# Armadillo 4.20

*Removing the armour: a naked animal*

{***W**ritten* ***b**y* **AndreaGeddon**}
{andreageddon@gmail.com}

[RET]
[www.reteam.org]

## INTRO

Armadillo is a nice animal, and a nice protection system! It is divided into two parts, the parent (debugger) and the child (debuggee). The code section is protected by a Copymem system and some code is emulated using nanomites. Also API emulation and redirection is used, a lot of nice features! The target I am using is Armadillo itself version 4.20. Let’s start reversing.

## STEP ONE: The ASM loader – Loading the loader

I start by having a general look at the PE, we can see that the import table seems to be correct (lots of API imports from kernel32, user32 and gdi32) but I don’t think it will be easy, so probably the real IT is loaded at runtime and is present elsewhere. Also, the section table is interesting

```
 entry point = 004c9000

  From     - To        Section  Size
 --------------------------------------
  00400000 - 00401000  PE       1000
  00401000 - 00448000  .text    47000
  00448000 - 0044B000  .rdata   3000
  0044B000 - 00479000  .data    2E000
  00479000 - 004C9000  .text1   50000
  004C9000 - 004D9000  .adata   10000
  004D9000 - 004F9000  .data1   20000
  004F9000 - 00639000  .pdata   140000
  00639000 - 007D0000  .rsrc    18E000
```

It seems that there are two executables joined in a unique PE. The entry point is located in the .adata section. This is the data section of the second executable. So we can assume that the initial code will be a pre loader that prepares the execution of the second executable, which resides in the .text1 section. Also this could mean that this second executable is the code of the protection core, and the first executable (.text) is the real application. Now that we have a general idea of the structure of the crypter, we can start tracing.

The beginning is as usual:

```
004C9000   PUSHAD
004C9001   CALL      004C9006
004C9006   POP       EBP
```

The address of the start of the loader is stored into ebp, so the loader can access static data in the loader referring to the displacement from ebp. This is done because a loader has no fixed addresses, its location is known only at runtime.
We soon find some polymorphic code:

```
004C9007   50                PUSH EAX
004C9008   51                PUSH ECX
004C9009   0FCA              BSWAP EDX
004C900B   F7D2              NOT EDX
004C900D   9C                PUSHFD
004C900E   F7D2              NOT EDX
004C9010   0FCA              BSWAP EDX
004C9012   EB 0F             JMP SHORT Armadill.004C9023
004C9014   B9 EB0FB8EB       MOV ECX,EBB80FEB
004C9019   07                POP ES
004C901A   B9 EB0F90EB       MOV ECX,EB900FEB
004C901F   08FD              OR CH,BH
004C9021   EB 0B             JMP SHORT Armadill.004C902E
004C9023   F2                PREFIX REPNE:
004C9024   EB F5             JMP SHORT Armadill.004C901B
004C9026   EB F6             JMP SHORT Armadill.004C901E
004C9028   F2                PREFIX REPNE:
004C9029   EB 08             JMP SHORT Armadill.004C9033
004C902B   FD                STD
004C902C   EB E9             JMP SHORT Armadill.004C9017
004C902E   F3                PREFIX REP:
004C902F   EB E4             JMP SHORT Armadill.004C9015
004C9031   FC                CLD
004C9032   E9 9D0FC98B       JMP 8C159FD4
004C9037   CA F7D1           RETF 0D1F7
004C903A   59                POP ECX
004C903B   58                POP EAX
```

This is usually the common form of the garbage code used by armadillo, there are some variants but the general structure is always similiar to this, so don’t waste time tracing these pieces of code, just skip them and focus on the real instructions. After this, we get to the first decryption routines

```
004C9157   XOR DWORD PTR DS:[EAX],EBX
004C9159   CMP DWORD PTR DS:[EAX],5478
004C915F   JE SHORT Armadill.004C9165
004C9161   XOR DWORD PTR DS:[EAX],EBX
004C9163   JMP SHORT Armadill.004C9151
004C9165   MOV ECX,9B4F
```

```
004C916A   AND EBX,0FF
004C9170   ADD EAX,3
004C9173   INC EAX
004C9174   XOR BYTE PTR DS:[EAX],BL
004C9176   DEC ECX
004C9177   JNZ SHORT Armadill.004C9173
004C9179   JMP SHORT Armadill.004C917F
```

This simply decrypts some of the following code (starting at 004c917f). We will encounter a lot of decryption routines. Now we see some anti-debug code:

```
004C917F   PUSH EAX
004C9180   CALL Armadill.004C91A9
004C9185   MOV ECX,DWORD PTR SS:[ESP+C]
004C9189   ADD DWORD PTR DS:[ECX+B8],2
004C9190   XOR EAX,EAX
004C9192   MOV DWORD PTR DS:[ECX+4],EAX
004C9195   MOV DWORD PTR DS:[ECX+8],EAX
004C9198   MOV DWORD PTR DS:[ECX+C],EAX
004C919B   MOV DWORD PTR DS:[ECX+10],EAX
004C919E   MOV DWORD PTR DS:[ECX+14],EAX
004C91A1   MOV DWORD PTR DS:[ECX+18],155
004C91A8   RETN
004C91A9   XOR EAX,EAX
004C91AB   PUSH DWORD PTR FS:[EAX]
004C91AE   MOV DWORD PTR FS:[EAX],ESP
004C91B1   XOR EAX,EAX
004C91B3   POP DWORD PTR DS:[EAX]
004C91B5   POP DWORD PTR FS:[0]
```

This code sets an exception handler (located at 004c9185) and at the line 004c91b3 it accesses a zero address to trigger the seh. In the seh we can see that the code zeroes registers (in the context provided by exception handler parameter) dr0, dr1, dr2, dr3, dr6 and sets in dr7 the bits corresponding to L0 – L3 and LE. In practice this interfers with the debugger, avoiding hardware memory breakpoints which rely on dr0-dr3. After this the exception handler quits returning the exception continuable value (0), so the execution is restored to the line 004c91b5 and the program continues running. Note that in the line 004c91b5 the unlinking of the exception record made with opcode 64:67:8F06 0000 sometimes causes troubles to ollydebug, in other packers I got troubles due to a massive use of these superfluous opcodes.
We continue on and we find some other junk code:

```
004C91C1   ...
...
004C91E7   POPAD
004C91E8   MOV EBX,DWORD PTR SS:[EBP+EDI*4+9CBA]
004C91EF   ADD EBX,EBP
004C91F1   MOV EDX,205
```

```
004C91F6   ADD EDX,EBP
004C91F8   PUSH EDX
004C91F9   CALL Armadill.004C9235
004C91FE   CALL Armadill.004C968C
004C9203   JMP EBX
004C9205   CMP EDI,4
004C9208   JNZ SHORT Armadill.004C91C1
```

This code dispatches the execution using a “call table” located at ebp+09CBA. EDI is the counter, we execute four functions. All this code is just used to decrypt the code that follows. At 004C91E8 in ebx is stored the pointer to the function taken from the function array, pointer which is relative to the loader starting offset, so ebp must be added to obtain the real address of the function. The if we trace the call to 004C9235 we find just a “JMP EBX” there, so the code jumps to the function.
The various functions are quite easy:

```
1°
  004D2C7F   MOV EAX,238
  004D2C84   RETN

2°
  004D2BFD   ADD EAX,EBP
  004D2BFF   RETN

3°
  004D2C44   MOV ECX,98AF
  004D2C49   RETN

4°
  004D2B0D   INC BYTE PTR DS:[EAX]
  004D2B44   ADD BYTE PTR DS:[EAX],CL
  004D2B6C   ROR BYTE PTR DS:[EAX],6
  004D2BA4   XOR BYTE PTR DS:[EAX],CL
  004D2BCC   INC EAX
  004D2BCD   DEC ECX
  004D2BCE   TEST ECX,ECX
  004D2BD0   JNZ Armadill.004D2AE7
  004D2BD6   RETN
```

This is the code without all the garbage polymorphic stuff. Basically it gets the relative pointer of the encrypted block in eax, relocates it by adding the loader base address, gets the size of the encrypted data in ecx, and then goes to the decrypt cycle, which performs a inc-add-ror-xor loop on each byte. The code being decrypted is located at 004C9238 and its length is 98AF bytes. You will find a lot of these loops, all with this similar structure (there are some variants), so just avoid tracing them and go directly to the following code that will be decrypted (usually you just need to go beyond the cmp edi, xx).
After another call table of eight functions, we arrive to:

```
004C9894   MOV ESI,EAX
004C9896   AND ESI,FFFF0000
004C989C   MOV EBX,0BF1
004C98A1   ADD EBX,EBP
004C98A3   CALL EBX
004C98A5   PUSHAD
```

EBX is the address of a function that returns the base address of kernel32, its parameter is passed in esi and is the return address of the code that called the entry point of the exe from kernel32. This return address was incremented by 0x13800, I don’t know why, probably for some win9x compatibility problem. Of course the kernel32 base address is calculated by the code and not using GetModuleHandle or LoadLibrary. As we expect, after this the code:

```
004C999B   PUSH EAX
004C999C   CALL Armadill.004C9B04
```

finds the address of GetProcAddress,

```
004C9A40   PUSH EAX
004C9A41   CALL Armadill.004C9B04
```

finds the address of LoadLibraryA

```
004C9ADD   CALL EAX
```

makes a call to LoadLibraryA to have user32 handle

```
004C9AF0   PUSH EAX
004C9AF1   MOV EAX,DWORD PTR SS:[EBP+1608]
004C9AF7   CALL EAX
```

get the address of FindWindowA. Now

```
004C9EDD   INC BYTE PTR DS:[ESI]
004C9EDF   ROR BYTE PTR DS:[ESI],4
004C9EE2   XOR BYTE PTR DS:[ESI],CL
004C9EE4   INC ESI
004C9EE5   LOOPD SHORT Armadill.004C9EDD
004C9EE7   POPAD
```

this code decrypts a pseudo import table:

```
004CA471  49 73 44 65 62 75 67 67 65 72 50 72 65 73 65 6E
IsDebuggerPresen
004CA481  74 00 57 72 69 74 65 50 72 6F 63 65 73 73 4D 65
t.WriteProcessMe
004CA491  6D 6F 72 79 00 52 65 61 64 50 72 6F 63 65 73 73
mory.ReadProcess
```

```
004CA4A1  4D 65 6D 6F 72 79 00 57 61 69 74 46 6F 72 44 65
Memory.WaitForDe
004CA4B1  62 75 67 45 76 65 6E 74 00 47 65 74 56 65 72 73
bugEvent.GetVers
004CA4C1  69 6F 6E 00 47 65 74 4D 6F 64 75 6C 65 48 61 6E
ion.GetModuleHan
004CA4D1  64 6C 65 41 00 47 65 74 43 6F 6D 6D 61 6E 64 4C
dleA.GetCommandL
004CA4E1  69 6E 65 41 00 47 65 74 50 72 6F 63 41 64 64 72
ineA.GetProcAddr
004CA4F1  65 73 73 00 4C 6F 61 64 4C 69 62 72 61 72 79 41
ess.LoadLibraryA
004CA501  00 47 65 74 53 74 61 72 74 75 70 49 6E 66 6F 41
.GetStartupInfoA
004CA511  00 57 72 69 74 65 46 69 6C 65 00 53 65 74 45 6E
.WriteFile.SetEn
004CA521  76 69 72 6F 6E 6D 65 6E 74 56 61 72 69 61 62 6C
vironmentVariabl
004CA531  65 41 00 47 65 74 45 6E 76 69 72 6F 6E 6D 65 6E
eA.GetEnvironmen
004CA541  74 56 61 72 69 61 62 6C 65 41 00 47 65 74 43 75
tVariableA.GetCu
004CA551  72 72 65 6E 74 50 72 6F 63 65 73 73 49 64 00 53
rrentProcessId.S
004CA561  6C 65 65 70 00 43 72 65 61 74 65 46 69 6C 65 41
leep.CreateFileA
004CA571  00 43 72 65 61 74 65 54 6F 6F 6C 68 65 6C 70 33
.CreateToolhelp3
004CA581  32 53 6E 61 70 73 68 6F 74 00 50 72 6F 63 65 73
2Snapshot.Proces
004CA591  73 33 32 46 69 72 73 74 00 50 72 6F 63 65 73 73
s32First.Process
004CA5A1  33 32 4E 65 78 74 00 43 6C 6F 73 65 48 61 6E 64
32Next.CloseHand
004CA5B1  6C 65 00 00 00 00 00 00 00 00 00 00 00 00 00 00
le..............
```

Now every API address is stored in the import array, which starts at [EBP + 15B4]. I am talking of import array because this is not a real import table. It’s just a list of functions that will be used by the loader. Note that in every API about first five bytes are checked

```
004C9F51   MOV EDI,EAX
004C9F53   MOV ECX,4
004C9F58   MOV EAX,660
004C9F5D   SHR EAX,3
004C9F60   REPNE SCAS BYTE PTR ES:[EDI]
004C9F62   TEST ECX,ECX
004C9F64   JE SHORT Armadill.004C9F7D
```

0x660 shr 3 = 0xCC, so if you want to set a breakpoint on these api’s you will need to put it a few lines after the API entry point, otherwise armadillo will set an internal error status value

```
004C9F6E   MOV DWORD PTR SS:[EBP+9620],6675636B
```

6675636B = ascii for “fuck”, so the execution will terminate. Whenever you see the code keeping that error value, it means that you are in the wrong place! After the import addresses are built, the import names are re-crypted. Here we come:

```
004CA81C   CALL DWORD PTR SS:[EBP+15F0]
```

an interesting call to CreateFileA to open the file \\.\BCMDMCCP. This should be the driver of SpywareBlaster, so maybe armadillo has tampering problems with it. Now there is a really annoying part!

There are about ONE HUNDRED decryption layers; all are made with the call method we have seen above. Also seh are used among decryption routines. Note that the first of these layers will trigger a seh that will be resumed at line 004d25c5; where there is an invalid op code. Ollydbg will panice with this.  You will need to trace it with shift-f9 to go on without problems. If you want you can trace all the hundred layers, but if you want to save your precious time then after all these layers there is a call to GetVersion, so just break on it and it’s done. Get out from GetVersion and we come to:

```
004CEB6A   CPUID
004CEB6C   RDTSC
```

The important thing is RDTSC. Soon after this we have a seh trigger.
Inside we find:

```
004CF297   CPUID
004CF299   RDTSC
004CF29B   SUB EAX,DWORD PTR SS:[ESP]
004CF29E   ADD ESP,4
004CF2A1   CMP EAX,5FFFFFF
004CF2A6   JB SHORT Armadill.004CF2AF
004CF2A8   ADD DWORD PTR DS:[ECX+B8],67
004CF2AF   XOR EAX,EAX
004CF2B1   RETN
```

another RDTSC. This instruction reads the timestamp counter of the CPU (that is, the number of cycles). So, in the seh it reads again the timestamp counter and subtracts from it the previously read timestamp value. If the difference is bigger than 0x5FFFFFFF then too much time is passed from one rdtsc to the other, and this means that someone is probably tracing the code (since tracing is

much more slower than real execution), so it tries to do something on context.eip, but the ecx pointer is zeroed, giving you another exception inside the exception handler itself.
Ok the end is near! We arrive to:

```
004CF499   ROR BYTE PTR DS:[EBX],CL
004CF49B   XOR BYTE PTR DS:[EBX],CL
004CF49D   INC EBX
004CF49E   DEC ECX
004CF49F   TEST ECX,ECX
004CF4A1   JNZ SHORT Armadill.004CF499
```

which is the code that decrypts the .text1 section.  This is the code section of the protection. This is first layer.
We find the second at:

```
004CF5F5   XOR BYTE PTR DS:[EBX],AL
004CF5F7   INC EBX
004CF5F8   DEC ECX
004CF5F9   TEST ECX,ECX
004CF5FB   JNZ SHORT Armadill.004CF5F5
```

You can now look at 00479000 and you will see all the code is decrypted. So we must trace just a few instructions and we find:

```
004CF904   POPAD
004CF905   JMP EBX
```

where ebx is 004B4BE3. This is a jump to the code in the .text1 section, so the boring ASM loader is now finished! You can see in this entry point the standard msvc code for WinMain, so you now can dump the executable and disassemble it and you will see the disassembly of the protection code. Let’s go for it.

## STEP 2: Fatherland – a parent takes care of his child

Armadillo is now running the code that will become the parent process. The same code will also split into the child process which we will see it later. First of all we let the armadillo run under Ollydbg.  We will get a crash of olly itself: it seems nowadays everyone is using this trick, which consists of sending a malformed debug string (lots of %s) which will cause a buffer overflow. So to avoid any problems, put a breakpoint on OutputDebugStringA and modify the string parameter that is passed to it into a simple string (for example “hi\0” ). There are two calls to OutputDebugStringA. Ok, at the beginning the loader is checking some string (INFO, REGISTER and so on) they are the parameters passed to the command line, going on we will find this:

```
004A07CD   MOV EAX,DWORD PTR DS:[4D93B4]
004A07D2   XOR EAX,DWORD PTR DS:[4D93C8]
```

```
004A07D8   PUSH EAX
004A07D9   CALL DWORD PTR DS:[<&KERNEL32.GetCurrent>;
kernel32.GetCurrentProcessId
004A07DF   PUSH EAX
004A07E0   PUSH Armadill.004D9674                  ; ASCII
"%X::DA%08X"
004A07E5   LEA ECX,DWORD PTR SS:[EBP-128]
004A07EB   PUSH ECX
004A07EC   CALL Armadill.004B47AA
004A07F1   ADD ESP,10
004A07F4   LEA EDX,DWORD PTR SS:[EBP-128]
004A07FA   PUSH EDX
004A07FB   PUSH 0
004A07FD   PUSH 1F0001
004A0802   CALL DWORD PTR DS:[<&KERNEL32.OpenMutexA>;
kernel32.OpenMutexA
004A0808   TEST EAX,EAX
004A080A   JE SHORT Armadill.004A0810
004A080C   MOV BYTE PTR SS:[EBP-24],0
004A0810   MOV EAX,DWORD PTR SS:[EBP-24]
```

It uses the string %X::DA%08X and transforms it in **PID**::DA**NUMBER** where PID is the process id of the actual current (parent) process and number is usually a fixed number given from the line 004A07CD (it is xored). The final string is so composed, for example for me it is 938::DAC858ADB2. Of course pid will be different at every execution. This string will be the mutex name used for the OpenMutexA function. Why does it try to open a mutex that has never been created??? Simple, this mutex will be checked also in the child process, so this is the point where the process knows if it must be launched in parent or child mode. If the mutex does not exist, the process will be parent, otherwise it will be child. Now we are going to trace the parent process, later we will see how to trace the child process. We soon find other similar code:

```
004A0C04   CALL DWORD PTR DS:[<&KERNEL32.OpenMutexA>;
kernel32.OpenMutexA
004A0C0A   TEST EAX,EAX
004A0C0C   JNZ Armadill.004A0E8C
```

Same as above, there are these two checks on the mutex before the code splits to child / parent flow. After this there is a call to IsDebuggerPresent, so it would be better if you set the PEB.BeingDebugged flag to zero every time you want to trace armadillo.
NOTE:
On winxpsp2 the PEB and TEB are no more at fixed addresses: usually PEB was always at 7FFDF000 and first TEB was 0x1000 bytes before PEB. Now they are in random addresses, so to find the PEB when the process starts you can look at TEB.Peb (TEB + 0x30), since the TEB address is indicated near the FS segment address base in Ollydbg.

Soon after this we see:

```
004A0D55   MOV EAX,DWORD PTR FS:[30]
004A0D5B   MOVZX EAX,BYTE PTR DS:[EAX+2]
004A0D5F   OR AL,AL
004A0D61   JNZ SHORT Armadill.004A0D81
```

A direct check to PEB.BeingDebugged without passing from IsDebuggerPresent, so don’t use breaks on such an API. Now let’s enter in this call.

```
004A0E3C   push    ecx
004A0E3D   call    004A35D8
```

In the following lines you see some calculus, just skip it, we will analyze it later; for now keep in mind what you have seen in those lines, that is a CreateFileMapping and some memory working on it and pointers adjustment. We come to a fundamental code:

```
004A4B05   CALL DWORD PTR DS:[<&KERNEL32.GetModuleF>;
kernel32.GetModuleFileNameW
004A4B0B   TEST EAX,EAX
004A4B0D   JNZ SHORT Armadill.004A4B16
004A4B0F   XOR AL,AL
004A4B11   JMP Armadill.004A74D2
004A4B16   MOV EAX,DWORD PTR DS:[4E0310]
004A4B1B   PUSH EAX
004A4B1C   LEA ECX,DWORD PTR SS:[EBP-7B4]
004A4B22   PUSH ECX
004A4B23   PUSH 0
004A4B25   PUSH 0
004A4B27   PUSH 4
004A4B29   PUSH 1
004A4B2B   PUSH 0
004A4B2D   PUSH 0
004A4B2F   CALL DWORD PTR DS:[<&KERNEL32.GetCommand>;
kernel32.GetCommandLineW
004A4B35   PUSH EAX
004A4B36   LEA EDX,DWORD PTR SS:[EBP-770]
004A4B3C   PUSH EDX
004A4B3D   CALL DWORD PTR DS:[<&KERNEL32.CreateProc>;
kernel32.CreateProcessW
```

Armadillo re-executes itself. The process being created is suspended but it’s not created in debug mode. In the call

```
004A4C01   CALL Armadill.004A906D
```

Armadillo with ReadProcessMemory reads two bytes from the child process (the two bytes at entry point) and changes them into EB FE with a WriteProcessMemory, which is a self jumping instruction. This is made so that the child process once running will loop its execution on the entry point. Once we get out from that call we find

```
004A4C13   CALL Armadill.004A929B
```

This is just a ResumeThread, Sleep, SuspendThread, used to run the program until it loops on the entry point.

```
004A4C25   CALL DWORD PTR DS:[<&KERNEL32.ResumeThre>;
kernel32.ResumeThread
004A4C2B   MOV EAX,DWORD PTR DS:[4E0310]
004A4C30   MOV ECX,DWORD PTR DS:[EAX+8]
004A4C33   PUSH ECX
004A4C34   CALL DWORD PTR DS:[<&KERNEL32.DebugActiv>;
kernel32.DebugActiveProcess
004A4C3A   MOV DWORD PTR SS:[EBP-20],EAX
004A4C3D   MOV EDX,DWORD PTR DS:[4E0310]
004A4C43   MOV EAX,DWORD PTR DS:[EDX+4]
004A4C46   PUSH EAX
004A4C47   CALL DWORD PTR DS:[<&KERNEL32.SuspendThr>;
kernel32.SuspendThread
004A4C4D   MOV ECX,DWORD PTR SS:[EBP-3C]
004A4C50   PUSH ECX
004A4C51   PUSH 0
004A4C53   MOV EDX,DWORD PTR DS:[4E0310]
004A4C59   PUSH EDX
004A4C5A   CALL Armadill.004A906D
```

The parent process attaches as a debugger to the child process. The call at 004A4C5A uses that function again to restore the bytes at the entry point removing the self-jumping op code. Again there is some calculus that for now we will skip until we arrive to

```
004A4DD9   CALL DWORD PTR DS:[<&KERNEL32.WaitForDeb>;
kernel32.WaitForDebugEvent
004A4DDF   TEST EAX,EAX
004A4DE1   JE Armadill.004A7493
```

OK, the parent process is waiting for the child events, so of course we are going to look what kind of events armadillo is interested in, and why. Just in case you miss the event constant identifiers, here it is a little list:

```
--------------------------------------------------------
EXCEPTION_DEBUG_EVENT        1
CREATE_THREAD_DEBUG_EVENT    2
CREATE_PROCESS_DEBUG_EVENT   3
EXIT_THREAD_DEBUG_EVENT      4
EXIT_PROCESS_DEBUG_EVENT     5
LOAD_DLL_DEBUG_EVENT         6
UNLOAD_DLL_DEBUG_EVENT       7
OUTPUT_DEBUG_STRING_EVENT    8
```

```
RIP_EVENT                   9

EXCEPTION_GUARD_PAGE                  80000001
EXCEPTION_DATATYPE_MISALIGNMENT       80000002
EXCEPTION_BREAKPOINT                  80000003
EXCEPTION_SINGLE_STEP                 80000004

EXCEPTION_ACCESS_VIOLATION            C0000005
EXCEPTION_IN_PAGE_ERROR               C0000006
EXCEPTION_ILLEGAL_INSTRUCTION         C000001D
EXCEPTION_NONCONTINUABLE_EXCEPTION    C0000025
EXCEPTION_INVALID_DISPOSITION         C0000026
--------------------------------------------------------
```

in [ebp-0A24] you have the event type, so following the code and eliminating all the garbage you can make a map of the event handlers:

```
004a4f1f - EXCEPTION_DEBUG_EVENT
    004a4f67 - EXCEPTION_GUARD_PAGE
    004a5635 - EXCEPTION_ACCESS_VIOLATION
    004a5adf - EXCEPTION_BREAKPOINT
    004a6668 - STATUS_INVALID_HANDLE
    004a6668 - EXCEPTION_STACK_OVERFLOW
    004a6668 - STATUS_HANDLE_NOT_CLOSEABLE

004a6682 - CREATE_THREAD_DEBUG_EVENT

004a670b - EXIT_THREAD_DEBUG_EVENT

004a676c - EXIT_PROCESS_DEBUG_EVENT

004a78d6 - OUTPUT_DEBUG_STRING_EVENT

004a68ed - RIP_EVENT

004a6901 - CREATE_PROCESS_DEBUG_EVENT

004a6e0b - LOAD_DLL_DEBUG_EVENT

004a7446 - ContinueDebugEvent
```

the ones we are interested in are the EXCEPTION_GUARD_PAGE and EXCEPTION_BREAKPOINT. The rest are not important for our purposes, they regard the normal debugging cycle.  For example, the createthread event keeps track of newly created threads and stores their handles, and so on. So for now let’s examine the event sequence, until we see the use of the two handlers we are interested in:

```
CREATE_PROCESS
LOAD_DLL
```

```
LOAD_DLL
LOAD_DLL
LOAD_DLL
CREATE_THREAD
EXCEPTION_DEBUG_EVENT  EXCEPTION_BREAKPOINT (first time its
ignored)
------------------------------------------------------------------
--
EXIT_THREAD_DEBUG_EVENT  calls exit thread handler
EXCEPTION_ACCESS_VIOLATION  (violations in the loader, pop fs:eax
and other useless stuff)
   does nothing, goes to continuedebugevent
EXCEPTION_ACCESS_VIOLATION (4cca84, pop fs:eax in loader)
   does nothing, goes to continuedebugevent
CREATE_THREAD
LOAD_DLL (005b1800 uxtheme.dll)
LOAD_DLL (77be0000 msvcrt)
LOAD_DLL (77f40000 advapi32)
LOAD_DLL (77da0000 rpcrt4)
LOAD_DLL (5d4d0000 comctl32.dll)
LOAD_DLL (76360000 comdlg32.dll)
LOAD_DLL (77e90000 shlwapi.dll)
LOAD_DLL (7c9d0000 shell32.dll)
LOAD_DLL (773a0000 comctl32.dll relocated)
LOAD_DLL (770f0000 oleaut32.dll)
LOAD_DLL (774b0000 ole32.dll)
  call to OutputDebugString of the father
  call to OutputDebugString of the father
LOAD_DLL (71a30000 ws2_32.dll)
LOAD_DLL (71a20000 ws2help.dll)
LOAD_DLL (66BB0000 inetmib1.dll)
LOAD_DLL (72d60000 iphlpapi.dll)
LOAD_DLL (71ef0000 snmpapi.dll)
LOAD_DLL (71a50000 wsock32.dll)
LOAD_DLL (76d00000 mprapi.dll)
LOAD_DLL (77c90000 activeds.dll)
LOAD_DLL (76dd0000 adsldpc.dll)
LOAD_DLL (5bc70000 netapi32.dll)
LOAD_DLL (76f20000 wldap32.dll)
LOAD_DLL (76ae0000 atl.dll)
LOAD_DLL (76e40000 rtutils.dll)
LOAD_DLL (71b80000 samlib.dll)
LOAD_DLL (778f0000 setupapi.dll)
EXCEPTION_ACCESS_VIOLATION 00e4e065 in security.dll (ghost dll)
  .text:1002E063                xor     eax, eax
  .text:1002E065                mov     [eax], eax
  father does nothing
EXCEPTION_ACCESS_VIOLATION  00e4e2ca  same as above
EXCEPTION_ACCESS_VIOLATION  same as above, and so all the
following
EXCEPTION_ACCESS_VIOLATION  00e4e065
EXIT_THREAD
```

```
EXCEPTION_ACCESS_VIOLATION  00e4e065
EXCEPTION_ACCESS_VIOLATION  00e4e065
EXCEPTION_ACCESS_VIOLATION  00e4e065
EXCEPTION_ACCESS_VIOLATION  00e4e065
EXCEPTION_ACCESS_VIOLATION  00e4e065
LOAD_DLL (73390000 msvbvm60.dll)
EXCEPTION_ACCESS_VIOLATION  00e4e4fb
EXCEPTION_ACCESS_VIOLATION  00e4edbf
EXCEPTION_ACCESS_VIOLATION  00e4eff0
EXCEPTION_ACCESS_VIOLATION  00e4f221
EXCEPTION_ACCESS_VIOLATION  00e4e065
EXCEPTION_ACCESS_VIOLATION  ooe4f452
EXCEPTION_ACCESS_VIOLATION  00e4e065
EXCEPTION_ACCESS_VIOLATION  00e4f683
EXCEPTION_ACCESS_VIOLATION  00e4e2ca
EXCEPTION_ACCESS_VIOLATION  00e4e065
EXCEPTION_ACCESS_VIOLATION   "
LOAD_DLL (77bd0000 version.dll)
EXCEPTION_ACCESS_VIOLATION  00e4e065
EXCEPTION_ACCESS_VIOLATION  00e4e065
EXCEPTION_ACCESS_VIOLATION  00e4e065
EXCEPTION_ACCESS_VIOLATION  00e4e065
EXCEPTION_ACCESS_VIOLATION  00e4e065
EXCEPTION_ACCESS_VIOLATION  00e4e065
EXCEPTION_ACCESS_VIOLATION  00e4e065
EXCEPTION_ACCESS_VIOLATION  00e4e065
EXCEPTION_ACCESS_VIOLATION  00e4e4fb
EXCEPTION_ACCESS_VIOLATION  00e4e065
EXCEPTION_ACCESS_VIOLATION  00e4e065
EXCEPTION_ACCESS_VIOLATION  00e4e72c
EXCEPTION_ACCESS_VIOLATION  00e4e95d
--------------------------------------------------------
EXCEPTION_GUARD_PAGE  00441cc0! first exception, we found OEP!
```

Okay you should have a similar event sequence. All the DLL's you see are probably loaded when the Armadillo is building the REAL import table. The first exception breakpoint event is ignored from the parent as it's just the normal DebugBreak call inside the newly created process. It's far from the program code itself. The interesting thing is the first GUARD_PAGE exception that we have, at address 00441CC0.

**STEP 3: COPYMEM2 – swimming in a page pool**

If we look at the memory of the child process at 00441CC0 we see:

```
seg000:00441CC0                push    ecx
seg000:00441CC1                mov     ebp, [edi]
seg000:00441CC3                mov     dl, 0FBh
seg000:00441CC5                push    404D13h
seg000:00441CCA                stosd
```

```
seg000:00441CCB                 test    [esi], bl
seg000:00441CCD                 inc     esp
seg000:00441CCE                 retn
```

nonsense code! It is obvious that the page is encrypted. If we look at the process memory regions, for example with Lord-PE, we see that all code sections have the attribute GUARD_PAGE, except the one we are looking now. GUARD_PAGE in fact is a “one-shot access alarm” (I love this definition in the msdn!). However, all pages are in GUARD_PAGE, this means ALL PAGES ARE ENCRYPTED, and they will shoot a GUARD_PAGE notification when executed the first time. So this 00441CC0 is the FIRST line executed from the original code section.  This means that this is the original entry point of the packed program. The problem now is: how do we dump the program? If we let it run we are not sure that ALL pages will be decrypted, and in fact you can try and see that even executing the program some pages remain encrypted. The simplest idea is: we force armadillo to decrypt all pages. We must examine the PAGE_GUARD handler.
I will paste the most important lines of a generic page decryption:

```
004A4FB9   MOV ECX,DWORD PTR DS:[EAX+24]    ; get exception
address

004A4FC2   CMP DWORD PTR DS:[4E032C],0      ; check if the number
of the decrypted pages is 0

004A5187   MOV ECX,DWORD PTR SS:[EBP-A3C]   ; ecx = fault address
004A518D   SUB ECX,DWORD PTR DS:[4E0314]    ; ecx -= 00401000
004A5193   SHR ECX,0C                       ; ecx = page number
004A5196   MOV DWORD PTR SS:[EBP-A34],ECX

004A5363   CMP DWORD PTR SS:[EBP-A34],0     ; check that
004A536A   JL Armadill.004A5618             ; 0 <= page number <=
47
004A5370   MOV ECX,DWORD PTR SS:[EBP-A34]
004A5376   CMP ECX,DWORD PTR DS:[4E0328]
004A537C   JGE Armadill.004A5618
```

We see a variable (004e032c) here that keeps the number of decrypted pages (break on this handler more times and you see that number being increased), then the page number is calculated from the virtual address, so the program can use it as an index for the page array in the code section. There is some calculus then stepping into two calls there is the code that we need

```
004A7E49   PUSH EAX                         ; old protection
004A7E4A   PUSH 4                           ; PAGE_READWRITE
004A7E4C   PUSH 1000                        ; size of the region
004A7E51   MOV ECX,DWORD PTR SS:[EBP-14]
```

```
004A7E54   PUSH ECX                         ; virtual address of
the faulting guarded page
004A7E55   MOV EDX,DWORD PTR DS:[4E0310]
004A7E5B   MOV EAX,DWORD PTR DS:[EDX]
004A7E5D   PUSH EAX                         ; child process handle
004A7E5E   CALL DWORD PTR DS:[<&KERNEL32.VirtualPro>;
kernel32.VirtualProtectEx
```

The page of the faulting address is turned back in the PAGE_READWRITE attribute.

```
004A7EE1   LEA ECX,DWORD PTR SS:[EBP-8]
004A7EE4   PUSH ECX                         ; ptr to number of
bytes read
004A7EE5   PUSH 1000                        ; number o bytes to
read
004A7EEA   MOV EDX,DWORD PTR DS:[4E033C]
004A7EF0   PUSH EDX                         ; pBuffer
004A7EF1   MOV EAX,DWORD PTR SS:[EBP-14]
004A7EF4   PUSH EAX                         ; virtual address of
the faulting page
004A7EF5   MOV ECX,DWORD PTR DS:[4E0310]
004A7EFB   MOV EDX,DWORD PTR DS:[ECX]
004A7EFD   PUSH EDX                         ; child process handle
004A7EFE   CALL DWORD PTR DS:[<&KERNEL32.ReadProces>;
kernel32.ReadProcessMemory
```

The parent process reads the memory of the page that was accessed and was in GUARD_PAGE protection.

```
004A88D2   CMP EDX,DWORD PTR SS:[EBP-28]
004A88D5   JNB SHORT Armadill.004A88EF
004A88D7   MOV EAX,DWORD PTR SS:[EBP-24]
004A88DA   MOV CL,BYTE PTR DS:[EAX]
004A88DC   XOR CL,BYTE PTR SS:[EBP-20]
004A88DF   MOV EDX,DWORD PTR SS:[EBP-24]
004A88E2   MOV BYTE PTR DS:[EDX],CL
004A88E4   MOV EAX,DWORD PTR SS:[EBP-24]
004A88E7   ADD EAX,1
004A88EA   MOV DWORD PTR SS:[EBP-24],EAX
004A88ED   JMP SHORT Armadill.004A88CF
```

This is the cycle that decrypts the memory read from the faulting page. It’s not the only one, another one is at

```
004A89F3   MOV EAX,DWORD PTR SS:[EBP-4]
004A89F6   CMP EAX,DWORD PTR SS:[EBP-C]
004A89F9   JNB SHORT Armadill.004A8A13
004A89FB   MOV ECX,DWORD PTR SS:[EBP-4]
004A89FE   MOV EDX,DWORD PTR DS:[ECX]
004A8A00   XOR EDX,DWORD PTR SS:[EBP-2C]
004A8A03   MOV EAX,DWORD PTR SS:[EBP-4]
```

```
004A8A06   MOV DWORD PTR DS:[EAX],EDX
004A8A08   MOV ECX,DWORD PTR SS:[EBP-4]
004A8A0B   ADD ECX,4
004A8A0E   MOV DWORD PTR SS:[EBP-4],ECX
004A8A11   JMP SHORT Armadill.004A89F3
```

Once the page is decrypted, it is written back to the process

```
004A8CD6   LEA EDX,DWORD PTR SS:[EBP-8]
004A8CD9   PUSH EDX                           ; number of bytes
written
004A8CDA   PUSH 1000                          ; number of bytes to
write
004A8CDF   MOV EAX,DWORD PTR DS:[4E033C]
004A8CE4   PUSH EAX                           ; pBuffer
004A8CE5   MOV ECX,DWORD PTR SS:[EBP-14]
004A8CE8   PUSH ECX                           ; address of the
faulting page
004A8CE9   MOV EDX,DWORD PTR DS:[4E0310]
004A8CEF   MOV EAX,DWORD PTR DS:[EDX]
004A8CF1   PUSH EAX                           ; handle of the
child process
004A8CF2   CALL DWORD PTR DS:[<&KERNEL32.WriteProce>;
kernel32.WriteProcessMemory
```

Finally, the attribute of the page is changed again:

```
004A8D93   LEA ECX,DWORD PTR SS:[EBP-18]
004A8D96   PUSH ECX
004A8D97   MOV EDX,DWORD PTR SS:[EBP-10]
004A8D9A   PUSH EDX
004A8D9B   PUSH 1000
004A8DA0   MOV EAX,DWORD PTR SS:[EBP-14]
004A8DA3   PUSH EAX
004A8DA4   MOV ECX,DWORD PTR DS:[4E0310]
004A8DAA   MOV EDX,DWORD PTR DS:[ECX]
004A8DAC   PUSH EDX
004A8DAD   CALL DWORD PTR DS:[<&KERNEL32.VirtualPro>;
kernel32.VirtualProtectEx
```

The page is changed to PAGE_EXECUTE_READ.
Now see this interesting compare:

```
004A7ABA   MOV EDX,DWORD PTR DS:[4E032C]      ; number of
decrypted pages
004A7AC0   CMP EDX,DWORD PTR DS:[4D97E4]      ; pool threshold
004A7AC6   JLE Armadill.004A7CA5
```

If the number of decrypted pages is below the threshold, then the execution goes to the ContinueDebugEvent so that the exception is repaired, and the child process continues execution with the decrypted page. If the number of the decrypted pages goes above the threshold, then armadillo checks which pages are decrypted and how often they are used, then it re-encrypts them back to the process so that the number of decrypted pages is always under the threshold value. With this system the program is NEVER completely decrypted in memory, but at max only the page pool is decrypted at the same time. The page pool is the number of pages allowed to be simultaneously decrypted and present in memory. Usually the pool size is the half of the total number of code section pages. This means we can’t dump the executable. Besides, if we try to dump it, usually the dumpers will fail because of the GUARD_PAGE attribute interfering with the ReadProcessMemory used to dump the executable image.

The first thing I did was to write a decrypter for the pages, but the decryption uses random keys that change at every execution, so the simplest solution is to force the routine to decrypt all pages, then dump the decrypted child process. To do this, we simply raise the threshold value so it is never reached, then we modify the code to loop for every page, so, start debugging Armadillo, and arrive at the first PAGE_GUARD exception. I executed all of the handler until it’s end, at address 004a7446. Now the entry point page at 00441000 si decrypted, and we write our shell code.
I used the following steps:

[004d97e4] is set to 0x100. This is the threshold value, since the program has 0x48 code pages the re-encryption will never occur.
[004a7482] is set to 00400000. This is the page address that each time is passed as the faulting address.
I searched every occurrence of the faulting EIP given for this exception, and found it in these memory locations

0012bc40
0012bc4c
0012bc50
0012cd80
0012cd8c
0012cd90
0012cdbc

With every iteration the shell code must overwrite these with our page address (004a7482).
Here is the shell code:

```
004A7446   MOV EAX,DWORD PTR DS:[4A7484]   ; get page address
004A744B   ADD EAX,1000                    ; goto next page
004A7450   MOV DWORD PTR DS:[4A7484],EAX   ; set next page address
to all faultin address locations
004A7455   MOV DWORD PTR DS:[12BC40],EAX
004A745A   MOV DWORD PTR DS:[12BC4C],EAX
004A745F   MOV DWORD PTR DS:[12BC50],EAX
004A7464   MOV DWORD PTR DS:[12CD80],EAX
```

```
004A7469   MOV DWORD PTR DS:[12CD8C],EAX
004A746E   MOV DWORD PTR DS:[12CD90],EAX
004A7473   MOV DWORD PTR DS:[12CDBC],EAX
004A7478   CMP EAX,Armadill.00441000        ; loop from 00401000
to 00441000
004A747D   JLE Armadill.004A4F67            ; jump back to the
beginning of the page_guard handler
004A747F   JMP 004A7446                     ; we arrive here when
all pages before entry point are decrypted
```

This loop will cycle for all the pages until the entry point page, which is already decrypted, so we must not execute the handler on it. You just need to change a bit of the shell code to decrypt pages from the page after the entry point to the end of the section:

```
004A7446   MOV EAX,DWORD PTR DS:[4A7484]    ; now this starts
being 00441000 (entry point page)
004A744B   ADD EAX,1000                     ; and we go to next
page
004A7450   MOV DWORD PTR DS:[4A7484],EAX
004A7455   MOV DWORD PTR DS:[12BC40],EAX
004A745A   MOV DWORD PTR DS:[12BC4C],EAX
004A745F   MOV DWORD PTR DS:[12BC50],EAX
004A7464   MOV DWORD PTR DS:[12CD80],EAX
004A7469   MOV DWORD PTR DS:[12CD8C],EAX
004A746E   MOV DWORD PTR DS:[12CD90],EAX
004A7473   MOV DWORD PTR DS:[12CDBC],EAX
004A7478   CMP EAX,Armadill.00448000        ; loop until last code
section page
004A747D   JLE Armadill.004A4F67
```

After this shell code we finally will have decrypted all the code section. We can now dump the process and we have the code and data sections decrypted. Of course the exe will not run we must still fix some things.

**STEP 4: IMPORT TABLE – Rebuilding after war**

We fix the entry point of the dumped PE (oep is 00441CC0), and disassemble the dumped exe.
First thing I see is just the ep:

```
.text:00441CC0 55                      push    ebp             ;
sub_441CC0
.text:00441CC1 8B EC                   mov     ebp, esp
.text:00441CC3 6A FF                   push    0FFFFFFFFh
.text:00441CC5 68 D0 95 44 00          push    offset unk_4495D0
.text:00441CCA 68 5C 1A 44 00          push    offset
unknown_libname_1
.text:00441CCF 64 A1 00 00 00 00       mov     eax, large fs:0
```

```
.text:00441CD5 50                          push    eax
.text:00441CD6 64 89 25 00 00 00 00        mov     large fs:0, esp
.text:00441CDD 83 EC 58                    sub     esp, 58h
.text:00441CE0 53                          push    ebx
.text:00441CE1 56                          push    esi
.text:00441CE2 57                          push    edi
.text:00441CE3 89 65 E8                    mov     [ebp+var_18], esp
.text:00441CE6 FF 15 54 31 FC 00           call    dword ptr
ds:0FC3154h
```

This is again the standard entry point of a visual studio compiled exe. The call we see here should be a call to GetVersion, but instead of the API address, we find that horrible 0FC3154. This means that API addresses are redirected using a wrapper or something like this.

Also, we see from the op code that the import table information is completely destroyed, we don't have the FirstThunk or OriginalFirstThunk information.  Everything relies on the memory bridge at 0FC3154. You can look at the code and see that a lot of imports are redirected towards that memory bridge. So what are we waiting for? Let's dump that bridge from the child process. You can use LordPE to see the child process memory regions and their size. The region we are dumping goes from 00FC2000 to 00FCB000. It's just a data region, so with IDA you can transform all the bytes in dwords. I used this little script

```
static DwordZone(Start, End)
{
   auto i;
   for(i=Start; i<End; I = i+4)
   {
      MakeDword(i);
   }
}
```

Lad it as an idc file and call it from idc command window passing starting and ending addresses. All the bridge bytes will be dworded. We see a lot of zeroed memory, a lot of strange nonsense dwords, the interesting thing is this little piece:

```
 garbage
seg000:00FC3050                 dd 0E865B5h    ; redirection /
emulation
seg000:00FC3054                 dd 0E8665Ch    ; redirection /
emulation
seg000:00FC3058                 dd 77D1C2B2h   ; direct thunk
seg000:00FC305C                 dd 0E865F9h    ; redirection /
emulation
seg000:00FC3060                 dd 0E865ACh    ; redirection /
emulation
seg000:00FC3064                 dd 77D1FD80h   ; direct thunk
```

```
 ...
seg000:00FC35A8                 dd 77D18F9Dh   ; direct thunk
seg000:00FC35AC                 dd 0E865BBh    ; redirection /
emulation
seg000:00FC35B0                 dd 0E8A46Ah    ; redirection /
emulation
seg000:00FC35B4                 dd 0E8673Eh    ; redirection /
emulation
garbage
```

This is our IAT. We see API thunks (those 77xxxxxx addresses), and some other strange 00E8xxxx thunk. This makes me think that some api's are called directly, some others are wrapped by another memory layer (the 00E8* one). Let's look again with LordPE at the child process memory. This time our attention should be attracted by the memory around this bridge:

```
address     size        protect      state
-------------------------------------------
00E70000    00001000    RW           COMMIT
00E71000    00033000    XRW          COMMIT
00EA4000    00003000    R            COMMIT
00EA7000    00013000    RW           COMMIT
00EBA000    00007000    R            COMMIT
```

The first block is of 0x1000 bytes, then the second is an executable block... it seems this is a dll! We can look at the module list of the child process, but there is no module at imagebase 00E70000, probably this is a dll that armadillo loaded by itself. Let's dump it!

We look into LordPE and it is a normal dll, it has imports and exports. In the export window and we also see its name string: "security.dll". There is only one export: SetFunctionAddresses. Ok, let's disassemble this dll and look at 0E8A452:

```
.text:00E8A452 sub_E8A452      proc near               ; DATA
XREF: .data:00EA8F64
.text:00E8A452                 mov     eax, dword_EB7D60
.text:00E8A457                 retn
.text:00E8A457 sub_E8A452      endp
```

where

```
.data:00EB7D60 dword_EB7D60    dd 0A280105h
```

This function just returns the value A280105, which is the value returned by GetVersion. This is API emulation. Luckily there are only a few emulated APIs. Other kinds of APIs are wrapped with some types of wrappers; for example:

```
.text:00E899FE                 push    ebp
```

```
.text:00E899FF                 mov     ebp, esp
.text:00E89A01                 push    ecx
.text:00E89A02                 push    ebx
.text:00E89A03                 push    esi
.text:00E89A04                 push    edi
.text:00E89A05                 pusha
.text:00E89A06                 mov     edx, dword_EB7D58
.text:00E89A0C                 add     edx, 64h
.text:00E89A0F                 call    edx                 ; call
GetTickCount
.text:00E89A11                 mov     edx, dword_EB7D14
.text:00E89A17                 add     edx, 64h
.text:00E89A1A                 mov     ecx, 5
.text:00E89A1F
.text:00E89A1F loc_E89A1F:                             ; CODE
XREF: sub_E899FE+26
.text:00E89A1F                 cmp     byte ptr [edx], 0CCh
.text:00E89A22                 jz      short loc_E89A34
.text:00E89A24                 loop    loc_E89A1F
.text:00E89A26                 push    [ebp+arg_C]
.text:00E89A29                 push    [ebp+arg_8]
.text:00E89A2C                 push    [ebp+arg_4]
.text:00E89A2F                 push    [ebp+arg_0]
.text:00E89A32                 call    edx             ; call
MessageBoxA
.text:00E89A34
.text:00E89A34 loc_E89A34:                             ; CODE
XREF: sub_E899FE+24
.text:00E89A34                 mov     [ebp+var_4], eax
.text:00E89A37                 popa
.text:00E89A38                 mov     eax, [ebp+var_4]
.text:00E89A3B                 pop     edi
.text:00E89A3C                 pop     esi
.text:00E89A3D                 pop     ebx
.text:00E89A3E                 leave
.text:00E89A3F                 retn    10
```

This wrapper calls MessageBoxA, before it calls GetTickCount to fool automatic tracers. First five bytes of the API are checked to see if there is a breakpoint on it (0xCC byte). API addresses are stored in the security dll but are subtracted by 0x64 bytes. That’s why you see those add edx, 64h. In another kind of wrapper the API call is encrypted instead:

```
.text:00E8667A                 push    ebp
.text:00E8667B                 mov     ebp, esp
.text:00E8667D                 push    ebx
.text:00E8667E                 push    esi
.text:00E8667F                 mov     esi,
ds:EnterCriticalSection
.text:00E86685                 push    edi
.text:00E86686                 push    offset unk_EB7058
```

```
.text:00E8668B                  call    esi ; EnterCriticalSection
.text:00E8668D                  push    offset unk_EB7070
.text:00E86692                  call    esi ; EnterCriticalSection
 ...
.text:00E867A5                  xchg    eax, ecx
; encrypted block
.text:00E867A6                  push    edi
.text:00E867A7                  pop     edi
.text:00E867A8                  nop
.text:00E867A9                  xchg    eax, ecx
.text:00E867AA                  cmp     [esi+2CE2EBCBh], esi
.text:00E867B0                  sbb     cl, dl
.text:00E867B2                  popf
.text:00E867B3                  and     eax, 0F52E793Ch
.text:00E867B8                  rep aas
.text:00E867BA                  lodsd
.text:00E867BB                  xchg    eax, ebp
.text:00E867BC                  mov     ecx, 0F87CD732h
.text:00E867C1                  in      al, dx
 ...
.text:00E86842                  mov     esi,
ds:LeaveCriticalSection
.text:00E86848                  push    offset unk_EB7070
.text:00E8684D                  call    esi ; LeaveCriticalSection
.text:00E8684F                  push    offset unk_EB7058
.text:00E86854                  call    esi ; LeaveCriticalSection
.text:00E86856                  mov     eax, edi
.text:00E86858                  pop     edi
.text:00E86859                  pop     esi
.text:00E8685A                  pop     ebx
.text:00E8685B                  pop     ebp
.text:00E8685C                  retn    4
```

there are calls to Enter/LeaveCriticalSections (again it fools automatic tracers), then the call to the real API (in this case this wrapper calls LoadLibraryA) is in an encrypted code block. It is decrypted at runtime, the call is executed and then the block is re-crypted. There are some other simpler variants of these wrappers, where there are no tricks, or just some calls to time function. If you can't figure out what API is called you can assemble a call to the wrapper in the child process and see where it goes. This introduces a new problem!

**STEP 4.1: SECURITY.DLL – tracing the child process**

Having this self loaded DLL in the child process means we need to debug it. How can we debug the child if the father is part of its execution? Well we can at least debug the loader until the oep of the application. Then PAGE_GUARD and BREAKPOINT faults will not be handled and the program won't run, but that's not a problem for now. First of all start the parent with olly. We break on

WaitForDebugEvent and let it run. Ok, we break on the first debug event, that is CREATE_PROCESS. Coming out from the API we are at

```
004A4DD9   CALL DWORD PTR DS:[<&KERNEL32.WaitForDeb>;
kernel32.WaitForDebugEvent
004A4DDF   TEST EAX,EAX
004A4DE1   JE Armadill.004A7493
004A4DE7   MOV EAX,DWORD PTR SS:[EBP-204]
```

If we trace the CREATE_PROCESS event handler we find

```
004A0802   CALL DWORD PTR DS:[<&KERNEL32.OpenMutexA>;
kernel32.OpenMutexA
004A0808   TEST EAX,EAX
004A080A   JE SHORT Armadill.004A0810
```

going on we see

```
004A6948   PUSH EAX   ; mutex name
004A6949   PUSH 0
004A694B   PUSH 0
004A694D   CALL DWORD PTR DS:[<&KERNEL32.CreateMute>;
kernel32.CreateMutexA
```

where the mutex name is PID::DANUMBER, and the mutex we have seen above. So reassuming we have:

```
Start Armadillo Execution
Check for PID::DANUMBER mutex. Does it exist:
  NO:  go on,
       start armadillo again in debug mode,
       trap CREATE_PROCESS and in this handler create
PID::DANUMBER
       where PID is the child process PID.
  YES: this execution must switch to child mode,
       load security.dll,
       prepare other stuff for copymem and nanomites
       and then run the original program.
```

also two other mutexes are created, named DILLOOEP and DILLOCREATE. We execute this handler so that per-process initialization is done correctly. Note that we also encounter this line

```
004A6989   MOV EDX,DWORD PTR DS:[ECX+4]
004A698C   PUSH EDX
004A698D   CALL DWORD PTR DS:[<&KERNEL32.ResumeThre>;
kernel32.ResumeThread
```

It resumes the main thread of the application. You must not execute it, just set edx to zero so the ResumeThread will fail. Why? Simple.  Our idea is to detach the parent debugger, and attach ourselves as debugger. If we resume the thread, when we detach, the debugger will be shut down, and the thread suspend count will go to zero, running the thread before we can attach to it. Once this resume is failed, we let the Armadillo run, and wait for the second break on WaitForDebugEvent. We exit from the function, and when at the line 004A4DDF we assemble

```
004A4DDF   PUSH 0A64              ; pid of your child process
004A4DE4   CALL kernel32.DebugActiveProcessStop
```

Now the parent is detached. Don't close the parent debugged process. We open another instance of olly and attach to the child process (use its pid to recognize it). We will break on ntdll.DbgBreakPoint. We can set a bpx on program entry point (004C9000), and we must go to view->threads and resume the thread, because when we detached we missed the ContinueDebugEvent, so the thread suspend count was not updated.

Okay, once you resume the thread, you can run the child and it will break on the entry point. Now we can step through the child. We directly set a break on OpenMutexA, so we go to the parent / child forking code. We have two breaks, on the code we have just seen in step 2:

```
004A07FA   PUSH EDX
004A07FB   PUSH 0
004A07FD   PUSH 1F0001
004A0802   CALL DWORD PTR DS:[<&KERNEL32.OpenMutexA>;
kernel32.OpenMutexA
...
004A0BFC   PUSH EAX
004A0BFD   PUSH 0
004A0BFF   PUSH 1F0001
004A0C04   CALL DWORD PTR DS:[<&KERNEL32.OpenMutexA>;
kernel32.OpenMutexA
```

If you correctly executed the CREATE_PROCESS event handler in the parent, these function will return valid handles. So now that the mutexes are valid, the execution will go to the child flow. First thing we find is this call is:

```
004A0EAC   CALL Armadill.0049F70B
```

Step over it (remember that you must have a break on OutputDebugStringA to avoid olly crash) and you will see that after this call the dll is mapped and ready to be used (look in view->memory and see the region in 00E70000). We continue and see some calls to the dll, until we get to

```
004A0ED3   CALL DWORD PTR DS:[4DFE30]
```

This is a call to the dll, then if you let the program run you will arrive at

```
00441CC0   INC EDI
00441CC1   JMP FAR D984:09ED26B8
00441CC8   PUSH ESI
00441CC9   POPAD
00441CCA   CMP AL,10
```

This is the entry point, breaking on access. Now you can assemble the calls to the redirected apis, and trace them if needed. Of course you can trace the dll loading, just enter in the call to 0049F70B. The code allocates memory for the dll then maps it, then there is a call to:

```
0049F7E4   PUSH 0
0049F7E6   PUSH 1
0049F7E8   MOV EDX,DWORD PTR DS:[4E003C]
0049F7EE   PUSH EDX
0049F7EF   CALL DWORD PTR DS:[4E0040]    ;  dll entry point
```

the address 00EA31DD, which is the entry point of the dll. If you dump the dll, the image base will be 0x10000000, change it to 00E70000 so when disassembling it you will have the same memory addresses you see in the child process. NOTE that 0x00E70000 is a dynamic allocation, it will probably be a different address on other computers. After this there is a call to the only exported API from that dll

```
0049F844   PUSH ECX
0049F845   PUSH Armadill.004A1948
0049F84A   PUSH Armadill.0049F6F9
0049F84F   PUSH Armadill.004A1CD7
0049F854   PUSH Armadill.0047CF90
0049F859   PUSH Armadill.0047C32E
0049F85E   PUSH Armadill.0047BF2B
0049F863   PUSH Armadill.0047BED2
0049F868   PUSH Armadill.0047BEB8
0049F86D   PUSH Armadill.0047B7D0
0049F872   MOV EDX,DWORD PTR DS:[4E0158]        ;
Armadill.00400000
0049F878   PUSH EDX
0049F879   CALL DWORD PTR SS:[EBP-58]
;SetFunctionAddresses
```

it sets some function pointer.

```
0049F95B   LEA ECX,DWORD PTR SS:[EBP-3C]
0049F95E   PUSH ECX
0049F95F   CALL DWORD PTR SS:[EBP-10]
```

Here there are the calls to OutputDebugStringA. First thing I want to know is emulated apis. I know that there is the emulation code for GetVersion, we have seen above that the GetVersion value is placed at 00EB7D60, so we go to the call to the security.dll entry point, and set a memory breakpoint on 00EB7D60. We will break into

```
00E95AC3   CALL DWORD PTR DS:[EA4224]              ;
kernel32.GetVersion
00E95AC9   MOV DWORD PTR DS:[EB7D60],EAX
```

Okay, that’s what we wanted to know. If we continue stepping we find some other emulated api

```
00E95B1A   CALL DWORD PTR DS:[EA4220]              ;
kernel32.GetCommandLineA
00E95B20   MOV DWORD PTR DS:[EB7D64],EAX
...
```

Our emulation information is built, now we know emulated apis:

```
.data:00EB7D5C dword_EB7D5C    dd 140000h              ; DATA
XREF: sub_E8A458 r
.data:00EB7D5C                                         ;
GetProcessHeap
.data:00EB7D60 dword_EB7D60    dd 0A280105h            ; DATA
XREF: sub_E8A452 r
.data:00EB7D60                                         ;
GetVersion
.data:00EB7D64 dword_EB7D64    dd 142378h              ; DATA
XREF: sub_E8A45E r
.data:00EB7D64                                         ;
GetCommandLineA
.data:00EB7D68 dword_EB7D68    dd 4E4h                 ; DATA
XREF: sub_E8A464 r
.data:00EB7D68                                         ; GetACP
```

only four. We are lucky. In the same way we find other iat information:

```
00E954BC   MOV EAX,DWORD PTR DS:[ EA422C ]
00E954C1   SUB EAX,64
00E954C4   MOV DWORD PTR DS:[EB7CFC],EAX           ;
kernel32.7C80FEC9

00E954D4   MOV EAX,DWORD PTR DS:[EA41E0]
00E954D9   SUB EAX,64
00E954DC   MOV DWORD PTR DS:[EB7D00],EAX           ;
kernel32.7C8100B5
...
```

and so on.

## STEP 4.2: IMPORTS – Analysis and rebuilding

Now we know the emulated apis, and we know how to determine redirected apis. What we need to do is change the address of the redirection bridge with the address of the real API thunk in the program’s original code section.
We have:

```
 ORIGINAL CODE                BRIDGE            SECURITY.DLL
 ------------------------------------------------------------
 dword ptr ds:0FC3154h  -->  Redirection  -->  API wrapper
                             API Thunk         API emulation
```

The first thing we do is scan the original code and see what imports in the code are redirected to the security.dll. I have lost the code of such a scanner, but you can easly adapt the following code of the rebuilder to make it. Well, once the scan is complete you have a list of 0x00E8xxxx addresses that are used in the code section. With the previous step we have seen how to find the API to which they point to, so we will have the following list:

| | |
|---|---|
| 0x00E86EAE | GetEnvironmentVariableA |
| 0x00E880CB | WriteFile |
| 0x00e899fe | MessageBoxA |
| 0x00e8667a | LoadLibraryA |
| 0x00e85aa1 | GetProcAddress |
| 0x00e89e64 | RegCreateKeyExA |
| 0x00e898b2 | EndDialog |
| 0x00e8a74c | DialogBoxParamA |
| 0x00e89a42 | GetWindowTextA |
| 0x00e88414 | CloseHandle |
| 0x00e8a23e | RegQueryValueA |
| 0x00e8944d | FindFirstFileA |
| 0x00e87a04 | CreateFileA |
| 0x00e87e8e | ReadFile |
| 0x00e882a0 | SetFilePointer |
| 0x00e874f6 | CreateThread |
| 0x00e89975 | GetModuleHandleA |
| 0x00e8749c | ExitThread |
| 0x00e88357 | GetFileSize |
| 0x00e88659 | CreateFileMappingA |
| 0x00e88980 | MapViewOfFile |
| 0x00e88aef | UnmapViewOfFile |
| 0x00e89915 | GlobalLock |
| 0x00e898e5 | GlobalUnlock |
| 0x00e87398 | ExitProcess |
| 0x00e8a6e8 | CreateDialogParamA |
| 0x00e8a46a | FindResourceA |
| 0x00e89945 | SetHandleCount |
| 0x00e87451 | TerminateProcess |

```
     0x00e8a452  GetVersion
     0x00e8a45e  GetCommandLineA
     0x00e871a2  GetEnvironmentStringsW
     0x00e86fac  GetEnvironmentStringsA
     0x00e88493  GetFileType
     0x00e8a464  GetACP
```

Now it’s done. We have everything we need to rebuild the import table.
First of all we add a new section to build the IAT

```
 SECTION   VAddress   Vsize      ROffset    Rsize      FLAGS
 ---------------------------------------------------------------
 .myiat    003C7000   00005000   003C7000   00005000   E00000E0
```

Then in the first 0x200 bytes we put the import table itself, with all descriptors. I made descriptors for all the dlls of the process, though they will not be in the import table, so remember to delete the unwanted ones after the rebuilding.

```
->Import Table
   1. ImageImportDescriptor:
    OriginalFirstThunk:  0x003CB200
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBFF0  ("ntdll.dll")
    FirstThunk:          0x003CB200

    Ordinal/Hint API name
    ------------ ---------------------------------------

   2. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C7400
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBFE0  ("kernel32.dll")
    FirstThunk:          0x003C7400

    Ordinal/Hint API name
    ------------ ---------------------------------------

   3. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C7600
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBFD0  ("user32.dll")
    FirstThunk:          0x003C7600

    Ordinal/Hint API name
    ------------ ---------------------------------------

   4. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C7800
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBFC0  ("gdi32.dll")
    FirstThunk:          0x003C7800

    Ordinal/Hint API name
    ------------ ---------------------------------------

   5. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C7A00
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBFB0  ("uxtheme.dll")
    FirstThunk:          0x003C7A00

    Ordinal/Hint API name
    ------------ ---------------------------------------

   6. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C7C00
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBFA0  ("msvcrt.dll")
    FirstThunk:          0x003C7C00

    Ordinal/Hint API name
    ------------ ---------------------------------------

   7. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C7E00
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBF90  ("advapi32.dll")
    FirstThunk:          0x003C7E00

    Ordinal/Hint API name
    ------------ ---------------------------------------

   8. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C8000
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBF80  ("rpcrt4.dll")
    FirstThunk:          0x003C8000

    Ordinal/Hint API name
    ------------ ---------------------------------------

   9. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C8200
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBF70  ("comctl32.dll")
    FirstThunk:          0x003C8200

    Ordinal/Hint API name
    ------------ ---------------------------------------

   10. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C8400
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBF60  ("comdlg32.dll")
    FirstThunk:          0x003C8400

    Ordinal/Hint API name
    ------------ ---------------------------------------

   11. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C8600
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBF50  ("shlwapi.dll")
    FirstThunk:          0x003C8600

    Ordinal/Hint API name
    ------------ ---------------------------------------

   12. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C8800
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBF40  ("shell32.dll")
    FirstThunk:          0x003C8800

    Ordinal/Hint API name
    ------------ ---------------------------------------

   13. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C8A00
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBF30  ("comctl32.dll")
    FirstThunk:          0x003C8A00

    Ordinal/Hint API name
    ------------ ---------------------------------------

   14. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C8C00
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBF20  ("oleaut32.dll")
    FirstThunk:          0x003C8C00

    Ordinal/Hint API name
    ------------ ---------------------------------------

   15. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C8E00
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBF10  ("ole32.dll")
    FirstThunk:          0x003C8E00

    Ordinal/Hint API name
    ------------ ---------------------------------------

   16. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C9000
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBF00  ("ws2_32.dll")
    FirstThunk:          0x003C9000

    Ordinal/Hint API name
    ------------ ---------------------------------------

   17. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C9200
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBEF0  ("ws2help.dll")
    FirstThunk:          0x003C9200

    Ordinal/Hint API name
    ------------ ---------------------------------------
   18. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C9400
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBEE0  ("inetmib1.dll")
    FirstThunk:          0x003C9400

    Ordinal/Hint API name
    ------------ ---------------------------------------

   19. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C9600
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBED0  ("iphlpapi.dll")
    FirstThunk:          0x003C9600

    Ordinal/Hint API name
    ------------ ---------------------------------------

   20. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C9800
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBEC0  ("snmpapi.dll")
    FirstThunk:          0x003C9800

    Ordinal/Hint API name
    ------------ ---------------------------------------

   21. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C9A00
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBEB0  ("wsock32.dll")
    FirstThunk:          0x003C9A00

    Ordinal/Hint API name
    ------------ ---------------------------------------

   22. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C9C00
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBEA0  ("mp3api.dll")
    FirstThunk:          0x003C9C00
    Ordinal/Hint API name
    ------------ ---------------------------------------

   23. ImageImportDescriptor:
    OriginalFirstThunk:  0x003C9E00
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBE90  ("activeds.dll")
    FirstThunk:          0x003C9E00

    Ordinal/Hint API name
    ------------ ---------------------------------------

   24. ImageImportDescriptor:
    OriginalFirstThunk:  0x003CA000
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBE80  ("adsldpc.dll")
    FirstThunk:          0x003CA000

    Ordinal/Hint API name
    ------------ ---------------------------------------

   25. ImageImportDescriptor:
    OriginalFirstThunk:  0x003CA200
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBE70  ("netapi32.dll")
    FirstThunk:          0x003CA200

    Ordinal/Hint API name
    ------------ ---------------------------------------

   26. ImageImportDescriptor:
    OriginalFirstThunk:  0x003CA400
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBE60  ("wldap32.dll")
    FirstThunk:          0x003CA400

    Ordinal/Hint API name
    ------------ ---------------------------------------

   27. ImageImportDescriptor:
    OriginalFirstThunk:  0x003CA600
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
```

```
    ForwarderChain:      0x00000000
    Name:                0x003CBE50  ("atl.dll")
    FirstThunk:          0x003CA600

    Ordinal/Hint API name
    ------------ ---------------------------------------

   28. ImageImportDescriptor:
    OriginalFirstThunk:  0x003CA800
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBE40  ("rtutils.dll")
    FirstThunk:          0x003CA800

    Ordinal/Hint API name
    ------------ ---------------------------------------

   29. ImageImportDescriptor:
    OriginalFirstThunk:  0x003CAA00
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBE30  ("samlib.dll")
    FirstThunk:          0x003CAA00

    Ordinal/Hint API name
    ------------ ---------------------------------------

   30. ImageImportDescriptor:
    OriginalFirstThunk:  0x003CAC00
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBE20  ("setupapi.dll")
    FirstThunk:          0x003CAC00

    Ordinal/Hint API name
    ------------ ---------------------------------------

   31. ImageImportDescriptor:
    OriginalFirstThunk:  0x003CAE00
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBE10  ("msvbvm60.dll")
    FirstThunk:          0x003CAE00

    Ordinal/Hint API name
    ------------ ---------------------------------------

   32. ImageImportDescriptor:
    OriginalFirstThunk:  0x003CB000
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x003CBE00  ("version.dll")
    FirstThunk:          0x003CB000

    Ordinal/Hint API name
    ------------ ---------------------------------------

   33. ImageImportDescriptor:
    OriginalFirstThunk:  0x000D902C
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x000E2378  ("KERNEL32.dll")
    FirstThunk:          0x000D902C

    Ordinal/Hint API name
    ------------ ---------------------------------------
    Bad RVA in thunk !
    ...

   34. ImageImportDescriptor:
    OriginalFirstThunk:  0x000D91D0
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x000E265E  ("USER32.dll")
    FirstThunk:          0x000D91D0

    Ordinal/Hint API name
    ------------ ---------------------------------------
    Bad RVA in thunk !
    ...

   35. ImageImportDescriptor:
    OriginalFirstThunk:  0x000D9000
    TimeDateStamp:       0x00000000  (GMT: Thu Jan 01 00:00:00
1970)
    ForwarderChain:      0x00000000
    Name:                0x000E2706  ("GDI32.dll")
    FirstThunk:          0x000D9000

    Ordinal/Hint API name
    ------------ ---------------------------------------
    Bad RVA in thunk !
    ...
```

The last three import descriptors are the ones that originally were in the IT.
Here is the code for the rebuilder:

```
------------ file data.h ---------------------------------------
-------------------------------

#define NUMBER_OF_THUNKS 35
#define M_KERNEL32           0
#define M_USER32             1
#define M_ADVAPI32           2

#define OFF_START        0x00001000
#define OFF_END              0x00048000

typedef struct _THUNK {
     DWORD     DllThunk;
     DWORD     OriginalThunk;
     char ApiName[32];
     int       Module;
} THUNK;

THUNK ThunkData[NUMBER_OF_THUNKS] = {\
     0x00E86EAE, 0, "GetEnvironmentVariableA", M_KERNEL32,\
     0x00E880CB, 0, "WriteFile", M_KERNEL32,\
     0x00e899fe, 0, "MessageBoxA", M_USER32,\
     0x00e8667a, 0, "LoadLibraryA", M_KERNEL32,\
     0x00e85aa1, 0, "GetProcAddress", M_KERNEL32,\
     0x00e89e64, 0, "RegCreateKeyExA", M_ADVAPI32,\
     0x00e898b2, 0, "EndDialog", M_USER32,\
     0x00e8a74c, 0, "DialogBoxParamA", M_USER32,\
     0x00e89a42, 0, "GetWindowTextA", M_USER32,\
     0x00e88414, 0, "CloseHandle", M_KERNEL32,\
     0x00e8a23e, 0, "RegQueryValueA", M_ADVAPI32,\
     0x00e8944d, 0, "FindFirstFileA", M_KERNEL32,\
     0x00e87a04, 0, "CreateFileA", M_KERNEL32,\
     0x00e87e8e, 0, "ReadFile", M_KERNEL32,\
     0x00e882a0, 0, "SetFilePointer", M_KERNEL32,\
     0x00e874f6, 0, "CreateThread", M_KERNEL32,\
     0x00e89975, 0, "GetModuleHandleA", M_KERNEL32,\
     0x00e8749c, 0, "ExitThread", M_KERNEL32,\
     0x00e88357, 0, "GetFileSize", M_KERNEL32,\
     0x00e88659, 0, "CreateFileMappingA", M_KERNEL32,\
     0x00e88980, 0, "MapViewOfFile", M_KERNEL32,\
     0x00e88aef, 0, "UnmapViewOfFile", M_KERNEL32,\
     0x00e89915, 0, "GlobalLock", M_KERNEL32,\
     0x00e898e5, 0, "GlobalUnlock", M_KERNEL32,\
     0x00e87398, 0, "ExitProcess", M_KERNEL32,\
     0x00e8a6e8, 0, "CreateDialogParamA", M_USER32,\
     0x00e8a46a, 0, "FindResourceA", M_KERNEL32,\
     0x00e89945, 0, "SetHandleCount", M_KERNEL32,\
     0x00e87451, 0, "TerminateProcess", M_KERNEL32,\
     0x00e8a452, 0, "GetVersion", M_KERNEL32,\
     0x00e8a45e, 0, "GetCommandLineA", M_KERNEL32,\
     0x00e871a2, 0, "GetEnvironmentStringsW", M_KERNEL32,\
     0x00e86fac, 0, "GetEnvironmentStringsA", M_KERNEL32,\
     0x00e88493, 0, "GetFileType", M_KERNEL32,\
     0x00e8a464, 0, "GetACP", M_KERNEL32};

#define NUMBER_OF_IMPORTS     32
#define IAT_FIRST             0x003C7200

typedef struct _MAP_IAT {
     DWORD     ImageBase;
     DWORD     ImageSize;
     DWORD     IatEntry;
} MAP_IAT;

MAP_IAT IatMap[NUMBER_OF_IMPORTS] = {\
/* ntdll */         0x7C910000, 0x000b6000, IAT_FIRST + 0x4000,\
/* kernel32 */ 0x7C800000, 0x000FF000, IAT_FIRST + 0x0200,\
/* user32 */        0x77d10000, 0x00090000, IAT_FIRST + 0x0400,\
/* gdi32 */         0x77e40000, 0x00046000, IAT_FIRST + 0x0600,\
/* uxtheme */       0x5b180000, 0x00038000, IAT_FIRST + 0x0800,\
/* msvcrt */        0x77be0000, 0x00058000, IAT_FIRST + 0x0a00,\
/* advapi32 */ 0x77f40000, 0x000ab000, IAT_FIRST + 0x0c00,\
/* rpcrt4 */        0x77da0000, 0x00091000, IAT_FIRST + 0x0e00,\
/* comctl32 */ 0x5d4d0000, 0x00097000, IAT_FIRST + 0x1000,\
/* comdlg32 */ 0x76360000, 0x0004a000, IAT_FIRST + 0x1200,\
/* shlwapi */       0x77e90000, 0x00076000, IAT_FIRST + 0x1400,\
/* shell32 */       0x7c9d0000, 0x0081b000, IAT_FIRST + 0x1600,\
/* comctl32 relocated */ 0x773a0000, 0x00102000, IAT_FIRST +
0x1800,\
/* oleaut32 */ 0x770f0000, 0x0008c000, IAT_FIRST + 0x1a00,\
/* ole32 */         0x774b0000, 0x0013d000, IAT_FIRST + 0x1c00,\
/* ws2_32 */        0x71a30000, 0x00017000, IAT_FIRST + 0x1e00,\
/* ws2help */       0x71a20000, 0x00008000, IAT_FIRST + 0x2000,\
/* inetmib1 */ 0x66bb0000, 0x0000b000, IAT_FIRST + 0x2200,\
/* iphlpapi */ 0x76d20000, 0x00019000, IAT_FIRST + 0x2400,\
/* snmpapi */       0x71ef0000, 0x00008000, IAT_FIRST + 0x2600,\
/* wsock32 */       0x71a50000, 0x0000a000, IAT_FIRST + 0x2800,\
/* mp3api */        0x76d00000, 0x00018000, IAT_FIRST + 0x2a00,\
/* activeds */ 0x77c90000, 0x00032000, IAT_FIRST + 0x2c00,\
/* adsldpc */       0x76dd0000, 0x00025000, IAT_FIRST + 0x2e00,\
/* netapi32 */ 0x5bc70000, 0x00054000, IAT_FIRST + 0x3000,\
/* wldap32 */       0x76f20000, 0x0002d000, IAT_FIRST + 0x3200,\
/* atl */           0x76ae0000, 0x00011000, IAT_FIRST + 0x3400,\
/* rtutils */       0x76e40000, 0x0000e000, IAT_FIRST + 0x3600,\
/* samlib */        0x71b80000, 0x00013000, IAT_FIRST + 0x3800,\
/* setupapi */ 0x778f0000, 0x000F7000, IAT_FIRST + 0x3a00,\
/* msvbvm60 */ 0x73390000, 0x00154000, IAT_FIRST + 0x3c00,\
/* version */       0x77bd0000, 0x00008000, IAT_FIRST + 0x3e00,\
};

------------- file main.c ----------------------------------------
----------------------
```

```
#include <windows.h>
#include "data.h"

#define FILE_TARGET      "Armadillo.exe"
#define OFF_START             0x00001000
#define OFF_END               0x00048000
#define EXE_IMAGE_BASE        0x00400000

#define BRIDGE           "bridge.dmp"
#define BRIDGE_BASE      0x00F20000
#define IAT_START             0x00FC3048
#define IAT_END               0x00FC35B4          //first non thunk
address

#define CALLS                 50                  //number of calls
in call list

#define RESULT_FILE      "armadillofix.exe"

int WINAPI WinMain(HINSTANCE Instance, HINSTANCE PreInst, LPSTR
CmdLine, int CmdShow)
{
     HANDLE TargetFile, BridgeFile, ResultFile;
     DWORD Dword, FileSize, BridgeSize, Address, *Pointer, Thunk,
*Bpointer;
     BYTE *TargetBuffer, *BridgeBuffer;
     int i, j;
     HINSTANCE Libraries[3];

     DWORD *IatApi;
     BOOL AddCall;

     //
     // initialize library handles for emulated apis
     //
     Libraries[M_KERNEL32] = LoadLibrary("kernel32.dll");
     Libraries[M_USER32] = LoadLibrary("user32.dll");
     Libraries[M_ADVAPI32] = LoadLibrary("advapi32.dll");

     //
     // open dumped exe file and read it
     //
     TargetFile = CreateFile(FILE_TARGET, GENERIC_READ,
FILE_SHARE_READ, NULL,
                   OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, NULL);
     FileSize = GetFileSize(TargetFile, &Dword);
     TargetBuffer = (BYTE*)malloc(FileSize);
     ReadFile(TargetFile, TargetBuffer, FileSize, &Dword, NULL);

     //
     // open bridge file and read it
     //
     BridgeFile = CreateFile(BRIDGE, GENERIC_READ, FILE_SHARE_READ,
NULL,
                    OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, NULL);
     BridgeSize = GetFileSize(BridgeFile, &Dword);
     BridgeBuffer = (BYTE*)malloc(BridgeSize);
     ReadFile(BridgeFile, BridgeBuffer, BridgeSize, &Dword, NULL);

     //
     // scan code section to find imports in memory bridge
     //
     for(i = OFF_START; I < (OFF_END - 4); i++)
     {
          Pointer = (DWORD*)&(TargetBuffer[i]);
          Address = *Pointer;
          if((Address >= IAT_START) && (Address < IAT_END) &&
(Address % 4 == 0))
          {
               //
               // address is in iat and is correctly 4-aligned
               // read thunk from memory bridge
               //
               Address -= BRIDGE_BASE;
               Bpointer = (DWORD*)&(BridgeBuffer[Address]);
               Thunk = *Bpointer;      //we get the import address
               if((Thunk > 0x00E85000) && (Thunk < 0x00EB0000))
               {
                    //
                    // API is emulated, obtain its pointer
                    //
                    for(j = 0; j < NUMBER_OF_THUNKS; j++)
                    {
                         if(Thunk == ThunkData[j].DllThunk)
                         {
                              Thunk =
(DWORD)GetProcAddress(Libraries[ThunkData[j].Module],
ThunkData[j].ApiName);
                              break;
                         }
                    }
               }

               //
               // write the thunk in the iat
               //
               for(j = 0; j < NUMBER_OF_THUNKS; j++)
               {
                    //
                    // determine in which API the address is
                    //
                    if((Thunk > IatMap[j].ImageBase) && (Thunk <
(IatMap[j].ImageBase + IatMap[j].ImageSize)))
```

```
                    {
                         // once located, add the address to the
iat entry
                         IatApi = (DWORD*)(TargetBuffer +
IatMap[j].IatEntry);
                         AddCall = TRUE;
                         while(*IatApi != 0)
                         {
                              if(*IatApi == Thunk)
                              {
                                   // API is already present in
iat, do nothing
                                   AddCall = FALSE;
                                   break;
                              }
                              IatApi++;
                         }
                         if(AddCall)
                         {
                              // API was not found in the iat, add
it in the iat array
                              *IatApi = Thunk;
                         }
                         // fix the code to call this iat entry
instead that bridge
                         *Pointer = (DWORD)((BYTE*)IatApi -
TargetBuffer + EXE_IMAGE_BASE);    //VIRTUAL ADDRESS, not RVA
                         break;
                    }
               }
          }
     }

     //
     // write fixed exe
     //
     ResultFile = CreateFile(RESULT_FILE, GENERIC_READ +
GENERIC_WRITE, FILE_SHARE_READ,
                    NULL, CREATE_ALWAYS, FILE_ATTRIBUTE_NORMAL,
NULL);
     WriteFile(ResultFile, TargetBuffer, FileSize, &Dword, NULL);

     //
     // close handles, free memory and exit
     //
     free(BridgeBuffer);
     free(TargetBuffer);
     CloseHandle(TargetFile);
     CloseHandle(BridgeFile);
     CloseHandle(ResultFile);
     return NULL;
}
```

------------------------------------------------------------------
-------------------------------

The program scans the code section to find all addresses relative to the memory bridge (0x00FC3*). When found, it looks in the bridge for the bytes at the given address. If these bytes are not an API thunk then they are the address of the security.dll, so from our associative array it extracts the API thunk corresponding to the security.dll wrapper. Once the API thunk is found, the program writes it in our newly created import table. As a result we will obtain an executable with all imports pointing to the correct API addresses. We created the import table so that the pointer of the OriginalFirstThunk is the same of the FirstThunk. Now that we have the correct import table and the correct import address table, we can use any import rebuilder, and our OriginalFirstThunk will be created. Now we have a perfect import table.

## STEP 5: NANOMITES – Here comes the pain!

The program is dumped, decrypted and rebuilt, but it still is not running. Let's have a look at the code, everything is ok, the imports are ok, let's look at the WinMain:

```
00401D03 55                     push    ebp
00401D04 8B EC                  mov     ebp, esp
00401D06 81 EC 38 0C 00 00      sub     esp, 0C38h
00401D0C 53                     push    ebx
00401D0D 56                     push    esi
00401D0E 57                     push    edi
00401D0F CC                     int     3
00401D10 D4 C6                  aam     0C6h
00401D12 00 0B                  add     [ebx], cl
00401D14 8B 45 08               mov     eax, [ebp+8]
00401D17 33 DB                  xor     ebx, ebx
00401D19 88 5D FF               mov     [ebp-1], bl
00401D1C 88 5D FD               mov     [ebp-3], bl
```

int 3? What is it doing there? If we look around we find a lot of those int 3's. Of course this means we need to debug the parent process in the EXCEPTION_BREAKPOINT handler. If we have a fast look, we break on that handler (004a5adf), we see a call to GetThreadContext which gives us the context.eip = 00401D10, then at the end of the handler we have a call to SetThreadContext, where context.eip is being set to 00401D14. So basically int 3 is just a change of eip, a jump. Now we must analyse the handler; here is the code of it, with no garbage and divided into blocks:

------------------
BLOCK 1

```
------------------

.text1:004A5ADF                 mov     dword ptr [ebp-1178h], 10h
.text1:004A5AE9                 mov     eax, ds:dword_4D9378
.text1:004A5AEE                 xor     eax, ds:dword_4D93B4
.text1:004A5AF4                 xor     eax, ds:dword_4D937C
.text1:004A5AFA                 mov     [ebp-1174h], eax
.text1:004A5B00                 mov     ecx, ds:dword_4D938C
.text1:004A5B06                 xor     ecx, ds:dword_4D9370
.text1:004A5B0C                 xor     ecx, ds:dword_4D93B8
.text1:004A5B12                 mov     [ebp-1170h], ecx
.text1:004A5B18                 mov     edx, ds:dword_4D9384
.text1:004A5B1E                 xor     edx, ds:dword_4D93C8
.text1:004A5B24                 xor     edx, ds:dword_4D93F0
.text1:004A5B2A                 mov     [ebp-116Ch], edx
.text1:004A5B30                 mov     eax, ds:dword_4D93E8
.text1:004A5B35                 xor     eax, ds:dword_4D93C0
.text1:004A5B3B                 xor     eax, ds:dword_4D93AC
.text1:004A5B41                 mov     [ebp-1168h], eax
.text1:004A5B47                 mov     ecx, ds:dword_4D93EC
.text1:004A5B4D                 xor     ecx, ds:dword_4D9398
.text1:004A5B53                 xor     ecx, ds:dword_4D9374
.text1:004A5B59                 mov     [ebp-1164h], ecx
.text1:004A5B5F                 mov     edx, ds:dword_4D93E4
.text1:004A5B65                 xor     edx, ds:dword_4D93A8
.text1:004A5B6B                 xor     edx, ds:dword_4D93F4
.text1:004A5B71                 mov     [ebp-1160h], edx
.text1:004A5B77                 mov     eax, ds:dword_4D9378
.text1:004A5B7C                 xor     eax, ds:dword_4D938C
.text1:004A5B82                 xor     eax, ds:dword_4D9384
.text1:004A5B88                 xor     eax, ds:dword_4D93D0
.text1:004A5B8E                 mov     [ebp-115Ch], eax
.text1:004A5B94                 mov     ecx, ds:dword_4D93B4
.text1:004A5B9A                 xor     ecx, ds:dword_4D9370
.text1:004A5BA0                 xor     ecx, ds:dword_4D93C8
.text1:004A5BA6                 xor     ecx, ds:dword_4D93E4
.text1:004A5BAC                 mov     [ebp-1158h], ecx
.text1:004A5BB2                 mov     edx, ds:dword_4D93E8
.text1:004A5BB8                 xor     edx, ds:dword_4D93EC
.text1:004A5BBE                 xor     edx, ds:dword_4D93E4
.text1:004A5BC4                 xor     edx, ds:dword_4D93A8
.text1:004A5BCA                 mov     [ebp-1154h], edx
.text1:004A5BD0                 mov     eax, ds:dword_4D93C0
.text1:004A5BD5                 xor     eax, ds:dword_4D9398
.text1:004A5BDB                 xor     eax, ds:dword_4D93A8
.text1:004A5BE1                 xor     eax, ds:dword_4D9378
.text1:004A5BE7                 mov     [ebp-1150h], eax
.text1:004A5BED                 mov     ecx, ds:dword_4D9378
.text1:004A5BF3                 xor     ecx, ds:dword_4D93EC
.text1:004A5BF9                 xor     ecx, ds:dword_4D938C
.text1:004A5BFF                 mov     [ebp-114Ch], ecx
.text1:004A5C05                 mov     edx, ds:dword_4D93B4
```

```
.text1:004A5C0B                 xor     edx, ds:dword_4D93C0
.text1:004A5C11                 xor     edx, ds:dword_4D9384
.text1:004A5C17                 mov     [ebp-1148h], edx
.text1:004A5C1D                 mov     eax, ds:dword_4D938C
.text1:004A5C22                 xor     eax, ds:dword_4D93E8
.text1:004A5C28                 xor     eax, ds:dword_4D93E4
.text1:004A5C2E                 mov     [ebp-1144h], eax
.text1:004A5C34                 mov     ecx, ds:dword_4D9370
.text1:004A5C3A                 xor     ecx, ds:dword_4D93C8
.text1:004A5C40                 xor     ecx, ds:dword_4D93E8
.text1:004A5C46                 mov     [ebp-1140h], ecx
.text1:004A5C4C                 mov     edx, ds:dword_4D9378
.text1:004A5C52                 xor     edx, ds:dword_4D9370
.text1:004A5C58                 xor     edx, ds:dword_4D93E8
.text1:004A5C5E                 xor     edx, ds:dword_4D93EC
.text1:004A5C64                 mov     [ebp-113Ch], edx
.text1:004A5C6A                 mov     eax, ds:dword_4D93B4
.text1:004A5C6F                 xor     eax, ds:dword_4D9384
.text1:004A5C75                 xor     eax, ds:dword_4D93C0
.text1:004A5C7B                 xor     eax, ds:dword_4D9398
.text1:004A5C81                 mov     [ebp-1138h], eax
.text1:004A5C87                 xor     ecx, ecx
.text1:004A5C89                 mov     cl, ds:FirstBreak
.text1:004A5C8F                 test    ecx, ecx
.text1:004A5C91                 jz      ContinueEvent

.text1:004A61E4                 push    2CCh
.text1:004A61E9                 push    0
.text1:004A61EB                 lea     edx, [ebp-1468h]
.text1:004A61F1                 push    edx
.text1:004A61F2                 call    AllocBuffer
.text1:004A61F7                 add     esp, 0Ch
.text1:004A61FA                 mov     dword ptr [ebp-1468h],
10001h
.text1:004A6204                 lea     eax, [ebp-1468h]
.text1:004A620A                 push    eax
.text1:004A620B                 mov     ecx, [ebp-1194h]
.text1:004A6211                 push    ecx
.text1:004A6212                 call    ds:GetThreadContext

---------------------
BLOCK 2
--------------------

.text1:004A625C                 mov     dword ptr [ebp-146Ch], 0
.text1:004A6266                 push    0FFFFFFFFh
.text1:004A6268                 push    4
.text1:004A626A                 lea     edx, [ebp-13B0h]
.text1:004A6270                 push    edx
.text1:004A6271                 call    GetFirstParameter
.text1:004A6276                 add     esp, 0Ch
.text1:004A6279                 mov     [ebp-FirstParameterV], eax
```

```
------------------------
block 3
------------------------

.text1:004A627F                 mov     eax, [ebp-FirstParameterV]
.text1:004A6285                 xor     edx, edx
.text1:004A6287                 mov     ecx, 10h
.text1:004A628C                 div     ecx
.text1:004A628E                 mov     [ebp-119Ch], edx
.text1:004A6294                 mov     edx, [ebp-13B0h]
.text1:004A629A                 push    edx
.text1:004A629B                 mov     eax, [ebp-119Ch]
.text1:004A62A1                 call    ds:off_4DDD3C[eax*4]
.text1:004A62A8                 add     esp, 4
.text1:004A62AB                 mov     [ebp-146Ch], eax

------------------------
block 4
------------------------

.text1:004A62B1                 mov     dword ptr [ebp-1470h], 0
.text1:004A62BB                 mov     ecx, [ebp-119Ch]
.text1:004A62C1                 mov     edx,
ds:dword_4E0230[ecx*4]
.text1:004A62C8                 mov     [ebp-1190h], edx
.text1:004A62CE
.text1:004A62CE loc_4A62CE:
.text1:004A62CE                 mov     eax, [ebp-1470h]
.text1:004A62D4                 cmp     eax, [ebp-1190h]
.text1:004A62DA                 jge     short near ptr byte_4A6338
.text1:004A62DC
.text1:004A62DC loc_4A62DC:
.text1:004A62DC                 mov     eax, [ebp-1190h]
.text1:004A62E2                 sub     eax, [ebp-1470h]
.text1:004A62E8                 cdq
.text1:004A62E9                 sub     eax, edx
.text1:004A62EB                 sar     eax, 1
.text1:004A62ED                 mov     ecx, [ebp-1470h]
.text1:004A62F3                 add     ecx, eax
.text1:004A62F5                 mov     [ebp-1474h], ecx
.text1:004A62FB                 mov     edx, [ebp-119Ch]
.text1:004A6301                 mov     eax,
ds:dword_4E01D0[edx*4]
.text1:004A6308                 mov     ecx, [ebp-1474h]
.text1:004A630E                 mov     edx, [ebp-146Ch]
.text1:004A6314                 cmp     edx, [eax+ecx*4]
.text1:004A6317                 jbe     short loc_4A632A
.text1:004A6319                 mov     eax, [ebp-1474h]
.text1:004A631F                 add     eax, 1
.text1:004A6322                 mov     [ebp-1470h], eax
.text1:004A6328                 jmp     short loc_4A6336
```

```
.text1:004A632A
.text1:004A632A loc_4A632A:
.text1:004A632A                 mov     ecx, [ebp-1474h]
.text1:004A6330                 mov     [ebp-1190h], ecx
.text1:004A6336
.text1:004A6336 loc_4A6336:
.text1:004A6336                 jmp     short loc_4A62CE

------------------------
 block 5
------------------------

.text1:004A635E                 mov     edx, [ebp-119Ch]
.text1:004A6364                 mov     eax,
ds:dword_4E01D0[edx*4]
.text1:004A636B                 mov     ecx, [ebp-1470h]
.text1:004A6371                 mov     edx, [eax+ecx*4]
.text1:004A6374                 cmp     edx, [ebp-146Ch]
.text1:004A637A                 jnz     ContinueEvent

------------------------
 block 6
------------------------

.text1:004A63D7                 mov     eax, [ebp-119Ch]
.text1:004A63DD                 mov     ecx,
ds:dword_4E0270[eax*4]
.text1:004A63E4                 mov     edx, [ebp-1470h]
.text1:004A63EA                 mov     eax, [ecx+edx*4]
.text1:004A63ED                 mov     [ebp-1488h], eax
.text1:004A63F3                 mov     ecx, [ebp-13A8h] ; eflags
.text1:004A63F9                 and     ecx, 0FD7h
.text1:004A63FF                 mov     [ebp-1478h], ecx
.text1:004A6405                 mov     edx, [ebp-1488h]
.text1:004A640B
.text1:004A640B loc_4A640B:
.text1:004A640B                 and     edx, 0FF000000h
.text1:004A6411                 shr     edx, 18h
.text1:004A6414                 mov     [ebp-1484h], edx
.text1:004A641A                 mov     eax, [ebp-1488h]
.text1:004A6420                 and     eax, 0FFFFFFh
.text1:004A6425                 mov     [ebp-1480h], eax
.text1:004A642B                 mov     ecx, [ebp-13BCh]
.text1:004A6431                 push    ecx
.text1:004A6432                 mov     edx, [ebp-1478h]
.text1:004A6438                 push    edx             ; eflags
.text1:004A6439                 mov     eax, [ebp-1480h]
.text1:004A643F                 push    eax
.text1:004A6440                 mov     ecx, [ebp-1484h]
.text1:004A6446                 call    ds:off_4D97E8[ecx*4]
.text1:004A644D                 add     esp, 0Ch
.text1:004A6450                 mov     [ebp-147Ch], eax
```

```
.text1:004A6456                 mov     edx, [ebp-147Ch]
.text1:004A645C                 and     edx, 1
.text1:004A645F                 test    edx, edx
.text1:004A6461                 jz      near ptr
SkipSecondCalculus

----------------------
 block 7
---------------------

.text1:004A648D SecondCalculus:
.text1:004A648D                 mov     eax, [ebp-119Ch]
.text1:004A6493                 mov     ecx,
ds:dword_4E0190[eax*4]
.text1:004A649A                 mov     eax, [ebp-1470h]
.text1:004A64A0                 xor     edx, edx
.text1:004A64A2                 mov     esi, 10h
.text1:004A64A7                 div     esi
.text1:004A64A9                 mov     eax, [ebp-1470h]
.text1:004A64AF                 mov     ecx, [ecx+eax*4]
.text1:004A64B2                 xor     ecx, [ebp+edx*4-1174h]
.text1:004A64B9                 mov     edx, [ebp-13B0h]
.text1:004A64BF                 add     edx, ecx
.text1:004A64C1                 mov     [ebp-13B0h], edx
                           goto setthreadcontext
.text1:004A64C1 ; --------------------------------------------------
---------------------------
.text1:004A64C7                 db 51h, 0Fh, 0C9h, 0F7h, 0D1h,
50h, 0F7h, 0D0h, 0B8h, 6Dh ; goto 004a65d6

----------------------
 block 8
---------------------

.text1:004A6515 SkipSecondCalculus
.text1:004A6515          db 70h, 7, 7Ch, 3, 0EBh, 5, 0E8h, 74h,
0FBh, 0EBh, 0F9h
.text1:004A6515
.text1:004A6520 ; --------------------------------------------------
---------------------------
.text1:004A6520                 mov     eax, [ebp-119Ch]
.text1:004A6526                 mov     ecx,
ds:dword_4E02B8[eax*4]
.text1:004A652D                 mov     edx, [ebp-1470h]
.text1:004A6533                 xor     eax, eax
.text1:004A6535                 mov     al, [ecx+edx]
.text1:004A6538                 mov     ecx, [ebp-13B0h]
.text1:004A653E                 add     ecx, eax
.text1:004A6540                 mov     [ebp-13B0h], ecx
                            goto setthreadcontext

setthreadcontext:
  updates information in the context, goto ContinueDebugEvent
ContinueEvent:
  continue the debugging cycle
```

All these code blocks use static or dynamic arrays of data and functions, some functions are executed dynamically. Maybe using some image will clarify the situation:

| |
|---|
| BLOCK 1<br>-----------------------------<br><br>Makes a static calculus, which is always the same, and gets the context of the faulting instruction |

| | | |
|---|---|---|
| BLOCK 2<br>-----------------------------<br><br>Calls a function that given the context.eip address calculates a 32bit number using an array of data | ---> | Array of 0x400 bytes |

| | | | | |
|---|---|---|---|---|
| BLOCK 3<br>-----------------------------<br><br>Uses an array of 16 functions to compute a 32bit number from context.eip | ---> | Array of 16 functions, each function uses a max of 4 different subfunctions | ---> | 4 sub functions |

| | | |
|---|---|---|
| BLOCK 4<br>-----------------------------<br><br>Uses an array of 16 8-bit values to extract a byte from an array containing 16 dynamic memory data arrays | ---> | Array of 16 8-bit values (stored as dwords) |
| | | Array of 16 dynamic memory blocks containing the data |

| | | |
|---|---|---|
| BLOCK 5 | | Array of 16 dynamic |

| | | |
|---|---|---|
| ------------------------------ Just makes a check using an array of 16 dynamic memory blocks | ---> | memory arrays |

| | | | | |
|---|---|---|---|---|
| BLOCK 6 ----------------------------- Uses an array of 16 dynamic memory blocks to extract the index of the function that must be executed from an array of 256 functions | ---> | Array of 256 static functions, each one calls some stati subfunctions, and other dynamic functions determined dynamically | ---> | array of 256 dynamic functions, which call a sublayer of functions |

| | | |
|---|---|---|
| BLOCK 7 ----------------------------- extracts a dword from an array of 16 memory arrays, then xores this dword with another dword taken from an array of other 16 dwords, the obtained number is the number of bytes that will be jumped | ---> | Array of 16 dynamic memory data arrays |
| | | Array of 16 dword used for xoring |

| | | |
|---|---|---|
| BLOCK 8 ----------------------------- uses an array of 16 dynamic memory arrays to extract the number of bytes that will be jumped | ---> | Array of 16 dynamic data arrays containing number of displacement bytes |

Okay this is the structure of the nanomites processing. We see that the EIP and EFLAGS from the context of the child process are used in the calculus, so what is now clear is that nanomites are used to emulate jumps, both conditional or unconditional. Every 0xCC in the code represents a jump. What we can do now is try to fix every 0xCC to its original opcode, or make an emulator of these jumps. Fixing the code is not easy, we should code a scanner which supports instruction tracing. Armadillo infact can't recognize if a 0xCC is really a nonomite or not, so for example if we scan the code section and we meet the bytes of the instruction

```
mov eax, 0xCC
```

we would trash the original instruction trying to fix this non-nanomite. Even having a disassembling engine, it would be easy to make mistakes and patch wrong 0xCCs. So what I did is coded the emulator. I took all the functions of the breakpoint handler, and inlined them in C, then dumped all the data used by the emulation blocks and the functions. It needed to be modified a little, but with some time the work is done. Once I had all the emulation functions, I just had to code a parent process which makes emulation:

```
------------ file main.c ---------------------------------------------------------------------------
////////////////////////////////////////////
//
//      NANOMULATOR
//
// nanomites emulator for armadillo 4.20
// written by AndreaGeddon
//
////////////////////////////////////////////

#include <windows.h>
#include "core.h"

#define PROCESS_NAME     "emul.exe"

typedef struct _THREADS {
     DWORD     ThreadId;
     HANDLE    ThreadHandle;
} THREADS;

int WINAPI WinMain(HINSTANCE Instance, HINSTANCE PreInst, LPSTR
CmdLine, int CmdShow)
{
     STARTUPINFO Si;
     PROCESS_INFORMATION Pi;
     DEBUG_EVENT Event;
     BOOL FirstBreak = TRUE, DebugLoop = TRUE;
     DWORD Continue;
     CONTEXT Context;
     THREADS   Threads[32];        //warning
     int j, NumberOfThreads = 0;
     ////////////////////
     //block 1 vars
     ////////////////////
```

```
	DWORD ebp1178h;
	DWORD ebp1174h;
	DWORD ebp1170h;
	DWORD ebp116Ch;
	DWORD ebp1168h;
	DWORD ebp1164h;
	DWORD ebp1160h;
	DWORD ebp115Ch;
	DWORD ebp1158h;
	DWORD ebp1154h;
	DWORD ebp1150h;
	DWORD ebp114Ch;
	DWORD ebp1148h;
	DWORD ebp1144h;
	DWORD ebp1140h;
	DWORD ebp113Ch;
	DWORD ebp1138h;
	/////////////////////
	// block 2 vars
	/////////////////////
	DWORD ebp146Ch;
	DWORD ebp1198h;		//first param in block 2
	/////////////////////
	// block 3
	/////////////////////
	DWORD ebp119Ch, ebp13B0h;
	/////////////////////
	// block 4
	/////////////////////
	DWORD ebp1474h, ebp1470h, ebp1190h;
	/////////////////////
	// block 5
	/////////////////////
	DWORD ebp147Ch, ebp13BCh, ebp1480h, ebp1484h, ebp1478h,
ebp13A8h, ebp1488h;
	/////////////////////
	// block 7
	////////////////////

	//
	// create debugged process
	//
	memset(&Si, NULL, sizeof(STARTUPINFO));
	memset(&Pi, NULL, sizeof(PROCESS_INFORMATION));

	CreateProcess(PROCESS_NAME, NULL, NULL, NULL, TRUE,
DEBUG_PROCESS, NULL, NULL,
				&Si, &Pi);

	Threads[NumberOfThreads].ThreadHandle = Pi.hThread;
	Threads[NumberOfThreads].ThreadId = Pi.dwThreadId;
	NumberOfThreads++;
```

```
     //
     // debug cycle for debugged process
     //
     while(DebugLoop)
     {
          WaitForDebugEvent(&Event, INFINITE);

          switch(Event.dwDebugEventCode)
          {
               case EXCEPTION_DEBUG_EVENT:

     switch(Event.u.Exception.ExceptionRecord.ExceptionCode)
                    {
                         case EXCEPTION_BREAKPOINT:
                              if(FirstBreak)
                              {
                                   //
                                   // this is first breakpoint in
the process, do nothing
                                   //
                                   FirstBreak = FALSE;
                                   Continue = DBG_CONTINUE;
                              }
                              else
                              {
                                   //
                                   // fixup context.eip
                                   //
///////////////////////////////////////////
// ASM emulation routine
///////////////////////////////////////////
                                   ///////////////////////////
                                   // block 1
                                   ///////////////////////////
                                   __asm
                                   {
                                     mov     dword ptr [ebp1178h],
10h
                                     mov     eax, ds:B1V1
                                     xor     eax, ds:B1V2
                                     xor     eax, ds:B1V3
                                     mov     [ebp1174h], eax
                                     mov     ecx, ds:B1V4
                                     xor     ecx, ds:B1V5
                                     xor     ecx, ds:B1V6
                                     mov     [ebp1170h], ecx
                                     mov     edx, ds:B1V7
                                     xor     edx, ds:B1V8
                                     xor     edx, ds:B1V9
                                     mov     [ebp116Ch], edx
                                     mov     eax, ds:B1V10
```

```
 xor     eax, ds:B1V11
 xor     eax, ds:B1V12
 mov     [ebp1168h], eax
 mov     ecx, ds:B1V13
 xor     ecx, ds:B1V14
 xor     ecx, ds:B1V15
 mov     [ebp1164h], ecx
 mov     edx, ds:B1V16
 xor     edx, ds:B1V17
 xor     edx, ds:B1V18
 mov     [ebp1160h], edx
 mov     eax, ds:B1V1
 xor     eax, ds:B1V4
 xor     eax, ds:B1V7
 xor     eax, ds:B1V19
 mov     [ebp115Ch], eax
 mov     ecx, ds:B1V2
 xor     ecx, ds:B1V5
 xor     ecx, ds:B1V8
 xor     ecx, ds:B1V16
 mov     [ebp1158h], ecx
 mov     edx, ds:B1V10
 xor     edx, ds:B1V13
 xor     edx, ds:B1V16
 xor     edx, ds:B1V17
 mov     [ebp1154h], edx
 mov     eax, ds:B1V11
 xor     eax, ds:B1V14
 xor     eax, ds:B1V17
 xor     eax, ds:B1V1
 mov     [ebp1150h], eax
 mov     ecx, ds:B1V1
 xor     ecx, ds:B1V13
 xor     ecx, ds:B1V4
 mov     [ebp114Ch], ecx
mov     edx, ds:B1V2
 xor     edx, ds:B1V11
 xor     edx, ds:B1V7
 mov     [ebp1148h], edx
 mov     eax, ds:B1V4
 xor     eax, ds:B1V10
 xor     eax, ds:B1V16
 mov     [ebp1144h], eax
 mov     ecx, ds:B1V5
 xor     ecx, ds:B1V8
 xor     ecx, ds:B1V10
 mov     [ebp1140h], ecx
 mov     edx, ds:B1V1
 xor     edx, ds:B1V5
 xor     edx, ds:B1V10
 xor     edx, ds:B1V13
 mov     [ebp113Ch], edx
```

```
                                  mov     eax, ds:B1V2
                                  xor     eax, ds:B1V7
                                  xor     eax, ds:B1V11
                                  xor     eax, ds:B1V14
                                  mov     [ebp1138h], eax
                                }
                                ////////////////////////////
                                // end block 1
                                ////////////////////////////
                                //
                                // find thread handle and fixup
                                //
                                for(j = 0; j < NumberOfThreads;
j++)
                                {
                                     if(Event.dwThreadId ==
Threads[j].ThreadId)
                                     {
                                          break;
                                     }
                                }
                                memset(&Context, NULL,
sizeof(CONTEXT));
                                Context.ContextFlags =
CONTEXT_FULL | CONTEXT_DEBUG_REGISTERS;

     GetThreadContext(Threads[j].ThreadHandle, &Context);
                                //initialization
                                ebp13B0h = Context.Eip;

                                ////////////////////////////
                                // block 2
                                ////////////////////////////
                                ebp146Ch = 0;
                                ebp1198h =
Block2Func1(&(Context.Eip), 4, 0xffffffff);

                                ////////////////////////////
                                // block 3
                                ////////////////////////////
                                __asm
                                {
                                  mov     eax, [ebp1198h]
                                  xor     edx, edx
                                  mov     ecx, 10h
                                  div     ecx
                                  mov     [ebp119Ch], edx
                                  mov     edx, [ebp13B0h]
                                  push    edx
                                  mov     eax, [ebp119Ch]
                                  call
ds:Block3Func1Data1[eax*4]
```

```
                                                add     esp, 4
                                                mov     [ebp146Ch], eax
                                              }

                                              /////////////////////////////
                                              // block 4
                                              /////////////////////////////
                                              __asm
                                              {
                                                mov     dword ptr [ebp1470h], 
0
                                                mov     ecx, [ebp119Ch]
                                                mov     edx, dword ptr 
Block4Data1[ecx*4]
                                                mov     [ebp1190h], edx
B4Label1:
                                                mov     eax, [ebp1470h]
                                                cmp     eax, [ebp1190h]
                                                jge     B4Label5
                                                mov     eax, [ebp1190h]
                                                sub     eax, [ebp1470h]
                                                cdq
                                                sub     eax, edx
                                                sar     eax, 1
                                                mov     ecx, [ebp1470h]
                                                add     ecx, eax
                                                mov     [ebp1474h], ecx
                                                mov     edx, [ebp119Ch]
                                                mov     eax, dword ptr 
Block4Data2[edx*4]
                                                mov     ecx, [ebp1474h]
                                                mov     edx, [ebp146Ch]
                                                cmp     edx, [eax+ecx*4]
                                                jbe     B4Label3
                                                mov     eax, [ebp1474h]
                                                add     eax, 1
                                                mov     [ebp1470h], eax
                                                jmp     B4Label4
B4Label3:
                                                mov     ecx, [ebp1474h]
                                                mov     [ebp1190h], ecx
B4Label4:
                                                jmp     B4Label1
B4Label5:
                                                    nop
                                              }
                                              ////////////////////////////
                                              // block 5
                                              ////////////////////////////
                                              __asm
                                              {
                                                mov     edx, [ebp119Ch]
```

```
                                  mov     eax, dword ptr
Block4Data2[edx*4]
                                  mov     ecx, [ebp1470h]
                                  mov     edx, [eax+ecx*4]
                                  cmp     edx, [ebp146Ch]
                                  jnz     ContinueDebugging
                                }
                                ////////////////////////////
                                // block 6
                                ////////////////////////////
                                ebp13A8h = Context.EFlags;
                                ebp13BCh = 0;
                                __asm
                                {
                                  mov     eax, [ebp119Ch]
                                  mov     ecx, dword ptr
Block6Data1[eax*4]
                                  mov     edx, [ebp1470h]
                                  mov     eax, [ecx+edx*4]
                                  mov     [ebp1488h], eax
                                  mov     ecx, [ebp13A8h] ;
eflags
                                  and     ecx, 0FD7h
                                  mov     [ebp1478h], ecx
                                  mov     edx, [ebp1488h]
                                  and     edx, 0FF000000h
                                  shr     edx, 18h
                                  mov     [ebp1484h], edx
                                  mov     eax, [ebp1488h]
                                  and     eax, 0FFFFFFh
                                  mov     [ebp1480h], eax
                                  mov     ecx, [ebp13BCh]
                                  push    ecx
                                  mov     edx, [ebp1478h]
                                  push    edx             ;
eflags
                                  mov     eax, [ebp1480h]
                                  push    eax
                                  mov     ecx, [ebp1484h]
                                  call    Block6Funcs[ecx*4]
                                  add     esp, 0Ch
                                  mov     [ebp147Ch], eax
                                  mov     edx, [ebp147Ch]
                                  and     edx, 1
                                  test    edx, edx
                                  jz      SkipSecondCalculus
                                }

                                ///////////////////////////////
                                // block 7
                                ///////////////////////////////
                                __asm
```

```
                                  {
//SecondCalculus:
                                    mov     eax, [ebp119Ch]
                                    mov     ecx, dword ptr
B7Array[eax*4]
                                    mov     eax, [ebp1470h]
                                    xor     edx, edx
                                    mov     esi, 10h
                                    div     esi
                                    mov     eax, [ebp1470h]
                                    mov     ecx, [ecx+eax*4]
//                                  xor     ecx, [ebp+edx*4-1174h]
                                        xor  ecx, dword ptr
[B7StackArray+edx*4]
                                    mov     edx, [ebp13B0h]
                                    add     edx, ecx
                                    mov     [ebp13B0h], edx
                                        jmp  ContinueDebugging
                                  }
                                  ///////////////////////////////
                                  // block 8
                                  ///////////////////////////////
                                  __asm
                                  {
SkipSecondCalculus:
                                    mov     eax, [ebp119Ch]
                                     mov     ecx, dword ptr
B8Array[eax*4]
                                    mov     edx, [ebp1470h]
                                    xor     eax, eax
                                    mov     al, [ecx+edx]
                                    mov     ecx, [ebp13B0h]
                                    add     ecx, eax
                                    mov     [ebp13B0h], ecx
                                  }

///////////////////////////////////////////
// end new ASM emulation routine
///////////////////////////////////////////
ContinueDebugging:
                                  Context.Eip = ebp13B0h;

     SetThreadContext(Threads[j].ThreadHandle, &Context);   ///
                                  Continue = DBG_CONTINUE;
                                  break;
                             }
                             break;

                        default:
                             Continue = DBG_EXCEPTION_NOT_HANDLED;
                   }
                   break;
```

```
               case CREATE_THREAD_DEBUG_EVENT:
                    Threads[NumberOfThreads].ThreadId =
Event.dwThreadId;
                    Threads[NumberOfThreads].ThreadHandle =
Event.u.CreateThread.hThread;
                    NumberOfThreads++;
                    Continue = DBG_CONTINUE;
                    break;

               case EXIT_PROCESS_DEBUG_EVENT:
                    MessageBox(NULL, "Exit process", "End", MB_OK);
                    DebugLoop = FALSE;
                    Continue = DBG_CONTINUE;
                    break;

               default:
                    Continue = DBG_CONTINUE;
                    break;
          }
          ContinueDebugEvent(Event.dwProcessId, Event.dwThreadId,
Continue);
     }
     return 0;
}

------------------------------------------------------------------
--------------------------------
```

Using this simple loader the program will finally run.
The following are the files needed by the loader.  They can be easily included in any other project: core.h is the header file, core.cpp contains the emulator functions, coredata.cpp contains all the data arrays needed. If you use this code you just need to dump the correct data arrays. Note that the files are quite big.

```
------------- file core.h ---------------------------------------
--------------------------------
//////////////////////////////////////////////////
//     NANOMULATOR
// armadillo 4.20 nanomites core emulator
// written by andreageddon
//
// uses core.cpp -> functions
//      coredata.cpp -> data tables used by functions
//////////////////////////////////////////////////
#include <windows.h>

/////////////////////////////
// BLOCK1
/////////////////////////////
```

```
extern DWORD B1V1;
extern DWORD B1V2;
extern DWORD B1V3;
extern DWORD B1V4;
extern DWORD B1V5;
extern DWORD B1V6;
extern DWORD B1V7;
extern DWORD B1V8;
extern DWORD B1V9;
extern DWORD B1V10;
extern DWORD B1V11;
extern DWORD B1V12;
extern DWORD B1V13;
extern DWORD B1V14;
extern DWORD B1V15;
extern DWORD B1V16;
extern DWORD B1V17;
extern DWORD B1V18;
extern DWORD B1V19;


////////////////////////////
// BLOCK 2
////////////////////////////

extern BYTE Block2Func2Data1[];
void Block2Func2(void);
DWORD Block2Func1(DWORD *Address, DWORD Param1, DWORD Param2);

////////////////////////////
// BLOCK 3
////////////////////////////

void Block3Func0(void);
void Block3Func1(void);
void Block3Func2(void);
void Block3Func3(void);
void Block3Func4(void);
void Block3Func5(void);
void Block3Func6(void);
void Block3Func7(void);
void Block3Func8(void);
void Block3Func9(void);
void Block3FuncA(void);
void Block3FuncB(void);
void Block3FuncC(void);
void Block3FuncD(void);
void Block3FuncE(void);
void Block3FuncF(void);
extern void* Block3Func1Data1[];

//////////////////////////////
```

```
// block 4
//////////////////////////////
extern BYTE Block4Data1[];
extern void* Block4Data2[];

//////////////////////////////
// block 6
//////////////////////////////

void sub_48DC2D(void);
void sub_48DD35(void);
void sub_48DE3D(void);
void sub_48DF45(void);
void sub_48E04D(void);
void sub_48E155(void);
void sub_48E25D(void);
void sub_48E364(void);
void sub_48E46C(void);
void sub_48E574(void);
void sub_48E67C(void);
void sub_48E783(void);
void sub_48E88B(void);
void sub_48E993(void);
void sub_48EA9B(void);
void sub_48EBA3(void);
void sub_48ECAB(void);
void sub_48EDB3(void);
void sub_48EEBB(void);
void sub_48EFC3(void);
void sub_48F0CB(void);
void sub_48F1D3(void);
void sub_48F2DB(void);
void sub_48F3E3(void);
void sub_48F4EB(void);
void sub_48F5F3(void);
void sub_48F6FB(void);
void sub_48F803(void);
void sub_48F90B(void);
void sub_48FA13(void);
void sub_48FB1B(void);
void sub_48FC20(void);
void sub_48FD28(void);
void sub_48FE2F(void);
void sub_48FF37(void);
void sub_49003F(void);
void sub_490147(void);
void sub_49024F(void);
void sub_490356(void);
void sub_49045E(void);
void sub_490566(void);
void sub_49066E(void);
void sub_490776(void);
void sub_49087E(void);
void sub_490983(void);
void sub_490A8B(void);
void sub_490B93(void);
void sub_490C9B(void);
void sub_490DA3(void);
void sub_490EAB(void);
void sub_490FB3(void);
void sub_4910BB(void);
void sub_4911C3(void);
void sub_4912CB(void);
void sub_4913D3(void);
void sub_4914DB(void);
void sub_4915E3(void);
void sub_4916EB(void);
void sub_4917F3(void);
void sub_4918FB(void);
void sub_491A03(void);
void sub_491B0B(void);
void sub_491C13(void);
void sub_491D1B(void);
void sub_491E20(void);
void sub_491F28(void);
void sub_492030(void);
void sub_492138(void);
void sub_492240(void);
void sub_492345(void);
void sub_49244D(void);
void sub_492555(void);
void sub_49265C(void);
void sub_492764(void);
void sub_49286C(void);
void sub_492974(void);
void sub_492A7C(void);
void sub_492B81(void);
void sub_492C89(void);
void sub_492D91(void);
void sub_492E99(void);
void sub_492FA1(void);
void sub_4930A9(void);
void sub_4931B1(void);
void sub_4932B9(void);
void sub_4933C1(void);
void sub_4934C9(void);
void sub_4935D1(void);
void sub_4936D9(void);
void sub_4937E1(void);
void sub_4938E9(void);
void sub_4939F1(void);
void sub_493AF9(void);
void sub_493C01(void);
void sub_493D09(void);
void sub_493E11(void);
void sub_493F19(void);
void sub_494021(void);
void sub_494129(void);
void sub_494231(void);
void sub_494339(void);
void sub_494441(void);
void sub_494549(void);
void sub_494651(void);
void sub_494759(void);
void sub_494861(void);
void sub_494969(void);
void sub_494A71(void);
void sub_494B79(void);
void sub_494C81(void);
void sub_494D89(void);
void sub_494E91(void);
void sub_494F99(void);
void sub_4950A1(void);
void sub_4951A9(void);
void sub_4952B1(void);
void sub_4953B9(void);
void sub_4954BE(void);
void sub_4955C6(void);
void sub_4956CE(void);
void sub_4957D6(void);
void sub_4958DE(void);
void sub_4959E6(void);
void sub_495AEE(void);
void sub_495BF6(void);
void sub_495CFE(void);
void sub_495E06(void);
void sub_495F0E(void);
void sub_496016(void);
void sub_49611E(void);
void sub_496226(void);
void sub_49632E(void);
void sub_496436(void);
void sub_49653E(void);
void sub_496645(void);
void sub_49674D(void);
void sub_496855(void);
void sub_49695C(void);
void sub_496A64(void);
void sub_496B6C(void);
void sub_496C74(void);
void sub_496D7C(void);
void sub_496E84(void);
void sub_496F8C(void);
void sub_497094(void);
void sub_49719C(void);
void sub_4972A4(void);
void sub_4973AC(void);
void sub_4974B4(void);
void sub_4975BC(void);
void sub_4976C1(void);
void sub_4977C9(void);
void sub_4978D1(void);
void sub_4979D9(void);
void sub_497AE1(void);
void sub_497BE9(void);
void sub_497CF1(void);
void sub_497DF9(void);
void sub_497F01(void);
void sub_498009(void);
void sub_498111(void);
void sub_498219(void);
void sub_498321(void);
void sub_498429(void);
void sub_498531(void);
void sub_498639(void);
void sub_498740(void);
void sub_498848(void);
void sub_498950(void);
void sub_498A58(void);
void sub_498B5D(void);
void sub_498C65(void);
void sub_498D6D(void);
void sub_498E75(void);
void sub_498F7D(void);
void sub_499085(void);
void sub_49918D(void);
void sub_499295(void);
void sub_49939D(void);
void sub_4994A5(void);
void sub_4995AA(void);
void sub_4996B2(void);
void sub_4997BA(void);
void sub_4998C2(void);
void sub_4999CA(void);
void sub_499AD2(void);
void sub_499BDA(void);
void sub_499CDF(void);
void sub_499DE7(void);
void sub_499EEF(void);
void sub_499FF7(void);
void sub_49A0FF(void);
void sub_49A207(void);
void sub_49A30F(void);
void sub_49A417(void);
void sub_49A51F(void);
void sub_49A627(void);
void sub_49A72F(void);
void sub_49A836(void);
void sub_49A93E(void);
void sub_49AA46(void);
void sub_49AB4E(void);
void sub_49AC56(void);
void sub_49AD5E(void);
void sub_49AE66(void);
void sub_49AF6E(void);
void sub_49B076(void);
void sub_49B17E(void);
void sub_49B286(void);
void sub_49B38E(void);
void sub_49B496(void);
void sub_49B59E(void);
void sub_49B6A6(void);
void sub_49B7AE(void);
void sub_49B8B6(void);
void sub_49B9BE(void);
void sub_49BAC6(void);
void sub_49BBCE(void);
void sub_49BCD6(void);
void sub_49BDDE(void);
void sub_49BEE6(void);
void sub_49BFEE(void);
void sub_49C0F6(void);
void sub_49C1FB(void);
void sub_49C303(void);
void sub_49C40B(void);
void sub_49C512(void);
void sub_49C61A(void);
void sub_49C722(void);
void sub_49C82A(void);
void sub_49C932(void);
void sub_49CA3A(void);
void sub_49CB42(void);
void sub_49CC4A(void);
void sub_49CD52(void);
void sub_49CE5A(void);
void sub_49CF62(void);
void sub_49D06A(void);
void sub_49D172(void);
void sub_49D27A(void);
void sub_49D382(void);
void sub_49D48A(void);
void sub_49D592(void);
void sub_49D69A(void);
void sub_49D7A2(void);
void sub_49D8AA(void);
void sub_49D9B2(void);
void sub_49DABA(void);
void sub_49DBC2(void);
void sub_49DCCA(void);
void sub_49DDD2(void);
void sub_49DED7(void);
void sub_49DFDF(void);
void sub_49E0E7(void);
void sub_49E1EF(void);
void sub_49E2F7(void);

void SixBlock0(void);
void SixBlock1(void);
void SixBlock2(void);
void SixBlock3(void);
void SixBlock4(void);
void SixBlock5(void);
void SixBlock6(void);
void SixBlock7(void);
void SixBlock8(void);
void SixBlock9(void);
void SixBlockA(void);
void SixBlockB(void);
void SixBlockC(void);
void SixBlockD(void);
void SixBlockE(void);
void SixBlockF(void);

extern void* Block6Data1[];
extern void* Block6Funcs[];
extern BYTE dword_4DF3C0[];
extern BYTE dword_4D92CC[];
extern void* off_4DDCDC[];
extern BYTE byte_4DDBA0[];
extern BYTE dword_552CA9[];

void sub_48D4EE(void);
void sub_48D2A4(void);
void sub_482654(void);
void sub_48A696(void);
void sub_48B8EB(void);
void sub_48B4C8(void);
void sub_47F71C(void);
void sub_48DA6B(void);
void sub_48A34A(void);
void sub_48778D(void);
void sub_48BCA8(void);
void sub_489D38(void);
void sub_486E92(void);
void sub_485AD2(void);
void sub_48A4A6(void);
void sub_484B20(void);
void sub_48873D(void);
void sub_48D647(void);
void sub_48649F(void);
void sub_4857D3(void);
void sub_48A2BE(void);
void sub_48ABB6(void);
void sub_489EBA(void);
void sub_486D34(void);
void sub_48AEC3(void);
void sub_4850F1(void);
void sub_4821A5(void);
void sub_487EE3(void);
void sub_4896E4(void);
void sub_48BAC0(void);
void sub_484F0B(void);
void sub_48D8AF(void);
void sub_487F9B(void);
void sub_48B85F(void);
void sub_485051(void);
void sub_47F7E9(void);
void sub_485593(void);
void sub_48468F(void);
void sub_4847E1(void);
void sub_487692(void);
void sub_482781(void);
void sub_485315(void);
void sub_486128(void);
void sub_48B379(void);
void sub_4838D4(void);
void sub_488E61(void);
void sub_489A35(void);
void sub_48A589(void);
void sub_483784(void);
void sub_48B048(void);
void sub_48830C(void);
void sub_48D6F8(void);
void sub_48363D(void);
void sub_4808ED(void);
void sub_4837FF(void);
void sub_489F92(void);
void sub_48572F(void);
void sub_486088(void);
void sub_48C479(void);
void sub_484526(void);
void sub_47FD40(void);
void sub_4851C2(void);
void sub_482511(void);
void sub_4872DC(void);
void sub_486C52(void);
void sub_489299(void);
void sub_485EFB(void);
void sub_4867A8(void);
void sub_483D1B(void);
void sub_48D987(void);
void sub_48D37A(void);
void sub_482347(void);
void sub_47FE0F(void);
void sub_48B144(void);
void sub_48C809(void);
void sub_48ADAB(void);
void sub_484DC4(void);
void sub_48A8A9(void);
void sub_488671(void);
void sub_489AE0(void);
void sub_48A3DB(void);
void sub_488260(void);
void sub_4859B7(void);
void sub_485898(void);
void sub_48BF8D(void);
void sub_487BED(void);
void sub_483136(void);
void sub_485479(void);
void sub_48B41E(void);
void sub_489500(void);
void sub_4853D4(void);
void sub_480513(void);
void sub_480753(void);
void sub_48027A(void);
void sub_48C558(void);
void sub_48818D(void);
void sub_4826EF(void);
void sub_489C62(void);
void sub_486362(void);
void sub_4868CA(void);
void sub_48C8AB(void);
void sub_483C44(void);
void sub_48CE93(void);
void sub_4898DB(void);
void sub_4806A8(void);
void sub_48839F(void);
void sub_484D34(void);
void sub_48C5F7(void);
void sub_48903D(void);
void sub_482AC5(void);
void sub_4879C2(void);
void sub_4848BA(void);
void sub_487CA9(void);
void sub_48565C(void);
void sub_485C4D(void);
void sub_48AAED(void);
void sub_482867(void);
void sub_484BCA(void);
void sub_483549(void);
void sub_480ABC(void);
void sub_48654F(void);
void sub_489214(void);
void sub_48A749(void);
void sub_485280(void);
void sub_482FB4(void);
void sub_482EFE(void);
void sub_47FA7F(void);
void sub_4803F9(void);
void sub_485CCB(void);
void sub_48DAF1(void);
void sub_485A3D(void);
void sub_4840CF(void);
void sub_487137(void);
void sub_482930(void);
void sub_48CAF8(void);
void sub_48D461(void);
void sub_484315(void);
void sub_4833EE(void);
void sub_48A015(void);
void sub_488F97(void);
void sub_483208(void);
void sub_488CA2(void);
void sub_487B44(void);
void sub_4823C8(void);
void sub_48097F(void);
void sub_4829D3(void);
void sub_480350(void);
void sub_4845BB(void);
void sub_484488(void);
void sub_48B65C(void);
void sub_48A993(void);
void sub_487DEA(void);
void sub_482E3D(void);
void sub_488055(void);
void sub_48349D(void);
void sub_484E68(void);
void sub_488108(void);
void sub_48246C(void);
void sub_482C9E(void);
void sub_488475(void);
void sub_486299(void);
void sub_489BB2(void);
void sub_48BED9(void);
void sub_48DBAA(void);
void sub_483EB0(void);
void sub_484003(void);
void sub_48682F(void);
void sub_480022(void);
void sub_487043(void);
void sub_487383(void);
void sub_488C12(void);
void sub_489798(void);
void sub_4861C8(void);
void sub_47FCB2(void);
void sub_48A0FB(void);
void sub_4889F5(void);
void sub_48A60F(void);
void sub_48CFE9(void);
void sub_483F62(void);
void sub_47FBF8(void);
void sub_48493A(void);
void sub_48C274(void);
void sub_4899A6(void);
void sub_48BE35(void);
void sub_4885F8(void);
void sub_4836E0(void);
void sub_487916(void);
void sub_487445(void);
void sub_48C973(void);
void sub_48640A(void);
void sub_48B2BC(void);
void sub_488F0A(void);
void sub_48A1B3(void);
void sub_48C180(void);
void sub_483349(void);
void sub_48AF5D(void);
void sub_483976(void);
void sub_48CF27(void);
void sub_489832(void);
void sub_48BB6B(void);
void sub_48D1F8(void);
void sub_4832AE(void);
void sub_48B761(void);
void sub_48D098(void);
void sub_47F9AE(void);
void sub_484A99(void);
void sub_48B9EF(void);
void sub_488B2B(void);
void sub_48CDAE(void);
void sub_47FF6D(void);
void sub_48BC05(void);
void sub_483B62(void);
void sub_48854E(void);
void sub_48912A(void);
void sub_4888F0(void);
void sub_482B40(void);
void sub_485E46(void);
void sub_47F8EF(void);
void sub_487D4D(void);
void sub_4875A3(void);
void sub_48C3D7(void);
void sub_48D7F5(void);
void sub_4874F5(void);
void sub_486A5F(void);
void sub_48C776(void);
void sub_48D11F(void);
void sub_48B1EC(void);
void sub_48C332(void);
void sub_484752(void);
void sub_48CC9F(void);
void sub_48B5B5(void);
void sub_48AA4C(void);
void sub_4865F8(void);
void sub_4807FD(void);
void sub_488AA6(void);
void sub_480A1B(void);
void sub_4801E3(void);
void sub_484255(void);
void sub_4843C4(void);
void sub_48A7DD(void);
void sub_48C6A1(void);
void sub_485BB2(void);
void sub_480491(void);
void sub_484C97(void);
void sub_48259B(void);
void sub_484F93(void);
void sub_488803(void);
void sub_482D75(void);
void sub_4866AC(void);
void sub_4822B2(void);
void sub_47FB2E(void);
void sub_489419(void);

//block 6 dynamic functions sub data
extern DWORD dword_4D9374;
extern DWORD dword_4D9370;
extern DWORD dword_4D937C;
extern DWORD dword_4D9380;
extern DWORD dword_4D93AC;
extern DWORD dword_4D93B0;
extern DWORD dword_4D93A8;
extern DWORD dword_4D9378;
extern DWORD dword_4D93A4;
extern DWORD dword_4D9388;
extern DWORD dword_4D938C;
extern DWORD dword_4D9394;
extern DWORD dword_4D9398;
extern DWORD dword_4D939C;
extern DWORD dword_4D93A0;
extern DWORD dword_4D9390;
extern DWORD dword_4D9384;

extern DWORD unk_552CA9;

//block 6 dynamic functions sub functions layer 1
extern void* off_4DDC9C;
extern void* off_4DDCA0;
extern void* off_4DDCA4;
extern void* off_4DDCA8;
extern void* off_4DDCAC;
extern void* off_4DDCB0;
extern void* off_4DDCB4;
extern void* off_4DDCB8;
extern void* off_4DDCBC;
extern void* off_4DDCC0;
extern void* off_4DDCC4;
extern void* off_4DDCC8;
extern void* off_4DDCCC;
extern void* off_4DDCD0;
extern void* off_4DDCD4;
extern void* off_4DDCD8;
extern void* off_4DDCDC_2;
extern void* off_4DDCE0;
extern void* off_4DDCE4;
extern void* off_4DDCE8;
extern void* off_4DDCEC;
extern void* off_4DDCF0;
extern void* off_4DDCF4;
extern void* off_4DDCF8;
extern void* off_4DDCFC;
extern void* off_4DDD00;
extern void* off_4DDD04;
extern void* off_4DDD08;
extern void* off_4DDD0C;
extern void* off_4DDD10;
extern void* off_4DDD14;
extern void* off_4DDD18;



void sub_47CFB0(void);
void sub_47D0EF(void);
void sub_47D2B5(void);
void sub_47D4F9(void);
void sub_47D6CE(void);
void sub_47D9DB(void);
void sub_47DCDA(void);
void sub_47DE96(void);
void sub_47E17A(void);
void sub_47E2FC(void);
void sub_47E508(void);
void sub_47E746(void);
void sub_47E9B4(void);
void sub_47EB90(void);
void sub_47EE6B(void);
void sub_47F09F(void);
void sub_47D04F(void);
void sub_47D1D2(void);
void sub_47D3D7(void);
void sub_47D5E3(void);
void sub_47D854(void);
void sub_47DB59(void);
void sub_47DDB8(void);
void sub_47E008(void);
void sub_47E23B(void);
void sub_47E402(void);
void sub_47E627(void);
void sub_47E87C(void);
void sub_47EAA2(void);
void sub_47ECFE(void);
void sub_47EF85(void);
void sub_47F22B(void);


DWORD B6OriginalFuncs[];
void* Block6DynamicFuncs[];



////////////////////////////
// block 7
////////////////////////////

void* B7Array[];
BYTE B7StackArray[];


////////////////////////////
// block 8
////////////////////////////

void* B8Array[];
------------------------------------------------------------------
--------------------------------



------------- file coredata.cpp ----------------------------------
--------------------------------
//////////////////////////////////////////////////
//     NANOMULATOR
// armadillo 4.20 nanomites core emulator
// written by andreageddon
//
// data used by core.cpp functions
//////////////////////////////////////////////////
#include <windows.h>
#include "core.h"

////////////////////////////////////////////
// block 1
////////////////////////////////////////////
DWORD B1V1 = 0xB8EA996B;
DWORD B1V2 = 0xA27B7849;
DWORD B1V3 = 0xDDC1D886;
DWORD B1V4 = 0xB9E1AC81;
DWORD B1V5 = 0xBA510D2C;
DWORD B1V6 = 0xF61E987;
DWORD B1V7 = 0x6A073847;
DWORD B1V8 = 0x6A23D5FB;
DWORD B1V9 = 0x79D514E2;
DWORD B1V10 = 0x3A612A59;
DWORD B1V11 = 0xB81D9989;
DWORD B1V12 = 0x338F1DEE;
DWORD B1V13 = 0x5217B725;
DWORD B1V14 = 0xD70E4E5A;
DWORD B1V15 = 0xBD3CAC82;
DWORD B1V16 = 0xA8396520;
DWORD B1V17 = 0xE4505EBF;
DWORD B1V18 = 0x6F1A9142;
DWORD B1V19 = 0x2BA925AD;

////////////////////////////////////////////
// block 2
////////////////////////////////////////////

BYTE Block2Func2Data1[] = {
/*0000:*/ 0x00, 0x00, 0x00, 0x00, 0x96, 0x30, 0x07, 0x77, 0x2C,
0x61, 0x0E, 0xEE, 0xBA, 0x51, 0x09, 0x99,
/*0010:*/ 0x19, 0xC4, 0x6D, 0x07, 0x8F, 0xF4, 0x6A, 0x70, 0x35,
0xA5, 0x63, 0xE9, 0xA3, 0x95, 0x64, 0x9E,
/*0020:*/ 0x32, 0x88, 0xDB, 0x0E, 0xA4, 0xB8, 0xDC, 0x79, 0x1E,
0xE9, 0xD5, 0xE0, 0x88, 0xD9, 0xD2, 0x97,
/*0030:*/ 0x2B, 0x4C, 0xB6, 0x09, 0xBD, 0x7C, 0xB1, 0x7E, 0x07,
0x2D, 0xB8, 0xE7, 0x91, 0x1D, 0xBF, 0x90,
/*0040:*/ 0x64, 0x10, 0xB7, 0x1D, 0xF2, 0x20, 0xB0, 0x6A, 0x48,
0x71, 0xB9, 0xF3, 0xDE, 0x41, 0xBE, 0x84,
/*0050:*/ 0x7D, 0xD4, 0xDA, 0x1A, 0xEB, 0xE4, 0xDD, 0x6D, 0x51,
0xB5, 0xD4, 0xF4, 0xC7, 0x85, 0xD3, 0x83,
/*0060:*/ 0x56, 0x98, 0x6C, 0x13, 0xC0, 0xA8, 0x6B, 0x64, 0x7A,
0xF9, 0x62, 0xFD, 0xEC, 0xC9, 0x65, 0x8A,
/*0070:*/ 0x4F, 0x5C, 0x01, 0x14, 0xD9, 0x6C, 0x06, 0x63, 0x63,
0x3D, 0x0F, 0xFA, 0xF5, 0x0D, 0x08, 0x8D,
/*0080:*/ 0xC8, 0x20, 0x6E, 0x3B, 0x5E, 0x10, 0x69, 0x4C, 0xE4,
0x41, 0x60, 0xD5, 0x72, 0x71, 0x67, 0xA2,
/*0090:*/ 0xD1, 0xE4, 0x03, 0x3C, 0x47, 0xD4, 0x04, 0x4B, 0xFD,
0x85, 0x0D, 0xD2, 0x6B, 0xB5, 0x0A, 0xA5,
/*00A0:*/ 0xFA, 0xA8, 0xB5, 0x35, 0x6C, 0x98, 0xB2, 0x42, 0xD6,
0xC9, 0xBB, 0xDB, 0x40, 0xF9, 0xBC, 0xAC,
/*00B0:*/ 0xE3, 0x6C, 0xD8, 0x32, 0x75, 0x5C, 0xDF, 0x45, 0xCF,
0x0D, 0xD6, 0xDC, 0x59, 0x3D, 0xD1, 0xAB,
/*00C0:*/ 0xAC, 0x30, 0xD9, 0x26, 0x3A, 0x00, 0xDE, 0x51, 0x80,
0x51, 0xD7, 0xC8, 0x16, 0x61, 0xD0, 0xBF,
/*00D0:*/ 0xB5, 0xF4, 0xB4, 0x21, 0x23, 0xC4, 0xB3, 0x56, 0x99,
0x95, 0xBA, 0xCF, 0x0F, 0xA5, 0xBD, 0xB8,
/*00E0:*/ 0x9E, 0xB8, 0x02, 0x28, 0x08, 0x88, 0x05, 0x5F, 0xB2,
0xD9, 0x0C, 0xC6, 0x24, 0xE9, 0x0B, 0xB1,
/*00F0:*/ 0x87, 0x7C, 0x6F, 0x2F, 0x11, 0x4C, 0x68, 0x58, 0xAB,
0x1D, 0x61, 0xC1, 0x3D, 0x2D, 0x66, 0xB6,
/*0100:*/ 0x90, 0x41, 0xDC, 0x76, 0x06, 0x71, 0xDB, 0x01, 0xBC,
0x20, 0xD2, 0x98, 0x2A, 0x10, 0xD5, 0xEF,
/*0110:*/ 0x89, 0x85, 0xB1, 0x71, 0x1F, 0xB5, 0xB6, 0x06, 0xA5,
0xE4, 0xBF, 0x9F, 0x33, 0xD4, 0xB8, 0xE8,
/*0120:*/ 0xA2, 0xC9, 0x07, 0x78, 0x34, 0xF9, 0x00, 0x0F, 0x8E,
0xA8, 0x09, 0x96, 0x18, 0x98, 0x0E, 0xE1,
/*0130:*/ 0xBB, 0x0D, 0x6A, 0x7F, 0x2D, 0x3D, 0x6D, 0x08, 0x97,
0x6C, 0x64, 0x91, 0x01, 0x5C, 0x63, 0xE6,
/*0140:*/ 0xF4, 0x51, 0x6B, 0x6B, 0x62, 0x61, 0x6C, 0x1C, 0xD8,
0x30, 0x65, 0x85, 0x4E, 0x00, 0x62, 0xF2,
/*0150:*/ 0xED, 0x95, 0x06, 0x6C, 0x7B, 0xA5, 0x01, 0x1B, 0xC1,
0xF4, 0x08, 0x82, 0x57, 0xC4, 0x0F, 0xF5,
/*0160:*/ 0xC6, 0xD9, 0xB0, 0x65, 0x50, 0xE9, 0xB7, 0x12, 0xEA,
0xB8, 0xBE, 0x8B, 0x7C, 0x88, 0xB9, 0xFC,
/*0170:*/ 0xDF, 0x1D, 0xDD, 0x62, 0x49, 0x2D, 0xDA, 0x15, 0xF3,
0x7C, 0xD3, 0x8C, 0x65, 0x4C, 0xD4, 0xFB,
/*0180:*/ 0x58, 0x61, 0xB2, 0x4D, 0xCE, 0x51, 0xB5, 0x3A, 0x74,
0x00, 0xBC, 0xA3, 0xE2, 0x30, 0xBB, 0xD4,
/*0190:*/ 0x41, 0xA5, 0xDF, 0x4A, 0xD7, 0x95, 0xD8, 0x3D, 0x6D,
0xC4, 0xD1, 0xA4, 0xFB, 0xF4, 0xD6, 0xD3,
/*01A0:*/ 0x6A, 0xE9, 0x69, 0x43, 0xFC, 0xD9, 0x6E, 0x34, 0x46,
0x88, 0x67, 0xAD, 0xD0, 0xB8, 0x60, 0xDA,
/*01B0:*/ 0x73, 0x2D, 0x04, 0x44, 0xE5, 0x1D, 0x03, 0x33, 0x5F,
0x4C, 0x0A, 0xAA, 0xC9, 0x7C, 0x0D, 0xDD,
/*01C0:*/ 0x3C, 0x71, 0x05, 0x50, 0xAA, 0x41, 0x02, 0x27, 0x10,
0x10, 0x0B, 0xBE, 0x86, 0x20, 0x0C, 0xC9,
/*01D0:*/ 0x25, 0xB5, 0x68, 0x57, 0xB3, 0x85, 0x6F, 0x20, 0x09,
0xD4, 0x66, 0xB9, 0x9F, 0xE4, 0x61, 0xCE,
/*01E0:*/ 0x0E, 0xF9, 0xDE, 0x5E, 0x98, 0xC9, 0xD9, 0x29, 0x22,
0x98, 0xD0, 0xB0, 0xB4, 0xA8, 0xD7, 0xC7,
/*01F0:*/ 0x17, 0x3D, 0xB3, 0x59, 0x81, 0x0D, 0xB4, 0x2E, 0x3B,
0x5C, 0xBD, 0xB7, 0xAD, 0x6C, 0xBA, 0xC0,
/*0200:*/ 0x20, 0x83, 0xB8, 0xED, 0xB6, 0xB3, 0xBF, 0x9A, 0x0C,
0xE2, 0xB6, 0x03, 0x9A, 0xD2, 0xB1, 0x74,
/*0210:*/ 0x39, 0x47, 0xD5, 0xEA, 0xAF, 0x77, 0xD2, 0x9D, 0x15,
0x26, 0xDB, 0x04, 0x83, 0x16, 0xDC, 0x73,
/*0220:*/ 0x12, 0x0B, 0x63, 0xE3, 0x84, 0x3B, 0x64, 0x94, 0x3E,
0x6A, 0x6D, 0x0D, 0xA8, 0x5A, 0x6A, 0x7A,
/*0230:*/ 0x0B, 0xCF, 0x0E, 0xE4, 0x9D, 0xFF, 0x09, 0x93, 0x27,
0xAE, 0x00, 0x0A, 0xB1, 0x9E, 0x07, 0x7D,
/*0240:*/ 0x44, 0x93, 0x0F, 0xF0, 0xD2, 0xA3, 0x08, 0x87, 0x68,
0xF2, 0x01, 0x1E, 0xFE, 0xC2, 0x06, 0x69,
/*0250:*/ 0x5D, 0x57, 0x62, 0xF7, 0xCB, 0x67, 0x65, 0x80, 0x71,
0x36, 0x6C, 0x19, 0xE7, 0x06, 0x6B, 0x6E,
/*0260:*/ 0x76, 0x1B, 0xD4, 0xFE, 0xE0, 0x2B, 0xD3, 0x89, 0x5A,
0x7A, 0xDA, 0x10, 0xCC, 0x4A, 0xDD, 0x67,
/*0270:*/ 0x6F, 0xDF, 0xB9, 0xF9, 0xF9, 0xEF, 0xBE, 0x8E, 0x43,
0xBE, 0xB7, 0x17, 0xD5, 0x8E, 0xB0, 0x60,
/*0280:*/ 0xE8, 0xA3, 0xD6, 0xD6, 0x7E, 0x93, 0xD1, 0xA1, 0xC4,
0xC2, 0xD8, 0x38, 0x52, 0xF2, 0xDF, 0x4F,
/*0290:*/ 0xF1, 0x67, 0xBB, 0xD1, 0x67, 0x57, 0xBC, 0xA6, 0xDD,
0x06, 0xB5, 0x3F, 0x4B, 0x36, 0xB2, 0x48,
/*02A0:*/ 0xDA, 0x2B, 0x0D, 0xD8, 0x4C, 0x1B, 0x0A, 0xAF, 0xF6,
0x4A, 0x03, 0x36, 0x60, 0x7A, 0x04, 0x41,
/*02B0:*/ 0xC3, 0xEF, 0x60, 0xDF, 0x55, 0xDF, 0x67, 0xA8, 0xEF,
0x8E, 0x6E, 0x31, 0x79, 0xBE, 0x69, 0x46,
/*02C0:*/ 0x8C, 0xB3, 0x61, 0xCB, 0x1A, 0x83, 0x66, 0xBC, 0xA0,
0xD2, 0x6F, 0x25, 0x36, 0xE2, 0x68, 0x52,
/*02D0:*/ 0x95, 0x77, 0x0C, 0xCC, 0x03, 0x47, 0x0B, 0xBB, 0xB9,
0x16, 0x02, 0x22, 0x2F, 0x26, 0x05, 0x55,
/*02E0:*/ 0xBE, 0x3B, 0xBA, 0xC5, 0x28, 0x0B, 0xBD, 0xB2, 0x92,
0x5A, 0xB4, 0x2B, 0x04, 0x6A, 0xB3, 0x5C,
/*02F0:*/ 0xA7, 0xFF, 0xD7, 0xC2, 0x31, 0xCF, 0xD0, 0xB5, 0x8B,
0x9E, 0xD9, 0x2C, 0x1D, 0xAE, 0xDE, 0x5B,
/*0300:*/ 0xB0, 0xC2, 0x64, 0x9B, 0x26, 0xF2, 0x63, 0xEC, 0x9C,
0xA3, 0x6A, 0x75, 0x0A, 0x93, 0x6D, 0x02,
/*0310:*/ 0xA9, 0x06, 0x09, 0x9C, 0x3F, 0x36, 0x0E, 0xEB, 0x85,
0x67, 0x07, 0x72, 0x13, 0x57, 0x00, 0x05,
/*0320:*/ 0x82, 0x4A, 0xBF, 0x95, 0x14, 0x7A, 0xB8, 0xE2, 0xAE,
0x2B, 0xB1, 0x7B, 0x38, 0x1B, 0xB6, 0x0C,
/*0330:*/ 0x9B, 0x8E, 0xD2, 0x92, 0x0D, 0xBE, 0xD5, 0xE5, 0xB7,
0xEF, 0xDC, 0x7C, 0x21, 0xDF, 0xDB, 0x0B,
/*0340:*/ 0xD4, 0xD2, 0xD3, 0x86, 0x42, 0xE2, 0xD4, 0xF1, 0xF8,
0xB3, 0xDD, 0x68, 0x6E, 0x83, 0xDA, 0x1F,
/*0350:*/ 0xCD, 0x16, 0xBE, 0x81, 0x5B, 0x26, 0xB9, 0xF6, 0xE1,
0x77, 0xB0, 0x6F, 0x77, 0x47, 0xB7, 0x18,
/*0360:*/ 0xE6, 0x5A, 0x08, 0x88, 0x70, 0x6A, 0x0F, 0xFF, 0xCA,
0x3B, 0x06, 0x66, 0x5C, 0x0B, 0x01, 0x11,
/*0370:*/ 0xFF, 0x9E, 0x65, 0x8F, 0x69, 0xAE, 0x62, 0xF8, 0xD3,
0xFF, 0x6B, 0x61, 0x45, 0xCF, 0x6C, 0x16,
/*0380:*/ 0x78, 0xE2, 0x0A, 0xA0, 0xEE, 0xD2, 0x0D, 0xD7, 0x54,
0x83, 0x04, 0x4E, 0xC2, 0xB3, 0x03, 0x39,
/*0390:*/ 0x61, 0x26, 0x67, 0xA7, 0xF7, 0x16, 0x60, 0xD0, 0x4D,
0x47, 0x69, 0x49, 0xDB, 0x77, 0x6E, 0x3E,
/*03A0:*/ 0x4A, 0x6A, 0xD1, 0xAE, 0xDC, 0x5A, 0xD6, 0xD9, 0x66,
0x0B, 0xDF, 0x40, 0xF0, 0x3B, 0xD8, 0x37,
/*03B0:*/ 0x53, 0xAE, 0xBC, 0xA9, 0xC5, 0x9E, 0xBB, 0xDE, 0x7F,
0xCF, 0xB2, 0x47, 0xE9, 0xFF, 0xB5, 0x30,
/*03C0:*/ 0x1C, 0xF2, 0xBD, 0xBD, 0x8A, 0xC2, 0xBA, 0xCA, 0x30,
0x93, 0xB3, 0x53, 0xA6, 0xA3, 0xB4, 0x24,
/*03D0:*/ 0x05, 0x36, 0xD0, 0xBA, 0x93, 0x06, 0xD7, 0xCD, 0x29,
0x57, 0xDE, 0x54, 0xBF, 0x67, 0xD9, 0x23,
/*03E0:*/ 0x2E, 0x7A, 0x66, 0xB3, 0xB8, 0x4A, 0x61, 0xC4, 0x02,
0x1B, 0x68, 0x5D, 0x94, 0x2B, 0x6F, 0x2A,
/*03F0:*/ 0x37, 0xBE, 0x0B, 0xB4, 0xA1, 0x8E, 0x0C, 0xC3, 0x1B,
0xDF, 0x05, 0x5A, 0x8D, 0xEF, 0x02, 0x2D,
/*0400:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

/////////////////
// block 3
```

```
/////////////////

void* Block3Func1Data1[] = {
          Block3Func0,
          Block3Func1,
          Block3Func2,
          Block3Func3,
          Block3Func4,
          Block3Func5,
          Block3Func6,
          Block3Func7,
          Block3Func8,
          Block3Func9,
          Block3FuncA,
          Block3FuncB,
          Block3FuncC,
          Block3FuncD,
          Block3FuncE,
          Block3FuncF
};


/////////////////////
// block 4
/////////////////////

BYTE B3D2_0[] = {
/*00C9F8:*/ 0xC3, 0xB1, 0x04, 0xF2, 0x8B, 0x91, 0x11, 0xF2,
/*00CA00:*/ 0x99, 0x91, 0x11, 0xF2, 0x8B, 0xA1, 0x11, 0xF2, 0x99,
0xA1, 0x11, 0xF2, 0xDC, 0xA1, 0x13, 0xF2,
/*00CA10:*/ 0xE8, 0xF1, 0x13, 0xF2, 0xB7, 0x81, 0x15, 0xF2, 0xE0,
0x81, 0x15, 0xF2, 0x8B, 0x91, 0x15, 0xF2,
/*00CA20:*/ 0xCF, 0x91, 0x15, 0xF2, 0xE0, 0xB1, 0x15, 0xF2, 0xE9,
0xF1, 0x15, 0xF2, 0x89, 0x41, 0x16, 0xF2,
/*00CA30:*/ 0x98, 0x91, 0x17, 0xF2, 0xBE, 0xF1, 0x17, 0xF2, 0x8D,
0xD1, 0x18, 0xF2, 0xD5, 0xE1, 0x19, 0xF2,
/*00CA40:*/ 0x98, 0xA1, 0x1B, 0xF2, 0xCE, 0xA1, 0x1B, 0xF2, 0xF3,
0xB1, 0x1B, 0xF2, 0xAC, 0xC1, 0x1B, 0xF2,
/*00CA50:*/ 0xE9, 0xC1, 0x1B, 0xF2, 0xAC, 0xF1, 0x1B, 0xF2, 0xE9,
0xF1, 0x1D, 0xF2, 0xBD, 0x11, 0x1E, 0xF2,
/*00CA60:*/ 0xF9, 0x21, 0x1E, 0xF2, 0x8C, 0xD1, 0x1E, 0xF2, 0xA9,
0x61, 0x1F, 0xF2, 0xDD, 0x91, 0x1F, 0xF2,
/*00CA70:*/ 0xAB, 0xC1, 0x31, 0xF2, 0xE1, 0xC1, 0x32, 0xF2, 0x9E,
0x91, 0x33, 0xF2, 0xE6, 0xB1, 0x33, 0xF2,
/*00CA80:*/ 0xD2, 0xE1, 0x33, 0xF2, 0xFD, 0xF1, 0x33, 0xF2, 0xF5,
0x81, 0x35, 0xF2, 0xEF, 0xC1, 0x35, 0xF2,
/*00CA90:*/ 0x97, 0xE1, 0x35, 0xF2, 0x82, 0xA1, 0x36, 0xF2, 0xB8,
0xC1, 0x39, 0xF2, 0xAB, 0xF1, 0x39, 0xF2,
/*00CAA0:*/ 0x8C, 0x91, 0x3B, 0xF2, 0xAB, 0xC1, 0x3D, 0xF2, 0xCE,
0xE1, 0x3E, 0xF2, 0x8D, 0xA1, 0x3F, 0xF2,
/*00CAB0:*/ 0xAA, 0xC1, 0x3F, 0xF2, 0xBC, 0x71, 0x50, 0xF2, 0xAA,
0xE1, 0x52, 0xF2, 0xAD, 0x91, 0x53, 0xF2,
/*00CAC0:*/ 0xBF, 0x91, 0x53, 0xF2, 0xBF, 0xA1, 0x53, 0xF2, 0x99,
0xC1, 0x53, 0xF2, 0xFB, 0x91, 0x55, 0xF2,
/*00CAD0:*/ 0xB1, 0x91, 0x58, 0xF2, 0x85, 0xF1, 0x5A, 0xF2, 0xD5,
0xB1, 0x5B, 0xF2, 0xF4, 0x91, 0x5C, 0xF2,
/*00CAE0:*/ 0x8B, 0xC1, 0x5D, 0xF2, 0xF3, 0xE1, 0x5D, 0xF2, 0xD1,
0x21, 0x5F, 0xF2, 0x8B, 0xF1, 0x5F, 0xF2,
/*00CAF0:*/ 0x8A, 0x81, 0x72, 0xF2, 0xA4, 0x91, 0x72, 0xF2, 0xE8,
0xD1, 0x72, 0xF2, 0xFA, 0xD1, 0x72, 0xF2,
/*00CB00:*/ 0x8D, 0xC1, 0x73, 0xF2, 0xC9, 0xC1, 0x73, 0xF2, 0xD3,
0xB1, 0x75, 0xF2, 0x9D, 0x21, 0x76, 0xF2,
/*00CB10:*/ 0x8C, 0xC1, 0x77, 0xF2, 0x8A, 0xB1, 0x78, 0xF2, 0x90,
0xF1, 0x78, 0xF2, 0xC7, 0xF1, 0x78, 0xF2,
/*00CB20:*/ 0xDB, 0xC1, 0x79, 0xF2, 0xDB, 0xF1, 0x79, 0xF2, 0xAA,
0xA1, 0x7B, 0xF2, 0xE7, 0xD1, 0x7D, 0xF2,
/*00CB30:*/ 0xB1, 0xE1, 0x7D, 0xF2, 0x99, 0xB1, 0x7E, 0xF2, 0x83,
0xF1, 0x7E, 0xF2, 0x96, 0x81, 0x7F, 0xF2,
/*00CB40:*/ 0xD2, 0x81, 0x7F, 0xF2, 0xFD, 0xA1, 0x7F, 0xF2, 0x8D,
0xB1, 0x88, 0xF2, 0x9D, 0x91, 0x92, 0xF2,
/*00CB50:*/ 0xFF, 0xC1, 0x92, 0xF2, 0xF9, 0xB1, 0x93, 0xF2, 0xE2,
0xC1, 0x93, 0xF2, 0xA5, 0x21, 0x96, 0xF2,
/*00CB60:*/ 0x94, 0xD1, 0x96, 0xF2, 0xF6, 0x81, 0x9A, 0xF2, 0xC4,
0xA1, 0x9B, 0xF2, 0xCD, 0xE1, 0x9B, 0xF2,
/*00CB70:*/ 0xD7, 0xA1, 0x9D, 0xF2, 0xAF, 0xB1, 0x9D, 0xF2, 0xB5,
0xC1, 0x9D, 0xF2, 0xA5, 0x21, 0x9E, 0xF2,
/*00CB80:*/ 0xB4, 0xF1, 0x9F, 0xF2, 0x99, 0x71, 0xB1, 0xF2, 0xA9,
0xB1, 0xB1, 0xF2, 0x99, 0x41, 0xB3, 0xF2,
/*00CB90:*/ 0xBA, 0xB1, 0xB3, 0xF2, 0xCA, 0xE1, 0xB3, 0xF2, 0xF7,
0xF1, 0xB3, 0xF2, 0xB9, 0x51, 0xB4, 0xF2,
/*00CBA0:*/ 0x86, 0xA1, 0xB5, 0xF2, 0xBB, 0xB1, 0xB5, 0xF2, 0xF6,
0xF1, 0xB5, 0xF2, 0xAE, 0xF1, 0xB6, 0xF2,
/*00CBB0:*/ 0xA8, 0x81, 0xB7, 0xF2, 0x8F, 0xD1, 0xB7, 0xF2, 0xCA,
0xD1, 0xB9, 0xF2, 0xDE, 0x91, 0xBA, 0xF2,
/*00CBC0:*/ 0xAE, 0xC1, 0xBA, 0xF2, 0xED, 0x81, 0xBD, 0xF2, 0xC3,
0xA1, 0xBD, 0xF2, 0xF7, 0xF1, 0xBD, 0xF2,
/*00CBD0:*/ 0xBA, 0x91, 0xD0, 0xF2, 0xA0, 0xD1, 0xD2, 0xF2, 0xB2,
0xE1, 0xD2, 0xF2, 0xF6, 0xE1, 0xD2, 0xF2,
/*00CBE0:*/ 0xF6, 0xD1, 0xD6, 0xF2, 0xFD, 0x71, 0xD7, 0xF2, 0xC4,
0xF1, 0xD7, 0xF2, 0xA0, 0xE1, 0xD8, 0xF2,
/*00CBF0:*/ 0xE2, 0x91, 0xD9, 0xF2, 0xAF, 0xD1, 0xD9, 0xF2, 0xBD,
0xE1, 0xD9, 0xF2, 0xDE, 0x81, 0xDB, 0xF2,
/*00CC00:*/ 0x9B, 0xB1, 0xDD, 0xF2, 0xB2, 0xA1, 0xF1, 0xF2, 0xA3,
0x71, 0xF2, 0xF2, 0x92, 0x81, 0xF2, 0xF2,
/*00CC10:*/ 0xC3, 0xF1, 0xF3, 0xF2, 0x84, 0x21, 0xF4, 0xF2, 0xC9,
0x51, 0xF6, 0xF2, 0xDB, 0x61, 0xF6, 0xF2,
/*00CC20:*/ 0xAD, 0x41, 0xF7, 0xF2, 0xE4, 0xA1, 0xF9, 0xF2, 0x86,
0xC1, 0xF9, 0xF2, 0xF7, 0x91, 0xFB, 0xF2,
/*00CC30:*/ 0x87, 0xF1, 0xFB, 0xF2, 0x97, 0x11, 0xFE, 0xF2, 0xD0,
0xF1, 0xFF, 0xF2, 0x0D, 0xF0, 0xAD, 0xBA,
/*00CC40:*/ 0x0D, 0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*00CC50:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};
```

```
BYTE B3D2_1[] = {
/*00D1E8:*/ 0x29, 0xD7, 0x1C, 0x20, 0x49, 0xD4, 0x9A, 0x20,
/*00D1F0:*/ 0x29, 0xD4, 0xC8, 0x21, 0xC9, 0xD7, 0x8C, 0x22, 0x89,
0xD4, 0x1A, 0x23, 0x89, 0xD7, 0x3E, 0x23,
/*00D200:*/ 0x51, 0xD4, 0x9E, 0x23, 0x09, 0xD5, 0x2A, 0x28, 0x49,
0xD6, 0x2C, 0x28, 0x49, 0xD4, 0x9C, 0x28,
/*00D210:*/ 0x89, 0xD5, 0xEC, 0x28, 0x49, 0xD7, 0x08, 0x29, 0xE9,
0xD5, 0xBE, 0x29, 0x99, 0xD7, 0xEC, 0x29,
/*00D220:*/ 0xA9, 0xD4, 0x2C, 0x2A, 0xA9, 0xD5, 0x08, 0x2B, 0xF9,
0xD6, 0x0E, 0x2B, 0x89, 0xD5, 0x88, 0x2B,
/*00D230:*/ 0xA9, 0xD4, 0x9C, 0x2B, 0xD1, 0xD5, 0xAA, 0x2B, 0x49,
0xD7, 0xDC, 0x2B, 0x41, 0xD4, 0x18, 0x30,
/*00D240:*/ 0xE1, 0xD4, 0x1E, 0x30, 0xA1, 0xD6, 0x1C, 0x31, 0xE1,
0xD6, 0x1E, 0x31, 0xE9, 0xD6, 0x5E, 0x31,
/*00D250:*/ 0x91, 0xD7, 0x68, 0x31, 0xE1, 0xD7, 0xCA, 0x31, 0xA1,
0xD4, 0xCC, 0x31, 0x59, 0xD7, 0xEC, 0x31,
/*00D260:*/ 0x09, 0xD7, 0xEE, 0x31, 0x61, 0xD6, 0xF8, 0x31, 0xC1,
0xD5, 0xFA, 0x31, 0xE1, 0xD6, 0x1A, 0x32,
/*00D270:*/ 0x69, 0xD4, 0x6C, 0x32, 0xC9, 0xD6, 0xDA, 0x32, 0x59,
0xD4, 0xEC, 0x32, 0x49, 0xD7, 0x58, 0x33,
/*00D280:*/ 0xA9, 0xD5, 0xFC, 0x34, 0x29, 0xD4, 0x3A, 0x36, 0x69,
0xD6, 0x2E, 0x38, 0x71, 0xD5, 0x4A, 0x38,
/*00D290:*/ 0x39, 0xD5, 0x68, 0x38, 0x99, 0xD5, 0x6E, 0x38, 0xE9,
0xD7, 0x7C, 0x38, 0x09, 0xD6, 0xAC, 0x38,
/*00D2A0:*/ 0x49, 0xD6, 0xAE, 0x38, 0x89, 0xD7, 0xBE, 0x38, 0x81,
0xD4, 0xDA, 0x38, 0x81, 0xD4, 0xFA, 0x38,
/*00D2B0:*/ 0xF1, 0xD6, 0x18, 0x39, 0xF9, 0xD6, 0x38, 0x39, 0xE1,
0xD5, 0x3C, 0x39, 0x41, 0xD5, 0x5A, 0x39,
/*00D2C0:*/ 0x69, 0xD4, 0x6E, 0x39, 0xE1, 0xD5, 0x7C, 0x39, 0xB9,
0xD4, 0x8A, 0x39, 0xD1, 0xD6, 0x98, 0x39,
/*00D2D0:*/ 0x39, 0xD5, 0xB8, 0x39, 0x61, 0xD5, 0xDA, 0x39, 0x59,
0xD5, 0x1E, 0x3A, 0xC9, 0xD7, 0x28, 0x3A,
/*00D2E0:*/ 0x99, 0xD7, 0x2A, 0x3A, 0x29, 0xD7, 0x2C, 0x3A, 0x89,
0xD4, 0x2E, 0x3A, 0x61, 0xD7, 0x2E, 0x3A,
/*00D2F0:*/ 0x29, 0xD4, 0x48, 0x3A, 0x89, 0xD7, 0x4A, 0x3A, 0x31,
0xD7, 0x6C, 0x3A, 0xB9, 0xD4, 0x8E, 0x3A,
/*00D300:*/ 0x99, 0xD5, 0xDA, 0x3A, 0x79, 0xD6, 0xDA, 0x3A, 0xD9,
0xD6, 0xFC, 0x3A, 0x91, 0xD6, 0xFE, 0x3A,
/*00D310:*/ 0x79, 0xD5, 0x0E, 0x3B, 0x21, 0xD6, 0x48, 0x3B, 0xA9,
0xD4, 0x7E, 0x3B, 0x81, 0xD7, 0x9A, 0x3B,
/*00D320:*/ 0xD1, 0xD4, 0x9C, 0x3B, 0x49, 0xD5, 0xAE, 0x3B, 0x41,
0xD6, 0xCA, 0x3B, 0xA1, 0xD6, 0xCE, 0x3B,
/*00D330:*/ 0x81, 0xD4, 0xDE, 0x3B, 0xF9, 0xD5, 0xE8, 0x3B, 0x41,
0xD5, 0xEE, 0x3B, 0x89, 0xD5, 0x3C, 0x60,
/*00D340:*/ 0x69, 0xD5, 0xA8, 0x61, 0xA9, 0xD5, 0x28, 0x62, 0x59,
0xD6, 0x28, 0x62, 0x69, 0xD7, 0x3C, 0x62,
/*00D350:*/ 0x11, 0xD6, 0xBA, 0x63, 0x81, 0xD7, 0xCA, 0x68, 0x01,
0xD5, 0x2C, 0x69, 0xA9, 0xD6, 0x2E, 0x69,
/*00D360:*/ 0x89, 0xD5, 0xAA, 0x69, 0x91, 0xD6, 0xCE, 0x69, 0xA9,
0xD5, 0x0E, 0x6A, 0x89, 0xD7, 0x1E, 0x6A,
/*00D370:*/ 0xE9, 0xD5, 0x2C, 0x6A, 0x89, 0xD6, 0x8A, 0x6A, 0xE9,
0xD7, 0xDC, 0x6A, 0x69, 0xD6, 0x5E, 0x6B,
/*00D380:*/ 0x09, 0xD6, 0x9C, 0x6B, 0x99, 0xD4, 0x6A, 0x70, 0x31,
0xD5, 0x68, 0x71, 0x11, 0xD7, 0x78, 0x71,
/*00D390:*/ 0x91, 0xD4, 0xBA, 0x71, 0x61, 0xD4, 0xBE, 0x71, 0x09,
0xD6, 0xEC, 0x71, 0x91, 0xD5, 0x6A, 0x72,
/*00D3A0:*/ 0xD9, 0xD6, 0x6C, 0x72, 0xA1, 0xD7, 0x7A, 0x72, 0xC1,
0xD4, 0x9C, 0x72, 0x81, 0xD4, 0xBE, 0x72,
/*00D3B0:*/ 0x29, 0xD7, 0xDC, 0x72, 0xC9, 0xD7, 0x48, 0x73, 0x61,
0xD7, 0x4E, 0x73, 0xE9, 0xD4, 0xAC, 0x73,
/*00D3C0:*/ 0x29, 0xD7, 0x5C, 0x74, 0xE9, 0xD5, 0x9C, 0x76, 0x51,
0xD5, 0x1C, 0x78, 0x39, 0xD7, 0x2E, 0x78,
/*00D3D0:*/ 0xE9, 0xD6, 0x3E, 0x78, 0x21, 0xD4, 0x4A, 0x78, 0x39,
0xD4, 0x4A, 0x78, 0xE9, 0xD5, 0x5A, 0x78,
/*00D3E0:*/ 0x71, 0xD4, 0x68, 0x78, 0x59, 0xD6, 0x78, 0x78, 0x11,
0xD5, 0x7E, 0x78, 0xF9, 0xD4, 0x8E, 0x78,
/*00D3F0:*/ 0xA1, 0xD4, 0xAC, 0x78, 0x01, 0xD4, 0xCA, 0x78, 0xF1,
0xD7, 0xCA, 0x78, 0xC9, 0xD6, 0xFE, 0x78,
/*00D400:*/ 0xE1, 0xD7, 0x1A, 0x79, 0x29, 0xD6, 0x2A, 0x79, 0x81,
0xD6, 0x2C, 0x79, 0xC9, 0xD6, 0x2E, 0x79,
/*00D410:*/ 0x59, 0xD4, 0x38, 0x79, 0x59, 0xD7, 0x5C, 0x79, 0x71,
0xD6, 0x68, 0x79, 0xF9, 0xD4, 0x7E, 0x79,
/*00D420:*/ 0x19, 0xD5, 0x8E, 0x79, 0x21, 0xD7, 0x9E, 0x79, 0x49,
0xD6, 0xA8, 0x79, 0x19, 0xD6, 0xCA, 0x79,
/*00D430:*/ 0x89, 0xD4, 0xDC, 0x79, 0x01, 0xD5, 0xEE, 0x79, 0x29,
0xD7, 0xFE, 0x79, 0x99, 0xD6, 0x08, 0x7A,
/*00D440:*/ 0xA1, 0xD4, 0x58, 0x7A, 0x71, 0xD6, 0x6C, 0x7A, 0xB9,
0xD4, 0x78, 0x7A, 0xE9, 0xD4, 0x7A, 0x7A,
/*00D450:*/ 0x21, 0xD7, 0x9A, 0x7A, 0x71, 0xD4, 0x9C, 0x7A, 0x21,
0xD4, 0x9E, 0x7A, 0x09, 0xD6, 0xAE, 0x7A,
/*00D460:*/ 0x21, 0xD4, 0xBE, 0x7A, 0x51, 0xD5, 0xE8, 0x7A, 0xE1,
0xD5, 0xEE, 0x7A, 0x79, 0xD7, 0x08, 0x7B,
/*00D470:*/ 0x81, 0xD4, 0x28, 0x7B, 0x61, 0xD7, 0x48, 0x7B, 0x29,
0xD4, 0x4E, 0x7B, 0xB9, 0xD6, 0x58, 0x7B,
/*00D480:*/ 0xD1, 0xD4, 0x6A, 0x7B, 0x21, 0xD7, 0x6A, 0x7B, 0x31,
0xD7, 0x6A, 0x7B, 0x89, 0xD7, 0x6C, 0x7B,
/*00D490:*/ 0x59, 0xD6, 0x7C, 0x7B, 0xE9, 0xD5, 0x7E, 0x7B, 0x09,
0xD6, 0x7E, 0x7B, 0x79, 0xD6, 0x9C, 0x7B,
/*00D4A0:*/ 0x99, 0xD6, 0xB8, 0x7B, 0x21, 0xD5, 0xBA, 0x7B, 0x29,
0xD6, 0xBE, 0x7B, 0xE1, 0xD4, 0xCA, 0x7B,
/*00D4B0:*/ 0x81, 0xD6, 0xD8, 0x7B, 0x99, 0xD6, 0xD8, 0x7B, 0xC1,
0xD5, 0xDE, 0x7B, 0x01, 0xD7, 0xEA, 0x7B,
/*00D4C0:*/ 0x11, 0xD7, 0xEA, 0x7B, 0x99, 0xD5, 0xFC, 0x7B, 0x79,
0xD6, 0xFC, 0x7B, 0x0D, 0xF0, 0xAD, 0xBA,
/*00D4D0:*/ 0x0D, 0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*00D4E0:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B3D2_2[] = {
/*00DBD8:*/ 0xFB, 0x1C, 0x28, 0x00, 0xA9, 0x00, 0xB8, 0x00,
/*00DBE0:*/ 0xAB, 0x0C, 0xB8, 0x00, 0xB8, 0x08, 0x38, 0x01, 0xB8,
0x04, 0xB8, 0x01, 0xCB, 0x14, 0xB8, 0x01,
/*00DBF0:*/ 0xBA, 0x15, 0x08, 0x02, 0xD9, 0x19, 0x18, 0x02, 0xD8,
0x18, 0x28, 0x02, 0xEA, 0x0D, 0x88, 0x02,
/*00DC00:*/ 0xE9, 0x15, 0xA8, 0x02, 0xEB, 0x19, 0xA8, 0x02, 0xE9,
0x1D, 0xB8, 0x02, 0xAB, 0x19, 0x18, 0x03,
/*00DC10:*/ 0xDB, 0x04, 0x28, 0x03, 0xF8, 0x15, 0x38, 0x03, 0xB9,
0x0D, 0x98, 0x03, 0xB8, 0x0C, 0xA8, 0x03,
/*00DC20:*/ 0xC9, 0x18, 0x08, 0x04, 0xB8, 0x0C, 0x18, 0x04, 0xCA,
0x15, 0x38, 0x04, 0xA9, 0x15, 0xA8, 0x04,
/*00DC30:*/ 0xF9, 0x09, 0xB8, 0x04, 0xDA, 0x0D, 0x08, 0x05, 0xFB,
0x01, 0x98, 0x05, 0xC9, 0x18, 0x08, 0x06,
/*00DC40:*/ 0xA8, 0x05, 0x18, 0x06, 0xD8, 0x05, 0x28, 0x06, 0xD8,
0x09, 0xA8, 0x06, 0xC9, 0x09, 0x38, 0x07,
/*00DC50:*/ 0xB8, 0x15, 0x38, 0x07, 0xC8, 0x19, 0x88, 0x07, 0xD9,
0x11, 0xB8, 0x07, 0xFA, 0x1D, 0x18, 0x08,
/*00DC60:*/ 0xEB, 0x0C, 0x08, 0x09, 0xB9, 0x1C, 0x18, 0x09, 0xAB,
0x08, 0x98, 0x09, 0xBB, 0x1C, 0x98, 0x09,
/*00DC70:*/ 0xFB, 0x09, 0x38, 0x0A, 0xEB, 0x1D, 0x38, 0x0A, 0xF8,
0x1D, 0x98, 0x0A, 0xF8, 0x00, 0x28, 0x0B,
/*00DC80:*/ 0xFA, 0x0C, 0x28, 0x0B, 0xFA, 0x00, 0x18, 0x0C, 0x89,
0x14, 0x38, 0x0D, 0xF8, 0x19, 0x08, 0x0E,
/*00DC90:*/ 0xB9, 0x10, 0x28, 0x0E, 0xFA, 0x04, 0x38, 0x0F, 0xFA,
0x1D, 0xA8, 0x0F, 0xFA, 0x10, 0x18, 0x10,
/*00DCA0:*/ 0xBB, 0x11, 0x28, 0x10, 0xF9, 0x1D, 0x28, 0x10, 0xBA,
0x14, 0x88, 0x10, 0xB9, 0x0C, 0x18, 0x11,
/*00DCB0:*/ 0xDB, 0x0D, 0x08, 0x12, 0xFA, 0x18, 0x08, 0x12, 0xD9,
0x09, 0x18, 0x12, 0xBA, 0x0D, 0x18, 0x12,
/*00DCC0:*/ 0xD9, 0x0D, 0x88, 0x12, 0xB9, 0x0C, 0xA8, 0x12, 0xA9,
0x10, 0x08, 0x13, 0xB8, 0x10, 0x28, 0x13,
/*00DCD0:*/ 0xAA, 0x15, 0x28, 0x13, 0xBA, 0x14, 0x38, 0x13, 0xA8,
0x15, 0xA8, 0x13, 0xDB, 0x10, 0xB8, 0x13,
/*00DCE0:*/ 0xB8, 0x1C, 0x18, 0x14, 0xBB, 0x19, 0x38, 0x14, 0xD8,
0x1D, 0x38, 0x14, 0xFA, 0x09, 0x88, 0x14,
/*00DCF0:*/ 0xBB, 0x1D, 0x18, 0x15, 0xAB, 0x10, 0x88, 0x15, 0xBB,
0x0C, 0x98, 0x15, 0xBB, 0x11, 0x28, 0x16,
/*00DD00:*/ 0xAB, 0x09, 0xA8, 0x16, 0xDA, 0x00, 0x08, 0x17, 0xD8,
0x11, 0x08, 0x17, 0xDA, 0x1D, 0x08, 0x17,
/*00DD10:*/ 0xDB, 0x10, 0xB8, 0x17, 0xEB, 0x01, 0xB8, 0x18, 0xB9,
0x19, 0xB8, 0x18, 0xA9, 0x10, 0x08, 0x19,
/*00DD20:*/ 0xFB, 0x00, 0xA8, 0x1A, 0xF9, 0x04, 0x08, 0x1B, 0xF8,
0x18, 0x38, 0x1B, 0xBB, 0x0C, 0x98, 0x1B,
/*00DD30:*/ 0xF8, 0x09, 0xB8, 0x1B, 0xF8, 0x0D, 0x28, 0x1D, 0xF9,
0x08, 0x88, 0x1D, 0x8B, 0x04, 0xB8, 0x1D,
/*00DD40:*/ 0xF8, 0x18, 0x88, 0x1E, 0x89, 0x19, 0x88, 0x1E, 0xDB,
0x01, 0x38, 0x1F, 0xAA, 0x04, 0xA8, 0x80,
/*00DD50:*/ 0xBA, 0x10, 0xA8, 0x80, 0xFA, 0x18, 0xB8, 0x80, 0xBA,
0x10, 0x18, 0x81, 0xB9, 0x08, 0x38, 0x81,
/*00DD60:*/ 0xFB, 0x00, 0xA8, 0x81, 0x8A, 0x01, 0xA8, 0x81, 0xEA,
0x15, 0x28, 0x82, 0xDB, 0x05, 0xA8, 0x82,
/*00DD70:*/ 0xF9, 0x00, 0x28, 0x83, 0xB9, 0x04, 0x08, 0x84, 0xFB,
0x1D, 0x18, 0x84, 0xA8, 0x04, 0x28, 0x84,
/*00DD80:*/ 0xD9, 0x1C, 0x08, 0x85, 0xC9, 0x00, 0x18, 0x85, 0xFA,
0x0D, 0x18, 0x85, 0xA8, 0x0C, 0x88, 0x85,
/*00DD90:*/ 0xDA, 0x15, 0xA8, 0x85, 0xDB, 0x14, 0x28, 0x86, 0xAB,
0x10, 0x88, 0x86, 0xA8, 0x08, 0xA8, 0x86,
/*00DDA0:*/ 0xD9, 0x01, 0xB8, 0x86, 0xB8, 0x14, 0x08, 0x87, 0xF9,
0x19, 0x08, 0x88, 0xBA, 0x09, 0x38, 0x88,
/*00DDB0:*/ 0xFB, 0x0C, 0x98, 0x88, 0xB8, 0x05, 0x88, 0x89, 0xE9,
0x18, 0xA8, 0x89, 0xFA, 0x1C, 0x28, 0x8A,
/*00DDC0:*/ 0xF9, 0x00, 0x28, 0x8B, 0xBB, 0x08, 0xB8, 0x8B, 0x88,
0x10, 0xA8, 0x8D, 0xF8, 0x1C, 0xA8, 0x8E,
/*00DDD0:*/ 0xF9, 0x08, 0x38, 0x8F, 0xFB, 0x19, 0x38, 0x8F, 0xB9,
0x01, 0x18, 0x90, 0xFB, 0x10, 0x18, 0x90,
/*00DDE0:*/ 0xBB, 0x14, 0x88, 0x90, 0xD8, 0x1C, 0x08, 0x92, 0xE8,
0x14, 0x28, 0x92, 0xAB, 0x1D, 0x38, 0x93,
/*00DDF0:*/ 0xD9, 0x00, 0x88, 0x93, 0xDA, 0x0D, 0x08, 0x94, 0xFA,
0x1D, 0xA8, 0x94, 0xA8, 0x0D, 0xB8, 0x94,
/*00DE00:*/ 0xFB, 0x09, 0x38, 0x95, 0xDB, 0x0C, 0x88, 0x95, 0xF8,
0x1D, 0x98, 0x95, 0xDA, 0x18, 0xA8, 0x95,
/*00DE10:*/ 0xFB, 0x18, 0xB8, 0x95, 0xAB, 0x11, 0x08, 0x96, 0xBB,
0x10, 0x18, 0x96, 0xA9, 0x11, 0x88, 0x96,
/*00DE20:*/ 0x8B, 0x05, 0x88, 0x97, 0xDA, 0x0D, 0xB8, 0x97, 0xBB,
0x09, 0x88, 0x98, 0xB8, 0x19, 0xB8, 0x98,
/*00DE30:*/ 0xEA, 0x18, 0x98, 0x99, 0xBB, 0x10, 0xA8, 0x99, 0xF8,
0x1D, 0x28, 0x9A, 0xF8, 0x19, 0xB8, 0x9A,
/*00DE40:*/ 0xFA, 0x08, 0xB8, 0x9C, 0xFA, 0x11, 0x98, 0x9D, 0xFB,
0x05, 0x08, 0x9E, 0xFB, 0x18, 0x08, 0x9E,
/*00DE50:*/ 0xF8, 0x15, 0x38, 0x9E, 0xF8, 0x1D, 0x98, 0x9F, 0xF9,
0x1C, 0xA8, 0x9F, 0xF9, 0x14, 0xB8, 0x9F,
/*00DE60:*/ 0xEB, 0x08, 0xA8, 0xA6, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*00DE70:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B3D2_3[] = {
/*00E498:*/ 0x5D, 0x86, 0x3F, 0x00, 0x59, 0x97, 0x79, 0x00,
/*00E4A0:*/ 0x5D, 0x93, 0x7A, 0x00, 0x59, 0x97, 0x5D, 0x01, 0x5D,
0x84, 0xDA, 0x03, 0x59, 0x91, 0xDD, 0x03,
/*00E4B0:*/ 0x59, 0x87, 0x9E, 0x04, 0x59, 0x80, 0xF8, 0x04, 0x59,
0x93, 0x39, 0x05, 0x5D, 0x93, 0x3B, 0x05,
/*00E4C0:*/ 0x59, 0x94, 0x3F, 0x05, 0x59, 0x97, 0x3C, 0x06, 0x59,
0x87, 0xBA, 0x07, 0x5D, 0x80, 0xDE, 0x07,
/*00E4D0:*/ 0x5D, 0x80, 0x9F, 0x08, 0x5D, 0x8C, 0xB8, 0x08, 0x59,
0x8F, 0xFD, 0x08, 0x59, 0x98, 0x3D, 0x09,
/*00E4E0:*/ 0x5D, 0x82, 0x7A, 0x09, 0x59, 0x81, 0x7F, 0x09, 0x59,
0x88, 0xBF, 0x09, 0x5D, 0x85, 0x38, 0x0A,
/*00E4F0:*/ 0x5D, 0x87, 0xB9, 0x0A, 0x59, 0x92, 0xDE, 0x0A, 0x59,
0x8D, 0x38, 0x0B, 0x5D, 0x8E, 0x3D, 0x0B,
/*00E500:*/ 0x59, 0x9F, 0x5B, 0x0B, 0x59, 0x80, 0xB9, 0x0B, 0x59,
0x80, 0xF9, 0x0B, 0x5D, 0x81, 0x1C, 0x0C,
/*00E510:*/ 0x5D, 0x8E, 0x3C, 0x0C, 0x59, 0x85, 0x7F, 0x0C, 0x59,
0x97, 0x78, 0x0D, 0x59, 0x8E, 0x7A, 0x0D,
/*00E520:*/ 0x59, 0x85, 0x7B, 0x0D, 0x5D, 0x80, 0x9E, 0x0D, 0x5D,
0x8B, 0xBF, 0x0D, 0x5D, 0x91, 0xFA, 0x0D,
/*00E530:*/ 0x5D, 0x95, 0xFB, 0x0D, 0x59, 0x90, 0x5A, 0x0E, 0x5D,
0x87, 0x9C, 0x0E, 0x5D, 0x91, 0x7A, 0x20,
/*00E540:*/ 0x59, 0x90, 0xFE, 0x20, 0x59, 0x85, 0xFB, 0x22, 0x59,
0x85, 0xFF, 0x23, 0x59, 0x91, 0x3D, 0x24,
/*00E550:*/ 0x59, 0x82, 0xFC, 0x25, 0x5D, 0x85, 0xBC, 0x26, 0x59,
0x81, 0xFB, 0x27, 0x5D, 0x99, 0x3C, 0x28,
/*00E560:*/ 0x5D, 0x9F, 0xFC, 0x28, 0x59, 0x83, 0x1F, 0x29, 0x59,
0x99, 0x3A, 0x29, 0x5D, 0x85, 0xDD, 0x29,
/*00E570:*/ 0x5D, 0x80, 0x3E, 0x2A, 0x5D, 0x8B, 0x3B, 0x2B, 0x59,
0x9B, 0xDB, 0x2B, 0x5D, 0x87, 0x1D, 0x2C,
/*00E580:*/ 0x5D, 0x88, 0x3D, 0x2C, 0x59, 0x86, 0x99, 0x2C, 0x5D,
0x85, 0x9C, 0x2C, 0x5D, 0x86, 0xDB, 0x2C,
/*00E590:*/ 0x5D, 0x84, 0x5E, 0x2D, 0x5D, 0x8F, 0x7F, 0x2D, 0x5D,
0x82, 0x9E, 0x2D, 0x59, 0x83, 0x1E, 0x2E,
/*00E5A0:*/ 0x59, 0x87, 0x7F, 0x2E, 0x59, 0x81, 0xDF, 0x2E, 0x59,
0x82, 0xF8, 0x2E, 0x5D, 0x80, 0x3F, 0x2F,
/*00E5B0:*/ 0x59, 0x86, 0x9D, 0x2F, 0x5D, 0x98, 0xBB, 0x2F, 0x59,
0x87, 0x78, 0x40, 0x5D, 0x97, 0xFC, 0x41,
/*00E5C0:*/ 0x5D, 0x83, 0x3B, 0x42, 0x59, 0x84, 0x3F, 0x42, 0x5D,
0x83, 0x7F, 0x43, 0x5D, 0x86, 0xF8, 0x43,
/*00E5D0:*/ 0x5D, 0x9A, 0x3C, 0x44, 0x59, 0x80, 0x7B, 0x44, 0x5D,
0x86, 0x99, 0x44, 0x5D, 0x85, 0x9E, 0x44,
/*00E5E0:*/ 0x5D, 0x93, 0xBC, 0x44, 0x59, 0x93, 0xBE, 0x44, 0x59,
0x82, 0xFE, 0x45, 0x5D, 0x90, 0xFB, 0x46,
/*00E5F0:*/ 0x59, 0x9E, 0x3A, 0x48, 0x5D, 0x8F, 0x3C, 0x48, 0x59,
0x8B, 0x3F, 0x48, 0x59, 0x8D, 0xBF, 0x48,
/*00E600:*/ 0x5D, 0x85, 0xDB, 0x48, 0x5D, 0x86, 0xFC, 0x48, 0x59,
0x8E, 0xFC, 0x49, 0x5D, 0x83, 0x1B, 0x4A,
/*00E610:*/ 0x5D, 0x92, 0x1F, 0x4A, 0x5D, 0x87, 0x3A, 0x4A, 0x5D,
0x91, 0x58, 0x4A, 0x59, 0x86, 0xBE, 0x4A,
/*00E620:*/ 0x59, 0x89, 0xBE, 0x4A, 0x5D, 0x83, 0x3F, 0x4B, 0x59,
0x86, 0x9A, 0x4B, 0x59, 0x9F, 0xB8, 0x4B,
/*00E630:*/ 0x5D, 0x9C, 0xBD, 0x4B, 0x5D, 0x84, 0x18, 0x4C, 0x5D,
0x83, 0x1A, 0x4D, 0x59, 0x88, 0x39, 0x4D,
/*00E640:*/ 0x5D, 0x80, 0x7D, 0x4D, 0x5D, 0x9B, 0xBE, 0x4D, 0x59,
0x87, 0x5D, 0x4E, 0x5D, 0x86, 0xD9, 0x4E,
/*00E650:*/ 0x59, 0x85, 0x98, 0x4F, 0x59, 0x86, 0x3F, 0x60, 0x5D,
0x9B, 0x5E, 0x60, 0x59, 0x91, 0xDB, 0x60,
/*00E660:*/ 0x59, 0x83, 0xF8, 0x60, 0x59, 0x87, 0xF9, 0x60, 0x59,
0x92, 0xFC, 0x60, 0x5D, 0x80, 0xB9, 0x61,
/*00E670:*/ 0x59, 0x83, 0xFC, 0x61, 0x59, 0x81, 0x59, 0x62, 0x59,
0x95, 0xDA, 0x62, 0x5D, 0x85, 0x7E, 0x63,
/*00E680:*/ 0x59, 0x8C, 0x9C, 0x63, 0x59, 0x87, 0xFD, 0x63, 0x5D,
0x84, 0xB9, 0x64, 0x59, 0x95, 0xBF, 0x64,
/*00E690:*/ 0x59, 0x81, 0x18, 0x65, 0x59, 0x9F, 0x1C, 0x65, 0x59,
0x87, 0xF8, 0x65, 0x5D, 0x90, 0x3A, 0x66,
/*00E6A0:*/ 0x59, 0x80, 0xBA, 0x66, 0x59, 0x80, 0xDA, 0x66, 0x5D,
0x95, 0xFD, 0x66, 0x5D, 0x91, 0xF8, 0x67,
/*00E6B0:*/ 0x59, 0x85, 0x18, 0x68, 0x5D, 0x82, 0x7C, 0x68, 0x59,
0x83, 0x98, 0x68, 0x59, 0x85, 0x38, 0x6A,
/*00E6C0:*/ 0x5D, 0x85, 0x3A, 0x6A, 0x5D, 0x8E, 0x3B, 0x6A, 0x59,
0x82, 0x3A, 0x6B, 0x59, 0x8E, 0x3D, 0x6B,
/*00E6D0:*/ 0x59, 0x87, 0xFD, 0x6B, 0x59, 0x8E, 0x38, 0x6D, 0x59,
0x81, 0x78, 0x6D, 0x59, 0x96, 0xFC, 0x6D,
/*00E6E0:*/ 0x59, 0x89, 0x3A, 0x6E, 0x59, 0x81, 0x5C, 0x6E, 0x59,
0x83, 0x9D, 0x6E, 0x5D, 0x87, 0xFE, 0x6E,
/*00E6F0:*/ 0x5D, 0x9C, 0x39, 0x6F, 0x59, 0x82, 0x7F, 0x6F, 0x59,
0x84, 0xDF, 0x6F, 0x59, 0x97, 0xBC, 0x88,
/*00E700:*/ 0x59, 0x92, 0xBB, 0xAD, 0x59, 0x96, 0x3D, 0xEC, 0x5D,
0x91, 0x3D, 0xED, 0x0D, 0xF0, 0xAD, 0xBA,
/*00E710:*/ 0x0D, 0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*00E720:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B3D2_4[] = {
/*00ED18:*/ 0x96, 0x3C, 0x82, 0xF9, 0x92, 0x5D, 0x82, 0xF9,
/*00ED20:*/ 0x96, 0x3C, 0x89, 0xF9, 0x82, 0x31, 0x8E, 0xF9, 0x8E,
0xF0, 0x94, 0xF9, 0x96, 0x59, 0x96, 0xF9,
/*00ED30:*/ 0x9A, 0xF4, 0x98, 0xF9, 0x82, 0xF0, 0x9C, 0xF9, 0x92,
0x31, 0x9D, 0xF9, 0x9E, 0x38, 0xA2, 0xF9,
/*00ED40:*/ 0x8E, 0x5C, 0xA8, 0xF9, 0x8E, 0xD5, 0xA8, 0xF9, 0x82,
0x31, 0xB2, 0xF9, 0x9A, 0x75, 0xB6, 0xF9,
/*00ED50:*/ 0x8E, 0x55, 0xBC, 0xF9, 0x8E, 0xF9, 0xBC, 0xF9, 0x96,
0xB5, 0xBE, 0xF9, 0x96, 0x3C, 0xC2, 0xF9,
/*00ED60:*/ 0x86, 0x7D, 0xC3, 0xF9, 0x96, 0xFD, 0xC4, 0xF9, 0x96,
0x7C, 0xC9, 0xF9, 0x92, 0xF9, 0xD0, 0xF9,
/*00ED70:*/ 0x96, 0x35, 0xD6, 0xF9, 0x9A, 0x3C, 0xDE, 0xF9, 0x9E,
0xF8, 0xE2, 0xF9, 0x9A, 0xFC, 0xE2, 0xF9,
/*00ED80:*/ 0x96, 0xFC, 0xEA, 0xF9, 0x8E, 0xF1, 0xEE, 0xF9, 0x82,
0x31, 0xF2, 0xF9, 0x86, 0x3D, 0xF4, 0xF9,
/*00ED90:*/ 0x86, 0xF4, 0xF4, 0xF9, 0x86, 0x75, 0xF9, 0xF9, 0x8E,
0xF0, 0xFC, 0xF9, 0x82, 0x70, 0xFF, 0xF9,
/*00EDA0:*/ 0x82, 0x34, 0x82, 0xFB, 0x92, 0x35, 0x88, 0xFB, 0x8A,
0x70, 0x8A, 0xFB, 0x82, 0x59, 0x90, 0xFB,
/*00EDB0:*/ 0x9E, 0xFD, 0x92, 0xFB, 0x8A, 0x54, 0x98, 0xFB, 0x96,
0xB8, 0x9C, 0xFB, 0x8A, 0xF9, 0x9E, 0xFB,
/*00EDC0:*/ 0x86, 0x39, 0xA1, 0xFB, 0x82, 0x3D, 0xA1, 0xFB, 0x8E,
0xD9, 0xA4, 0xFB, 0x8A, 0xF8, 0xA4, 0xFB,
/*00EDD0:*/ 0x82, 0x35, 0xA7, 0xFB, 0x86, 0xB0, 0xAA, 0xFB, 0x82,
0xB4, 0xAA, 0xFB, 0x86, 0x5D, 0xAC, 0xFB,
/*00EDE0:*/ 0x92, 0xB4, 0xAD, 0xFB, 0x92, 0xB4, 0xB2, 0xFB, 0x92,
0xF4, 0xB2, 0xFB, 0x8E, 0xFD, 0xB6, 0xFB,
/*00EDF0:*/ 0x86, 0x38, 0xB8, 0xFB, 0x96, 0x30, 0xB9, 0xFB, 0x92,
0x34, 0xB9, 0xFB, 0x9A, 0x55, 0xBA, 0xFB,
/*00EE00:*/ 0x9A, 0xF0, 0xBA, 0xFB, 0x9E, 0x59, 0xBC, 0xFB, 0x9A,
0xD4, 0xBC, 0xFB, 0x82, 0xF4, 0xBE, 0xFB,
/*00EE10:*/ 0x96, 0x31, 0xBF, 0xFB, 0x9E, 0x35, 0xC0, 0xFB, 0x86,
0x39, 0xC2, 0xFB, 0x82, 0x3D, 0xC2, 0xFB,
/*00EE20:*/ 0x86, 0x71, 0xC4, 0xFB, 0x86, 0xDD, 0xC4, 0xFB, 0x8A,
0x38, 0xC7, 0xFB, 0x9E, 0x35, 0xCB, 0xFB,
/*00EE30:*/ 0x8E, 0xF5, 0xCC, 0xFB, 0x8E, 0x3C, 0xD3, 0xFB, 0x8E,
0x7C, 0xD8, 0xFB, 0x86, 0x70, 0xDD, 0xFB,
/*00EE40:*/ 0x82, 0x7D, 0xE1, 0xFB, 0x8E, 0xBD, 0xE2, 0xFB, 0x82,
0x34, 0xEA, 0xFB, 0x82, 0xFD, 0xEA, 0xFB,
/*00EE50:*/ 0x8E, 0x35, 0xEF, 0xFB, 0x8A, 0x38, 0xF0, 0xFB, 0x8E,
0x50, 0xF0, 0xFB, 0x96, 0x39, 0xF2, 0xFB,
/*00EE60:*/ 0x8E, 0x51, 0xF6, 0xFB, 0x8E, 0x7D, 0xF6, 0xFB, 0x8E,
0x35, 0xFB, 0xFB, 0x8A, 0x34, 0x80, 0xFD,
/*00EE70:*/ 0x96, 0x35, 0x82, 0xFD, 0x96, 0x3D, 0x84, 0xFD, 0x8A,
0xF5, 0x86, 0xFD, 0x9E, 0x71, 0x8A, 0xFD,
/*00EE80:*/ 0x86, 0xFC, 0x8E, 0xFD, 0x92, 0x31, 0x96, 0xFD, 0x96,
0x59, 0x96, 0xFD, 0x86, 0x51, 0x97, 0xFD,
/*00EE90:*/ 0x86, 0x91, 0x97, 0xFD, 0x9A, 0x35, 0xA2, 0xFD, 0x9A,
0x7D, 0xA4, 0xFD, 0x9E, 0xF9, 0xA4, 0xFD,
/*00EEA0:*/ 0x8A, 0x34, 0xA8, 0xFD, 0x8A, 0xF4, 0xA8, 0xFD, 0x8E,
0x78, 0xAE, 0xFD, 0x82, 0xF9, 0xB4, 0xFD,
/*00EEB0:*/ 0x92, 0xD4, 0xBE, 0xFD, 0x8E, 0xF8, 0xD2, 0xFD, 0x9E,
0x70, 0xD8, 0xFD, 0x9A, 0xB4, 0xD8, 0xFD,
/*00EEC0:*/ 0x8A, 0xFD, 0xE3, 0xFD, 0x8A, 0xF5, 0xE5, 0xFD, 0x86,
0x7C, 0xE6, 0xFD, 0x86, 0xF5, 0xE6, 0xFD,
/*00EED0:*/ 0x8E, 0x70, 0xE8, 0xFD, 0x8E, 0xB0, 0xE8, 0xFD, 0x82,
0x39, 0xEB, 0xFD, 0x86, 0xB4, 0xEB, 0xFD,
/*00EEE0:*/ 0x96, 0x74, 0xEC, 0xFD, 0x9E, 0xB0, 0xF0, 0xFD, 0x9A,
0xB4, 0xF0, 0xFD, 0x82, 0x30, 0xF4, 0xFD,
/*00EEF0:*/ 0x82, 0x79, 0xF4, 0xFD, 0x82, 0x31, 0xF9, 0xFD, 0x8A,
0x3D, 0xFC, 0xFD, 0x96, 0x38, 0x83, 0xFF,
/*00EF00:*/ 0x96, 0x71, 0x88, 0xFF, 0x96, 0xF1, 0x88, 0xFF, 0x8E,
0xFD, 0x8A, 0xFF, 0x8A, 0x1D, 0x93, 0xFF,
/*00EF10:*/ 0x9E, 0x3C, 0x94, 0xFF, 0x82, 0x34, 0x96, 0xFF, 0x82,
0x99, 0x9B, 0xFF, 0x86, 0x9C, 0x9D, 0xFF,
/*00EF20:*/ 0x8A, 0x55, 0x9E, 0xFF, 0x8A, 0xF9, 0x9E, 0xFF, 0x9E,
0x75, 0x9F, 0xFF, 0x8A, 0x55, 0xA2, 0xFF,
/*00EF30:*/ 0x9E, 0x7D, 0xA5, 0xFF, 0x86, 0xF8, 0xAC, 0xFF, 0x8E,
0xFC, 0xB0, 0xFF, 0x8E, 0xF4, 0xB6, 0xFF,
/*00EF40:*/ 0x9A, 0x71, 0xB7, 0xFF, 0x86, 0xF1, 0xB8, 0xFF, 0x9A,
0x39, 0xBA, 0xFF, 0x9E, 0x7D, 0xBA, 0xFF,
/*00EF50:*/ 0x8A, 0x31, 0xBB, 0xFF, 0x8E, 0x3C, 0xBB, 0xFF, 0x8E,
0x50, 0xBB, 0xFF, 0x9A, 0x71, 0xBC, 0xFF,
/*00EF60:*/ 0x86, 0xF0, 0xBE, 0xFF, 0x92, 0x59, 0xBF, 0xFF, 0x86,
0x30, 0xC2, 0xFF, 0x82, 0x34, 0xC2, 0xFF,
/*00EF70:*/ 0x82, 0x7C, 0xC4, 0xFF, 0x8E, 0xFC, 0xC7, 0xFF, 0x8A,
0x30, 0xCA, 0xFF, 0x86, 0x38, 0xCF, 0xFF,
/*00EF80:*/ 0x86, 0x5D, 0xCF, 0xFF, 0x86, 0x31, 0xD0, 0xFF, 0x9A,
0xB8, 0xD4, 0xFF, 0x9E, 0xBD, 0xD9, 0xFF,
/*00EF90:*/ 0x96, 0x38, 0xDC, 0xFF, 0x8A, 0x30, 0xDE, 0xFF, 0x86,
0x30, 0xE1, 0xFF, 0x8A, 0x71, 0xE4, 0xFF,
/*00EFA0:*/ 0x86, 0x38, 0xF3, 0xFF, 0x86, 0x38, 0xF8, 0xFF, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*00EFB0:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B3D2_5[] = {
```

```
/*00F5D8:*/ 0xD3, 0x08, 0x03, 0x01, 0xD3, 0x89, 0x16, 0x01,
/*00F5E0:*/ 0xD3, 0x06, 0x1A, 0x01, 0xD3, 0x84, 0x1A, 0x01, 0x53,
0x84, 0x1B, 0x01, 0xD3, 0x84, 0x1C, 0x01,
/*00F5F0:*/ 0x93, 0x86, 0x58, 0x01, 0x93, 0x82, 0x12, 0x05, 0xD3,
0x0E, 0x1B, 0x05, 0x53, 0x8C, 0x1C, 0x05,
/*00F600:*/ 0x93, 0x02, 0x42, 0x05, 0x53, 0x8E, 0x4C, 0x05, 0x13,
0x0C, 0x5A, 0x05, 0xD3, 0x88, 0x02, 0x09,
/*00F610:*/ 0x53, 0x07, 0x07, 0x09, 0x93, 0x09, 0x07, 0x09, 0xD3,
0x85, 0x0C, 0x09, 0xD3, 0x8B, 0x17, 0x09,
/*00F620:*/ 0x93, 0x0A, 0x1A, 0x09, 0xD3, 0x84, 0x45, 0x09, 0x13,
0x05, 0x47, 0x09, 0x13, 0x84, 0x52, 0x09,
/*00F630:*/ 0x13, 0x84, 0x58, 0x09, 0xD3, 0x08, 0x5A, 0x09, 0x13,
0x01, 0x01, 0x0D, 0x53, 0x0F, 0x04, 0x0D,
/*00F640:*/ 0x13, 0x0C, 0x07, 0x0D, 0xD3, 0x81, 0x44, 0x0D, 0x93,
0x00, 0x45, 0x0D, 0x13, 0x82, 0x4A, 0x0D,
/*00F650:*/ 0x13, 0x03, 0x57, 0x0D, 0xD3, 0x00, 0x03, 0x11, 0x53,
0x82, 0x06, 0x11, 0x93, 0x8C, 0x06, 0x11,
/*00F660:*/ 0xD3, 0x8F, 0x07, 0x11, 0xD3, 0x00, 0x0B, 0x11, 0x13,
0xCB, 0x0B, 0x11, 0xD3, 0x0F, 0x57, 0x11,
/*00F670:*/ 0x13, 0x83, 0x57, 0x11, 0x93, 0x8A, 0x12, 0x15, 0x53,
0x86, 0x4C, 0x15, 0xD3, 0x88, 0x5A, 0x15,
/*00F680:*/ 0xD3, 0x0A, 0x5C, 0x15, 0xD3, 0x80, 0x02, 0x19, 0x13,
0x4B, 0x0E, 0x19, 0x53, 0x83, 0x16, 0x19,
/*00F690:*/ 0xD3, 0x83, 0x17, 0x19, 0x13, 0x8D, 0x17, 0x19, 0xD3,
0x8C, 0x45, 0x19, 0xD3, 0x81, 0x47, 0x19,
/*00F6A0:*/ 0xD3, 0x0E, 0x4B, 0x19, 0x13, 0x82, 0x4B, 0x19, 0x93,
0x8C, 0x53, 0x19, 0x13, 0x0E, 0x58, 0x19,
/*00F6B0:*/ 0x13, 0x8C, 0x58, 0x19, 0x53, 0x82, 0x5D, 0x19, 0x93,
0x08, 0x45, 0x1D, 0x93, 0x06, 0x56, 0x1D,
/*00F6C0:*/ 0x53, 0x08, 0x5A, 0x1D, 0xD3, 0x8D, 0x4F, 0x21, 0x53,
0x0C, 0x5F, 0x21, 0x93, 0x08, 0x04, 0x25,
/*00F6D0:*/ 0x13, 0x89, 0x12, 0x25, 0xD3, 0x09, 0x5F, 0x25, 0xD3,
0x86, 0x5F, 0x25, 0xD3, 0x0E, 0x57, 0x29,
/*00F6E0:*/ 0xD3, 0x0E, 0x5F, 0x29, 0x53, 0x88, 0x1E, 0x2D, 0x93,
0x06, 0x46, 0x2D, 0x93, 0x06, 0x4A, 0x2D,
/*00F6F0:*/ 0x93, 0x84, 0x4E, 0x2D, 0x53, 0x8A, 0x4E, 0x2D, 0xD3,
0x09, 0x58, 0x31, 0xD3, 0x00, 0x1E, 0x35,
/*00F700:*/ 0xD3, 0x02, 0x4E, 0x35, 0xD3, 0x0F, 0x4E, 0x35, 0x53,
0x02, 0x1E, 0x3D, 0x13, 0x0C, 0x1F, 0x3D,
/*00F710:*/ 0x93, 0x8C, 0x46, 0x3D, 0x53, 0x82, 0x4E, 0x3D, 0xD3,
0x08, 0x0D, 0x41, 0xD3, 0x87, 0x0D, 0x41,
/*00F720:*/ 0x13, 0x08, 0x10, 0x41, 0xD3, 0x06, 0x12, 0x41, 0x93,
0x08, 0x1F, 0x41, 0xD3, 0x09, 0x40, 0x41,
/*00F730:*/ 0xD3, 0x04, 0x46, 0x41, 0xD3, 0x0D, 0x0E, 0x45, 0x93,
0x8F, 0x12, 0x45, 0x93, 0x8D, 0x4E, 0x45,
/*00F740:*/ 0xD3, 0x02, 0x5C, 0x45, 0x13, 0x04, 0x0A, 0x49, 0x93,
0x88, 0x1A, 0x49, 0x93, 0x8A, 0x42, 0x49,
/*00F750:*/ 0x13, 0x08, 0x43, 0x49, 0x93, 0x84, 0x53, 0x49, 0x53,
0x08, 0x5B, 0x49, 0x53, 0x08, 0x5D, 0x49,
/*00F760:*/ 0xD3, 0x8C, 0x4A, 0x4D, 0x93, 0x0E, 0x54, 0x4D, 0x93,
0x0E, 0x5E, 0x4D, 0x93, 0x0E, 0x04, 0x51,
/*00F770:*/ 0xD3, 0x0E, 0x12, 0x51, 0xD3, 0x0E, 0x16, 0x51, 0x93,
0x02, 0x43, 0x51, 0xD3, 0x08, 0x02, 0x55,
/*00F780:*/ 0x13, 0x84, 0x04, 0x55, 0xD3, 0x06, 0x15, 0x55, 0x93,
0x0A, 0x4C, 0x55, 0x53, 0x04, 0x4E, 0x55,
/*00F790:*/ 0x93, 0x88, 0x4E, 0x55, 0xD3, 0x0A, 0x54, 0x55, 0x53,
0x88, 0x55, 0x55, 0x13, 0x86, 0x56, 0x55,
/*00F7A0:*/ 0x13, 0x04, 0x5E, 0x55, 0x93, 0x04, 0x5F, 0x55, 0x13,
0x01, 0x00, 0x59, 0x53, 0x02, 0x0B, 0x59,
/*00F7B0:*/ 0x13, 0x0C, 0x0C, 0x59, 0x53, 0x01, 0x14, 0x59, 0x13,
0x8C, 0x52, 0x59, 0x53, 0x00, 0x53, 0x59,
/*00F7C0:*/ 0x93, 0x8C, 0x55, 0x59, 0x93, 0x09, 0x02, 0x5D, 0x53,
0x0A, 0x02, 0x5D, 0x53, 0x85, 0x02, 0x5D,
/*00F7D0:*/ 0x93, 0x04, 0x04, 0x5D, 0x93, 0x04, 0x0C, 0x5D, 0x93,
0x07, 0x13, 0x5D, 0x13, 0x06, 0x5F, 0x5D,
/*00F7E0:*/ 0x93, 0x0D, 0x0F, 0x61, 0x93, 0x03, 0x1E, 0x61, 0xD3,
0x8D, 0x4F, 0x61, 0x13, 0x8D, 0x52, 0x61,
/*00F7F0:*/ 0x13, 0x07, 0x0E, 0x6D, 0xD3, 0x8A, 0x49, 0x6D, 0x53,
0x08, 0x4E, 0x6D, 0x53, 0x84, 0x0F, 0x71,
/*00F800:*/ 0x13, 0x8B, 0x4F, 0x71, 0x93, 0x0C, 0x4F, 0x75, 0xD3,
0x8E, 0x53, 0x75, 0x13, 0x09, 0x1E, 0x79,
/*00F810:*/ 0x53, 0x05, 0x4F, 0x79, 0x13, 0x87, 0x5F, 0x79, 0x13,
0x0F, 0x0E, 0x7D, 0x0D, 0xF0, 0xAD, 0xBA,
/*00F820:*/ 0x0D, 0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*00F830:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B3D2_6[] = {
/*0048:*/ 0x04, 0x0C, 0xFA, 0x00, 0x0C, 0x00, 0x7C, 0x04,
/*0050:*/ 0x8C, 0x0C, 0xFC, 0x05, 0x0C, 0x08, 0x7C, 0x06, 0x8C,
0x19, 0x7E, 0x07, 0x04, 0x08, 0x7B, 0x08,
/*0060:*/ 0x0C, 0x10, 0x7C, 0x09, 0x8C, 0x14, 0x7C, 0x09, 0x0C,
0x18, 0x7E, 0x0B, 0x8C, 0x05, 0xFC, 0x0B,
/*0070:*/ 0x0C, 0x04, 0xFB, 0x0C, 0x8C, 0x11, 0x7B, 0x0E, 0x0C,
0x04, 0x7B, 0x0F, 0x0C, 0x05, 0x78, 0x10,
/*0080:*/ 0x04, 0x08, 0x7B, 0x10, 0x0C, 0x0D, 0x78, 0x12, 0x8C,
0x14, 0xFC, 0x12, 0x0C, 0x14, 0xFB, 0x19,
/*0090:*/ 0x0C, 0x08, 0xFC, 0x1D, 0x8C, 0x08, 0x78, 0x1E, 0x0C,
0x08, 0x7C, 0x1E, 0x0C, 0x0C, 0x7E, 0x1E,
/*00A0:*/ 0x84, 0x18, 0xFB, 0x1E, 0x8C, 0x04, 0xFC, 0x1F, 0x0C,
0x04, 0xFE, 0x1F, 0x8C, 0x08, 0x7A, 0x20,
/*00B0:*/ 0x8C, 0x1D, 0x78, 0x21, 0x0C, 0x00, 0xFE, 0x21, 0x0C,
0x04, 0x7E, 0x22, 0x04, 0x10, 0xFD, 0x22,
/*00C0:*/ 0x8C, 0x11, 0x7A, 0x23, 0x04, 0x04, 0xFC, 0x24, 0x0C,
0x1C, 0x7B, 0x26, 0x04, 0x0D, 0x78, 0x29,
/*00D0:*/ 0x8C, 0x1C, 0x7C, 0x2D, 0x0C, 0x01, 0x7E, 0x2E, 0x0C,
0x18, 0xFE, 0x2E, 0x84, 0x01, 0x7A, 0x33,
/*00E0:*/ 0x84, 0x09, 0xF9, 0x39, 0x0C, 0x04, 0x7E, 0x3A, 0x04,
0x0C, 0xF8, 0x3E, 0x0C, 0x09, 0x7F, 0x42,
/*00F0:*/ 0x84, 0x09, 0xFB, 0x44, 0x8C, 0x01, 0xFE, 0x48, 0x8C,
0x0D, 0x7C, 0x49, 0x84, 0x08, 0x7B, 0x4B,
/*0100:*/ 0x8C, 0x1C, 0xFE, 0x4B, 0x04, 0x05, 0x78, 0x4E, 0x04,
0x05, 0x7C, 0x4E, 0x0C, 0x11, 0xFB, 0x56,
/*0110:*/ 0x0C, 0x1C, 0xFB, 0x58, 0x04, 0x04, 0xFA, 0x59, 0x8C,
0x08, 0x7E, 0x5D, 0x8C, 0x15, 0xFC, 0x5D,
/*0120:*/ 0x0C, 0x0C, 0xFA, 0x60, 0x0C, 0x00, 0x7C, 0x61, 0x0C,
0x00, 0xFC, 0x62, 0x04, 0x1D, 0x78, 0x67,
/*0130:*/ 0x0C, 0x18, 0xFE, 0x6D, 0x04, 0x1C, 0xFA, 0x70, 0x84,
0x0D, 0xFE, 0x71, 0x84, 0x18, 0x78, 0x73,
/*0140:*/ 0x8C, 0x01, 0x7E, 0x75, 0x0C, 0x04, 0x78, 0x79, 0x0C,
0x04, 0xF8, 0x7A, 0x0C, 0x00, 0xFC, 0x7A,
/*0150:*/ 0x8C, 0x11, 0xFA, 0x7B, 0x84, 0x1C, 0xFF, 0x7B, 0x0C,
0x1D, 0x7A, 0x83, 0x04, 0x08, 0xF8, 0x84,
/*0160:*/ 0x84, 0x1D, 0xFC, 0x85, 0x0C, 0x1C, 0xF8, 0x8C, 0x8C,
0x10, 0x78, 0x8D, 0x8C, 0x0D, 0x78, 0x8E,
/*0170:*/ 0x0C, 0x10, 0xFE, 0x8E, 0x0C, 0x05, 0xFC, 0x8F, 0x0C,
0x05, 0x7C, 0x94, 0x0C, 0x05, 0x7E, 0x94,
/*0180:*/ 0x8C, 0x0D, 0x78, 0x96, 0x8C, 0x1C, 0x7A, 0x97, 0x0C,
0x05, 0xFC, 0x97, 0x8C, 0x08, 0xF8, 0x99,
/*0190:*/ 0x0C, 0x0C, 0xFE, 0x99, 0x04, 0x0C, 0xFA, 0xA2, 0x8C,
0x04, 0xFA, 0xA5, 0x0C, 0x15, 0x78, 0xA7,
/*01A0:*/ 0x0C, 0x0D, 0x7A, 0xA8, 0x8C, 0x14, 0xFA, 0xA8, 0x8C,
0x14, 0xFC, 0xA8, 0x8C, 0x01, 0xF8, 0xA9,
/*01B0:*/ 0x8C, 0x01, 0xFA, 0xA9, 0x8C, 0x10, 0x7A, 0xAB, 0x84,
0x0D, 0xFE, 0xAE, 0x0C, 0x14, 0xFE, 0xB0,
/*01C0:*/ 0x0C, 0x0D, 0xF8, 0xB3, 0x04, 0x14, 0x7C, 0xB6, 0x84,
0x15, 0x7C, 0xBA, 0x8C, 0x10, 0xFB, 0xBB,
/*01D0:*/ 0x8C, 0x04, 0xFC, 0xBD, 0x8C, 0x1D, 0xF8, 0xBE, 0x8C,
0x19, 0xFE, 0xBE, 0x84, 0x18, 0xFB, 0xC1,
/*01E0:*/ 0x8C, 0x0C, 0xFE, 0xC2, 0x0C, 0x1D, 0xFA, 0xC3, 0x0C,
0x19, 0xFE, 0xC3, 0x84, 0x0C, 0x78, 0xC4,
/*01F0:*/ 0x0C, 0x01, 0x7F, 0xC4, 0x04, 0x19, 0xF8, 0xC6, 0x8C,
0x05, 0xF9, 0xC7, 0x0C, 0x10, 0xFC, 0xCD,
/*0200:*/ 0x0C, 0x10, 0x7E, 0xCE, 0x04, 0x09, 0x78, 0xD0, 0x8C,
0x0D, 0x7A, 0xD5, 0x0C, 0x10, 0xFC, 0xD5,
/*0210:*/ 0x0C, 0x14, 0xFE, 0xD5, 0x8C, 0x18, 0xFE, 0xD7, 0x8C,
0x1D, 0x7A, 0xD8, 0x8C, 0x0C, 0x7A, 0xD9,
/*0220:*/ 0x0C, 0x08, 0x7E, 0xD9, 0x0C, 0x15, 0xF8, 0xD9, 0x0C,
0x1D, 0xFA, 0xDB, 0x0C, 0x09, 0xFB, 0xDD,
/*0230:*/ 0x0C, 0x0D, 0xFB, 0xDD, 0x0C, 0x1C, 0x7B, 0xE1, 0x84,
0x15, 0x7C, 0xE1, 0x84, 0x0C, 0xFC, 0xE1,
/*0240:*/ 0x04, 0x15, 0xF8, 0xE2, 0x0C, 0x18, 0xFB, 0xE2, 0x8C,
0x11, 0x7A, 0xE4, 0x0C, 0x05, 0x7E, 0xE9,
/*0250:*/ 0x0C, 0x18, 0x7E, 0xEA, 0x0C, 0x10, 0xFC, 0xEB, 0x0C,
0x14, 0x7A, 0xF0, 0x04, 0x15, 0x78, 0xF9,
/*0260:*/ 0x0C, 0x1C, 0x7B, 0xF9, 0x04, 0x08, 0x78, 0xFA, 0x8C,
0x04, 0xFE, 0xFE, 0x8C, 0x0C, 0x78, 0xFF,
/*0270:*/ 0x8C, 0x15, 0xFC, 0xFF, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*0280:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B3D2_7[] = {
```

```
/*0720:*/ 0x36, 0x54, 0x3A, 0x90, 0x32, 0x3E, 0x3E, 0x92, 0x3C,
0xA4, 0x34, 0xA0, 0x38, 0x54, 0x35, 0xA0,
/*0730:*/ 0x34, 0x9E, 0x38, 0xA0, 0x38, 0xBE, 0x38, 0xA0, 0x38,
0xBE, 0x3D, 0xA0, 0x3C, 0xA4, 0x3E, 0xA0,
/*0740:*/ 0x32, 0x94, 0x30, 0xA1, 0x3C, 0x3C, 0x35, 0xA1, 0x30,
0xF6, 0x35, 0xA1, 0x3E, 0xAC, 0x38, 0xA1,
/*0750:*/ 0x30, 0xEE, 0x3B, 0xA1, 0x36, 0x64, 0x3C, 0xA1, 0x32,
0x7E, 0x3C, 0xA1, 0x38, 0xCC, 0x3E, 0xA1,
/*0760:*/ 0x38, 0x26, 0x3F, 0xA1, 0x38, 0xCC, 0x3F, 0xA1, 0x34,
0x04, 0x30, 0xA2, 0x3E, 0x5C, 0x30, 0xA2,
/*0770:*/ 0x3A, 0x5E, 0x30, 0xA2, 0x36, 0x7E, 0x30, 0xA2, 0x30,
0xF4, 0x31, 0xA2, 0x38, 0x3C, 0x33, 0xA2,
/*0780:*/ 0x3C, 0xD4, 0x33, 0xA2, 0x30, 0xEC, 0x33, 0xA2, 0x30,
0xF4, 0x33, 0xA2, 0x32, 0x8E, 0x36, 0xA2,
/*0790:*/ 0x3E, 0xAE, 0x36, 0xA2, 0x36, 0x7E, 0x39, 0xA2, 0x3A,
0xB4, 0x39, 0xA2, 0x38, 0xD6, 0x3F, 0xA2,
/*07A0:*/ 0x34, 0xF6, 0x3F, 0xA2, 0x3E, 0x2E, 0x30, 0xA3, 0x3E,
0x36, 0x30, 0xA3, 0x32, 0xE4, 0x33, 0xA3,
/*07B0:*/ 0x32, 0xFC, 0x35, 0xA3, 0x38, 0x4E, 0x3B, 0xA3, 0x30,
0x9E, 0x3B, 0xA3, 0x38, 0xBC, 0x3B, 0xA3,
/*07C0:*/ 0x3E, 0xDC, 0x3D, 0xA3, 0x3E, 0x2E, 0x3E, 0xA3, 0x30,
0x6C, 0x3E, 0xA3, 0x32, 0xE4, 0x3E, 0xA3,
/*07D0:*/ 0x3C, 0x54, 0x3F, 0xA3, 0x3A, 0xDE, 0x3F, 0xA3, 0x30,
0x5E, 0x3A, 0xA8, 0x36, 0xCC, 0x3A, 0xA8,
/*07E0:*/ 0x32, 0xD6, 0x3A, 0xA8, 0x3E, 0xF6, 0x3A, 0xA8, 0x30,
0x46, 0x3E, 0xA8, 0x3A, 0x86, 0x39, 0xA9,
/*07F0:*/ 0x38, 0x16, 0x3B, 0xA9, 0x32, 0xA6, 0x36, 0xAA, 0x36,
0x56, 0x38, 0xAA, 0x3A, 0x9C, 0x38, 0xAA,
/*0800:*/ 0x3E, 0x9E, 0x3A, 0xAA, 0x36, 0xA4, 0x3A, 0xAA, 0x38,
0x14, 0x3B, 0xAA, 0x3E, 0xF4, 0x38, 0xAB,
/*0810:*/ 0x36, 0xD6, 0x3A, 0xAB, 0x38, 0x7E, 0x3D, 0xAB, 0x36,
0x46, 0x31, 0xB0, 0x32, 0x5C, 0x31, 0xB0,
/*0820:*/ 0x36, 0xAC, 0x31, 0xB0, 0x3E, 0x7C, 0x32, 0xB0, 0x34,
0x3C, 0x35, 0xB0, 0x38, 0xF6, 0x36, 0xB0,
/*0830:*/ 0x32, 0x2E, 0x32, 0xB1, 0x36, 0xC6, 0x32, 0xB1, 0x3A,
0x0C, 0x38, 0xB1, 0x38, 0x84, 0x3D, 0xB1,
/*0840:*/ 0x38, 0x9C, 0x3D, 0xB1, 0x34, 0x54, 0x30, 0xB2, 0x30,
0xA4, 0x30, 0xB2, 0x36, 0xDC, 0x32, 0xB2,
/*0850:*/ 0x36, 0x2E, 0x35, 0xB2, 0x30, 0xBC, 0x3D, 0xB2, 0x32,
0xB4, 0x33, 0xB3, 0x34, 0xCC, 0x34, 0xB3,
/*0860:*/ 0x32, 0x46, 0x38, 0xB3, 0x3E, 0x94, 0x3B, 0xB3, 0x38,
0xF4, 0x3B, 0xB3, 0x38, 0x06, 0x3E, 0xB3,
/*0870:*/ 0x3E, 0x54, 0x3B, 0xB8, 0x30, 0x8E, 0x33, 0xB9, 0x38,
0xB4, 0x35, 0xB9, 0x3E, 0x24, 0x3B, 0xBA,
/*0880:*/ 0x3E, 0x3C, 0x3B, 0xBA, 0x32, 0x6E, 0x3B, 0xBB, 0x3E,
0x24, 0x39, 0xD1, 0x3E, 0x26, 0x3A, 0xD2,
/*0890:*/ 0x36, 0xDC, 0x31, 0xE0, 0x3C, 0x9C, 0x32, 0xE0, 0x34,
0x4C, 0x33, 0xE0, 0x3C, 0x6E, 0x35, 0xE0,
/*08A0:*/ 0x32, 0x2C, 0x36, 0xE0, 0x32, 0x34, 0x36, 0xE0, 0x3E,
0x14, 0x39, 0xE0, 0x30, 0x4E, 0x39, 0xE0,
/*08B0:*/ 0x30, 0x56, 0x39, 0xE0, 0x3C, 0x6E, 0x39, 0xE0, 0x30,
0xA4, 0x39, 0xE0, 0x36, 0xDC, 0x39, 0xE0,
/*08C0:*/ 0x34, 0x54, 0x3B, 0xE0, 0x38, 0x9E, 0x3B, 0xE0, 0x3A,
0x0E, 0x3C, 0xE0, 0x3E, 0x14, 0x3C, 0xE0,
/*08D0:*/ 0x3A, 0x0E, 0x3E, 0xE0, 0x3E, 0x14, 0x3E, 0xE0, 0x34,
0xA6, 0x3E, 0xE0, 0x3E, 0xE6, 0x3F, 0xE0,
/*08E0:*/ 0x3A, 0xFC, 0x3F, 0xE0, 0x3A, 0x96, 0x30, 0xE1, 0x38,
0xF4, 0x30, 0xE1, 0x3E, 0x7E, 0x31, 0xE1,
/*08F0:*/ 0x38, 0xF4, 0x31, 0xE1, 0x3C, 0xEE, 0x32, 0xE1, 0x3C,
0xF6, 0x33, 0xE1, 0x3E, 0x8C, 0x39, 0xE1,
/*0900:*/ 0x30, 0x24, 0x3E, 0xE1, 0x30, 0x3C, 0x3E, 0xE1, 0x3C,
0xEE, 0x3E, 0xE1, 0x3E, 0x8E, 0x31, 0xE2,
/*0910:*/ 0x30, 0xCC, 0x32, 0xE2, 0x3E, 0x7C, 0x38, 0xE2, 0x32,
0xAE, 0x39, 0xE2, 0x34, 0xD6, 0x39, 0xE2,
/*0920:*/ 0x38, 0xEE, 0x3B, 0xE2, 0x3C, 0xF4, 0x3B, 0xE2, 0x32,
0x44, 0x3C, 0xE2, 0x38, 0xEE, 0x3C, 0xE2,
/*0930:*/ 0x38, 0xF6, 0x3C, 0xE2, 0x3E, 0x7C, 0x3D, 0xE2, 0x36,
0xAC, 0x3E, 0xE2, 0x3E, 0xE4, 0x32, 0xE3,
/*0940:*/ 0x36, 0xC6, 0x33, 0xE3, 0x36, 0xDE, 0x33, 0xE3, 0x36,
0xC6, 0x36, 0xE3, 0x38, 0x6E, 0x38, 0xE3,
/*0950:*/ 0x30, 0xA6, 0x38, 0xE3, 0x3A, 0xE6, 0x3D, 0xE3, 0x32,
0x2E, 0x3E, 0xE3, 0x38, 0x84, 0x3F, 0xE3,
/*0960:*/ 0x36, 0x06, 0x39, 0xE8, 0x38, 0xDC, 0x3A, 0xE9, 0x34,
0x0C, 0x30, 0xEA, 0x3E, 0xA6, 0x38, 0xEA,
/*0970:*/ 0x3A, 0x4E, 0x3A, 0xEA, 0x30, 0x0E, 0x3B, 0xEA, 0x38,
0x34, 0x3B, 0xEA, 0x30, 0x64, 0x3A, 0xEB,
/*0980:*/ 0x36, 0x7E, 0x31, 0xF0, 0x3A, 0x5E, 0x32, 0xF0, 0x3A,
0xC6, 0x32, 0xF1, 0x3E, 0xDC, 0x32, 0xF1,
/*0990:*/ 0x32, 0xFC, 0x35, 0xF1, 0x36, 0xFC, 0x32, 0xF2, 0x32,
0x14, 0x38, 0xF2, 0x3E, 0x34, 0x3D, 0xF2,
/*09A0:*/ 0x3E, 0x5E, 0x32, 0xF3, 0x36, 0x64, 0x32, 0xF3, 0x32,
0x7E, 0x38, 0xF3, 0x36, 0x96, 0x38, 0xF3,
/*09B0:*/ 0x3C, 0x16, 0x36, 0xF8, 0x30, 0xDC, 0x37, 0xF8, 0x3A,
0x76, 0x3B, 0xF8, 0x30, 0xC4, 0x3B, 0xF8,
/*09C0:*/ 0x3A, 0xF4, 0x3B, 0xFA, 0x3E, 0xF6, 0x3B, 0xFA, 0x3A,
0x06, 0x3D, 0xFA, 0x3E, 0x84, 0x36, 0xFB,
/*09D0:*/ 0x3A, 0x9E, 0x3D, 0xFB, 0x0D, 0xF0, 0xAD, 0xBA, 0x0D,
0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*09E0:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B3D2_8[] = {
/*0FA8:*/ 0x1F, 0x06, 0x86, 0x0D, 0xAB, 0x05, 0xA6, 0x0D,
/*0FB0:*/ 0x88, 0x06, 0xA6, 0x0D, 0x2E, 0x06, 0x86, 0x0F, 0x94,
0x06, 0xA4, 0x0F, 0x6E, 0x07, 0xA6, 0x0F,
/*0FC0:*/ 0x25, 0x07, 0x80, 0x1D, 0x7C, 0x06, 0x82, 0x1D, 0x88,
0x04, 0x84, 0x1D, 0x72, 0x05, 0x86, 0x1D,
/*0FD0:*/ 0xE4, 0x05, 0x86, 0x1D, 0xBC, 0x04, 0xA4, 0x1D, 0xD8,
0x05, 0xA4, 0x1D, 0xAC, 0x04, 0xA6, 0x1D,
/*0FE0:*/ 0x9A, 0x07, 0x86, 0x1F, 0xDC, 0x05, 0xA2, 0x1F, 0x79,
0x06, 0xA4, 0x1F, 0x1B, 0x04, 0xA6, 0x1F,
/*0FF0:*/ 0x3D, 0x06, 0xA4, 0x2D, 0xF4, 0x04, 0xA6, 0x2D, 0x15,
0x06, 0x84, 0x2F, 0xA1, 0x05, 0xA2, 0x2F,
/*1000:*/ 0x14, 0x06, 0xA4, 0x2F, 0x74, 0x05, 0x84, 0x3D, 0xF4,
0x06, 0x84, 0x3D, 0x24, 0x04, 0xA2, 0x3D,
/*1010:*/ 0xD8, 0x06, 0xA2, 0x3D, 0x55, 0x05, 0x80, 0x3F, 0xC5,
0x06, 0x80, 0x3F, 0xDD, 0x06, 0x82, 0x3F,
/*1020:*/ 0xF8, 0x06, 0x82, 0x3F, 0xE0, 0x06, 0x84, 0x3F, 0x29,
0x04, 0x86, 0x3F, 0xB7, 0x04, 0x86, 0x3F,
/*1030:*/ 0xE0, 0x06, 0x86, 0x3F, 0x20, 0x04, 0xA2, 0x3F, 0xC4,
0x06, 0xA4, 0x3F, 0x44, 0x05, 0xA6, 0x3F,
/*1040:*/ 0x7C, 0x07, 0x86, 0x45, 0x21, 0x07, 0x80, 0x4D, 0x39,
0x07, 0x80, 0x4D, 0x94, 0x04, 0x84, 0x4D,
/*1050:*/ 0x39, 0x07, 0x84, 0x4D, 0x1A, 0x04, 0x86, 0x4D, 0x31,
0x07, 0x86, 0x4D, 0xFC, 0x05, 0x84, 0x4F,
/*1060:*/ 0x51, 0x06, 0x84, 0x4F, 0x4E, 0x05, 0xA0, 0x4F, 0x44,
0x04, 0x80, 0x5D, 0xC5, 0x07, 0xA0, 0x5D,
/*1070:*/ 0xE8, 0x07, 0xA2, 0x5D, 0x60, 0x04, 0xA4, 0x5D, 0xD0,
0x07, 0x86, 0x5F, 0x8D, 0x07, 0xA0, 0x6D,
/*1080:*/ 0xA0, 0x07, 0xA0, 0x6D, 0x49, 0x05, 0x82, 0x6F, 0x95,
0x05, 0x82, 0x7D, 0x30, 0x06, 0x82, 0x7D,
/*1090:*/ 0x0D, 0x06, 0x84, 0x7D, 0x8D, 0x05, 0x86, 0x7D, 0xE0,
0x04, 0xA2, 0x7D, 0x70, 0x07, 0xA2, 0x7D,
/*10A0:*/ 0xA1, 0x05, 0xA4, 0x7D, 0xA1, 0x05, 0xA6, 0x7D, 0x14,
0x06, 0xA6, 0x7D, 0x1C, 0x06, 0xA6, 0x7D,
/*10B0:*/ 0x34, 0x06, 0x82, 0x7F, 0x48, 0x07, 0x82, 0x7F, 0xD0,
0x04, 0x86, 0x7F, 0xB4, 0x05, 0x86, 0x7F,
/*10C0:*/ 0x35, 0x06, 0xA2, 0x7F, 0x35, 0x06, 0xA4, 0x7F, 0xC9,
0x04, 0xA6, 0x7F, 0x25, 0x06, 0xA6, 0x7F,
/*10D0:*/ 0xDF, 0x05, 0xA2, 0x8D, 0x51, 0x05, 0xA4, 0x8D, 0xA8,
0x07, 0x84, 0x8F, 0x84, 0x07, 0xA2, 0x8F,
/*10E0:*/ 0x94, 0x07, 0xA4, 0x8F, 0x3D, 0x06, 0x80, 0x9D, 0xF4,
0x04, 0x84, 0x9D, 0xC9, 0x04, 0x86, 0x9D,
/*10F0:*/ 0xBC, 0x05, 0xA2, 0x9D, 0x11, 0x06, 0xA2, 0x9D, 0xC0,
0x04, 0xA4, 0x9D, 0x39, 0x06, 0x80, 0x9F,
/*1100:*/ 0xCD, 0x04, 0x82, 0x9F, 0xE0, 0x04, 0x82, 0x9F, 0x8C,
0x05, 0x84, 0x9F, 0xB1, 0x05, 0x84, 0x9F,
/*1110:*/ 0x21, 0x06, 0x86, 0x9F, 0xB8, 0x05, 0xA2, 0x9F, 0x54,
0x07, 0xA4, 0x9F, 0xAC, 0x04, 0x84, 0xAD,
/*1120:*/ 0xC6, 0x06, 0x86, 0xAD, 0x85, 0x04, 0x80, 0xAF, 0x4A,
0x05, 0x80, 0xAF, 0x95, 0x04, 0x84, 0xAF,
/*1130:*/ 0xB0, 0x04, 0x86, 0xAF, 0xCA, 0x06, 0x86, 0xAF, 0x00,
0x05, 0x84, 0xBD, 0xD6, 0x04, 0xA6, 0xBD,
/*1140:*/ 0xDE, 0x04, 0xA6, 0xBD, 0x39, 0x05, 0x80, 0xBF, 0xB7,
0x05, 0x80, 0xBF, 0xA1, 0x06, 0x80, 0xBF,
/*1150:*/ 0x70, 0x04, 0x82, 0xBF, 0xB1, 0x06, 0x84, 0xBF, 0x85,
0x06, 0xA0, 0xBF, 0x4A, 0x07, 0xA0, 0xBF,
/*1160:*/ 0xDA, 0x04, 0xA6, 0xBF, 0xA0, 0x06, 0xA6, 0xBF, 0x4A,
0x04, 0xA0, 0xCD, 0xD4, 0x04, 0xA4, 0xCD,
/*1170:*/ 0xA8, 0x05, 0xA4, 0xCD, 0xD1, 0x04, 0x86, 0xCF, 0xE5,
0x04, 0xA0, 0xCF, 0x44, 0x05, 0x80, 0xDD,
/*1180:*/ 0xB8, 0x07, 0x82, 0xDD, 0x36, 0x07, 0x86, 0xDD, 0x8D,
0x07, 0x86, 0xDD, 0x31, 0x04, 0xA2, 0xDD,
/*1190:*/ 0x78, 0x05, 0xA4, 0xDD, 0x0C, 0x04, 0xA6, 0xDD, 0x29,
0x04, 0xA6, 0xDD, 0x4B, 0x06, 0xA6, 0xDD,
/*11A0:*/ 0xBC, 0x07, 0x82, 0xDF, 0x7D, 0x05, 0x84, 0xDF, 0x32,
0x07, 0x86, 0xDF, 0x90, 0x07, 0xA2, 0xDF,
/*11B0:*/ 0x00, 0x04, 0xA4, 0xDF, 0x08, 0x04, 0xA6, 0xDF, 0xDF,
0x05, 0xA6, 0xDF, 0xB8, 0x07, 0xA6, 0xE7,
/*11C0:*/ 0x95, 0x06, 0xA4, 0xED, 0x10, 0x05, 0x80, 0xEF, 0x34,
0x05, 0xA6, 0xEF, 0x20, 0x07, 0x80, 0xFD,
/*11D0:*/ 0xA0, 0x04, 0x84, 0xFD, 0x4A, 0x05, 0x86, 0xFD, 0x78,
0x06, 0xA2, 0xFD, 0x3F, 0x04, 0xA6, 0xFD,
/*11E0:*/ 0xFD, 0x05, 0x80, 0xFF, 0xF5, 0x05, 0x82, 0xFF, 0xC6,
0x06, 0x84, 0xFF, 0xD6, 0x06, 0x86, 0xFF,
/*11F0:*/ 0xEC, 0x05, 0xA2, 0xFF, 0x7C, 0x06, 0xA4, 0xFF, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*1200:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B3D2_9[] = {
/*16E0:*/ 0x63, 0x20, 0x39, 0x04, 0x43, 0x64, 0x58, 0x04, 0x42,
0x60, 0x78, 0x04, 0x40, 0x68, 0x98, 0x04,
/*16F0:*/ 0x41, 0x6C, 0xB8, 0x04, 0x43, 0xE8, 0xD8, 0x04, 0x60,
0x20, 0xD9, 0x04, 0x61, 0x24, 0xF9, 0x04,
/*1700:*/ 0x61, 0x2C, 0x19, 0x05, 0x60, 0x34, 0x78, 0x05, 0x63,
0xB4, 0x98, 0x05, 0x40, 0x70, 0xB9, 0x05,
/*1710:*/ 0x40, 0xB4, 0xD8, 0x05, 0x40, 0x6C, 0xF8, 0x05, 0x42,
0x78, 0xF9, 0x05, 0x40, 0x24, 0xF8, 0x09,
/*1720:*/ 0x60, 0x20, 0xB9, 0x20, 0x40, 0xF4, 0xB9, 0x20, 0x61,
0x34, 0xF8, 0x20, 0x40, 0x30, 0x39, 0x23,
/*1730:*/ 0x60, 0xE8, 0x38, 0x24, 0x41, 0xE0, 0x98, 0x24, 0x42,
0xA8, 0x99, 0x24, 0x42, 0xFC, 0x99, 0x24,
/*1740:*/ 0x62, 0x24, 0xB9, 0x24, 0x42, 0xF0, 0xB9, 0x24, 0x62,
0xE4, 0x18, 0x25, 0x61, 0xF4, 0x99, 0x25,
/*1750:*/ 0x62, 0xE4, 0xB8, 0x25, 0x60, 0x34, 0xD8, 0x25, 0x60,
0xEC, 0xF8, 0x25, 0x62, 0xF8, 0xF9, 0x25,
/*1760:*/ 0x42, 0x60, 0x39, 0x28, 0x61, 0xE0, 0x79, 0x28, 0x41,
0xA4, 0xB8, 0x28, 0x41, 0xF4, 0x78, 0x29,
/*1770:*/ 0x41, 0xBC, 0x99, 0x29, 0x41, 0x7C, 0x58, 0x2C, 0x60,
0xE4, 0x99, 0x2C, 0x42, 0xEC, 0xD9, 0x2C,
/*1780:*/ 0x42, 0x30, 0x39, 0x2D, 0x42, 0x74, 0x58, 0x2D, 0x43,
0x70, 0x78, 0x2D, 0x40, 0xF0, 0x98, 0x2D,
/*1790:*/ 0x41, 0xA0, 0xB8, 0x2D, 0x41, 0x64, 0xD9, 0x2D, 0x41,
0xA8, 0x19, 0x40, 0x60, 0x7C, 0x59, 0x41,
/*17A0:*/ 0x41, 0xF8, 0xD9, 0x41, 0x41, 0xB8, 0x18, 0x44, 0x43,
0xE8, 0x78, 0x44, 0x43, 0xA0, 0x99, 0x44,
/*17B0:*/ 0x63, 0xA0, 0x99, 0x44, 0x41, 0xB8, 0xB8, 0x44, 0x62,
0xB4, 0xD8, 0x44, 0x43, 0xBC, 0xD8, 0x44,
/*17C0:*/ 0x60, 0xAC, 0xF9, 0x44, 0x42, 0x64, 0x18, 0x45, 0x41,
0xA0, 0x39, 0x45, 0x63, 0x70, 0xF9, 0x45,
/*17D0:*/ 0x42, 0xE8, 0xD9, 0x49, 0x63, 0x38, 0x39, 0x4D, 0x61,
0x6C, 0xD8, 0x60, 0x40, 0x34, 0x18, 0x61,
/*17E0:*/ 0x42, 0x20, 0x19, 0x61, 0x62, 0x78, 0x39, 0x61, 0x41,
0x74, 0x59, 0x61, 0x60, 0xA4, 0x79, 0x61,
/*17F0:*/ 0x40, 0x38, 0x98, 0x61, 0x42, 0x20, 0xB9, 0x61, 0x41,
0x2C, 0xD9, 0x61, 0x60, 0x70, 0xD9, 0x61,
/*1800:*/ 0x61, 0x74, 0xF9, 0x61, 0x43, 0xF8, 0x19, 0x64, 0x42,
0xFC, 0x39, 0x64, 0x62, 0xEC, 0x58, 0x64,
/*1810:*/ 0x42, 0x60, 0x78, 0x64, 0x63, 0xF8, 0xB9, 0x64, 0x61,
0xA4, 0xF9, 0x64, 0x41, 0xF8, 0x19, 0x65,
/*1820:*/ 0x61, 0xA0, 0x39, 0x65, 0x43, 0xA4, 0x59, 0x65, 0x61,
0xE8, 0x78, 0x65, 0x62, 0x20, 0x79, 0x65,
/*1830:*/ 0x61, 0x74, 0x99, 0x65, 0x41, 0x68, 0xD8, 0x65, 0x61,
0xE8, 0xD8, 0x65, 0x40, 0x7C, 0xD8, 0x69,
/*1840:*/ 0x40, 0x20, 0xF8, 0x6C, 0x41, 0x20, 0x18, 0x6D, 0x43,
0x34, 0xB9, 0x6D, 0x63, 0x2C, 0xD8, 0x80,
/*1850:*/ 0x61, 0x38, 0x19, 0x84, 0x43, 0x30, 0x59, 0x84, 0x41,
0xA8, 0x78, 0x84, 0x42, 0x34, 0xD9, 0x84,
/*1860:*/ 0x40, 0xF8, 0xF8, 0x84, 0x42, 0x38, 0xF9, 0x84, 0x41,
0x2C, 0x18, 0x85, 0x41, 0xF8, 0x18, 0x85,
/*1870:*/ 0x42, 0x78, 0x58, 0x85, 0x42, 0xBC, 0x18, 0x8D, 0x63,
0x3C, 0x19, 0xA0, 0x61, 0xA4, 0x38, 0xA0,
/*1880:*/ 0x61, 0x34, 0xF9, 0xA0, 0x61, 0x3C, 0xB9, 0xA1, 0x61,
0xBC, 0xB9, 0xA1, 0x61, 0x20, 0xF8, 0xA1,
/*1890:*/ 0x60, 0x28, 0xF8, 0xA1, 0x62, 0xA4, 0x18, 0xA4, 0x63,
0x20, 0x38, 0xA4, 0x61, 0x60, 0x39, 0xA4,
/*18A0:*/ 0x40, 0xBC, 0x39, 0xA4, 0x63, 0xBC, 0x79, 0xA4, 0x62,
0x7C, 0x98, 0xA4, 0x42, 0xFC, 0x98, 0xA4,
/*18B0:*/ 0x61, 0xB4, 0x99, 0xA4, 0x43, 0xE0, 0x39, 0xA5, 0x41,
0xF4, 0x98, 0xA5, 0x42, 0xE8, 0x99, 0xA5,
/*18C0:*/ 0x43, 0xEC, 0xB9, 0xA5, 0x60, 0xEC, 0xF9, 0xA5, 0x41,
0x34, 0x78, 0xA8, 0x41, 0xBC, 0x38, 0xA9,
/*18D0:*/ 0x40, 0x74, 0x59, 0xAC, 0x41, 0x70, 0x79, 0xAC, 0x41,
0x38, 0x98, 0xAC, 0x41, 0x6C, 0x98, 0xAC,
/*18E0:*/ 0x42, 0xE4, 0x38, 0xAD, 0x43, 0x70, 0x79, 0xAD, 0x42,
0x30, 0x98, 0xAD, 0x41, 0x20, 0xB9, 0xAD,
/*18F0:*/ 0x40, 0xE0, 0xF8, 0xAD, 0x42, 0x74, 0xF9, 0xAD, 0x40,
0xE8, 0xF9, 0xC0, 0x63, 0xF4, 0x38, 0xC4,
/*1900:*/ 0x43, 0x20, 0x98, 0xC4, 0x63, 0xAC, 0xB8, 0xC4, 0x62,
0xB4, 0xD9, 0xC4, 0x61, 0xA4, 0xF8, 0xC4,
/*1910:*/ 0x42, 0x30, 0x19, 0xC5, 0x41, 0xA0, 0x38, 0xC5, 0x43,
0x7C, 0x78, 0xC5, 0x62, 0xF4, 0x78, 0xC5,
/*1920:*/ 0x41, 0x3C, 0x79, 0xC5, 0x40, 0x7C, 0x98, 0xC5, 0x60,
0xA8, 0x98, 0xC5, 0x40, 0x24, 0xB8, 0xC5,
/*1930:*/ 0x60, 0xA4, 0xB8, 0xC5, 0x62, 0xF4, 0xD8, 0xC5, 0x42,
0xA0, 0x39, 0xC9, 0x40, 0x68, 0x18, 0xCC,
/*1940:*/ 0x63, 0xF4, 0xB9, 0xCC, 0x43, 0x30, 0x39, 0xE0, 0x43,
0x2C, 0x78, 0xE0, 0x63, 0xAC, 0x78, 0xE0,
/*1950:*/ 0x41, 0x38, 0x79, 0xE0, 0x61, 0xB8, 0x79, 0xE0, 0x61,
0xA0, 0x58, 0xE1, 0x43, 0x34, 0x59, 0xE1,
/*1960:*/ 0x40, 0x24, 0x78, 0xE1, 0x63, 0xB8, 0x79, 0xE1, 0x62,
0x74, 0xB8, 0xE1, 0x60, 0x70, 0xD8, 0xE1,
/*1970:*/ 0x40, 0x28, 0xF8, 0xE1, 0x61, 0x74, 0xF8, 0xE1, 0x42,
0x70, 0x18, 0xE4, 0x43, 0xF8, 0x18, 0xE4,
/*1980:*/ 0x41, 0xB8, 0x19, 0xE4, 0x41, 0x34, 0x39, 0xE4, 0x60,
0x68, 0x39, 0xE4, 0x61, 0xE0, 0x39, 0xE4,
/*1990:*/ 0x62, 0xB8, 0x59, 0xE4, 0x61, 0xFC, 0x78, 0xE4, 0x43,
0x78, 0xB8, 0xE4, 0x61, 0x7C, 0xD8, 0xE4,
/*19A0:*/ 0x62, 0x60, 0xD9, 0xE4, 0x61, 0xA4, 0xF8, 0xE4, 0x43,
0x30, 0xF9, 0xE4, 0x62, 0xB8, 0xF9, 0xE4,
/*19B0:*/ 0x40, 0xA4, 0x18, 0xE5, 0x41, 0xF4, 0x38, 0xE5, 0x63,
0x60, 0x39, 0xE5, 0x40, 0xEC, 0x59, 0xE5,
/*19C0:*/ 0x43, 0xFC, 0x78, 0xE5, 0x60, 0x60, 0x79, 0xE5, 0x61,
0x74, 0x98, 0xE5, 0x60, 0x70, 0xB8, 0xE5,
/*19D0:*/ 0x61, 0x3C, 0xD9, 0xE5, 0x41, 0xBC, 0xD9, 0xE5, 0x40,
0xB8, 0xF9, 0xE5, 0x40, 0x3C, 0xB8, 0xEC,
/*19E0:*/ 0x42, 0x28, 0xB9, 0xEC, 0x42, 0x3C, 0x18, 0xED, 0x0D,
0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*19F0:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B3D2_A[] = {
/*2140:*/ 0x51, 0xB7, 0x1F, 0x21, 0xD7, 0xA7, 0x2A, 0x21, 0xD7,
0xAC, 0x53, 0x21, 0xD5, 0xB4, 0x56, 0x21,
/*2150:*/ 0xD1, 0xB5, 0x56, 0x21, 0x51, 0xAD, 0x5B, 0x21, 0x53,
0xAD, 0x5B, 0x21, 0x51, 0xBE, 0x5C, 0x21,
/*2160:*/ 0xD3, 0xAC, 0x71, 0x21, 0x51, 0xBF, 0x79, 0x21, 0x53,
0xA6, 0x7B, 0x21, 0x57, 0xAC, 0x7E, 0x21,
/*2170:*/ 0x53, 0xBE, 0x7E, 0x21, 0x55, 0xBF, 0x7E, 0x21, 0xD1,
0xAC, 0x8D, 0x21, 0xD0, 0xB7, 0x9D, 0x21,
/*2180:*/ 0xD3, 0xAC, 0xD3, 0x21, 0x55, 0xAC, 0xDB, 0x21, 0xD3,
0xA6, 0xF4, 0x21, 0xD3, 0xAD, 0xF6, 0x21,
/*2190:*/ 0x57, 0xAD, 0xFE, 0x21, 0xD3, 0xA6, 0x0B, 0x31, 0xD5,
0xA5, 0x13, 0x31, 0xD1, 0xA5, 0x14, 0x31,
/*21A0:*/ 0xD5, 0xB6, 0x14, 0x31, 0xD1, 0xAF, 0x16, 0x31, 0x51,
0xB7, 0x1C, 0x31, 0x53, 0xBD, 0x3C, 0x31,
/*21B0:*/ 0x51, 0xB7, 0x45, 0x31, 0xD7, 0xAC, 0x57, 0x31, 0x51,
0xBE, 0x5F, 0x31, 0xD1, 0xAE, 0x6A, 0x31,
/*21C0:*/ 0xD5, 0xAD, 0x72, 0x31, 0xD5, 0xBE, 0x72, 0x31, 0xD3,
0xBF, 0x75, 0x31, 0x51, 0xAD, 0x7D, 0x31,
/*21D0:*/ 0x50, 0xAF, 0x93, 0x31, 0x51, 0xAC, 0xA4, 0x31, 0xD1,
0xBD, 0xC8, 0x31, 0xD3, 0xA6, 0xF0, 0x31,
/*21E0:*/ 0xD7, 0xBF, 0xF5, 0x31, 0xD3, 0xA7, 0xF7, 0x31, 0xD5,
0xA5, 0x1B, 0x61, 0xD2, 0xB7, 0x34, 0x61,
/*21F0:*/ 0xD5, 0xA7, 0x5A, 0x61, 0xD1, 0xA7, 0x5D, 0x61, 0x55,
0xAC, 0x75, 0x61, 0x53, 0xA7, 0x77, 0x61,
/*2200:*/ 0xD3, 0xBF, 0x7D, 0x61, 0xD3, 0xBF, 0x86, 0x61, 0x51,
0xBC, 0xB3, 0x61, 0xD1, 0xB6, 0xB9, 0x61,
/*2210:*/ 0x51, 0xB7, 0xCA, 0x61, 0x51, 0xBE, 0xD7, 0x61, 0xD1,
0xAC, 0xDF, 0x61, 0x51, 0xB4, 0xF7, 0x61,
/*2220:*/ 0xD3, 0xBE, 0xFA, 0x61, 0xD5, 0xBF, 0xFD, 0x61, 0xD5,
0xBD, 0x1A, 0x71, 0xD3, 0xAE, 0x38, 0x71,
/*2230:*/ 0xD3, 0xAE, 0x3F, 0x71, 0xD7, 0xBE, 0x5C, 0x71, 0x51,
0xB7, 0x6C, 0x71, 0x55, 0xAC, 0x76, 0x71,
/*2240:*/ 0x51, 0xAD, 0x76, 0x71, 0x53, 0xBF, 0x76, 0x71, 0xD5,
0xAC, 0x7E, 0x71, 0xD1, 0xBE, 0x7E, 0x71,
/*2250:*/ 0xD3, 0xBF, 0x7E, 0x71, 0xD1, 0xA4, 0x9F, 0x71, 0x51,
0xBF, 0xA8, 0x71, 0x55, 0xAE, 0xB0, 0x71,
/*2260:*/ 0xD1, 0xA4, 0xBD, 0x71, 0x51, 0xAF, 0xCC, 0x71, 0x51,
0xB4, 0xD1, 0x71, 0x51, 0xBE, 0xD3, 0x71,
/*2270:*/ 0x53, 0xAC, 0xF6, 0x71, 0xD1, 0xBE, 0xF9, 0x71, 0xD5,
0xB5, 0xFB, 0x71, 0xD7, 0xB7, 0x06, 0xA1,
/*2280:*/ 0xD1, 0xAD, 0x1B, 0xA1, 0xD1, 0xAD, 0x1C, 0xA1, 0xD5,
0xBE, 0x1C, 0xA1, 0x51, 0xAD, 0x31, 0xA1,
/*2290:*/ 0xD1, 0xA7, 0x3B, 0xA1, 0xD1, 0xB5, 0x3B, 0xA1, 0xD1,
0xAD, 0x42, 0xA1, 0x55, 0xA5, 0x50, 0xA1,
/*22A0:*/ 0x51, 0xAE, 0x52, 0xA1, 0x57, 0xA4, 0x57, 0xA1, 0xD5,
0xB6, 0x58, 0xA1, 0xD1, 0xAE, 0x5A, 0xA1,
/*22B0:*/ 0xD5, 0xAF, 0x5A, 0xA1, 0xD1, 0xAF, 0x5D, 0xA1, 0xD1,
0xBD, 0x5D, 0xA1, 0x53, 0xAE, 0x70, 0xA1,
/*22C0:*/ 0x55, 0xAF, 0x70, 0xA1, 0x55, 0xAE, 0x77, 0xA1, 0xD1,
0xAE, 0x78, 0xA1, 0xD3, 0xBD, 0x78, 0xA1,
/*22D0:*/ 0xD7, 0xA4, 0x7A, 0xA1, 0xD3, 0xB6, 0x7A, 0xA1, 0xD5,
0xB6, 0x7D, 0xA1, 0xD3, 0xBC, 0x7F, 0xA1,
/*22E0:*/ 0x55, 0xA7, 0x91, 0xA1, 0x55, 0xAD, 0x94, 0xA1, 0x51,
0xBC, 0xD5, 0xA1, 0x55, 0xA5, 0xD7, 0xA1,
/*22F0:*/ 0x51, 0xBF, 0xEF, 0xA1, 0x53, 0xBD, 0xF0, 0xA1, 0x55,
0xA4, 0xF2, 0xA1, 0x51, 0xB6, 0xF5, 0xA1,
/*2300:*/ 0xD5, 0xAF, 0xFF, 0xA1, 0xD1, 0xBC, 0x59, 0xB1, 0xD7,
0xBC, 0x59, 0xB1, 0xD1, 0xA4, 0x5B, 0xB1,
/*2310:*/ 0xD1, 0xB5, 0x61, 0xB1, 0x51, 0xAD, 0x6B, 0xB1, 0x53,
0xB7, 0x71, 0xB1, 0xD1, 0xAE, 0x7B, 0xB1,
/*2320:*/ 0xD3, 0xAF, 0x7C, 0xB1, 0x51, 0xB6, 0x8D, 0xB1, 0x55,
0xB5, 0x95, 0xB1, 0xD5, 0xA7, 0x9A, 0xB1,
/*2330:*/ 0xD5, 0xA6, 0x9D, 0xB1, 0x55, 0xAD, 0xB2, 0xB1, 0x55,
0xBF, 0xB5, 0xB1, 0x53, 0xAF, 0xD1, 0xB1,
/*2340:*/ 0x51, 0xB6, 0xD4, 0xB1, 0x53, 0xAF, 0xD6, 0xB1, 0xD3,
0xAF, 0xD9, 0xB1, 0xD7, 0xB7, 0xDC, 0xB1,
/*2350:*/ 0xD5, 0xAE, 0xDE, 0xB1, 0x51, 0xB5, 0xE9, 0xB1, 0xD5,
0xBF, 0x17, 0xE1, 0x55, 0xA7, 0x1D, 0xE1,
/*2360:*/ 0xD1, 0xBD, 0x2D, 0xE1, 0xD5, 0xB5, 0x30, 0xE1, 0xD3,
0xBC, 0x51, 0xE1, 0xD5, 0xA5, 0x54, 0xE1,
/*2370:*/ 0xD5, 0xAE, 0x56, 0xE1, 0xD1, 0xAF, 0x56, 0xE1, 0x53,
0xA4, 0x5B, 0xE1, 0x57, 0xB6, 0x5C, 0xE1,
/*2380:*/ 0xD1, 0xBD, 0x73, 0xE1, 0xD5, 0xB6, 0x76, 0xE1, 0xD7,
0xB6, 0x76, 0xE1, 0x51, 0xA5, 0x79, 0xE1,
/*2390:*/ 0x57, 0xAF, 0xA2, 0xE1, 0xD5, 0xBE, 0xB5, 0xE1, 0x51,
0xBE, 0xBD, 0xE1, 0xD1, 0xA6, 0xCB, 0xE1,
/*23A0:*/ 0xD1, 0xBF, 0xCE, 0xE1, 0xD1, 0xBC, 0xD1, 0xE1, 0xD3,
0xA4, 0xD4, 0xE1, 0x51, 0xBC, 0xD9, 0xE1,
/*23B0:*/ 0x53, 0xBD, 0xD9, 0xE1, 0x51, 0xBF, 0xE4, 0xE1, 0xD1,
0xA7, 0xE9, 0xE1, 0xD1, 0xA4, 0xF1, 0xE1,
/*23C0:*/ 0xD1, 0xAE, 0xF3, 0xE1, 0x53, 0xAF, 0xFC, 0xE1, 0x53,
0xA5, 0xFE, 0xE1, 0xD5, 0xAD, 0x13, 0xF1,
/*23D0:*/ 0xD3, 0xBE, 0x14, 0xF1, 0xD1, 0xBF, 0x36, 0xF1, 0xD3,
0xAF, 0x55, 0xF1, 0xD7, 0xB6, 0x57, 0xF1,
/*23E0:*/ 0x53, 0xBD, 0x5A, 0xF1, 0xD5, 0xA4, 0x75, 0xF1, 0x51,
0xBD, 0x78, 0xF1, 0x57, 0xA5, 0x7A, 0xF1,
/*23F0:*/ 0x53, 0xB7, 0x7A, 0xF1, 0x51, 0xB7, 0x81, 0xF1, 0x51,
0xAE, 0x84, 0xF1, 0x57, 0xA4, 0x86, 0xF1,
/*2400:*/ 0xD7, 0xBC, 0x8C, 0xF1, 0x53, 0xA7, 0xC7, 0xF1, 0xD3,
0xAD, 0xCD, 0xF1, 0xD5, 0xA4, 0xD7, 0xF1,
/*2410:*/ 0x53, 0xA5, 0xDF, 0xF1, 0x55, 0xA5, 0xDF, 0xF1, 0x51,
0xB5, 0xE2, 0xF1, 0xD1, 0xAD, 0xE8, 0xF1,
/*2420:*/ 0xD3, 0xBC, 0xF7, 0xF1, 0x0D, 0xF0, 0xAD, 0xBA, 0x0D,
0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*2430:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B3D2_B[] = {
/*2B30:*/ 0x0E, 0x18, 0x00, 0x92, 0x9C, 0x48, 0x00, 0x92, 0x8F,
0x60, 0x00, 0x92, 0x95, 0x00, 0x01, 0x92,
/*2B40:*/ 0x1A, 0x70, 0x08, 0x92, 0x97, 0x34, 0x09, 0x92, 0x1D,
0x78, 0x09, 0x92, 0x0C, 0x04, 0x21, 0x92,
/*2B50:*/ 0x93, 0x2C, 0x21, 0x92, 0x0C, 0x64, 0x21, 0x92, 0x04,
0x68, 0x21, 0x92, 0x14, 0x78, 0x21, 0x92,
/*2B60:*/ 0x8C, 0x20, 0x28, 0x92, 0x13, 0x58, 0x29, 0x92, 0x02,
0x5C, 0x29, 0x92, 0x0C, 0x7C, 0x29, 0x92,
/*2B70:*/ 0x0A, 0x18, 0x40, 0x92, 0x9A, 0x54, 0x41, 0x92, 0x82,
0x60, 0x48, 0x92, 0x83, 0x24, 0x49, 0x92,
/*2B80:*/ 0x9C, 0x18, 0x61, 0x92, 0x0E, 0x48, 0x61, 0x92, 0x0C,
0x64, 0x61, 0x92, 0x92, 0x10, 0x68, 0x92,
/*2B90:*/ 0x0B, 0x14, 0x68, 0x92, 0x16, 0x1C, 0x68, 0x92, 0x9C,
0x50, 0x68, 0x92, 0x9E, 0x7C, 0x68, 0x92,
/*2BA0:*/ 0x0A, 0x50, 0x69, 0x92, 0x86, 0x30, 0x89, 0x92, 0x0E,
0x28, 0xA1, 0x92, 0x11, 0x24, 0xA8, 0x92,
/*2BB0:*/ 0x96, 0x70, 0xA8, 0x92, 0x97, 0x1C, 0xC0, 0x92, 0x18,
0x44, 0xC0, 0x92, 0x13, 0x40, 0xC1, 0x92,
/*2BC0:*/ 0x0F, 0x38, 0x00, 0x93, 0x1A, 0x3C, 0x00, 0x93, 0x06,
0x40, 0x00, 0x93, 0x04, 0x3C, 0x01, 0x93,
/*2BD0:*/ 0x95, 0x54, 0x01, 0x93, 0x8D, 0x00, 0x08, 0x93, 0x96,
0x24, 0x08, 0x93, 0x8A, 0x20, 0x20, 0x93,
/*2BE0:*/ 0x83, 0x08, 0x21, 0x93, 0x9C, 0x2C, 0x21, 0x93, 0x82,
0x34, 0x28, 0x93, 0x15, 0x68, 0x40, 0x93,
/*2BF0:*/ 0x9A, 0x00, 0x41, 0x93, 0x1A, 0x24, 0x48, 0x93, 0x1A,
0x44, 0x48, 0x93, 0x04, 0x44, 0x49, 0x93,
/*2C00:*/ 0x93, 0x60, 0x49, 0x93, 0x82, 0x2C, 0x60, 0x93, 0x8E,
0x40, 0x60, 0x93, 0x1E, 0x0C, 0x61, 0x93,
/*2C10:*/ 0x17, 0x14, 0x61, 0x93, 0x9C, 0x2C, 0x61, 0x93, 0x94,
0x40, 0x61, 0x93, 0x12, 0x60, 0x61, 0x93,
/*2C20:*/ 0x8F, 0x64, 0x61, 0x93, 0x1E, 0x6C, 0x61, 0x93, 0x84,
0x18, 0x68, 0x93, 0x16, 0x48, 0x68, 0x93,
/*2C30:*/ 0x07, 0x1C, 0x69, 0x93, 0x87, 0x70, 0x69, 0x93, 0x1E,
0x5C, 0xA0, 0x93, 0x18, 0x40, 0xA1, 0x93,
/*2C40:*/ 0x9F, 0x24, 0xC0, 0x93, 0x0A, 0x7C, 0xC1, 0x93, 0x9A,
0x28, 0xC8, 0x93, 0x12, 0x48, 0xC8, 0x93,
/*2C50:*/ 0x0E, 0x1C, 0xE1, 0x93, 0x90, 0x40, 0xE1, 0x93, 0x96,
0x44, 0xE8, 0x93, 0x0E, 0x04, 0xE9, 0x93,
/*2C60:*/ 0x9F, 0x5C, 0x21, 0x97, 0x1E, 0x00, 0x00, 0xD2, 0x96,
0x60, 0x00, 0xD2, 0x8A, 0x4C, 0x01, 0xD2,
/*2C70:*/ 0x03, 0x58, 0x01, 0xD2, 0x16, 0x5C, 0x01, 0xD2, 0x95,
0x20, 0x08, 0xD2, 0x04, 0x28, 0x08, 0xD2,
/*2C80:*/ 0x85, 0x60, 0x09, 0xD2, 0x8D, 0x6C, 0x09, 0xD2, 0x12,
0x0C, 0x20, 0xD2, 0x12, 0x6C, 0x20, 0xD2,
/*2C90:*/ 0x8B, 0x20, 0x29, 0xD2, 0x82, 0x38, 0x29, 0xD2, 0x92,
0x48, 0x29, 0xD2, 0x0B, 0x4C, 0x29, 0xD2,
/*2CA0:*/ 0x8E, 0x34, 0x49, 0xD2, 0x9A, 0x6C, 0x60, 0xD2, 0x0F,
0x34, 0x61, 0xD2, 0x86, 0x08, 0x68, 0xD2,
/*2CB0:*/ 0x92, 0x18, 0x68, 0xD2, 0x1F, 0x3C, 0x69, 0xD2, 0x8E,
0x54, 0x89, 0xD2, 0x9C, 0x20, 0xA0, 0xD2,
/*2CC0:*/ 0x1E, 0x78, 0xA8, 0xD2, 0x9E, 0x24, 0xA9, 0xD2, 0x0B,
0x64, 0xC0, 0xD2, 0x9B, 0x60, 0xC8, 0xD2,
/*2CD0:*/ 0x93, 0x6C, 0xC8, 0xD2, 0x8B, 0x70, 0xC8, 0xD2, 0x9E,
0x3C, 0xE1, 0xD2, 0x06, 0x54, 0xE9, 0xD2,
/*2CE0:*/ 0x06, 0x28, 0x00, 0xD3, 0x12, 0x38, 0x00, 0xD3, 0x07,
0x3C, 0x00, 0xD3, 0x8E, 0x48, 0x00, 0xD3,
/*2CF0:*/ 0x92, 0x54, 0x00, 0xD3, 0x1E, 0x04, 0x01, 0xD3, 0x14,
0x24, 0x01, 0xD3, 0x04, 0x1C, 0x08, 0xD3,
/*2D00:*/ 0x1D, 0x44, 0x09, 0xD3, 0x9D, 0x48, 0x09, 0xD3, 0x8D,
0x58, 0x09, 0xD3, 0x12, 0x58, 0x20, 0xD3,
/*2D10:*/ 0x13, 0x1C, 0x21, 0xD3, 0x8C, 0x54, 0x21, 0xD3, 0x0C,
0x20, 0x29, 0xD3, 0x8D, 0x38, 0x29, 0xD3,
/*2D20:*/ 0x92, 0x7C, 0x29, 0xD3, 0x1E, 0x04, 0x41, 0xD3, 0x9A,
0x08, 0x41, 0xD3, 0x8A, 0x30, 0x48, 0xD3,
/*2D30:*/ 0x1A, 0x4C, 0x48, 0xD3, 0x96, 0x4C, 0x48, 0xD3, 0x02,
0x00, 0x49, 0xD3, 0x8B, 0x74, 0x49, 0xD3,
/*2D40:*/ 0x80, 0x08, 0x60, 0xD3, 0x13, 0x2C, 0x60, 0xD3, 0x86,
0x44, 0x60, 0xD3, 0x04, 0x54, 0x61, 0xD3,
/*2D50:*/ 0x8A, 0x30, 0x68, 0xD3, 0x1B, 0x38, 0x68, 0xD3, 0x1F,
0x08, 0x69, 0xD3, 0x86, 0x0C, 0x89, 0xD3,
/*2D60:*/ 0x16, 0x58, 0xA0, 0xD3, 0x82, 0x14, 0xA1, 0xD3, 0x10,
0x14, 0xC0, 0xD3, 0x06, 0x48, 0xC0, 0xD3,
/*2D70:*/ 0x1E, 0x04, 0xC1, 0xD3, 0x0E, 0x74, 0xC1, 0xD3, 0x17,
0x7C, 0xC1, 0xD3, 0x87, 0x48, 0xC8, 0xD3,
/*2D80:*/ 0x92, 0x4C, 0xC8, 0xD3, 0x96, 0x54, 0xE0, 0xD3, 0x16,
0x08, 0xE1, 0xD3, 0x0E, 0x14, 0xE1, 0xD3,
/*2D90:*/ 0x0E, 0x74, 0xE1, 0xD3, 0x0E, 0x6C, 0xE9, 0xD3, 0x8F,
0x58, 0x28, 0xD6, 0x87, 0x18, 0x20, 0xD7,
/*2DA0:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B3D2_C[] = {
/*3380:*/ 0xC0, 0xB2, 0xA1, 0x01, 0x43, 0xA3, 0xA2, 0x01, 0x66,
0xB2, 0xA3, 0x01, 0xC6, 0x22, 0xA7, 0x01,
/*3390:*/ 0xE0, 0x22, 0xAA, 0x01, 0x62, 0x3B, 0xAA, 0x01, 0x63,
0x8B, 0xAE, 0x01, 0xE7, 0x02, 0xA3, 0x11,
/*33A0:*/ 0xE3, 0x82, 0xA4, 0x11, 0x47, 0x82, 0xA6, 0x11, 0x41,
0xBB, 0xA6, 0x11, 0xE3, 0x8B, 0xAA, 0x11,
/*33B0:*/ 0xE3, 0x9A, 0xAB, 0x11, 0xC2, 0x1B, 0xAC, 0x11, 0x63,
0x2B, 0xAC, 0x11, 0xE3, 0x33, 0xAC, 0x11,
/*33C0:*/ 0xE2, 0x2B, 0xAE, 0x11, 0xE2, 0x3A, 0xAE, 0x11, 0x41,
0xA2, 0xA6, 0x21, 0xE0, 0x93, 0xA7, 0x21,
/*33D0:*/ 0xC2, 0x12, 0xAD, 0x21, 0xE0, 0x32, 0xAF, 0x21, 0xE2,
0x9A, 0xA2, 0x31, 0xE6, 0xA2, 0xA3, 0x31,
/*33E0:*/ 0xE7, 0x02, 0xA7, 0x31, 0x46, 0x32, 0xA7, 0x31, 0x47,
0x22, 0xAB, 0x31, 0xE1, 0x8A, 0xAE, 0x31,
/*33F0:*/ 0xC6, 0x8B, 0xAF, 0x31, 0xE7, 0x02, 0xA0, 0x41, 0x67,
0x0A, 0xA1, 0x41, 0x42, 0x1A, 0xA1, 0x41,
/*3400:*/ 0xE1, 0x2A, 0xA1, 0x41, 0x47, 0x3A, 0xA3, 0x41, 0x60,
0x8A, 0xAA, 0x41, 0xC1, 0xAA, 0xAB, 0x41,
/*3410:*/ 0x43, 0x1A, 0xAD, 0x41, 0xC7, 0x92, 0xA0, 0x51, 0x61,
0x82, 0xA2, 0x51, 0xC0, 0x1A, 0xA4, 0x51,
/*3420:*/ 0x43, 0x1A, 0xA6, 0x51, 0xE2, 0x3B, 0xA7, 0x51, 0xC7,
0x22, 0xA8, 0x51, 0xC1, 0x1B, 0xA9, 0x51,
/*3430:*/ 0xC0, 0x02, 0xAB, 0x51, 0xC2, 0x03, 0xAB, 0x51, 0x61,
0x9A, 0xAD, 0x51, 0x43, 0xAB, 0xAE, 0x51,
/*3440:*/ 0x62, 0x93, 0xAF, 0x51, 0x23, 0x12, 0xA1, 0x61, 0x46,
0x22, 0xA4, 0x61, 0x42, 0x1A, 0xA5, 0x61,
/*3450:*/ 0xE2, 0x3A, 0xA9, 0x61, 0x40, 0x12, 0xAA, 0x61, 0xC0,
0xB2, 0xAD, 0x61, 0xE2, 0x82, 0xAE, 0x61,
/*3460:*/ 0xE2, 0x83, 0xAE, 0x61, 0xE6, 0xAB, 0xAF, 0x61, 0x42,
0x12, 0xA1, 0x71, 0xE1, 0x22, 0xA1, 0x71,
/*3470:*/ 0xC3, 0x12, 0xA3, 0x71, 0x47, 0x32, 0xA3, 0x71, 0xE3,
0x8A, 0xA6, 0x71, 0x63, 0x93, 0xA6, 0x71,
/*3480:*/ 0x42, 0xAA, 0xA6, 0x71, 0x65, 0xA2, 0xA9, 0x71, 0xE2,
0x8B, 0xAA, 0x71, 0x60, 0x92, 0xAB, 0x71,
/*3490:*/ 0xC3, 0xB2, 0xAB, 0x71, 0x41, 0x13, 0xAC, 0x71, 0x60,
0x2A, 0xAD, 0x71, 0x02, 0x82, 0xAD, 0x71,
/*34A0:*/ 0x43, 0x82, 0xA2, 0x81, 0xC3, 0x8A, 0xA2, 0x81, 0xE0,
0xA2, 0xA2, 0x81, 0xC3, 0x8B, 0xA3, 0x81,
/*34B0:*/ 0xE7, 0x2A, 0xA7, 0x81, 0xC3, 0x3B, 0xAA, 0x81, 0xC0,
0x8A, 0xAE, 0x81, 0x61, 0xBB, 0xAE, 0x81,
/*34C0:*/ 0x63, 0x12, 0xA2, 0x91, 0xC0, 0x22, 0xA3, 0x91, 0x66,
0x9A, 0xA7, 0x91, 0x64, 0x83, 0xA8, 0x91,
/*34D0:*/ 0xE2, 0x1A, 0xAE, 0x91, 0xE6, 0x32, 0xAF, 0x91, 0xE3,
0x12, 0xA3, 0xA1, 0xC6, 0xAB, 0xA5, 0xA1,
/*34E0:*/ 0x41, 0x82, 0xA6, 0xA1, 0xC7, 0xA2, 0xA7, 0xA1, 0xE0,
0xBA, 0xA9, 0xA1, 0x40, 0x82, 0xAA, 0xA1,
/*34F0:*/ 0x67, 0x93, 0xAA, 0xA1, 0xE3, 0xA2, 0xAA, 0xA1, 0x46,
0xAB, 0xAB, 0xA1, 0x63, 0x12, 0xAD, 0xA1,
/*3500:*/ 0x60, 0x1A, 0xAE, 0xA1, 0xE0, 0x12, 0xAF, 0xA1, 0x66,
0x33, 0xAF, 0xA1, 0xC0, 0x8A, 0xA1, 0xB1,
/*3510:*/ 0xC3, 0x22, 0xAA, 0xB1, 0x60, 0xBA, 0xAD, 0xB1, 0xE3,
0xAA, 0xAE, 0xB1, 0x67, 0x3A, 0xA1, 0xC1,
/*3520:*/ 0x43, 0x22, 0xA2, 0xC1, 0xE0, 0x02, 0xA3, 0xC1, 0xC2,
0x8A, 0xA6, 0xC1, 0xE3, 0xA2, 0xA6, 0xC1,
/*3530:*/ 0xC0, 0x9A, 0xA7, 0xC1, 0xC6, 0xB2, 0xA7, 0xC1, 0xC6,
0xAA, 0xA8, 0xC1, 0xE7, 0x82, 0xA9, 0xC1,
/*3540:*/ 0xC0, 0x92, 0xA9, 0xC1, 0x62, 0xAB, 0xAA, 0xC1, 0x62,
0x12, 0xAD, 0xC1, 0xC1, 0x22, 0xAD, 0xC1,
/*3550:*/ 0x42, 0x23, 0xAF, 0xC1, 0x41, 0x92, 0xA1, 0xD1, 0x62,
0xBA, 0xA1, 0xD1, 0xC2, 0x92, 0xA3, 0xD1,
/*3560:*/ 0x42, 0x9B, 0xA3, 0xD1, 0xC5, 0x03, 0xA9, 0xD1, 0x61,
0x13, 0xAA, 0xD1, 0xC0, 0x22, 0xAA, 0xD1,
/*3570:*/ 0xE3, 0x1A, 0xAB, 0xD1, 0xC3, 0x82, 0xAE, 0xD1, 0x43,
0x8A, 0xAE, 0xD1, 0x43, 0x8B, 0xAE, 0xD1,
/*3580:*/ 0xE0, 0xBB, 0xAE, 0xD1, 0xE2, 0xAB, 0xAF, 0xD1, 0xE2,
0xBA, 0xAF, 0xD1, 0x62, 0xB3, 0xA0, 0xE1,
/*3590:*/ 0x23, 0x22, 0xA1, 0xE1, 0x42, 0x92, 0xA3, 0xE1, 0x60,
0x1A, 0xA6, 0xE1, 0x60, 0x0B, 0xA7, 0xE1,
/*35A0:*/ 0xC7, 0x12, 0xA7, 0xE1, 0x46, 0x1A, 0xAB, 0xE1, 0x61,
0xB2, 0xAD, 0xE1, 0xE2, 0xA2, 0xAE, 0xE1,
/*35B0:*/ 0x41, 0x3B, 0xA2, 0xF1, 0x41, 0x93, 0xA4, 0xF1, 0x42,
0x9A, 0xA6, 0xF1, 0xE2, 0xBB, 0xAA, 0xF1,
/*35C0:*/ 0xC1, 0x82, 0xAB, 0xF1, 0x43, 0x8B, 0xAB, 0xF1, 0x43,
0x9B, 0xAB, 0xF1, 0xE0, 0xBA, 0xAB, 0xF1,
/*35D0:*/ 0xC1, 0x3A, 0xAD, 0xF1, 0x61, 0x13, 0xAE, 0xF1, 0x0D,
0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*35E0:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B3D2_D[] = {
/*3BA0:*/ 0x75, 0x28, 0x09, 0x58, 0x24, 0x98, 0x0D, 0x58, 0x34,
0x3A, 0x49, 0x58, 0x2C, 0xB8, 0x49, 0x58,
/*3BB0:*/ 0x64, 0xBA, 0x6D, 0x58, 0x74, 0x92, 0x79, 0x58, 0x6C,
0x1A, 0x7D, 0x58, 0x35, 0x28, 0x7D, 0x58,
/*3BC0:*/ 0x3C, 0x10, 0x89, 0x58, 0x34, 0x92, 0x8D, 0x58, 0x24,
0x98, 0x8D, 0x58, 0x2C, 0x12, 0x99, 0x58,
/*3BD0:*/ 0x25, 0x2A, 0x99, 0x58, 0x7D, 0x88, 0x99, 0x58, 0x35,
0xF8, 0xAD, 0x58, 0x25, 0xAA, 0xB9, 0x58,
/*3BE0:*/ 0x75, 0x8A, 0xC9, 0x58, 0x25, 0x00, 0xE9, 0x58, 0x25,
0x2A, 0xED, 0x58, 0x7D, 0xA8, 0xED, 0x58,
/*3BF0:*/ 0x24, 0xE0, 0x19, 0x59, 0x74, 0x9A, 0x1D, 0x59, 0x6C,
0xC2, 0x3D, 0x59, 0x3D, 0x22, 0x49, 0x59,
/*3C00:*/ 0x6D, 0x7A, 0x59, 0x59, 0x25, 0xA8, 0x59, 0x59, 0x74,
0xB0, 0x99, 0x59, 0x64, 0xB0, 0x9D, 0x59,
/*3C10:*/ 0x3C, 0x92, 0xBD, 0x59, 0x34, 0x6A, 0xCD, 0x59, 0x75,
0xA0, 0xCD, 0x59, 0x2D, 0x00, 0xD9, 0x59,
/*3C20:*/ 0x64, 0x32, 0xDD, 0x59, 0x65, 0x82, 0xDD, 0x59, 0x35,
0x88, 0xDD, 0x59, 0x35, 0x70, 0xE9, 0x59,
/*3C30:*/ 0x2D, 0xF2, 0xE9, 0x59, 0x75, 0x58, 0xF9, 0x59, 0x74,
0x92, 0xF9, 0x59, 0x75, 0xE0, 0x09, 0x5A,
/*3C40:*/ 0x7C, 0x88, 0x0D, 0x5A, 0x65, 0x32, 0x3D, 0x5A, 0x65,
0x32, 0x49, 0x5A, 0x65, 0x38, 0x4D, 0x5A,
/*3C50:*/ 0x65, 0x3A, 0x59, 0x5A, 0x3D, 0x30, 0x69, 0x5A, 0x7C,
0x80, 0x69, 0x5A, 0x3C, 0xA0, 0x69, 0x5A,
/*3C60:*/ 0x65, 0x98, 0x6D, 0x5A, 0x3C, 0xAA, 0x6D, 0x5A, 0x6D,
0x38, 0x7D, 0x5A, 0x25, 0xA2, 0x85, 0x5A,
/*3C70:*/ 0x7D, 0x92, 0xA9, 0x5A, 0x2D, 0xB2, 0xAD, 0x5A, 0x24,
0x88, 0xB9, 0x5A, 0x3C, 0x00, 0xBD, 0x5A,
/*3C80:*/ 0x2D, 0xBA, 0xBD, 0x5A, 0x65, 0x32, 0xC9, 0x5A, 0x35,
0x32, 0xCD, 0x5A, 0x7D, 0x9A, 0xCD, 0x5A,
/*3C90:*/ 0x7C, 0x08, 0xD9, 0x5A, 0x7D, 0xE2, 0xD9, 0x5A, 0x65,
0x30, 0xDD, 0x5A, 0x2D, 0x98, 0xDD, 0x5A,
/*3CA0:*/ 0x7D, 0xB2, 0xDD, 0x5A, 0x6C, 0x80, 0xED, 0x5A, 0x7C,
0xA8, 0xF9, 0x5A, 0x7D, 0x68, 0xFD, 0x5A,
/*3CB0:*/ 0x75, 0x9A, 0xFD, 0x5A, 0x7C, 0xD8, 0xFD, 0x5A, 0x25,
0xC0, 0x0D, 0x5B, 0x7C, 0xB8, 0x15, 0x5B,
/*3CC0:*/ 0x24, 0xA2, 0x49, 0x5B, 0x6C, 0x20, 0x4D, 0x5B, 0x7D,
0xB2, 0x5D, 0x5B, 0x3D, 0x18, 0x79, 0x5B,
/*3CD0:*/ 0x2D, 0x18, 0x89, 0x5B, 0x35, 0x90, 0x8D, 0x5B, 0x65,
0x9A, 0x8D, 0x5B, 0x3D, 0xAA, 0xA1, 0x5B,
/*3CE0:*/ 0x7C, 0x78, 0xA9, 0x5B, 0x3D, 0xE8, 0xA9, 0x5B, 0x7D,
0xE8, 0xA9, 0x5B, 0x24, 0xF0, 0xAD, 0x5B,
/*3CF0:*/ 0x6D, 0xEA, 0xB9, 0x5B, 0x7C, 0x7A, 0xBD, 0x5B, 0x75,
0x30, 0xD9, 0x5B, 0x7C, 0xAA, 0xED, 0x5B,
/*3D00:*/ 0x2D, 0x68, 0xF9, 0x5B, 0x24, 0x00, 0xFD, 0x5B, 0x3D,
0x12, 0xFD, 0x5B, 0x65, 0xB0, 0xFD, 0x5B,
/*3D10:*/ 0x64, 0x0A, 0x39, 0x5C, 0x34, 0x2A, 0x49, 0x5C, 0x6D,
0x1A, 0x5D, 0x5C, 0x64, 0x58, 0x5D, 0x5C,
/*3D20:*/ 0x3C, 0x22, 0x69, 0x5C, 0x34, 0x82, 0x8D, 0x5C, 0x3D,
0xB8, 0x99, 0x5C, 0x2D, 0xB8, 0x9D, 0x5C,
/*3D30:*/ 0x34, 0x02, 0xAD, 0x5C, 0x24, 0x08, 0xAD, 0x5C, 0x2C,
0x82, 0xCD, 0x5C, 0x3C, 0x80, 0xDD, 0x5C,
/*3D40:*/ 0x75, 0x3A, 0xE9, 0x5C, 0x6C, 0x02, 0xED, 0x5C, 0x2C,
0x22, 0xED, 0x5C, 0x75, 0x32, 0xF9, 0x5C,
/*3D50:*/ 0x25, 0x68, 0xFD, 0x5C, 0x7C, 0x5A, 0x09, 0x5D, 0x2D,
0x32, 0x4D, 0x5D, 0x7D, 0x3A, 0x59, 0x5D,
/*3D60:*/ 0x3D, 0xB2, 0x69, 0x5D, 0x7D, 0xB2, 0x69, 0x5D, 0x24,
0xF2, 0x8D, 0x5D, 0x64, 0x20, 0xC9, 0x5D,
/*3D70:*/ 0x3C, 0x7A, 0xFD, 0x5D, 0x64, 0x92, 0x3D, 0x5E, 0x74,
0x18, 0x69, 0x5E, 0x75, 0xA0, 0x79, 0x5E,
/*3D80:*/ 0x6D, 0x28, 0x7D, 0x5E, 0x3D, 0x22, 0x89, 0x5E, 0x2D,
0x28, 0x89, 0x5E, 0x35, 0xAA, 0x89, 0x5E,
/*3D90:*/ 0x24, 0x1A, 0x8D, 0x5E, 0x7D, 0x0A, 0x99, 0x5E, 0x2C,
0x18, 0xA9, 0x5E, 0x7C, 0x32, 0xA9, 0x5E,
/*3DA0:*/ 0x6D, 0x80, 0xB9, 0x5E, 0x2C, 0x1A, 0xBD, 0x5E, 0x2D,
0xAA, 0xBD, 0x5E, 0x3C, 0x3A, 0xCD, 0x5E,
/*3DB0:*/ 0x35, 0x00, 0xD9, 0x5E, 0x3C, 0x62, 0xD9, 0x5E, 0x7D,
0x88, 0xD9, 0x5E, 0x2D, 0xF8, 0xD9, 0x5E,
/*3DC0:*/ 0x7C, 0x12, 0xDD, 0x5E, 0x2D, 0x88, 0xDD, 0x5E, 0x24,
0x90, 0xDD, 0x5E, 0x64, 0x12, 0xE9, 0x5E,
/*3DD0:*/ 0x65, 0xA8, 0xED, 0x5E, 0x3C, 0xE8, 0xFD, 0x5E, 0x2C,
0x6A, 0x4D, 0x5F, 0x34, 0xE8, 0x4D, 0x5F,
/*3DE0:*/ 0x3D, 0x08, 0x79, 0x5F, 0x35, 0xB2, 0x81, 0x5F, 0x7D,
0x70, 0x99, 0x5F, 0x65, 0x70, 0xAD, 0x5F,
/*3DF0:*/ 0x34, 0xE2, 0xC9, 0x5F, 0x3C, 0x62, 0xD9, 0x5F, 0x75,
0x2A, 0xDD, 0x5F, 0x74, 0xBA, 0xDD, 0x5F,
/*3E00:*/ 0x74, 0x32, 0xED, 0x5F, 0x65, 0xAA, 0xF9, 0x5F, 0x3C,
0xE2, 0xF9, 0x5F, 0x0D, 0xF0, 0xAD, 0xBA,
/*3E10:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};
```

```
BYTE B3D2_E[] = {
/*43F0:*/ 0x1C, 0x18, 0x22, 0x01, 0xBC, 0x1C, 0x3D, 0x01, 0x9C,
0x97, 0x70, 0x01, 0x3C, 0x1E, 0x76, 0x01,
/*4400:*/ 0x3C, 0x91, 0x76, 0x01, 0x1C, 0x9D, 0x79, 0x01, 0x3C,
0x18, 0x8E, 0x01, 0x9C, 0x1C, 0x90, 0x01,
/*4410:*/ 0x1C, 0x98, 0x99, 0x01, 0xBC, 0x9C, 0xB0, 0x01, 0x3C,
0x92, 0xB1, 0x01, 0x1C, 0x1A, 0xB6, 0x01,
/*4420:*/ 0x3C, 0x15, 0xC5, 0x01, 0x3C, 0x95, 0xC5, 0x01, 0x9C,
0x94, 0xD3, 0x01, 0x3C, 0x92, 0xD4, 0x01,
/*4430:*/ 0x3C, 0x18, 0xDD, 0x01, 0x3C, 0x12, 0xE2, 0x01, 0xBC,
0x1C, 0xE2, 0x01, 0x3C, 0x1D, 0xE3, 0x01,
/*4440:*/ 0x1C, 0x9D, 0xF5, 0x01, 0x3C, 0x11, 0xFA, 0x01, 0x3C,
0x99, 0x1E, 0x05, 0x1C, 0x14, 0x26, 0x05,
/*4450:*/ 0xBC, 0x18, 0x28, 0x05, 0x3C, 0x96, 0x29, 0x05, 0x3C,
0x15, 0x30, 0x05, 0x1C, 0x10, 0x4A, 0x05,
/*4460:*/ 0x3C, 0x19, 0x4C, 0x05, 0x3C, 0x11, 0x5C, 0x05, 0x3C,
0x10, 0x6A, 0x05, 0x9C, 0x10, 0x7D, 0x05,
/*4470:*/ 0x9C, 0x1C, 0x8C, 0x05, 0x3C, 0x99, 0x93, 0x05, 0x9C,
0x9E, 0x94, 0x05, 0xBC, 0x9D, 0x9A, 0x05,
/*4480:*/ 0x1C, 0x10, 0xA3, 0x05, 0x3C, 0x19, 0xA4, 0x05, 0xBC,
0x96, 0xA4, 0x05, 0x1C, 0x95, 0xAA, 0x05,
/*4490:*/ 0x1C, 0x18, 0xB3, 0x05, 0x3C, 0x11, 0xB4, 0x05, 0xBC,
0x90, 0xB5, 0x05, 0xBC, 0x14, 0xEF, 0x05,
/*44A0:*/ 0x1C, 0x94, 0xF9, 0x05, 0xBC, 0x1C, 0xFF, 0x05, 0xBC,
0x19, 0x11, 0x09, 0x1C, 0x14, 0x1E, 0x09,
/*44B0:*/ 0x1C, 0x95, 0x28, 0x09, 0xBC, 0x1D, 0x2F, 0x09, 0x3C,
0x9A, 0x3F, 0x09, 0x3C, 0x96, 0x43, 0x09,
/*44C0:*/ 0x9C, 0x1D, 0x6B, 0x09, 0x9C, 0x98, 0x8A, 0x09, 0x1C,
0x15, 0x92, 0x09, 0x3C, 0x9C, 0x95, 0x09,
/*44D0:*/ 0x9C, 0x90, 0x9B, 0x09, 0xBC, 0x1D, 0xA3, 0x09, 0x3C,
0x14, 0xB3, 0x09, 0x3C, 0x10, 0xBB, 0x09,
/*44E0:*/ 0x3C, 0x9D, 0xC6, 0x09, 0x3C, 0x96, 0xCE, 0x09, 0x3C,
0x9E, 0xDE, 0x09, 0x1C, 0x9E, 0xFE, 0x09,
/*44F0:*/ 0x1C, 0x14, 0x05, 0x0C, 0x1C, 0x18, 0x1A, 0x0D, 0x1C,
0x16, 0x1B, 0x0D, 0x3C, 0x91, 0x78, 0x0D,
/*4500:*/ 0x1C, 0x10, 0x87, 0x0D, 0x9C, 0x14, 0x8E, 0x0D, 0x9C,
0x94, 0x8E, 0x0D, 0x9C, 0x9C, 0x9E, 0x0D,
/*4510:*/ 0x1C, 0x96, 0xA1, 0x0D, 0x3C, 0x10, 0xA6, 0x0D, 0x1C,
0x1C, 0xA8, 0x0D, 0x9C, 0x1C, 0xA9, 0x0D,
/*4520:*/ 0x3C, 0x9C, 0xBF, 0x0D, 0xBC, 0x11, 0xC3, 0x0D, 0x1C,
0x98, 0xC5, 0x0D, 0x1C, 0x99, 0xC5, 0x0D,
/*4530:*/ 0xBC, 0x19, 0xD3, 0x0D, 0x1C, 0x9A, 0xDD, 0x0D, 0x9C,
0x11, 0xE3, 0x0D, 0x3C, 0x96, 0xE4, 0x0D,
/*4540:*/ 0xBC, 0x15, 0xFD, 0x0D, 0xBC, 0x10, 0x0D, 0x81, 0x3C,
0x1D, 0x14, 0x81, 0x3C, 0x16, 0x4F, 0x81,
/*4550:*/ 0x1C, 0x96, 0x58, 0x81, 0x9C, 0x97, 0x58, 0x81, 0x9C,
0x98, 0x86, 0x81, 0xBC, 0x19, 0x90, 0x81,
/*4560:*/ 0x3C, 0x96, 0xA7, 0x81, 0x3C, 0x98, 0xA7, 0x81, 0xBC,
0x98, 0xA7, 0x81, 0x3C, 0x1D, 0xAF, 0x81,
/*4570:*/ 0x1C, 0x18, 0xB0, 0x81, 0x1C, 0x19, 0xB1, 0x81, 0x3C,
0x1E, 0xB6, 0x81, 0x3C, 0x95, 0xBF, 0x81,
/*4580:*/ 0x3C, 0x98, 0xC2, 0x81, 0xBC, 0x18, 0xC3, 0x81, 0x9C,
0x14, 0xCC, 0x81, 0x3C, 0x91, 0xD3, 0x81,
/*4590:*/ 0xBC, 0x14, 0xDB, 0x81, 0x3C, 0x14, 0xEC, 0x81, 0x1C,
0x91, 0x07, 0x85, 0x3C, 0x11, 0x10, 0x85,
/*45A0:*/ 0x1C, 0x99, 0x16, 0x85, 0xBC, 0x14, 0x2F, 0x85, 0x1C,
0x15, 0x82, 0x85, 0x1C, 0x9C, 0xA5, 0x85,
/*45B0:*/ 0x9C, 0x14, 0xB4, 0x85, 0x9C, 0x94, 0xB5, 0x85, 0x3C,
0x99, 0xBB, 0x85, 0x1C, 0x10, 0xBD, 0x85,
/*45C0:*/ 0x9C, 0x11, 0xBD, 0x85, 0x1C, 0x12, 0xC0, 0x85, 0x9C,
0x9C, 0xC1, 0x85, 0x1C, 0x16, 0xC8, 0x85,
/*45D0:*/ 0x3C, 0x10, 0xCE, 0x85, 0xBC, 0x93, 0xD6, 0x85, 0x3C,
0x16, 0xDE, 0x85, 0x1C, 0x1E, 0xEE, 0x85,
/*45E0:*/ 0x1C, 0x9C, 0xF7, 0x85, 0xBC, 0x98, 0x6D, 0x88, 0x1C,
0x9A, 0x37, 0x89, 0x1C, 0x91, 0x3F, 0x89,
/*45F0:*/ 0x3C, 0x1D, 0x54, 0x89, 0x1C, 0x15, 0x64, 0x89, 0x3C,
0x9A, 0x73, 0x89, 0x3C, 0x91, 0x7A, 0x89,
/*4600:*/ 0x9C, 0x9A, 0x8C, 0x89, 0x9C, 0x19, 0xA3, 0x89, 0x3C,
0x11, 0xA4, 0x89, 0x3C, 0x90, 0xA5, 0x89,
/*4610:*/ 0x3C, 0x11, 0xC0, 0x89, 0x1C, 0x99, 0xC7, 0x89, 0x3C,
0x14, 0xC8, 0x89, 0x1C, 0x10, 0xD7, 0x89,
/*4620:*/ 0x3C, 0x18, 0xE6, 0x89, 0xBC, 0x18, 0xE6, 0x89, 0x1C,
0x14, 0xE8, 0x89, 0xBC, 0x9D, 0xEE, 0x89,
/*4630:*/ 0x1C, 0x94, 0x7E, 0x8C, 0x3C, 0x9E, 0x02, 0x8D, 0x1C,
0x15, 0x2B, 0x8D, 0x3C, 0x19, 0x40, 0x8D,
/*4640:*/ 0x1C, 0x1C, 0x5E, 0x8D, 0x9C, 0x1C, 0x80, 0x8D, 0x1C,
0x92, 0x81, 0x8D, 0x9C, 0x18, 0x89, 0x8D,
/*4650:*/ 0x9C, 0x15, 0x91, 0x8D, 0x3C, 0x9C, 0x97, 0x8D, 0x9C,
0x14, 0xA6, 0x8D, 0x9C, 0x94, 0xA6, 0x8D,
/*4660:*/ 0x1C, 0x10, 0xAE, 0x8D, 0x9C, 0x10, 0xAF, 0x8D, 0x9C,
0x1C, 0xB7, 0x8D, 0x3C, 0x9E, 0xB8, 0x8D,
/*4670:*/ 0x9C, 0x9C, 0xD2, 0x8D, 0x3C, 0x10, 0xDC, 0x8D, 0xBC,
0x9C, 0xF2, 0x8D, 0x0D, 0xF0, 0xAD, 0xBA,
/*4680:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
/*43F0:*/ 0x1C, 0x18, 0x22, 0x01, 0xBC, 0x1C, 0x3D, 0x01, 0x9C,
0x97, 0x70, 0x01, 0x3C, 0x1E, 0x76, 0x01,
/*4400:*/ 0x3C, 0x91, 0x76, 0x01, 0x1C, 0x9D, 0x79, 0x01, 0x3C,
0x18, 0x8E, 0x01, 0x9C, 0x1C, 0x90, 0x01,
/*4410:*/ 0x1C, 0x98, 0x99, 0x01, 0xBC, 0x9C, 0xB0, 0x01, 0x3C,
0x92, 0xB1, 0x01, 0x1C, 0x1A, 0xB6, 0x01,
/*4420:*/ 0x3C, 0x15, 0xC5, 0x01, 0x3C, 0x95, 0xC5, 0x01, 0x9C,
0x94, 0xD3, 0x01, 0x3C, 0x92, 0xD4, 0x01,
/*4430:*/ 0x3C, 0x18, 0xDD, 0x01, 0x3C, 0x12, 0xE2, 0x01, 0xBC,
0x1C, 0xE2, 0x01, 0x3C, 0x1D, 0xE3, 0x01,
/*4440:*/ 0x1C, 0x9D, 0xF5, 0x01, 0x3C, 0x11, 0xFA, 0x01, 0x3C,
0x99, 0x1E, 0x05, 0x1C, 0x14, 0x26, 0x05,
/*4450:*/ 0xBC, 0x18, 0x28, 0x05, 0x3C, 0x96, 0x29, 0x05, 0x3C,
0x15, 0x30, 0x05, 0x1C, 0x10, 0x4A, 0x05,
/*4460:*/ 0x3C, 0x19, 0x4C, 0x05, 0x3C, 0x11, 0x5C, 0x05, 0x3C,
0x10, 0x6A, 0x05, 0x9C, 0x10, 0x7D, 0x05,
/*4470:*/ 0x9C, 0x1C, 0x8C, 0x05, 0x3C, 0x99, 0x93, 0x05, 0x9C,
0x9E, 0x94, 0x05, 0xBC, 0x9D, 0x9A, 0x05,
/*4480:*/ 0x1C, 0x10, 0xA3, 0x05, 0x3C, 0x19, 0xA4, 0x05, 0xBC,
0x96, 0xA4, 0x05, 0x1C, 0x95, 0xAA, 0x05,
/*4490:*/ 0x1C, 0x18, 0xB3, 0x05, 0x3C, 0x11, 0xB4, 0x05, 0xBC,
0x90, 0xB5, 0x05, 0xBC, 0x14, 0xEF, 0x05,
/*44A0:*/ 0x1C, 0x94, 0xF9, 0x05, 0xBC, 0x1C, 0xFF, 0x05, 0xBC,
0x19, 0x11, 0x09, 0x1C, 0x14, 0x1E, 0x09,
/*44B0:*/ 0x1C, 0x95, 0x28, 0x09, 0xBC, 0x1D, 0x2F, 0x09, 0x3C,
0x9A, 0x3F, 0x09, 0x3C, 0x96, 0x43, 0x09,
/*44C0:*/ 0x9C, 0x1D, 0x6B, 0x09, 0x9C, 0x98, 0x8A, 0x09, 0x1C,
0x15, 0x92, 0x09, 0x3C, 0x9C, 0x95, 0x09,
/*44D0:*/ 0x9C, 0x90, 0x9B, 0x09, 0xBC, 0x1D, 0xA3, 0x09, 0x3C,
0x14, 0xB3, 0x09, 0x3C, 0x10, 0xBB, 0x09,
/*44E0:*/ 0x3C, 0x9D, 0xC6, 0x09, 0x3C, 0x96, 0xCE, 0x09, 0x3C,
0x9E, 0xDE, 0x09, 0x1C, 0x9E, 0xFE, 0x09,
/*44F0:*/ 0x1C, 0x14, 0x05, 0x0C, 0x1C, 0x18, 0x1A, 0x0D, 0x1C,
0x16, 0x1B, 0x0D, 0x3C, 0x91, 0x78, 0x0D,
/*4500:*/ 0x1C, 0x10, 0x87, 0x0D, 0x9C, 0x14, 0x8E, 0x0D, 0x9C,
0x94, 0x8E, 0x0D, 0x9C, 0x9C, 0x9E, 0x0D,
/*4510:*/ 0x1C, 0x96, 0xA1, 0x0D, 0x3C, 0x10, 0xA6, 0x0D, 0x1C,
0x1C, 0xA8, 0x0D, 0x9C, 0x1C, 0xA9, 0x0D,
/*4520:*/ 0x3C, 0x9C, 0xBF, 0x0D, 0xBC, 0x11, 0xC3, 0x0D, 0x1C,
0x98, 0xC5, 0x0D, 0x1C, 0x99, 0xC5, 0x0D,
/*4530:*/ 0xBC, 0x19, 0xD3, 0x0D, 0x1C, 0x9A, 0xDD, 0x0D, 0x9C,
0x11, 0xE3, 0x0D, 0x3C, 0x96, 0xE4, 0x0D,
/*4540:*/ 0xBC, 0x15, 0xFD, 0x0D, 0xBC, 0x10, 0x0D, 0x81, 0x3C,
0x1D, 0x14, 0x81, 0x3C, 0x16, 0x4F, 0x81,
/*4550:*/ 0x1C, 0x96, 0x58, 0x81, 0x9C, 0x97, 0x58, 0x81, 0x9C,
0x98, 0x86, 0x81, 0xBC, 0x19, 0x90, 0x81,
/*4560:*/ 0x3C, 0x96, 0xA7, 0x81, 0x3C, 0x98, 0xA7, 0x81, 0xBC,
0x98, 0xA7, 0x81, 0x3C, 0x1D, 0xAF, 0x81,
/*4570:*/ 0x1C, 0x18, 0xB0, 0x81, 0x1C, 0x19, 0xB1, 0x81, 0x3C,
0x1E, 0xB6, 0x81, 0x3C, 0x95, 0xBF, 0x81,
/*4580:*/ 0x3C, 0x98, 0xC2, 0x81, 0xBC, 0x18, 0xC3, 0x81, 0x9C,
0x14, 0xCC, 0x81, 0x3C, 0x91, 0xD3, 0x81,
/*4590:*/ 0xBC, 0x14, 0xDB, 0x81, 0x3C, 0x14, 0xEC, 0x81, 0x1C,
0x91, 0x07, 0x85, 0x3C, 0x11, 0x10, 0x85,
/*45A0:*/ 0x1C, 0x99, 0x16, 0x85, 0xBC, 0x14, 0x2F, 0x85, 0x1C,
0x15, 0x82, 0x85, 0x1C, 0x9C, 0xA5, 0x85,
/*45B0:*/ 0x9C, 0x14, 0xB4, 0x85, 0x9C, 0x94, 0xB5, 0x85, 0x3C,
0x99, 0xBB, 0x85, 0x1C, 0x10, 0xBD, 0x85,
/*45C0:*/ 0x9C, 0x11, 0xBD, 0x85, 0x1C, 0x12, 0xC0, 0x85, 0x9C,
0x9C, 0xC1, 0x85, 0x1C, 0x16, 0xC8, 0x85,
/*45D0:*/ 0x3C, 0x10, 0xCE, 0x85, 0xBC, 0x93, 0xD6, 0x85, 0x3C,
0x16, 0xDE, 0x85, 0x1C, 0x1E, 0xEE, 0x85,
/*45E0:*/ 0x1C, 0x9C, 0xF7, 0x85, 0xBC, 0x98, 0x6D, 0x88, 0x1C,
0x9A, 0x37, 0x89, 0x1C, 0x91, 0x3F, 0x89,
/*45F0:*/ 0x3C, 0x1D, 0x54, 0x89, 0x1C, 0x15, 0x64, 0x89, 0x3C,
0x9A, 0x73, 0x89, 0x3C, 0x91, 0x7A, 0x89,
/*4600:*/ 0x9C, 0x9A, 0x8C, 0x89, 0x9C, 0x19, 0xA3, 0x89, 0x3C,
0x11, 0xA4, 0x89, 0x3C, 0x90, 0xA5, 0x89,
/*4610:*/ 0x3C, 0x11, 0xC0, 0x89, 0x1C, 0x99, 0xC7, 0x89, 0x3C,
0x14, 0xC8, 0x89, 0x1C, 0x10, 0xD7, 0x89,
/*4620:*/ 0x3C, 0x18, 0xE6, 0x89, 0xBC, 0x18, 0xE6, 0x89, 0x1C,
0x14, 0xE8, 0x89, 0xBC, 0x9D, 0xEE, 0x89,
/*4630:*/ 0x1C, 0x94, 0x7E, 0x8C, 0x3C, 0x9E, 0x02, 0x8D, 0x1C,
0x15, 0x2B, 0x8D, 0x3C, 0x19, 0x40, 0x8D,
/*4640:*/ 0x1C, 0x1C, 0x5E, 0x8D, 0x9C, 0x1C, 0x80, 0x8D, 0x1C,
0x92, 0x81, 0x8D, 0x9C, 0x18, 0x89, 0x8D,
/*4650:*/ 0x9C, 0x15, 0x91, 0x8D, 0x3C, 0x9C, 0x97, 0x8D, 0x9C,
0x14, 0xA6, 0x8D, 0x9C, 0x94, 0xA6, 0x8D,
/*4660:*/ 0x1C, 0x10, 0xAE, 0x8D, 0x9C, 0x10, 0xAF, 0x8D, 0x9C,
0x1C, 0xB7, 0x8D, 0x3C, 0x9E, 0xB8, 0x8D,
/*4670:*/ 0x9C, 0x9C, 0xD2, 0x8D, 0x3C, 0x10, 0xDC, 0x8D, 0xBC,
0x9C, 0xF2, 0x8D, 0x0D, 0xF0, 0xAD, 0xBA,
/*4680:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B3D2_F[] = {
/*4CB0:*/ 0x65, 0x80, 0x04, 0x48, 0x65, 0x09, 0x05, 0x48, 0x4F,
0x8B, 0x06, 0x48, 0x67, 0x0E, 0x09, 0x48,
/*4CC0:*/ 0x4D, 0x07, 0x80, 0x48, 0x4F, 0x88, 0x82, 0x48, 0x6C,
0x84, 0x83, 0x48, 0x4C, 0x02, 0x84, 0x48,
/*4CD0:*/ 0x4D, 0x8F, 0x86, 0x48, 0x67, 0x86, 0x8B, 0x48, 0x6E,
0x81, 0x8C, 0x48, 0x6C, 0x85, 0x8C, 0x48,
/*4CE0:*/ 0x67, 0x0E, 0x8D, 0x48, 0x65, 0x0B, 0x8E, 0x48, 0x4F,
0x0A, 0x00, 0x4A, 0x4D, 0x87, 0x05, 0x4A,
/*4CF0:*/ 0x4C, 0x0A, 0x87, 0x4A, 0x4F, 0x87, 0x01, 0x58, 0x6F,
0x02, 0x02, 0x58, 0x6D, 0x04, 0x05, 0x58,
/*4D00:*/ 0x6D, 0x8D, 0x08, 0x58, 0x6D, 0x07, 0x09, 0x58, 0x6F,
0x03, 0x0D, 0x58, 0x65, 0x07, 0x81, 0x58,
/*4D10:*/ 0x4E, 0x80, 0x86, 0x58, 0x65, 0x8E, 0x88, 0x58, 0x6F,
0x8B, 0x8B, 0x58, 0x4F, 0x86, 0x8E, 0x58,
/*4D20:*/ 0x4F, 0x8C, 0x02, 0x5A, 0x6F, 0x0A, 0x05, 0x5A, 0x6D,
0x0C, 0x06, 0x5A, 0x6F, 0x08, 0x0E, 0x5A,
/*4D30:*/ 0x4C, 0x8E, 0x86, 0x5A, 0x4D, 0x09, 0x01, 0x68, 0x6F,
0x88, 0x02, 0x68, 0x6F, 0x8B, 0x02, 0x68,
/*4D40:*/ 0x4D, 0x80, 0x0C, 0x68, 0x6D, 0x8F, 0x0E, 0x68, 0x4D,
0x0A, 0x81, 0x68, 0x4F, 0x0F, 0x82, 0x68,
/*4D50:*/ 0x6F, 0x02, 0x83, 0x68, 0x4F, 0x86, 0x83, 0x68, 0x6C,
0x02, 0x84, 0x68, 0x44, 0x0D, 0x86, 0x68,
/*4D60:*/ 0x64, 0x89, 0x86, 0x68, 0x6F, 0x89, 0x89, 0x68, 0x65,
0x04, 0x8C, 0x68, 0x4D, 0x08, 0x8E, 0x68,
/*4D70:*/ 0x65, 0x8C, 0x8E, 0x68, 0x6D, 0x84, 0x01, 0x6A, 0x46,
0x88, 0x04, 0x6A, 0x6F, 0x80, 0x0D, 0x6A,
/*4D80:*/ 0x4F, 0x05, 0x0E, 0x6A, 0x4F, 0x06, 0x0E, 0x6A, 0x47,
0x8F, 0x0F, 0x6A, 0x4D, 0x8B, 0x83, 0x6A,
/*4D90:*/ 0x4E, 0x89, 0x87, 0x6A, 0x45, 0x88, 0x8B, 0x6A, 0x4D,
0x0F, 0x84, 0x6E, 0x6D, 0x08, 0x03, 0x78,
/*4DA0:*/ 0x4D, 0x8C, 0x03, 0x78, 0x6D, 0x08, 0x0B, 0x78, 0x6F,
0x84, 0x0D, 0x78, 0x6F, 0x87, 0x0D, 0x78,
/*4DB0:*/ 0x67, 0x86, 0x0E, 0x78, 0x6F, 0x0C, 0x0F, 0x78, 0x4F,
0x88, 0x0F, 0x78, 0x4D, 0x8F, 0x0F, 0x78,
/*4DC0:*/ 0x47, 0x8A, 0x80, 0x78, 0x6E, 0x80, 0x82, 0x78, 0x67,
0x0E, 0x84, 0x78, 0x4D, 0x05, 0x86, 0x78,
/*4DD0:*/ 0x6C, 0x0D, 0x87, 0x78, 0x4D, 0x8D, 0x88, 0x78, 0x6D,
0x81, 0x8A, 0x78, 0x45, 0x8C, 0x8B, 0x78,
/*4DE0:*/ 0x4F, 0x02, 0x8E, 0x78, 0x4F, 0x0B, 0x02, 0x7A, 0x64,
0x8F, 0x05, 0x7A, 0x6F, 0x04, 0x8C, 0x7A,
/*4DF0:*/ 0x4F, 0x80, 0x8C, 0x7A, 0x6D, 0x0A, 0x08, 0xC8, 0x4F,
0x02, 0x0A, 0xC8, 0x6D, 0x0B, 0x0B, 0xC8,
/*4E00:*/ 0x6F, 0x0D, 0x0C, 0xC8, 0x6F, 0x0E, 0x0C, 0xC8, 0x4F,
0x00, 0x0D, 0xC8, 0x4F, 0x03, 0x0D, 0xC8,
/*4E10:*/ 0x4F, 0x89, 0x80, 0xC8, 0x4F, 0x88, 0x83, 0xC8, 0x6E,
0x0B, 0x84, 0xC8, 0x4F, 0x8A, 0x84, 0xC8,
/*4E20:*/ 0x6C, 0x86, 0x85, 0xC8, 0x65, 0x0A, 0x88, 0xC8, 0x6F,
0x87, 0x89, 0xC8, 0x65, 0x09, 0x8C, 0xC8,
/*4E30:*/ 0x6D, 0x0A, 0x8C, 0xC8, 0x4E, 0x8F, 0x8C, 0xC8, 0x47,
0x03, 0x8D, 0xC8, 0x6E, 0x81, 0x8D, 0xC8,
/*4E40:*/ 0x6D, 0x81, 0x8E, 0xC8, 0x4F, 0x80, 0x04, 0xCA, 0x6D,
0x01, 0x0F, 0xCA, 0x4E, 0x0C, 0x81, 0xCA,
/*4E50:*/ 0x67, 0x04, 0x88, 0xCA, 0x4D, 0x0A, 0x01, 0xD8, 0x6F,
0x88, 0x02, 0xD8, 0x6D, 0x06, 0x03, 0xD8,
/*4E60:*/ 0x47, 0x0D, 0x05, 0xD8, 0x4F, 0x0E, 0x05, 0xD8, 0x45,
0x81, 0x0B, 0xD8, 0x4E, 0x80, 0x0F, 0xD8,
/*4E70:*/ 0x6E, 0x04, 0x83, 0xD8, 0x4E, 0x0B, 0x85, 0xD8, 0x6E,
0x8F, 0x85, 0xD8, 0x65, 0x04, 0x8C, 0xD8,
/*4E80:*/ 0x6F, 0x8A, 0x8D, 0xD8, 0x4F, 0x8E, 0x04, 0xDA, 0x4D,
0x8A, 0x08, 0xDA, 0x47, 0x8C, 0x0F, 0xDA,
/*4E90:*/ 0x4E, 0x01, 0x81, 0xDA, 0x47, 0x05, 0x8A, 0xDA, 0x6D,
0x0E, 0x8C, 0xDA, 0x4F, 0x87, 0x01, 0xE8,
/*4EA0:*/ 0x45, 0x0A, 0x04, 0xE8, 0x4D, 0x0A, 0x08, 0xE8, 0x4D,
0x80, 0x09, 0xE8, 0x66, 0x07, 0x0A, 0xE8,
/*4EB0:*/ 0x45, 0x81, 0x0A, 0xE8, 0x65, 0x8C, 0x0B, 0xE8, 0x4D,
0x0B, 0x0F, 0xE8, 0x4F, 0x0D, 0x80, 0xE8,
/*4EC0:*/ 0x6E, 0x8C, 0x80, 0xE8, 0x6F, 0x00, 0x81, 0xE8, 0x45,
0x81, 0x82, 0xE8, 0x4F, 0x86, 0x82, 0xE8,
/*4ED0:*/ 0x4C, 0x0C, 0x84, 0xE8, 0x45, 0x80, 0x85, 0xE8, 0x45,
0x83, 0x85, 0xE8, 0x67, 0x8A, 0x8C, 0xE8,
/*4EE0:*/ 0x6E, 0x06, 0x8D, 0xE8, 0x45, 0x83, 0x8D, 0xE8, 0x4D,
0x00, 0x04, 0xEA, 0x6D, 0x87, 0x04, 0xEA,
/*4EF0:*/ 0x4F, 0x8D, 0x05, 0xEA, 0x4D, 0x01, 0x0F, 0xEA, 0x4E,
0x8B, 0x81, 0xEA, 0x65, 0x0C, 0x8A, 0xEA,
/*4F00:*/ 0x65, 0x86, 0x8B, 0xEA, 0x65, 0x80, 0x04, 0xF8, 0x65,
0x83, 0x04, 0xF8, 0x4F, 0x8A, 0x0D, 0xF8,
/*4F10:*/ 0x65, 0x09, 0x81, 0xF8, 0x45, 0x8E, 0x81, 0xF8, 0x65,
0x81, 0x83, 0xF8, 0x65, 0x82, 0x83, 0xF8,
/*4F20:*/ 0x67, 0x85, 0x83, 0xF8, 0x44, 0x02, 0x84, 0xF8, 0x67,
0x0D, 0x85, 0xF8, 0x45, 0x8D, 0x85, 0xF8,
/*4F30:*/ 0x4E, 0x8D, 0x86, 0xF8, 0x6C, 0x84, 0x87, 0xF8, 0x4D,
0x8C, 0x8A, 0xF8, 0x4D, 0x8F, 0x8A, 0xF8,
/*4F40:*/ 0x45, 0x05, 0x8B, 0xF8, 0x6D, 0x89, 0x04, 0xFA, 0x65,
0x8A, 0x08, 0xFA, 0x66, 0x02, 0x09, 0xFA,
/*4F50:*/ 0x47, 0x08, 0x0F, 0xFA, 0x6D, 0x89, 0x8C, 0xFA, 0x6F,
0x08, 0x84, 0xFE, 0x0D, 0xF0, 0xAD, 0xBA,
/*4F60:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE Block4Data1[] = {
0x91, 0x00, 0x00, 0x00,
0xB9, 0x00, 0x00, 0x00,
0xA3, 0x00, 0x00, 0x00,
0x9D, 0x00, 0x00, 0x00,
0xA4, 0x00, 0x00, 0x00,
0x91, 0x00, 0x00, 0x00,
0x8B, 0x00, 0x00, 0x00,
0xAD, 0x00, 0x00, 0x00,
0x94, 0x00, 0x00, 0x00,
0xC2, 0x00, 0x00, 0x00,
0xB9, 0x00, 0x00, 0x00,
0x9C, 0x00, 0x00, 0x00,
0x96, 0x00, 0x00, 0x00,
0x9B, 0x00, 0x00, 0x00,
0xA3, 0x00, 0x00, 0x00,
0xAB, 0x00, 0x00, 0x00,
};

void* Block4Data2[] = {
     B3D2_0,
     B3D2_1,
     B3D2_2,
     B3D2_3,
     B3D2_4,
     B3D2_5,
     B3D2_6,
     B3D2_7,
     B3D2_8,
     B3D2_9,
     B3D2_A,
     B3D2_B,
     B3D2_C,
     B3D2_D,
     B3D2_E,
     B3D2_F
};

//////////////////////
// block 6
//////////////////////

BYTE B6D2_0[] = {
/*00CC60:*/ 0xA8, 0x21, 0x62, 0x44, 0x70, 0xF3, 0x21, 0x92, 0x7B,
0xF8, 0xDA, 0x1E, 0xB2, 0x0E, 0xC0, 0x25,
/*00CC70:*/ 0x3B, 0x22, 0x07, 0x87, 0x7B, 0x07, 0x82, 0x94, 0x63,
0xD9, 0xBE, 0x5C, 0x0C, 0x31, 0xB7, 0x7D,
/*00CC80:*/ 0x1E, 0xE6, 0x3B, 0xD2, 0x03, 0x02, 0xB4, 0x2D, 0x38,
0x3C, 0x18, 0x72, 0x6E, 0x35, 0xFA, 0x9F,
/*00CC90:*/ 0xAB, 0xD1, 0xD3, 0x0E, 0xE0, 0x58, 0x7A, 0x6F, 0x5D,
0x65, 0xB5, 0x0A, 0x72, 0x38, 0xA6, 0x78,
/*00CCA0:*/ 0xC8, 0xF7, 0x44, 0xFE, 0xB5, 0x99, 0x8F, 0xB7, 0x2D,
0x4B, 0xA5, 0x82, 0xE6, 0xB0, 0xC1, 0x6B,
/*00CCB0:*/ 0xDE, 0x02, 0x1C, 0xBD, 0x34, 0x33, 0xF7, 0x21, 0x2D,
0x4B, 0xA5, 0x82, 0x8E, 0xB3, 0xD6, 0xCA,
/*00CCC0:*/ 0x6E, 0xA1, 0xEE, 0x7C, 0x0B, 0x91, 0x90, 0x54, 0xE4,
0x75, 0xCD, 0x29, 0x24, 0x0B, 0x90, 0xBC,
/*00CCD0:*/ 0x29, 0x78, 0x60, 0x9B, 0x31, 0xDB, 0xF4, 0x11, 0x3F,
0x86, 0x27, 0xB9, 0xC0, 0x94, 0x54, 0xC9,
/*00CCE0:*/ 0x8F, 0x08, 0x7C, 0x84, 0x1E, 0xAA, 0x3D, 0x3F, 0x3C,
0x6C, 0xB3, 0x5F, 0x3F, 0x86, 0x27, 0xB9,
/*00CCF0:*/ 0x38, 0x3C, 0x18, 0x72, 0x50, 0x45, 0x07, 0xF0, 0x18,
0x3A, 0x8E, 0x81, 0x32, 0x39, 0x17, 0x0C,
/*00CD00:*/ 0xD7, 0xAB, 0xF5, 0x95, 0x12, 0xD1, 0x59, 0x1B, 0x57,
0x89, 0xDA, 0xB2, 0x84, 0x8E, 0xCA, 0xB6,
/*00CD10:*/ 0xDD, 0x87, 0x4E, 0x1D, 0x1E, 0xAA, 0x3D, 0x3F, 0x50,
0xEE, 0xF3, 0x99, 0xD1, 0xC9, 0xD0, 0x2E,
/*00CD20:*/ 0x55, 0xAF, 0xCB, 0x7D, 0xC9, 0x9A, 0xC6, 0xDD, 0x3D,
0x76, 0x22, 0xB1, 0x30, 0xC5, 0x0D, 0x6C,
/*00CD30:*/ 0xC9, 0x17, 0xFF, 0x0E, 0x3B, 0x5C, 0x56, 0x54, 0xD2,
0x1D, 0x28, 0xFD, 0x88, 0x59, 0xC4, 0x4D,
/*00CD40:*/ 0x3E, 0x2C, 0x92, 0xD5, 0x93, 0x60, 0x00, 0x4C, 0xDC,
0xBD, 0x76, 0x19, 0x46, 0xC2, 0x68, 0x74,
/*00CD50:*/ 0xF9, 0xA8, 0xD0, 0xE2, 0xD8, 0x7C, 0x3C, 0x08, 0x99,
0x7B, 0x6F, 0xC5, 0x3F, 0xF5, 0x3C, 0x3E,
/*00CD60:*/ 0x78, 0xBE, 0x58, 0x61, 0xBF, 0xB4, 0xD2, 0x8F, 0xA9,
0x86, 0xA4, 0x12, 0x9C, 0xF1, 0x52, 0x83,
/*00CD70:*/ 0x3A, 0xBA, 0xED, 0xB6, 0x0B, 0x94, 0x45, 0xA7, 0x02,
0x44, 0x8C, 0xB9, 0xC4, 0xC9, 0xED, 0x05,
/*00CD80:*/ 0xFC, 0x8A, 0x20, 0xF0, 0x0B, 0x91, 0x90, 0x54, 0x69,
0x5E, 0x78, 0x07, 0x4B, 0x70, 0xE9, 0x42,
/*00CD90:*/ 0xD7, 0xAB, 0xF5, 0x95, 0x43, 0x6C, 0xEF, 0x92, 0xB2,
0x0E, 0xC0, 0x25, 0x3C, 0x6C, 0xB3, 0x5F,
/*00CDA0:*/ 0x8F, 0x08, 0x7C, 0x84, 0x23, 0xA4, 0x9B, 0x3E, 0xCB,
0x53, 0xD3, 0xFF, 0xA0, 0x18, 0x0B, 0x31,
/*00CDB0:*/ 0xF0, 0x36, 0x3C, 0x9C, 0x0B, 0xD9, 0x3A, 0x41, 0xB4,
0x69, 0x9F, 0x38, 0x29, 0x78, 0x60, 0x9B,
/*00CDC0:*/ 0x75, 0x18, 0x6F, 0xB8, 0x4F, 0xA6, 0xC1, 0x11, 0xB4,
0x69, 0x9F, 0x38, 0xA3, 0xAE, 0xC3, 0xFA,
/*00CDD0:*/ 0x50, 0x45, 0x07, 0xF0, 0xA7, 0x54, 0x4C, 0xAD, 0xC1,
0xF6, 0x2F, 0x66, 0xE0, 0x58, 0x7A, 0x6F,
/*00CDE0:*/ 0x31, 0xAF, 0x65, 0x25, 0x1B, 0xA6, 0x0E, 0x00, 0x4C,
0x79, 0xC2, 0xEB, 0xA4, 0xCC, 0xA3, 0x0A,
/*00CDF0:*/ 0x6A, 0xAB, 0x24, 0xB1, 0x7D, 0x8C, 0xFD, 0x17, 0xC6,
0xB3, 0x24, 0x7D, 0x88, 0x5D, 0xD3, 0xB7,
/*00CE00:*/ 0x06, 0xFD, 0x74, 0x0A, 0xA8, 0x5F, 0xB1, 0x36, 0xFD,
0x2F, 0x14, 0x90, 0x05, 0x6E, 0x54, 0x33,
/*00CE10:*/ 0x12, 0xE9, 0xB6, 0xBE, 0xE8, 0x2A, 0x38, 0x57, 0x86,
0xBD, 0xE9, 0x78, 0xFC, 0x8A, 0x20, 0xF0,
/*00CE20:*/ 0x38, 0x6D, 0xFD, 0x2D, 0xA4, 0xCC, 0xA3, 0x0A, 0x7F,
0x7E, 0x86, 0xF1, 0xA8, 0x21, 0x62, 0x44,
/*00CE30:*/ 0x3E, 0xF1, 0xB5, 0x10, 0x3B, 0x5C, 0x56, 0x54, 0x3E,
0x77, 0xA8, 0x9D, 0x30, 0xD9, 0x62, 0x05,
/*00CE40:*/ 0xD7, 0xAB, 0xF5, 0x95, 0xFC, 0x8A, 0x20, 0xF0, 0xD5,
0x80, 0xA1, 0x41, 0xDD, 0xB8, 0xF4, 0x8F,
/*00CE50:*/ 0xF5, 0xF5, 0xB0, 0x30, 0x9C, 0x26, 0xF2, 0x99, 0x75,
0x80, 0x94, 0x83, 0xF6, 0x90, 0xE9, 0xDC,
/*00CE60:*/ 0x7B, 0x07, 0x82, 0x94, 0x0B, 0x94, 0xDD, 0x44, 0x29,
0x78, 0x60, 0x9B, 0x17, 0x08, 0x0D, 0x3F,
/*00CE70:*/ 0xBF, 0x9D, 0xDE, 0x0F, 0x55, 0x59, 0xA8, 0xF2, 0x5E,
0x8C, 0x7B, 0xD8, 0xDF, 0xC7, 0x1C, 0x7D,
/*00CE80:*/ 0xC8, 0xF7, 0x44, 0xFE, 0xDF, 0xC7, 0x1C, 0x7D, 0x77,
0xA4, 0xCE, 0x28, 0xF3, 0x37, 0xA7, 0x4A,
/*00CE90:*/ 0xAC, 0xD9, 0x82, 0x35, 0x16, 0x42, 0x66, 0x51, 0x42,
0xFA, 0x83, 0x36, 0x55, 0xA1, 0xE1, 0x6B,
/*00CEA0:*/ 0x70, 0xF3, 0x21, 0x92, 0x0D, 0xF0, 0xAD, 0xBA, 0x0D,
0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*00CEB0:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B6D2_1[] = {
/*00D4F0:*/ 0x87, 0xE9, 0x8C, 0x3E, 0x0B, 0x91, 0x90, 0x54, 0xAD,
0x45, 0x93, 0x08, 0xA7, 0x54, 0x4C, 0xAD,
/*00D500:*/ 0xAC, 0xCD, 0x2D, 0x3A, 0x87, 0xA3, 0x0D, 0x5B, 0xB5,
0x0F, 0xE0, 0x5A, 0x4A, 0xEF, 0x3F, 0x65,
/*00D510:*/ 0x74, 0x48, 0x37, 0x81, 0x22, 0xE4, 0x16, 0xC9, 0x4F,
0xA6, 0xC1, 0x11, 0x53, 0xE5, 0xF5, 0x0D,
/*00D520:*/ 0xB9, 0xD7, 0x98, 0x99, 0x55, 0xA1, 0xE1, 0x6B, 0xA1,
0x44, 0x3D, 0x80, 0x5B, 0x25, 0xF1, 0x26,
/*00D530:*/ 0x93, 0x60, 0x00, 0x4C, 0xD7, 0xAB, 0xF5, 0x95, 0x0C,
0x94, 0x6A, 0x6E, 0xAD, 0xBE, 0xFA, 0xE0,
/*00D540:*/ 0x31, 0xDB, 0xF4, 0x11, 0x77, 0xA4, 0xCE, 0x28, 0x1A,
0x22, 0xCB, 0x38, 0xB9, 0xD8, 0x19, 0x8B,
/*00D550:*/ 0x3B, 0xE7, 0x29, 0xC7, 0xC9, 0x9A, 0xC6, 0xDD, 0xB9,
0xB0, 0x7A, 0x35, 0x59, 0x72, 0x6E, 0xA4,
/*00D560:*/ 0x4C, 0x79, 0xC2, 0xEB, 0x44, 0x09, 0xC7, 0xED, 0x44,
0x09, 0xC7, 0xED, 0xE8, 0x83, 0x83, 0x7C,
/*00D570:*/ 0xAD, 0xD2, 0xB8, 0x48, 0x3F, 0xF5, 0x3C, 0x3E, 0x0B,
0x91, 0x90, 0x54, 0x30, 0x52, 0xB4, 0xEB,
/*00D580:*/ 0x0B, 0x91, 0x90, 0x54, 0x30, 0x52, 0xB4, 0xEB, 0xAD,
0xBE, 0xFA, 0xE0, 0x01, 0x62, 0x83, 0xE7,
/*00D590:*/ 0x05, 0x0A, 0xA5, 0xE5, 0x5B, 0x3A, 0xD3, 0x95, 0x0E,
0xFF, 0x87, 0x5A, 0x95, 0xCB, 0x9B, 0xC3,
/*00D5A0:*/ 0xFC, 0x2C, 0x50, 0x09, 0xC5, 0x3F, 0x56, 0x06, 0x16,
0x31, 0xE8, 0x05, 0xBF, 0xB4, 0xD2, 0x8F,
/*00D5B0:*/ 0x28, 0x4C, 0x01, 0x90, 0x05, 0x6E, 0x54, 0x33, 0x32,
0x39, 0x17, 0x0C, 0xFC, 0xA4, 0x4A, 0xBD,
/*00D5C0:*/ 0x91, 0xB3, 0x84, 0xAB, 0xD2, 0x1D, 0x28, 0xFD, 0x58,
0x1F, 0xAC, 0x46, 0xA0, 0x3E, 0xD4, 0xB4,
/*00D5D0:*/ 0xC8, 0xF7, 0x44, 0xFE, 0x45, 0xCF, 0xD3, 0x8B, 0x04,
0x47, 0x46, 0x11, 0x50, 0xEE, 0xF3, 0x99,
/*00D5E0:*/ 0x4D, 0x01, 0x6B, 0x7A, 0x16, 0x42, 0x66, 0x51, 0xAE,
0xBA, 0x7D, 0xFF, 0x2D, 0xD1, 0x6B, 0xA4,
/*00D5F0:*/ 0x22, 0xA7, 0xE7, 0x6E, 0x60, 0x43, 0xCC, 0x0E, 0x74,
0x7A, 0x01, 0x61, 0x52, 0x14, 0xC3, 0xE8,
/*00D600:*/ 0x5D, 0x65, 0xB5, 0x0A, 0xEC, 0xC6, 0xB5, 0x4C, 0x6F,
0xEE, 0xE7, 0x2D, 0xB2, 0x85, 0x75, 0xC4,
/*00D610:*/ 0xC9, 0x9A, 0xC6, 0xDD, 0x55, 0xA1, 0xE1, 0x6B, 0x31,
0xEA, 0x4D, 0xAD, 0xBF, 0xB4, 0xD2, 0x8F,
/*00D620:*/ 0xC9, 0x9A, 0xC6, 0xDD, 0xF0, 0x25, 0xD0, 0x02, 0x57,
0x89, 0xDA, 0xB2, 0x4C, 0x2C, 0x53, 0x9D,
/*00D630:*/ 0x8D, 0x3B, 0x92, 0xB8, 0x58, 0x1F, 0xAC, 0x46, 0xC8,
0xF7, 0x44, 0xFE, 0x45, 0xCF, 0xD3, 0x8B,
/*00D640:*/ 0xE2, 0x6D, 0x41, 0xB7, 0xCF, 0x7E, 0x80, 0xB6, 0xA4,
0xCC, 0xA3, 0x0A, 0xE6, 0x05, 0x3C, 0xDF,
/*00D650:*/ 0xB9, 0xB0, 0x7A, 0x35, 0xFC, 0xB8, 0x3C, 0xEB, 0x7E,
0x03, 0x73, 0x1D, 0xB9, 0xB0, 0x7A, 0x35,
/*00D660:*/ 0xE0, 0x58, 0x7A, 0x6F, 0xFA, 0xCE, 0xAF, 0xA5, 0x0C,
0x48, 0x3A, 0x3A, 0x3E, 0x77, 0xA8, 0x9D,
/*00D670:*/ 0xA2, 0xAB, 0xA5, 0xF1, 0x0B, 0x91, 0x90, 0x54, 0xD4,
0xA8, 0xF0, 0xA6, 0x70, 0xF3, 0x21, 0x92,
/*00D680:*/ 0x53, 0xE5, 0xF5, 0x0D, 0x58, 0x1F, 0xAC, 0x46, 0xC0,
0x94, 0x54, 0xC9, 0x35, 0xC4, 0xB2, 0x1B,
/*00D690:*/ 0x41, 0x0F, 0x82, 0xFB, 0x7E, 0x03, 0x73, 0x1D, 0xB9,
0xD8, 0x08, 0x6A, 0x34, 0x2E, 0x7D, 0x84,
/*00D6A0:*/ 0xDF, 0xC7, 0x1C, 0x7D, 0xE6, 0x05, 0x3C, 0xDF, 0x0B,
0x91, 0x90, 0x54, 0x0B, 0xD9, 0x3A, 0x41,
/*00D6B0:*/ 0x3B, 0xFB, 0x95, 0x81, 0xF7, 0x33, 0x41, 0x1A, 0x76,
0x0E, 0x7F, 0x97, 0x91, 0xAB, 0xA7, 0x58,
/*00D6C0:*/ 0x69, 0x28, 0xA5, 0xF2, 0x5A, 0x6C, 0x52, 0x04, 0xC8,
0xF7, 0x44, 0xFE, 0x20, 0x2B, 0x8D, 0x19,
/*00D6D0:*/ 0x15, 0x5D, 0x3E, 0xDE, 0x48, 0x9B, 0x25, 0x2F, 0xD9,
0x27, 0xDA, 0xA5, 0xCF, 0x7E, 0x80, 0xB6,
/*00D6E0:*/ 0x6C, 0x3C, 0x44, 0x49, 0x1A, 0x22, 0xCB, 0x38, 0xAE,
0xBA, 0x7D, 0xFF, 0xDD, 0xD7, 0x3F, 0x58,
/*00D6F0:*/ 0x04, 0xEA, 0xA7, 0x28, 0x12, 0xD1, 0x59, 0x1B, 0x7B,
0xF8, 0xDA, 0x1E, 0xD8, 0x7C, 0x3C, 0x08,
/*00D700:*/ 0xE2, 0x6D, 0x41, 0xB7, 0xF3, 0x37, 0xA7, 0x4A, 0x23,
0xEB, 0x14, 0x0E, 0x45, 0xCF, 0x2B, 0x5E,
/*00D710:*/ 0xFC, 0xA4, 0x4A, 0xBD, 0xCB, 0xCC, 0xC1, 0xC4, 0x16,
0x42, 0x66, 0x51, 0x22, 0xE4, 0x16, 0xC9,
/*00D720:*/ 0x7B, 0xF8, 0xDA, 0x1E, 0xAD, 0xC9, 0xB2, 0x04, 0x83,
0xB9, 0x54, 0x82, 0x95, 0xCB, 0x9B, 0xC3,
/*00D730:*/ 0x4A, 0xDF, 0x4D, 0xF0, 0x20, 0x2B, 0x8D, 0x19, 0x85,
0x5E, 0xED, 0xBB, 0x92, 0x26, 0x3D, 0x12,
/*00D740:*/ 0x3F, 0xF5, 0x3C, 0x3E, 0x02, 0x44, 0x8C, 0xB9, 0xC4,
0x68, 0xAC, 0x83, 0xC5, 0x21, 0xEF, 0xFE,
/*00D750:*/ 0x4C, 0x79, 0xC2, 0xEB, 0x8C, 0xBA, 0x04, 0x05, 0x6D,
0x08, 0x98, 0x24, 0x0C, 0x48, 0x3A, 0x3A,
/*00D760:*/ 0xC3, 0x37, 0xAD, 0x8F, 0x23, 0xA4, 0x9B, 0x3E, 0xE3,
0xA6, 0xD3, 0x18, 0x2F, 0x13, 0xD5, 0xAE,
/*00D770:*/ 0x30, 0xF7, 0x73, 0xE2, 0x12, 0xE9, 0xB6, 0xBE, 0xA0,
0x4F, 0x6B, 0xA5, 0x74, 0x2F, 0xB8, 0xE3,
/*00D780:*/ 0x9F, 0x3A, 0xA0, 0xC3, 0x23, 0xA4, 0x9B, 0x3E, 0xE8,
0x83, 0x83, 0x7C, 0x9E, 0xE0, 0xC6, 0x8C,
/*00D790:*/ 0xC8, 0xF7, 0x44, 0xFE, 0x16, 0xCE, 0xEE, 0xEB, 0x30,
0xC5, 0x0D, 0x6C, 0x52, 0x1B, 0x4B, 0xEE,
/*00D7A0:*/ 0x88, 0x8D, 0xF5, 0x4B, 0x79, 0xE7, 0x73, 0x3D, 0x92,
0x26, 0x3D, 0x12, 0x12, 0xE9, 0xB6, 0xBE,
/*00D7B0:*/ 0x48, 0x2F, 0x23, 0x3B, 0x6C, 0x3C, 0x44, 0x49, 0x48,
0xF6, 0x2D, 0x5F, 0xF3, 0x37, 0xA7, 0x4A,
/*00D7C0:*/ 0x4A, 0xDF, 0x4D, 0xF0, 0x0E, 0xFF, 0x87, 0x5A, 0x9C,
0x26, 0xF2, 0x99, 0x05, 0xAD, 0xAB, 0x34,
/*00D7D0:*/ 0xD4, 0x56, 0x59, 0x6D, 0x0D, 0xF0, 0xAD, 0xBA, 0x0D,
0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*00D7E0:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B6D2_2[] = {
/*00DE80:*/ 0x30, 0xC5, 0x0D, 0x6C, 0x5E, 0xAC, 0xB2, 0x86, 0x33,
0xD9, 0x7D, 0xD3, 0x52, 0x1B, 0x4B, 0xEE,
/*00DE90:*/ 0x22, 0xA2, 0x25, 0xBA, 0xAD, 0x45, 0x93, 0x08, 0x85,
0x9D, 0xF0, 0x35, 0xD2, 0x1D, 0x28, 0xFD,
/*00DEA0:*/ 0xDD, 0xB8, 0xF4, 0x8F, 0x7A, 0x1A, 0xB6, 0x8B, 0xC2,
0xEF, 0x17, 0x8C, 0x91, 0xB3, 0x84, 0xAB,
/*00DEB0:*/ 0xA0, 0x4F, 0x6B, 0xA5, 0x00, 0x31, 0x68, 0x11, 0x4C,
0x79, 0xC2, 0xEB, 0x3E, 0x2C, 0x92, 0xD5,
/*00DEC0:*/ 0x30, 0xC5, 0x0D, 0x6C, 0x95, 0xCB, 0x9B, 0xC3, 0xE6,
0x05, 0x3C, 0xDF, 0xDA, 0x4D, 0x0A, 0x23,
/*00DED0:*/ 0x06, 0xFF, 0xC3, 0x32, 0xD8, 0x7C, 0x3C, 0x08, 0x30,
0x52, 0xB4, 0xEB, 0xF3, 0x37, 0xA7, 0x4A,
/*00DEE0:*/ 0x04, 0xDE, 0x4C, 0x7C, 0x18, 0xD8, 0x08, 0x89, 0x50,
0x45, 0x07, 0xF0, 0xE2, 0x6D, 0x41, 0xB7,
/*00DEF0:*/ 0x58, 0xF3, 0x41, 0xBF, 0x55, 0xA1, 0xE1, 0x6B, 0xFC,
0xB8, 0x3C, 0xEB, 0x28, 0x4C, 0x01, 0x90,
/*00DF00:*/ 0xD3, 0x37, 0xFF, 0xEE, 0x26, 0xAD, 0xDB, 0x83, 0x87,
0xA3, 0x0D, 0x5B, 0x59, 0x72, 0x6E, 0xA4,
/*00DF10:*/ 0x93, 0x60, 0x00, 0x4C, 0x0F, 0x10, 0x4F, 0x74, 0x77,
0xF6, 0x5E, 0xC9, 0x48, 0x05, 0x13, 0xBF,
/*00DF20:*/ 0x76, 0x0E, 0x7F, 0x97, 0xDD, 0x87, 0x4E, 0x1D, 0xE6,
0xC2, 0x63, 0x04, 0xA0, 0x4F, 0x6B, 0xA5,
/*00DF30:*/ 0x02, 0x44, 0x8C, 0xB9, 0x0C, 0x31, 0xB7, 0x7D, 0x3B,
0x5C, 0x56, 0x54, 0x12, 0x7A, 0x2C, 0xF2,
/*00DF40:*/ 0x3D, 0x72, 0xD9, 0x4A, 0x1E, 0xE6, 0x3B, 0xD2, 0xE6,
0x1C, 0xB6, 0x87, 0x4B, 0x70, 0xE9, 0x42,
/*00DF50:*/ 0xCC, 0x79, 0x89, 0x67, 0x05, 0x0A, 0xA5, 0xE5, 0xB2,
0x0E, 0xC0, 0x25, 0x5B, 0x1A, 0x7E, 0x64,
/*00DF60:*/ 0x9C, 0x08, 0x26, 0xCD, 0xB9, 0xD7, 0x98, 0x99, 0xB2,
0x99, 0xD9, 0x2C, 0xF6, 0x90, 0xE9, 0xDC,
/*00DF70:*/ 0xDD, 0x61, 0xC6, 0xD6, 0x59, 0xA6, 0x2F, 0x0B, 0x7F,
0x7E, 0x86, 0xF1, 0x43, 0x6C, 0xEF, 0x92,
/*00DF80:*/ 0x30, 0x52, 0xB4, 0xEB, 0xF4, 0x2A, 0x00, 0x0E, 0xD4,
0x0E, 0x6D, 0x00, 0x9E, 0xE0, 0xC6, 0x8C,
/*00DF90:*/ 0x7F, 0x7E, 0x86, 0xF1, 0xE8, 0x83, 0x83, 0x7C, 0xC8,
0xF7, 0x44, 0xFE, 0x95, 0xCB, 0x9B, 0xC3,
/*00DFA0:*/ 0x4D, 0x1E, 0x4D, 0x3C, 0x4D, 0x1E, 0x4D, 0x3C, 0x3B,
0x22, 0x07, 0x87, 0x57, 0x89, 0xDA, 0xB2,
/*00DFB0:*/ 0x26, 0xF9, 0xB9, 0xC4, 0x51, 0x07, 0x91, 0xBA, 0x3B,
0xFB, 0x95, 0x81, 0x8F, 0x08, 0x7C, 0x84,
/*00DFC0:*/ 0xCB, 0x41, 0x4C, 0xDA, 0x95, 0xCB, 0x9B, 0xC3, 0xD7,
0xE3, 0xE9, 0x28, 0x1E, 0xAA, 0x3D, 0x3F,
/*00DFD0:*/ 0x1B, 0xBE, 0xBA, 0x74, 0x8D, 0x09, 0x0F, 0xE5, 0x6A,
0xAB, 0x24, 0xB1, 0x99, 0x45, 0xAA, 0x43,
/*00DFE0:*/ 0xE2, 0x99, 0xC6, 0x51, 0xAD, 0x45, 0x93, 0x08, 0x1E,
0xAA, 0x3D, 0x3F, 0x9C, 0x26, 0xF2, 0x99,
/*00DFF0:*/ 0xFC, 0x8A, 0x20, 0xF0, 0xD2, 0x1D, 0x28, 0xFD, 0xC6,
0xB3, 0x24, 0x7D, 0x91, 0xB3, 0x84, 0xAB,
/*00E000:*/ 0xC2, 0xEF, 0x17, 0x8C, 0x3C, 0x6C, 0xB3, 0x5F, 0x6F,
0xEE, 0xE7, 0x2D, 0x87, 0xA3, 0x0D, 0x5B,
/*00E010:*/ 0x0C, 0x48, 0x3A, 0x3A, 0x49, 0xE4, 0x72, 0xD0, 0x66,
0x7B, 0xFC, 0x02, 0xD5, 0x80, 0xA1, 0x41,
/*00E020:*/ 0xA8, 0x21, 0x62, 0x44, 0xFC, 0x8A, 0x20, 0xF0, 0xC3,
0x68, 0x4D, 0xA0, 0x87, 0xA3, 0x0D, 0x5B,
/*00E030:*/ 0x65, 0xF8, 0xDE, 0x27, 0x6C, 0x4E, 0xC0, 0x23, 0x53,
0xE5, 0xF5, 0x0D, 0x05, 0xAD, 0xAB, 0x34,
/*00E040:*/ 0xD7, 0x9F, 0x82, 0x50, 0xD4, 0x56, 0x59, 0x6D, 0x9C,
0x26, 0xF2, 0x99, 0x74, 0x2F, 0xB8, 0xE3,
/*00E050:*/ 0xE9, 0xD6, 0xD1, 0xF2, 0x08, 0xBD, 0xC5, 0x11, 0xDE,
0x02, 0x1C, 0xBD, 0xF5, 0x3F, 0x9F, 0x2A,
/*00E060:*/ 0xFC, 0xB8, 0x3C, 0xEB, 0x30, 0xF7, 0x73, 0xE2, 0xC2,
0xEF, 0x17, 0x8C, 0x91, 0x24, 0x1D, 0x9F,
/*00E070:*/ 0x8C, 0xD7, 0x34, 0x6A, 0x9A, 0x17, 0xED, 0x45, 0x4C,
0x79, 0xC2, 0xEB, 0xD4, 0x56, 0x59, 0x6D,
/*00E080:*/ 0x16, 0xCE, 0xEE, 0xEB, 0x5B, 0x1A, 0x7E, 0x64, 0xA6,
0xAB, 0xA8, 0x1A, 0x82, 0x97, 0x72, 0x52,
/*00E090:*/ 0x65, 0xF8, 0xDE, 0x27, 0x9D, 0x62, 0x7B, 0x14, 0x63,
0xAE, 0xE3, 0x1C, 0x78, 0xBE, 0x58, 0x61,
/*00E0A0:*/ 0x60, 0x43, 0xCC, 0x0E, 0x6F, 0x07, 0xE8, 0x61, 0xE2,
0x99, 0xC6, 0x51, 0xC9, 0x17, 0xFF, 0x0E,
/*00E0B0:*/ 0xE0, 0x9C, 0x6D, 0x48, 0x52, 0xE5, 0x37, 0xD4, 0xE4,
0x75, 0xCD, 0x29, 0xF7, 0x95, 0x36, 0x9E,
/*00E0C0:*/ 0xFC, 0x2C, 0x50, 0x09, 0xE0, 0x58, 0x7A, 0x6F, 0xE4,
0x4E, 0x30, 0x3A, 0xBF, 0xB4, 0xD2, 0x8F,
/*00E0D0:*/ 0xCA, 0xE9, 0xF0, 0x51, 0x33, 0xD9, 0x7D, 0xD3, 0xFC,
0xB8, 0x3C, 0xEB, 0xEC, 0x08, 0x7B, 0x79,
/*00E0E0:*/ 0x92, 0x82, 0xF5, 0xB9, 0x9A, 0x17, 0xED, 0x45, 0xCB,
0x53, 0xD3, 0xFF, 0xCF, 0xC9, 0xD5, 0x98,
/*00E0F0:*/ 0x37, 0x83, 0x14, 0x26, 0x52, 0x14, 0xC3, 0xE8, 0xEB,
0x76, 0x06, 0x7B, 0xBF, 0xB4, 0xD2, 0x8F,
/*00E100:*/ 0x8C, 0x7F, 0x26, 0x4A, 0x85, 0x9D, 0xF0, 0x35, 0xB9,
0xB0, 0x7A, 0x35, 0x0D, 0xF0, 0xAD, 0xBA,
/*00E110:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B6D2_3[] = {
/*00E730:*/ 0x15, 0x04, 0x21, 0x64, 0xDD, 0x61, 0xC6, 0xD6, 0xA9,
0x86, 0xA4, 0x12, 0x06, 0xFF, 0xC3, 0x32,
/*00E740:*/ 0xA0, 0x4F, 0x6B, 0xA5, 0xAD, 0xBE, 0xFA, 0xE0, 0x06,
0x4B, 0xA5, 0x4C, 0xE8, 0x83, 0x83, 0x7C,
/*00E750:*/ 0x28, 0x4C, 0x01, 0x90, 0xA1, 0xDC, 0x17, 0xE0, 0x2D,
0x75, 0xCC, 0x80, 0x15, 0x5D, 0x3E, 0xDE,
/*00E760:*/ 0x75, 0x80, 0x94, 0x83, 0xCF, 0x7E, 0x80, 0xB6, 0x11,
0x68, 0xF0, 0x4A, 0xC8, 0xC4, 0xD0, 0xA3,
/*00E770:*/ 0xFA, 0xCE, 0xAF, 0xA5, 0x85, 0x9D, 0xF0, 0x35, 0x8E,
0x23, 0x09, 0xF3, 0x5D, 0x65, 0xB5, 0x0A,
/*00E780:*/ 0x89, 0xBB, 0xCE, 0xE7, 0x00, 0xBC, 0x1A, 0xC2, 0xF7,
0x33, 0x41, 0x1A, 0x7E, 0x03, 0x73, 0x1D,
/*00E790:*/ 0xE2, 0x9F, 0xD6, 0x27, 0x2F, 0x1F, 0xC8, 0xD8, 0xFC,
0x8A, 0x20, 0xF0, 0x69, 0x28, 0xA5, 0xF2,
/*00E7A0:*/ 0x67, 0x0B, 0x9D, 0x26, 0x43, 0xDF, 0xC5, 0x2E, 0x38,
0x6D, 0xFD, 0x2D, 0x5F, 0xF2, 0xFA, 0xF6,
/*00E7B0:*/ 0xD8, 0x99, 0x2C, 0xB7, 0x53, 0xE5, 0xF5, 0x0D, 0x38,
0x6D, 0xFD, 0x2D, 0xA4, 0xCC, 0xA3, 0x0A,
/*00E7C0:*/ 0xAC, 0xD9, 0x20, 0x28, 0xEC, 0x88, 0xC7, 0xDB, 0x63,
0xD9, 0xBE, 0x5C, 0x2B, 0xCE, 0x76, 0xE4,
/*00E7D0:*/ 0x05, 0xAD, 0xAB, 0x34, 0xBF, 0xB4, 0xD2, 0x8F, 0xD2,
0x1D, 0x28, 0xFD, 0xF5, 0xB7, 0xF4, 0x43,
/*00E7E0:*/ 0x23, 0xA4, 0x9B, 0x3E, 0x5B, 0x3A, 0xD3, 0x95, 0xE8,
0x83, 0x83, 0x7C, 0x24, 0x0B, 0x90, 0xBC,
/*00E7F0:*/ 0x28, 0x4C, 0x01, 0x90, 0xC8, 0xF7, 0x44, 0xFE, 0x88,
0x59, 0xC4, 0x4D, 0x0C, 0x31, 0xB7, 0x7D,
/*00E800:*/ 0x59, 0x03, 0xA8, 0x1A, 0xE3, 0xA6, 0xD3, 0x18, 0x4B,
0x91, 0x59, 0xAA, 0xDF, 0xC7, 0x1C, 0x7D,
/*00E810:*/ 0xAD, 0xBE, 0xFA, 0xE0, 0x01, 0xD7, 0x5F, 0xC0, 0xE9,
0xD6, 0xD1, 0xF2, 0x08, 0xBD, 0xC5, 0x11,
/*00E820:*/ 0xEC, 0xC6, 0xB5, 0x4C, 0x05, 0x6E, 0x54, 0x33, 0x28,
0x4C, 0x01, 0x90, 0x50, 0x09, 0xFA, 0x55,
/*00E830:*/ 0xCE, 0x1C, 0x58, 0x49, 0x00, 0xBC, 0x1A, 0xC2, 0x15,
0x04, 0x21, 0x64, 0x8E, 0x92, 0x63, 0xB0,
/*00E840:*/ 0x21, 0xD0, 0xE7, 0x04, 0xBC, 0x9B, 0x2F, 0xC1, 0x4D,
0x74, 0xF8, 0x85, 0xFD, 0x70, 0xEB, 0xC1,
/*00E850:*/ 0x28, 0x4C, 0x01, 0x90, 0x7C, 0x77, 0x4B, 0xD7, 0x70,
0x35, 0xAC, 0xDE, 0xEB, 0x76, 0x06, 0x7B,
/*00E860:*/ 0x83, 0x55, 0x2F, 0xE2, 0xCB, 0x41, 0x4C, 0xDA, 0xAD,
0xBE, 0xFA, 0xE0, 0x72, 0xCE, 0xA1, 0xE6,
/*00E870:*/ 0x52, 0x01, 0x30, 0xE2, 0x4A, 0xDF, 0x4D, 0xF0, 0x32,
0xB2, 0x2A, 0xFF, 0x55, 0x59, 0xA8, 0xF2,
/*00E880:*/ 0xB0, 0xF5, 0x9F, 0x48, 0xD7, 0x9F, 0x82, 0x50, 0x87,
0xA3, 0x0D, 0x5B, 0x10, 0x20, 0x84, 0x97,
/*00E890:*/ 0x7C, 0xC3, 0x8D, 0xB5, 0x4B, 0x70, 0xE9, 0x42, 0x75,
0x80, 0x94, 0x83, 0x51, 0x07, 0x91, 0xBA,
/*00E8A0:*/ 0x94, 0xE8, 0xA4, 0xD1, 0x68, 0x0E, 0xC1, 0x03, 0x95,
0xCB, 0x9B, 0xC3, 0x68, 0x36, 0xD9, 0x72,
/*00E8B0:*/ 0xB9, 0xB0, 0x7A, 0x35, 0x99, 0x45, 0xAA, 0x43, 0xBC,
0x9B, 0x2F, 0xC1, 0x22, 0xE4, 0x16, 0xC9,
/*00E8C0:*/ 0x9C, 0xF1, 0x52, 0x83, 0xFC, 0xA4, 0x4A, 0xBD, 0xD5,
0x80, 0xA1, 0x41, 0xD9, 0xB8, 0x25, 0xCB,
/*00E8D0:*/ 0x63, 0xAE, 0xE3, 0x1C, 0x44, 0x09, 0xC7, 0xED, 0x51,
0x07, 0x91, 0xBA, 0xC8, 0xF7, 0x44, 0xFE,
/*00E8E0:*/ 0x6A, 0xAB, 0x24, 0xB1, 0x75, 0x80, 0x94, 0x83, 0xAD,
0x45, 0x93, 0x08, 0x24, 0x0B, 0x90, 0xBC,
/*00E8F0:*/ 0x1A, 0xD8, 0xBC, 0x1F, 0x3E, 0x77, 0xA8, 0x9D, 0x3F,
0x86, 0x27, 0xB9, 0xF2, 0xD1, 0x99, 0xBC,
/*00E900:*/ 0x88, 0x8D, 0xF5, 0x4B, 0x05, 0xAD, 0xAB, 0x34, 0x3B,
0x22, 0x07, 0x87, 0xE8, 0x83, 0x83, 0x7C,
/*00E910:*/ 0xE0, 0x58, 0x7A, 0x6F, 0xA4, 0xCC, 0xA3, 0x0A, 0xE6,
0x05, 0x3C, 0xDF, 0x44, 0x4C, 0x5D, 0xB4,
/*00E920:*/ 0x69, 0x28, 0xA5, 0xF2, 0xCE, 0x1C, 0x58, 0x49, 0x38,
0x8D, 0x3D, 0x7F, 0x55, 0xA1, 0xE1, 0x6B,
/*00E930:*/ 0x59, 0x72, 0x6E, 0xA4, 0xC4, 0x28, 0x18, 0xE6, 0x8E,
0x11, 0x1D, 0xA3, 0x34, 0x2E, 0x7D, 0x84,
/*00E940:*/ 0x34, 0x2E, 0x7D, 0x84, 0x7E, 0xB6, 0xC1, 0x8E, 0xA7,
0x54, 0x4C, 0xAD, 0xE6, 0x1C, 0xB6, 0x87,
/*00E950:*/ 0xFD, 0x72, 0xC3, 0xA8, 0x7A, 0x97, 0x8F, 0xDF, 0xCA,
0xE9, 0xF0, 0x51, 0x28, 0x4C, 0x01, 0x90,
/*00E960:*/ 0x89, 0xBB, 0xCE, 0xE7, 0x35, 0x64, 0xCB, 0xAB, 0x84,
0xE6, 0x91, 0xA6, 0x7E, 0x03, 0x73, 0x1D,
/*00E970:*/ 0x0A, 0x4A, 0x75, 0x7B, 0xC0, 0x94, 0x54, 0xC9, 0x89,
0xBB, 0xCE, 0xE7, 0xB4, 0x69, 0x9F, 0x38,
/*00E980:*/ 0x67, 0x0B, 0x9D, 0x26, 0x04, 0xEA, 0xA7, 0x28, 0x52,
0x14, 0xC3, 0xE8, 0x48, 0xF6, 0x2D, 0x5F,
/*00E990:*/ 0x3F, 0xF5, 0x3C, 0x3E, 0xE6, 0x05, 0x3C, 0xDF, 0xE4,
0x61, 0x1D, 0x38, 0x59, 0x03, 0xA8, 0x1A,
/*00E9A0:*/ 0x9C, 0x26, 0xF2, 0x99, 0x0D, 0xF0, 0xAD, 0xBA, 0x0D,
0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*00E9B0:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B6D2_4[] = {
/*00EFC0:*/ 0x9E, 0xE0, 0xC6, 0x8C, 0x33, 0x94, 0x16, 0xD9, 0x8F,
0x08, 0x7C, 0x84, 0x6F, 0x1B, 0x63, 0xB3,
/*00EFD0:*/ 0xC8, 0xC4, 0xD0, 0xA3, 0xA2, 0x54, 0x67, 0x83, 0x9C,
0x26, 0xF2, 0x99, 0xD8, 0x99, 0x2C, 0xB7,
/*00EFE0:*/ 0xE6, 0xB0, 0xC1, 0x6B, 0x0C, 0x48, 0x3A, 0x3A, 0x4C,
0x79, 0xC2, 0xEB, 0x85, 0x9D, 0xF0, 0x35,
/*00EFF0:*/ 0xA0, 0x4F, 0x6B, 0xA5, 0xAD, 0xBE, 0xFA, 0xE0, 0x12,
0xD1, 0x59, 0x1B, 0x3C, 0x6C, 0xB3, 0x5F,
/*00F000:*/ 0xCF, 0x7E, 0x80, 0xB6, 0x38, 0x6D, 0xFD, 0x2D, 0x65,
0xF8, 0xDE, 0x27, 0xEC, 0xC6, 0xB5, 0x4C,
/*00F010:*/ 0x48, 0x05, 0x13, 0xBF, 0x15, 0x5D, 0x3E, 0xDE, 0x63,
0xAE, 0xE3, 0x1C, 0x02, 0x44, 0x8C, 0xB9,
/*00F020:*/ 0x1A, 0x22, 0xCB, 0x38, 0xD4, 0x0E, 0x6D, 0x00, 0x87,
0x35, 0xDF, 0x32, 0x5B, 0x25, 0xF1, 0x26,
/*00F030:*/ 0xE0, 0x9C, 0x6D, 0x48, 0x9E, 0x3A, 0xCF, 0x5B, 0x59,
0x72, 0x6E, 0xA4, 0x5B, 0x1A, 0x7E, 0x64,
/*00F040:*/ 0x70, 0x35, 0xAC, 0xDE, 0x01, 0xD7, 0x5F, 0xC0, 0x75,
0x80, 0x94, 0x83, 0xBA, 0x5F, 0x94, 0x98,
/*00F050:*/ 0xDF, 0xC7, 0x1C, 0x7D, 0x1B, 0xA6, 0x0E, 0x00, 0x86,
0xBD, 0xE9, 0x78, 0x77, 0xA4, 0xCE, 0x28,
/*00F060:*/ 0x0E, 0xFF, 0x87, 0x5A, 0xF4, 0xE3, 0x73, 0xBE, 0x83,
0xB9, 0x54, 0x82, 0xB2, 0x0E, 0xC0, 0x25,
/*00F070:*/ 0x7E, 0x03, 0x73, 0x1D, 0x88, 0x8D, 0xF5, 0x4B, 0xC2,
0xEF, 0x17, 0x8C, 0xFC, 0xA4, 0x4A, 0xBD,
/*00F080:*/ 0xD8, 0x7C, 0x3C, 0x08, 0x33, 0x67, 0x62, 0x42, 0x44,
0x09, 0xC7, 0xED, 0xA8, 0x5F, 0xB1, 0x36,
/*00F090:*/ 0x0C, 0x48, 0x3A, 0x3A, 0xF2, 0xD1, 0x99, 0xBC, 0xA0,
0x4F, 0x6B, 0xA5, 0x8E, 0xB3, 0xD6, 0xCA,
/*00F0A0:*/ 0x3D, 0x76, 0x22, 0xB1, 0x34, 0x2E, 0x7D, 0x84, 0x0B,
0x91, 0x90, 0x54, 0x02, 0x6F, 0xC8, 0xEA,
/*00F0B0:*/ 0x9C, 0x26, 0xF2, 0x99, 0xB6, 0x45, 0x9D, 0x23, 0x16,
0xCE, 0xEE, 0xEB, 0x3B, 0xE7, 0x29, 0xC7,
/*00F0C0:*/ 0x28, 0x4C, 0x01, 0x90, 0x12, 0xE9, 0xB6, 0xBE, 0xFA,
0xCE, 0xAF, 0xA5, 0x8C, 0xD7, 0x34, 0x6A,
/*00F0D0:*/ 0x59, 0x03, 0xA8, 0x1A, 0xA9, 0x86, 0xA4, 0x12, 0x24,
0x1A, 0xBF, 0x03, 0xC3, 0x37, 0xAD, 0x8F,
/*00F0E0:*/ 0x3D, 0x72, 0xD9, 0x4A, 0xD3, 0x81, 0x65, 0xE4, 0xA2,
0x54, 0x67, 0x83, 0x1C, 0xBF, 0x64, 0xE0,
/*00F0F0:*/ 0x55, 0x59, 0xA8, 0xF2, 0x72, 0xCE, 0xA1, 0xE6, 0x30,
0x52, 0xB4, 0xEB, 0x15, 0x04, 0x21, 0x64,
/*00F100:*/ 0x34, 0x2E, 0x7D, 0x84, 0x92, 0x26, 0x3D, 0x12, 0x66,
0x7B, 0xFC, 0x02, 0x45, 0xCF, 0xD3, 0x8B,
/*00F110:*/ 0x53, 0xE5, 0xF5, 0x0D, 0xCC, 0x79, 0x89, 0x67, 0x31,
0xAF, 0x65, 0x25, 0x24, 0x0B, 0x90, 0xBC,
/*00F120:*/ 0x2D, 0xEA, 0x0D, 0x1D, 0x16, 0xCE, 0xEE, 0xEB, 0x16,
0xCE, 0xEE, 0xEB, 0xC6, 0x71, 0xAD, 0x6F,
/*00F130:*/ 0x04, 0x47, 0x46, 0x11, 0x0B, 0x91, 0x90, 0x54, 0xD2,
0x1D, 0x28, 0xFD, 0x11, 0x43, 0xBF, 0x52,
/*00F140:*/ 0x34, 0x2E, 0x7D, 0x84, 0xD7, 0x9F, 0x82, 0x50, 0x0C,
0x48, 0x3A, 0x3A, 0x6C, 0x4E, 0xC0, 0x23,
/*00F150:*/ 0x22, 0xA2, 0x25, 0xBA, 0x52, 0x01, 0x30, 0xE2, 0x01,
0xFF, 0x73, 0x9C, 0x27, 0xC0, 0xD0, 0x4C,
/*00F160:*/ 0xE6, 0x1C, 0xB6, 0x87, 0x3D, 0x33, 0x64, 0x8E, 0x01,
0xD7, 0x5F, 0xC0, 0xFC, 0xB8, 0x3C, 0xEB,
/*00F170:*/ 0x8D, 0xCA, 0x5F, 0x62, 0xF4, 0xE3, 0x73, 0xBE, 0x70,
0x6E, 0x67, 0xDF, 0x16, 0xCE, 0xEE, 0xEB,
/*00F180:*/ 0x0A, 0x4A, 0x75, 0x7B, 0x59, 0x03, 0xA8, 0x1A, 0x8E,
0x92, 0x63, 0xB0, 0xBD, 0x8A, 0x9F, 0xA9,
/*00F190:*/ 0x46, 0x7A, 0x21, 0x47, 0x76, 0x0E, 0x7F, 0x97, 0x48,
0x2F, 0x23, 0x3B, 0x88, 0x8D, 0xF5, 0x4B,
/*00F1A0:*/ 0xD2, 0x93, 0xE6, 0x24, 0xDF, 0xC7, 0x1C, 0x7D, 0x7E,
0xB6, 0xC1, 0x8E, 0xD8, 0x2F, 0x7E, 0xCA,
/*00F1B0:*/ 0x5F, 0xF2, 0xFA, 0xF6, 0x01, 0x62, 0x83, 0xE7, 0x16,
0xCE, 0xEE, 0xEB, 0x15, 0x5D, 0x3E, 0xDE,
/*00F1C0:*/ 0x44, 0x09, 0xC7, 0xED, 0xA8, 0x21, 0x62, 0x44, 0x2D,
0xC1, 0xCA, 0x89, 0x63, 0xD9, 0xBE, 0x5C,
/*00F1D0:*/ 0x0A, 0x4A, 0x75, 0x7B, 0x8D, 0x3B, 0x92, 0xB8, 0x95,
0xCB, 0x9B, 0xC3, 0x9C, 0x9C, 0x45, 0xA8,
/*00F1E0:*/ 0x53, 0xE5, 0xF5, 0x0D, 0x74, 0x7A, 0x01, 0x61, 0xD0,
0x4A, 0x7A, 0xFE, 0x12, 0x45, 0xB9, 0xF4,
/*00F1F0:*/ 0x88, 0x5D, 0xD3, 0xB7, 0x78, 0xBE, 0x58, 0x61, 0x7F,
0x7E, 0x86, 0xF1, 0x6C, 0x4E, 0xC0, 0x23,
/*00F200:*/ 0x02, 0x44, 0x8C, 0xB9, 0x8E, 0xB3, 0xD6, 0xCA, 0x0F,
0x10, 0x4F, 0x74, 0x44, 0x09, 0xC7, 0xED,
/*00F210:*/ 0x3B, 0x5C, 0x56, 0x54, 0x12, 0x7A, 0x2C, 0xF2, 0x8E,
0x23, 0x09, 0xF3, 0x4F, 0xA6, 0xC1, 0x11,
/*00F220:*/ 0x9F, 0x3A, 0xA0, 0xC3, 0x05, 0x6E, 0x54, 0x33, 0x0B,
0x91, 0x90, 0x54, 0x3E, 0xF1, 0xB5, 0x10,
/*00F230:*/ 0x4C, 0x79, 0xC2, 0xEB, 0x85, 0x9D, 0xF0, 0x35, 0xFF,
0x15, 0x89, 0x5E, 0xA4, 0xCC, 0xA3, 0x0A,
/*00F240:*/ 0x53, 0xE5, 0xF5, 0x0D, 0xF0, 0x36, 0x3C, 0x9C, 0x59,
0x6B, 0x25, 0xE6, 0xDD, 0x61, 0xC6, 0xD6,
/*00F250:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B6D2_5[] = {
/*00F840:*/ 0x3F, 0x71, 0x9A, 0x6A, 0x59, 0xA6, 0x2F, 0x0B, 0x51,
0x07, 0x91, 0xBA, 0x7C, 0xC3, 0x8D, 0xB5,
/*00F850:*/ 0x8F, 0x08, 0x7C, 0x84, 0xFC, 0xA4, 0x4A, 0xBD, 0x7C,
0xC3, 0x8D, 0xB5, 0x77, 0xF6, 0x5E, 0xC9,
/*00F860:*/ 0xD8, 0x99, 0x2C, 0xB7, 0x59, 0xA6, 0x2F, 0x0B, 0x8D,
0xCA, 0x5F, 0x62, 0xC0, 0x00, 0x3D, 0xCA,
/*00F870:*/ 0xE0, 0x58, 0x7A, 0x6F, 0x48, 0x2F, 0x23, 0x3B, 0x05,
0xAD, 0xAB, 0x34, 0xB4, 0x69, 0x9F, 0x38,
/*00F880:*/ 0xC3, 0x37, 0xAD, 0x8F, 0x17, 0x08, 0x0D, 0x3F, 0xFD,
0x68, 0x02, 0x03, 0x65, 0xF8, 0xDE, 0x27,
/*00F890:*/ 0xDD, 0x87, 0x4E, 0x1D, 0xDE, 0x02, 0x1C, 0xBD, 0xB4,
0x69, 0x9F, 0x38, 0x2F, 0x13, 0xD5, 0xAE,
/*00F8A0:*/ 0xAD, 0x45, 0x93, 0x08, 0xA8, 0xB8, 0x38, 0x0A, 0x63,
0xAE, 0xE3, 0x1C, 0x0A, 0x4A, 0x75, 0x7B,
/*00F8B0:*/ 0x10, 0x20, 0x84, 0x97, 0x55, 0xAF, 0xCB, 0x7D, 0x7E,
0xB6, 0xC1, 0x8E, 0xC0, 0x94, 0x54, 0xC9,
/*00F8C0:*/ 0x23, 0xA4, 0x9B, 0x3E, 0xBB, 0x4B, 0x29, 0x9F, 0x31,
0xAF, 0x65, 0x25, 0x3B, 0x5C, 0x56, 0x54,
/*00F8D0:*/ 0xA8, 0x21, 0x62, 0x44, 0xFD, 0x72, 0xC3, 0xA8, 0xD8,
0x2F, 0x7E, 0xCA, 0xC9, 0x17, 0xFF, 0x0E,
/*00F8E0:*/ 0x70, 0x6E, 0x67, 0xDF, 0x7C, 0x77, 0x4B, 0xD7, 0xDF,
0xC7, 0x1C, 0x7D, 0x51, 0xE1, 0x15, 0xDD,
/*00F8F0:*/ 0x5B, 0x1A, 0x7E, 0x64, 0xD2, 0x93, 0xE6, 0x24, 0x91,
0xB3, 0x84, 0xAB, 0xF3, 0xF7, 0xB5, 0x7E,
/*00F900:*/ 0x28, 0x4C, 0x01, 0x90, 0x5F, 0xF2, 0xFA, 0xF6, 0xD9,
0xB8, 0x25, 0xCB, 0x35, 0x64, 0xCB, 0xAB,
/*00F910:*/ 0x02, 0x44, 0x8C, 0xB9, 0xE6, 0xD3, 0x65, 0xF8, 0x6F,
0xEE, 0xE7, 0x2D, 0x9E, 0xD8, 0xF8, 0x66,
/*00F920:*/ 0x2F, 0x67, 0xDA, 0xAC, 0x67, 0xA2, 0xA9, 0x96, 0x53,
0xE5, 0xF5, 0x0D, 0x93, 0x60, 0x00, 0x4C,
/*00F930:*/ 0x3C, 0x6C, 0xB3, 0x5F, 0xA7, 0x54, 0x4C, 0xAD, 0xC4,
0xC9, 0xED, 0x05, 0x55, 0xA1, 0xE1, 0x6B,
/*00F940:*/ 0x58, 0xF3, 0x41, 0xBF, 0x3B, 0xFB, 0x95, 0x81, 0x04,
0xEA, 0xA7, 0x28, 0x72, 0xCE, 0xA1, 0xE6,
/*00F950:*/ 0xC8, 0xC4, 0xD0, 0xA3, 0x35, 0xC4, 0xB2, 0x1B, 0x88,
0x5D, 0xD3, 0xB7, 0x4A, 0xDF, 0x4D, 0xF0,
/*00F960:*/ 0x87, 0xA3, 0x0D, 0x5B, 0xB4, 0x69, 0x9F, 0x38, 0xE6,
0x1C, 0xB6, 0x87, 0x7F, 0x45, 0xF7, 0x36,
/*00F970:*/ 0x65, 0x1C, 0x74, 0x7F, 0x4C, 0x79, 0xC2, 0xEB, 0x05,
0xAD, 0xAB, 0x34, 0x58, 0xF3, 0x41, 0xBF,
/*00F980:*/ 0xFD, 0x68, 0x02, 0x03, 0xB0, 0x93, 0x08, 0x65, 0xC3,
0x79, 0x85, 0xAB, 0x15, 0x52, 0x7C, 0xB8,
/*00F990:*/ 0x3E, 0x2C, 0x92, 0xD5, 0xAD, 0xBE, 0xFA, 0xE0, 0x16,
0x42, 0x66, 0x51, 0x0F, 0x10, 0x4F, 0x74,
/*00F9A0:*/ 0xA0, 0xAF, 0xBB, 0xFC, 0x95, 0xCB, 0x9B, 0xC3, 0x1E,
0xAA, 0x3D, 0x3F, 0xD3, 0x37, 0xFF, 0xEE,
/*00F9B0:*/ 0x57, 0x89, 0xDA, 0xB2, 0xEF, 0x02, 0xF1, 0x1F, 0x5D,
0x65, 0xB5, 0x0A, 0x01, 0xD7, 0x5F, 0xC0,
/*00F9C0:*/ 0xC8, 0xC4, 0xD0, 0xA3, 0x82, 0x97, 0x72, 0x52, 0x69,
0x28, 0xA5, 0xF2, 0x7C, 0x72, 0x7A, 0xCD,
/*00F9D0:*/ 0x1A, 0x22, 0xCB, 0x38, 0xB4, 0x1E, 0x49, 0xD5, 0xF4,
0x60, 0xD8, 0x66, 0x63, 0xD9, 0xBE, 0x5C,
/*00F9E0:*/ 0xF3, 0xF7, 0xB5, 0x7E, 0x04, 0x47, 0x46, 0x11, 0x8E,
0x11, 0x1D, 0xA3, 0x04, 0xEA, 0xA7, 0x28,
/*00F9F0:*/ 0x03, 0x59, 0x9A, 0x4E, 0x9E, 0xE0, 0xC6, 0x8C, 0x1A,
0x22, 0xCB, 0x38, 0x31, 0xDB, 0xF4, 0x11,
/*00FA00:*/ 0xD2, 0x1D, 0x28, 0xFD, 0xBF, 0xB4, 0xD2, 0x8F, 0xA0,
0x4F, 0x6B, 0xA5, 0x48, 0x05, 0x13, 0xBF,
/*00FA10:*/ 0x93, 0x60, 0x00, 0x4C, 0x9C, 0xF1, 0x52, 0x83, 0xE3,
0xA9, 0xC2, 0xC3, 0xD4, 0x56, 0x59, 0x6D,
/*00FA20:*/ 0x83, 0x55, 0x2F, 0xE2, 0xDD, 0x61, 0xC6, 0xD6, 0xD8,
0x7C, 0x3C, 0x08, 0x4F, 0xA6, 0xC1, 0x11,
/*00FA30:*/ 0xB2, 0x0E, 0xC0, 0x25, 0x0B, 0x91, 0x90, 0x54, 0x0B,
0xD9, 0x3A, 0x41, 0x2F, 0x13, 0xD5, 0xAE,
/*00FA40:*/ 0x4F, 0xA6, 0xC1, 0x11, 0x57, 0xB3, 0xF9, 0x8C, 0x5A,
0x4A, 0x0B, 0x7B, 0xEC, 0x88, 0xC7, 0xDB,
/*00FA50:*/ 0x65, 0xF8, 0xDE, 0x27, 0xAD, 0x45, 0x93, 0x08, 0x76,
0x0E, 0x7F, 0x97, 0x4F, 0xA6, 0xC1, 0x11,
/*00FA60:*/ 0x9E, 0xBD, 0x9F, 0x16, 0xC0, 0x94, 0x54, 0xC9, 0x5D,
0xB1, 0xE3, 0x15, 0x6F, 0xEE, 0xE7, 0x2D,
/*00FA70:*/ 0xD2, 0x1D, 0x28, 0xFD, 0x35, 0x64, 0xCB, 0xAB, 0x3E,
0x77, 0xA8, 0x9D, 0x40, 0xB3, 0x1B, 0x9E,
/*00FA80:*/ 0xE0, 0x58, 0x7A, 0x6F, 0x0D, 0xF0, 0xAD, 0xBA, 0x0D,
0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*00FA90:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B6D2_6[] = {
/*0290:*/ 0x9A, 0x45, 0xD2, 0xAA, 0x9E, 0x3A, 0xCF, 0x5B, 0xC6,
0x71, 0xAD, 0x6F, 0x58, 0x1F, 0xAC, 0x46,
/*02A0:*/ 0x5A, 0x87, 0x35, 0x2F, 0x15, 0x5D, 0x3E, 0xDE, 0x23,
0xA4, 0x9B, 0x3E, 0xD2, 0x1D, 0x28, 0xFD,
/*02B0:*/ 0xE0, 0x9C, 0x6D, 0x48, 0x63, 0x8F, 0xBC, 0x37, 0x31,
0xDB, 0xF4, 0x11, 0xC4, 0x28, 0x18, 0xE6,
/*02C0:*/ 0x59, 0x72, 0x6E, 0xA4, 0x38, 0x3C, 0x18, 0x72, 0x34,
0x2E, 0x7D, 0x84, 0xA8, 0xB8, 0x38, 0x0A,
/*02D0:*/ 0x6C, 0x3C, 0x44, 0x49, 0xC5, 0x0F, 0xC0, 0xC6, 0x53,
0x69, 0x4A, 0xC8, 0xC1, 0xF6, 0x2F, 0x66,
/*02E0:*/ 0x9E, 0xE0, 0xC6, 0x8C, 0xF4, 0x60, 0xD8, 0x66, 0x9C,
0x26, 0xF2, 0x99, 0x04, 0xEA, 0xA7, 0x28,
/*02F0:*/ 0x55, 0x59, 0xA8, 0xF2, 0x04, 0x47, 0x46, 0x11, 0x0F,
0x10, 0x4F, 0x74, 0x3E, 0x77, 0xA8, 0x9D,
/*0300:*/ 0x72, 0x38, 0xA6, 0x78, 0x44, 0x09, 0xC7, 0xED, 0xC5,
0x0F, 0xC0, 0xC6, 0x64, 0xA6, 0xD4, 0xF0,
/*0310:*/ 0xF6, 0x19, 0x5C, 0x46, 0xF4, 0x2A, 0x00, 0x0E, 0x53,
0xE5, 0xF5, 0x0D, 0xF4, 0x2A, 0x00, 0x0E,
/*0320:*/ 0x7B, 0xF8, 0xDA, 0x1E, 0x72, 0xCE, 0xA1, 0xE6, 0x20,
0x2B, 0x8D, 0x19, 0xC9, 0x17, 0xFF, 0x0E,
/*0330:*/ 0xE8, 0x83, 0x83, 0x7C, 0x4F, 0xA6, 0xC1, 0x11, 0x3E,
0x77, 0xA8, 0x9D, 0x88, 0x5D, 0xD3, 0xB7,
/*0340:*/ 0xBD, 0x8A, 0x9F, 0xA9, 0xAD, 0x45, 0x93, 0x08, 0x4B,
0x70, 0xE9, 0x42, 0x72, 0xCE, 0xA1, 0xE6,
/*0350:*/ 0xB9, 0xB0, 0x7A, 0x35, 0x78, 0xBE, 0x58, 0x61, 0x01,
0xD7, 0x5F, 0xC0, 0x48, 0xF6, 0x2D, 0x5F,
/*0360:*/ 0xE0, 0x58, 0x7A, 0x6F, 0x91, 0x24, 0x1D, 0x9F, 0xE6,
0x1C, 0xB6, 0x87, 0x5B, 0x3A, 0xD3, 0x95,
/*0370:*/ 0x25, 0x32, 0x51, 0x73, 0x05, 0xB0, 0x7F, 0x45, 0xF6,
0x90, 0xE9, 0xDC, 0x06, 0xFD, 0x74, 0x0A,
/*0380:*/ 0x95, 0xCB, 0x9B, 0xC3, 0x12, 0x7A, 0x2C, 0xF2, 0xD2,
0x1D, 0x28, 0xFD, 0x05, 0xAD, 0xAB, 0x34,
/*0390:*/ 0x22, 0xE4, 0x16, 0xC9, 0x4B, 0x91, 0x59, 0xAA, 0x6C,
0x4E, 0xC0, 0x23, 0x59, 0x6B, 0x25, 0xE6,
/*03A0:*/ 0x91, 0x24, 0x1D, 0x9F, 0xC0, 0xA2, 0x3F, 0xEA, 0x85,
0x9D, 0xF0, 0x35, 0x12, 0x45, 0xB9, 0xF4,
/*03B0:*/ 0x94, 0xE8, 0xA4, 0xD1, 0x17, 0x08, 0x0D, 0x3F, 0xF5,
0xF5, 0xB0, 0x30, 0x4C, 0x03, 0x12, 0x32,
/*03C0:*/ 0xE6, 0xD3, 0x65, 0xF8, 0xE4, 0x75, 0xCD, 0x29, 0x0C,
0x48, 0x3A, 0x3A, 0x5B, 0x1A, 0x7E, 0x64,
/*03D0:*/ 0x84, 0xE6, 0x91, 0xA6, 0x32, 0x39, 0x17, 0x0C, 0x0C,
0x48, 0x3A, 0x3A, 0xCF, 0xC9, 0xD5, 0x98,
/*03E0:*/ 0xE8, 0x83, 0x83, 0x7C, 0xD8, 0x7C, 0x3C, 0x08, 0x19,
0x76, 0xC1, 0xDE, 0x3F, 0x17, 0x18, 0xCB,
/*03F0:*/ 0x3C, 0x6C, 0xB3, 0x5F, 0xC6, 0x71, 0xAD, 0x6F, 0x70,
0x35, 0xAC, 0xDE, 0x84, 0xE6, 0x91, 0xA6,
/*0400:*/ 0x02, 0x44, 0x8C, 0xB9, 0x34, 0x2E, 0x7D, 0x84, 0xE5,
0x0C, 0x47, 0x24, 0xE4, 0x4E, 0x30, 0x3A,
/*0410:*/ 0x35, 0xC4, 0xB2, 0x1B, 0xF5, 0xB7, 0xF4, 0x43, 0x8E,
0x92, 0x63, 0xB0, 0xF0, 0x25, 0xD0, 0x02,
/*0420:*/ 0xE0, 0x9C, 0x6D, 0x48, 0xC8, 0xF7, 0x44, 0xFE, 0x6A,
0xAB, 0x24, 0xB1, 0x1E, 0xAA, 0x3D, 0x3F,
/*0430:*/ 0x7C, 0x77, 0x4B, 0xD7, 0xF2, 0xD1, 0x99, 0xBC, 0xFC,
0xB8, 0x3C, 0xEB, 0xE2, 0x6D, 0x41, 0xB7,
/*0440:*/ 0x88, 0x59, 0xC4, 0x4D, 0x65, 0xF8, 0xDE, 0x27, 0x0B,
0xD9, 0x3A, 0x41, 0x30, 0x52, 0xB4, 0xEB,
/*0450:*/ 0xAC, 0xD9, 0x82, 0x35, 0xD5, 0x80, 0xA1, 0x41, 0x04,
0xEA, 0xA7, 0x28, 0x69, 0x28, 0xA5, 0xF2,
/*0460:*/ 0x24, 0x0B, 0x90, 0xBC, 0x38, 0x3C, 0x18, 0x72, 0x55,
0xAF, 0xCB, 0x7D, 0x8C, 0x7F, 0x26, 0x4A,
/*0470:*/ 0x6C, 0xD3, 0xE5, 0x55, 0xC5, 0x21, 0xEF, 0xFE, 0xD5,
0x80, 0xA1, 0x41, 0x6D, 0x08, 0x98, 0x24,
/*0480:*/ 0xDF, 0xC7, 0x1C, 0x7D, 0xAD, 0xBE, 0xFA, 0xE0, 0x32,
0x39, 0x17, 0x0C, 0xE6, 0x05, 0x3C, 0xDF,
/*0490:*/ 0xCF, 0x7E, 0x80, 0xB6, 0x28, 0x4C, 0x01, 0x90, 0xCC,
0x79, 0x89, 0x67, 0xB2, 0x0E, 0xC0, 0x25,
/*04A0:*/ 0x8E, 0x92, 0x63, 0xB0, 0x12, 0xD1, 0x59, 0x1B, 0x8D,
0xCA, 0x5F, 0x62, 0xDC, 0xBD, 0x76, 0x19,
/*04B0:*/ 0x94, 0xAC, 0xDE, 0xF1, 0x03, 0x02, 0xB4, 0x2D, 0x58,
0xF3, 0x41, 0xBF, 0x0D, 0xF0, 0xAD, 0xBA,
/*04C0:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B6D2_7[] = {
/*09F8:*/ 0xA8, 0x21, 0x62, 0x44, 0x41, 0x0F, 0x82, 0xFB,
/*0A00:*/ 0x39, 0xAA, 0xBF, 0x6C, 0x57, 0x89, 0xDA, 0xB2, 0xFA,
0xCE, 0xAF, 0xA5, 0x51, 0x07, 0x91, 0xBA,
/*0A10:*/ 0xF4, 0x2A, 0x00, 0x0E, 0x28, 0xD3, 0xF0, 0xAA, 0x0C,
0x48, 0x3A, 0x3A, 0xF4, 0x2A, 0x00, 0x0E,
/*0A20:*/ 0x59, 0x72, 0x6E, 0xA4, 0xBF, 0xB4, 0xD2, 0x8F, 0x76,
0x0E, 0x7F, 0x97, 0x92, 0x82, 0xF5, 0xB9,
/*0A30:*/ 0x23, 0xA4, 0x9B, 0x3E, 0xC8, 0xF7, 0x44, 0xFE, 0x63,
0xAE, 0xE3, 0x1C, 0x03, 0x02, 0xB4, 0x2D,
/*0A40:*/ 0x69, 0x28, 0xA5, 0xF2, 0xD8, 0x7C, 0x3C, 0x08, 0xB9,
0xB0, 0x7A, 0x35, 0x19, 0x76, 0xC1, 0xDE,
/*0A50:*/ 0xCA, 0x62, 0xA6, 0x46, 0x8C, 0xBA, 0x04, 0x05, 0x3E,
0x2C, 0x92, 0xD5, 0x50, 0x45, 0x07, 0xF0,
/*0A60:*/ 0xD2, 0x43, 0x93, 0xFD, 0xA0, 0x18, 0x0B, 0x31, 0x6A,
0x19, 0xF9, 0x2F, 0xA0, 0x4F, 0x6B, 0xA5,
/*0A70:*/ 0x8D, 0x3B, 0x92, 0xB8, 0x8C, 0xBA, 0x04, 0x05, 0x34,
0x2E, 0x7D, 0x84, 0xE6, 0x1C, 0xB6, 0x87,
/*0A80:*/ 0x92, 0x26, 0x3D, 0x12, 0xCB, 0x41, 0x4C, 0xDA, 0x58,
0xF3, 0x41, 0xBF, 0x8C, 0xBA, 0x04, 0x05,
/*0A90:*/ 0xD5, 0x80, 0xA1, 0x41, 0x28, 0x4C, 0x01, 0x90, 0xD7,
0xE3, 0xE9, 0x28, 0x6F, 0x1B, 0x63, 0xB3,
/*0AA0:*/ 0x6E, 0x35, 0xFA, 0x9F, 0x50, 0xEE, 0xF3, 0x99, 0x27,
0xC0, 0xD0, 0x4C, 0x3D, 0x72, 0xD9, 0x4A,
/*0AB0:*/ 0x55, 0xAF, 0xCB, 0x7D, 0xFD, 0x68, 0x02, 0x03, 0xDB,
0xEA, 0xEC, 0x30, 0xEA, 0xA0, 0x14, 0x88,
/*0AC0:*/ 0xDC, 0xBD, 0x76, 0x19, 0xC8, 0xF7, 0x44, 0xFE, 0x55,
0xAF, 0xCB, 0x7D, 0x55, 0xA1, 0xE1, 0x6B,
/*0AD0:*/ 0x28, 0x4C, 0x01, 0x90, 0x16, 0xCE, 0xEE, 0xEB, 0x55,
0xAF, 0xCB, 0x7D, 0x59, 0x03, 0xA8, 0x1A,
/*0AE0:*/ 0x53, 0xE5, 0xF5, 0x0D, 0xEF, 0x02, 0xF1, 0x1F, 0xAE,
0xC0, 0x21, 0x9D, 0x9C, 0x26, 0xF2, 0x99,
/*0AF0:*/ 0x5B, 0x1A, 0x7E, 0x64, 0xDA, 0x4D, 0x0A, 0x23, 0xC4,
0x68, 0xAC, 0x83, 0xEB, 0x76, 0x06, 0x7B,
/*0B00:*/ 0xDC, 0xBD, 0x76, 0x19, 0x35, 0xC4, 0xB2, 0x1B, 0x12,
0xD1, 0x59, 0x1B, 0x12, 0xD1, 0x59, 0x1B,
/*0B10:*/ 0x0A, 0x4A, 0x75, 0x7B, 0xC8, 0xF7, 0x44, 0xFE, 0x8E,
0x92, 0x63, 0xB0, 0x70, 0xF3, 0x21, 0x92,
/*0B20:*/ 0x22, 0xA2, 0x25, 0xBA, 0x4C, 0x2C, 0x53, 0x9D, 0x01,
0x62, 0x83, 0xE7, 0xC8, 0xF7, 0x44, 0xFE,
/*0B30:*/ 0x12, 0x7A, 0x2C, 0xF2, 0xE6, 0x05, 0x3C, 0xDF, 0xB0,
0x93, 0x08, 0x65, 0xB9, 0xB0, 0x7A, 0x35,
/*0B40:*/ 0x0C, 0x48, 0x3A, 0x3A, 0xC8, 0xF7, 0x44, 0xFE, 0x7E,
0x03, 0x73, 0x1D, 0x5F, 0xE8, 0xC6, 0x9E,
/*0B50:*/ 0xC8, 0xF7, 0x44, 0xFE, 0x41, 0x0F, 0x82, 0xFB, 0x3F,
0x71, 0x9A, 0x6A, 0xB9, 0xB0, 0x7A, 0x35,
/*0B60:*/ 0xFC, 0x8A, 0x20, 0xF0, 0xE0, 0x58, 0x7A, 0x6F, 0x63,
0xD9, 0xBE, 0x5C, 0xF3, 0x37, 0xA7, 0x4A,
/*0B70:*/ 0x3D, 0x76, 0x22, 0xB1, 0xC9, 0x9A, 0xC6, 0xDD, 0x05,
0xAD, 0xAB, 0x34, 0xAC, 0xD9, 0x82, 0x35,
/*0B80:*/ 0x9E, 0xD8, 0xF8, 0x66, 0xE5, 0xF7, 0xC5, 0xBD, 0x04,
0xEA, 0xA7, 0x28, 0x8E, 0x23, 0x09, 0xF3,
/*0B90:*/ 0xC5, 0x0F, 0xC0, 0xC6, 0x85, 0x5E, 0xED, 0xBB, 0x04,
0x47, 0x46, 0x11, 0x74, 0x39, 0x6F, 0x2B,
/*0BA0:*/ 0xB5, 0x99, 0x8F, 0xB7, 0xEB, 0x76, 0x06, 0x7B, 0x86,
0xBD, 0xE9, 0x78, 0xD2, 0x1D, 0x28, 0xFD,
/*0BB0:*/ 0x20, 0x2B, 0x8D, 0x19, 0x65, 0xF8, 0xDE, 0x27, 0x91,
0x24, 0x1D, 0x9F, 0x31, 0xEA, 0x4D, 0xAD,
/*0BC0:*/ 0x5E, 0x8C, 0x7B, 0xD8, 0x35, 0x64, 0xCB, 0xAB, 0x4A,
0xDF, 0x4D, 0xF0, 0x53, 0xE5, 0xF5, 0x0D,
/*0BD0:*/ 0xA9, 0x20, 0xF4, 0x24, 0x84, 0x8E, 0xCA, 0xB6, 0xC2,
0xEF, 0x17, 0x8C, 0xB4, 0x69, 0x9F, 0x38,
/*0BE0:*/ 0x50, 0xEE, 0xF3, 0x99, 0xF4, 0xE3, 0x73, 0xBE, 0xB6,
0x14, 0x7E, 0x68, 0xE3, 0xA9, 0xC2, 0xC3,
/*0BF0:*/ 0x91, 0x24, 0x1D, 0x9F, 0x82, 0x97, 0x72, 0x52, 0x92,
0x26, 0x3D, 0x12, 0x3B, 0x5C, 0x56, 0x54,
/*0C00:*/ 0xC4, 0x46, 0x11, 0x47, 0x37, 0x83, 0x14, 0x26, 0x4B,
0x91, 0x59, 0xAA, 0x6E, 0x35, 0xFA, 0x9F,
/*0C10:*/ 0x75, 0x80, 0x94, 0x83, 0xCB, 0x53, 0xD3, 0xFF, 0x92,
0x26, 0x3D, 0x12, 0xA3, 0xAE, 0xC3, 0xFA,
/*0C20:*/ 0xC0, 0x94, 0x54, 0xC9, 0x68, 0x36, 0xD9, 0x72, 0x32,
0x39, 0x17, 0x0C, 0xFA, 0xCE, 0xAF, 0xA5,
/*0C30:*/ 0x10, 0x20, 0x84, 0x97, 0x01, 0xD7, 0x5F, 0xC0, 0xA7,
0x54, 0x4C, 0xAD, 0x31, 0xEA, 0x4D, 0xAD,
/*0C40:*/ 0xC8, 0xF7, 0x44, 0xFE, 0xFA, 0xCE, 0xAF, 0xA5, 0x4D,
0x74, 0xF8, 0x85, 0x55, 0xA1, 0xE1, 0x6B,
/*0C50:*/ 0xBE, 0x61, 0x2C, 0xF5, 0x56, 0xF8, 0x83, 0x82, 0x53,
0xE5, 0xF5, 0x0D, 0x05, 0xAD, 0xAB, 0x34,
/*0C60:*/ 0x02, 0x44, 0x8C, 0xB9, 0x1A, 0xD8, 0xBC, 0x1F, 0x0B,
0x91, 0x90, 0x54, 0x6F, 0xEE, 0xE7, 0x2D,
/*0C70:*/ 0x5D, 0x65, 0xB5, 0x0A, 0x2D, 0xEA, 0x0D, 0x1D, 0x04,
0xDE, 0x4C, 0x7C, 0xDC, 0xBD, 0x76, 0x19,
/*0C80:*/ 0x3F, 0xF5, 0x3C, 0x3E, 0x70, 0xF3, 0x21, 0x92, 0xA7,
0x54, 0x4C, 0xAD, 0x01, 0xD7, 0x5F, 0xC0,
/*0C90:*/ 0x3E, 0x77, 0xA8, 0x9D, 0x87, 0xA3, 0x0D, 0x5B, 0xE6,
0x05, 0x3C, 0xDF, 0x3F, 0x71, 0x9A, 0x6A,
/*0CA0:*/ 0x85, 0x9D, 0xF0, 0x35, 0x93, 0x60, 0x00, 0x4C, 0xA8,
0x21, 0x62, 0x44, 0x0D, 0xF0, 0xAD, 0xBA,
/*0CB0:*/ 0x0D, 0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*0CC0:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B6D2_8[] = {
/*1210:*/ 0x8D, 0x6C, 0x70, 0x47, 0x44, 0x09, 0xC7, 0xED, 0xC8,
0xC4, 0xD0, 0xA3, 0xDC, 0xBD, 0x76, 0x19,
/*1220:*/ 0x38, 0x8D, 0x3D, 0x7F, 0x0C, 0xA4, 0x6C, 0xFC, 0x2F,
0x67, 0xDA, 0xAC, 0x23, 0xEB, 0x14, 0x0E,
/*1230:*/ 0x6A, 0xAB, 0x24, 0xB1, 0xC4, 0x68, 0xAC, 0x83, 0xCB,
0x53, 0xD3, 0xFF, 0x88, 0x8D, 0xF5, 0x4B,
/*1240:*/ 0x6C, 0x4E, 0xC0, 0x23, 0x8F, 0x08, 0x7C, 0x84, 0x06,
0x8C, 0x28, 0x9D, 0xF4, 0x2A, 0x00, 0x0E,
/*1250:*/ 0x4A, 0xDF, 0x4D, 0xF0, 0xCC, 0x02, 0xEB, 0xA5, 0x12,
0xD1, 0x59, 0x1B, 0xFC, 0xA4, 0x4A, 0xBD,
/*1260:*/ 0x12, 0xD1, 0x59, 0x1B, 0xD8, 0x99, 0x2C, 0xB7, 0xC0,
0x94, 0x54, 0xC9, 0x03, 0x59, 0x9A, 0x4E,
/*1270:*/ 0xA3, 0xBD, 0x45, 0x9C, 0x30, 0x44, 0x25, 0xFF, 0x99,
0x7B, 0x6F, 0xC5, 0x2E, 0xBE, 0x99, 0x34,
/*1280:*/ 0x4C, 0x79, 0xC2, 0xEB, 0xC3, 0x79, 0x85, 0xAB, 0x05,
0xAD, 0xAB, 0x34, 0x2D, 0xEA, 0x0D, 0x1D,
/*1290:*/ 0x51, 0x07, 0x91, 0xBA, 0xE0, 0x32, 0x37, 0xB3, 0x2F,
0x13, 0xD5, 0xAE, 0xA9, 0x86, 0xA4, 0x12,
/*12A0:*/ 0x8C, 0xBA, 0x04, 0x05, 0x29, 0x78, 0x60, 0x9B, 0xFC,
0xB8, 0x3C, 0xEB, 0xF0, 0x36, 0x3C, 0x9C,
/*12B0:*/ 0xED, 0x06, 0x1B, 0xEA, 0x12, 0xD1, 0x59, 0x1B, 0xA3,
0xAE, 0xC3, 0xFA, 0xB2, 0x0E, 0xC0, 0x25,
/*12C0:*/ 0x05, 0x7B, 0xD6, 0xA2, 0xCB, 0xCC, 0xC1, 0xC4, 0x1B,
0xBE, 0xBA, 0x74, 0xDF, 0xC7, 0x1C, 0x7D,
/*12D0:*/ 0xBD, 0x8A, 0x9F, 0xA9, 0xD8, 0x99, 0x2C, 0xB7, 0x05,
0xAD, 0xAB, 0x34, 0x3D, 0x72, 0xD9, 0x4A,
/*12E0:*/ 0xEB, 0x76, 0x06, 0x7B, 0xAD, 0x45, 0x93, 0x08, 0xAC,
0xD9, 0x82, 0x35, 0x62, 0x0B, 0x86, 0x44,
/*12F0:*/ 0x6C, 0x3C, 0x44, 0x49, 0xC6, 0x71, 0xAD, 0x6F, 0x3B,
0x22, 0x07, 0x87, 0x1A, 0x22, 0xCB, 0x38,
/*1300:*/ 0xEE, 0x5A, 0x8F, 0x21, 0xA3, 0xBD, 0x45, 0x9C, 0xCE,
0x2D, 0x8A, 0xF8, 0x01, 0xD7, 0x5F, 0xC0,
/*1310:*/ 0x6A, 0x19, 0xF9, 0x2F, 0xAD, 0x45, 0x93, 0x08, 0xD2,
0x43, 0x93, 0xFD, 0x02, 0x6D, 0x90, 0x88,
/*1320:*/ 0x65, 0xF8, 0xDE, 0x27, 0x77, 0xC8, 0x8F, 0x75, 0x7A,
0x1A, 0xB6, 0x8B, 0x6A, 0xAB, 0x24, 0xB1,
/*1330:*/ 0x3E, 0x77, 0xA8, 0x9D, 0x29, 0x78, 0x60, 0x9B, 0x0B,
0x91, 0x90, 0x54, 0xC3, 0x37, 0xAD, 0x8F,
/*1340:*/ 0x3D, 0x72, 0xD9, 0x4A, 0x3E, 0x77, 0xA8, 0x9D, 0xEE,
0x5A, 0x8F, 0x21, 0x86, 0xBD, 0xE9, 0x78,
/*1350:*/ 0x4C, 0x79, 0xC2, 0xEB, 0x76, 0x0E, 0x7F, 0x97, 0xB0,
0xF5, 0x9F, 0x48, 0x58, 0x1F, 0xAC, 0x46,
/*1360:*/ 0x91, 0xB3, 0x84, 0xAB, 0x6A, 0xAB, 0x24, 0xB1, 0x0E,
0xFF, 0x87, 0x5A, 0x3B, 0xFB, 0x95, 0x81,
/*1370:*/ 0x22, 0xA7, 0xE7, 0x6E, 0x99, 0x45, 0xAA, 0x43, 0x6F,
0x1B, 0x63, 0xB3, 0x11, 0x43, 0xBF, 0x52,
/*1380:*/ 0x77, 0xA4, 0xCE, 0x28, 0x8C, 0xBA, 0x04, 0x05, 0x55,
0xA1, 0xE1, 0x6B, 0xDC, 0xBD, 0x76, 0x19,
/*1390:*/ 0xB9, 0xB0, 0x7A, 0x35, 0x05, 0x6E, 0x54, 0x33, 0xFC,
0xB8, 0x3C, 0xEB, 0x01, 0x62, 0x83, 0xE7,
/*13A0:*/ 0x6A, 0x19, 0xF9, 0x2F, 0xA0, 0x4F, 0x6B, 0xA5, 0xC3,
0x68, 0x4D, 0xA0, 0xA5, 0x1F, 0xF8, 0x3B,
/*13B0:*/ 0xC8, 0xF7, 0x44, 0xFE, 0xAE, 0xBA, 0x7D, 0xFF, 0x3E,
0xCA, 0xD9, 0x17, 0x52, 0x14, 0xC3, 0xE8,
/*13C0:*/ 0x34, 0x2E, 0x7D, 0x84, 0xFC, 0x8A, 0x20, 0xF0, 0xF4,
0xE3, 0x73, 0xBE, 0x17, 0x08, 0x0D, 0x3F,
/*13D0:*/ 0x02, 0x44, 0x8C, 0xB9, 0x3F, 0x17, 0x18, 0xCB, 0x7D,
0x8C, 0xFD, 0x17, 0xB5, 0x99, 0x8F, 0xB7,
/*13E0:*/ 0xAD, 0xBE, 0xFA, 0xE0, 0x5E, 0xAC, 0xB2, 0x86, 0x6C,
0x4E, 0xC0, 0x23, 0xC5, 0x3F, 0x56, 0x06,
/*13F0:*/ 0xB6, 0x14, 0x7E, 0x68, 0x74, 0x48, 0x37, 0x81, 0xD8,
0x99, 0x2C, 0xB7, 0xC0, 0x00, 0x3D, 0xCA,
/*1400:*/ 0xCA, 0x62, 0xA6, 0x46, 0x59, 0x03, 0xA8, 0x1A, 0xA0,
0x18, 0x0B, 0x31, 0x3D, 0x72, 0xD9, 0x4A,
/*1410:*/ 0xA4, 0x60, 0x41, 0xA8, 0x16, 0x42, 0x66, 0x51, 0xAB,
0xD1, 0xD3, 0x0E, 0xCF, 0x7E, 0x80, 0xB6,
/*1420:*/ 0x8E, 0x92, 0x63, 0xB0, 0xDF, 0xC7, 0x1C, 0x7D, 0xF4,
0xE3, 0x73, 0xBE, 0x01, 0x62, 0x83, 0xE7,
/*1430:*/ 0xAD, 0xD2, 0xB8, 0x48, 0x60, 0x43, 0xCC, 0x0E, 0x44,
0x67, 0x6E, 0x3D, 0x76, 0x0E, 0x7F, 0x97,
/*1440:*/ 0x05, 0x7B, 0xD6, 0xA2, 0xF5, 0xB7, 0xF4, 0x43, 0x4D,
0x01, 0x6B, 0x7A, 0x83, 0x55, 0x2F, 0xE2,
/*1450:*/ 0x87, 0xA3, 0x0D, 0x5B, 0x70, 0xF3, 0x21, 0x92, 0x55,
0xAF, 0xCB, 0x7D, 0x8D, 0x3B, 0x92, 0xB8,
/*1460:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B6D2_9[] = {
/*1A08:*/ 0x3B, 0x5C, 0x56, 0x54, 0x5F, 0xF2, 0xFA, 0xF6,
/*1A10:*/ 0x33, 0x94, 0x16, 0xD9, 0x65, 0x1C, 0x74, 0x7F, 0x55,
0x59, 0xA8, 0xF2, 0xA3, 0xBD, 0x45, 0x9C,
/*1A20:*/ 0x9A, 0x45, 0xD2, 0xAA, 0x97, 0xB2, 0x9C, 0x33, 0xEB,
0x76, 0x06, 0x7B, 0x33, 0x94, 0x16, 0xD9,
/*1A30:*/ 0xA7, 0x54, 0x4C, 0xAD, 0xEB, 0x76, 0x06, 0x7B, 0x23,
0xA4, 0x9B, 0x3E, 0x59, 0x6B, 0x25, 0xE6,
/*1A40:*/ 0x9E, 0x3A, 0xCF, 0x5B, 0xE6, 0x05, 0x3C, 0xDF, 0x16,
0xCE, 0xEE, 0xEB, 0x9D, 0x62, 0x7B, 0x14,
/*1A50:*/ 0x1B, 0xA6, 0x0E, 0x00, 0xC8, 0xF7, 0x44, 0xFE, 0xEE,
0x5A, 0x8F, 0x21, 0x94, 0xAC, 0xDE, 0xF1,
/*1A60:*/ 0x0B, 0x91, 0x90, 0x54, 0x16, 0xCE, 0xEE, 0xEB, 0x1B,
0xA6, 0x0E, 0x00, 0x03, 0x02, 0xB4, 0x2D,
/*1A70:*/ 0x04, 0x47, 0x46, 0x11, 0x75, 0x18, 0x6F, 0xB8, 0x92,
0x82, 0xF5, 0xB9, 0x61, 0xE8, 0x92, 0xBB,
/*1A80:*/ 0x57, 0x89, 0xDA, 0xB2, 0x04, 0x47, 0x46, 0x11, 0xC8,
0xF7, 0x44, 0xFE, 0x11, 0x68, 0xF0, 0x4A,
/*1A90:*/ 0x4D, 0x1F, 0x00, 0xFA, 0xAD, 0x45, 0x93, 0x08, 0xE6,
0x05, 0x3C, 0xDF, 0x52, 0xE5, 0x37, 0xD4,
/*1AA0:*/ 0xAD, 0xBE, 0xFA, 0xE0, 0x3D, 0x72, 0xD9, 0x4A, 0xC5,
0x0F, 0xC0, 0xC6, 0xBF, 0x5B, 0x21, 0x67,
/*1AB0:*/ 0x33, 0xD9, 0x7D, 0xD3, 0x5D, 0x20, 0x4E, 0x4E, 0x1B,
0xBE, 0xBA, 0x74, 0x1A, 0x46, 0xB5, 0xFF,
/*1AC0:*/ 0xC4, 0xC9, 0xED, 0x05, 0x01, 0x62, 0x83, 0xE7, 0xDC,
0xBD, 0x76, 0x19, 0x1A, 0x22, 0xCB, 0x38,
/*1AD0:*/ 0x2F, 0x1F, 0xC8, 0xD8, 0xEC, 0xC6, 0xB5, 0x4C, 0xF2,
0xD1, 0x99, 0xBC, 0xC9, 0x81, 0x66, 0xB7,
/*1AE0:*/ 0x34, 0x2E, 0x7D, 0x84, 0x7C, 0xC3, 0x8D, 0xB5, 0xDA,
0x68, 0x24, 0x6C, 0x75, 0x18, 0x6F, 0xB8,
/*1AF0:*/ 0xCE, 0x1C, 0x58, 0x49, 0xD4, 0x56, 0x59, 0x6D, 0xE0,
0x58, 0x7A, 0x6F, 0x7E, 0x03, 0x73, 0x1D,
/*1B00:*/ 0x1B, 0xA6, 0x0E, 0x00, 0x3B, 0xE7, 0x29, 0xC7, 0x19,
0x1A, 0x36, 0xD0, 0x8E, 0x92, 0x63, 0xB0,
/*1B10:*/ 0x0C, 0x48, 0x3A, 0x3A, 0x01, 0xD7, 0x5F, 0xC0, 0x01,
0x42, 0xCD, 0xC9, 0x17, 0x08, 0x0D, 0x3F,
/*1B20:*/ 0x8C, 0xBA, 0x04, 0x05, 0x58, 0xF3, 0x41, 0xBF, 0xD2,
0x1D, 0x28, 0xFD, 0x34, 0x2E, 0x7D, 0x84,
/*1B30:*/ 0x50, 0xEE, 0xF3, 0x99, 0x16, 0x31, 0xE8, 0x05, 0x48,
0xE6, 0x76, 0x2D, 0x91, 0x24, 0x1D, 0x9F,
/*1B40:*/ 0x78, 0xBE, 0x58, 0x61, 0xF4, 0xE3, 0x73, 0xBE, 0x69,
0xC6, 0xF8, 0x5D, 0x35, 0x64, 0xCB, 0xAB,
/*1B50:*/ 0x3F, 0x17, 0x18, 0xCB, 0x7B, 0x07, 0x82, 0x94, 0x5C,
0xAA, 0xAC, 0x53, 0x03, 0x59, 0x9A, 0x4E,
/*1B60:*/ 0x35, 0x64, 0xCB, 0xAB, 0x55, 0xA1, 0xE1, 0x6B, 0x23,
0xEB, 0x14, 0x0E, 0xEB, 0x76, 0x06, 0x7B,
/*1B70:*/ 0x29, 0xE1, 0x35, 0x00, 0xE6, 0x05, 0x3C, 0xDF, 0xB4,
0x69, 0x9F, 0x38, 0x3F, 0xF5, 0x3C, 0x3E,
/*1B80:*/ 0xE6, 0x1C, 0xB6, 0x87, 0x01, 0xD7, 0x5F, 0xC0, 0x03,
0x02, 0xB4, 0x2D, 0x33, 0x94, 0x16, 0xD9,
/*1B90:*/ 0x8E, 0x92, 0x63, 0xB0, 0x6A, 0xAB, 0x24, 0xB1, 0x68,
0x36, 0xD9, 0x72, 0x02, 0x44, 0x8C, 0xB9,
/*1BA0:*/ 0x0B, 0xD2, 0x73, 0x78, 0x26, 0xAF, 0x6F, 0xC0, 0x5E,
0x8C, 0x7B, 0xD8, 0x44, 0x09, 0xC7, 0xED,
/*1BB0:*/ 0xE3, 0xA6, 0xD3, 0x18, 0xF2, 0xD1, 0x99, 0xBC, 0xF0,
0x5F, 0xAF, 0xD7, 0x17, 0x08, 0x0D, 0x3F,
/*1BC0:*/ 0xCA, 0xE9, 0xF0, 0x51, 0xA1, 0xDC, 0x17, 0xE0, 0x70,
0x77, 0xF6, 0x7E, 0xAB, 0x8D, 0xD1, 0x39,
/*1BD0:*/ 0x37, 0x83, 0x14, 0x26, 0xAC, 0xD9, 0x82, 0x35, 0x84,
0xE6, 0x91, 0xA6, 0x12, 0xBC, 0x59, 0xD4,
/*1BE0:*/ 0xCB, 0xCC, 0xC1, 0xC4, 0xD7, 0x9F, 0x82, 0x50, 0x7A,
0x1A, 0xB6, 0x8B, 0x3F, 0x86, 0x27, 0xB9,
/*1BF0:*/ 0xC8, 0xF7, 0x44, 0xFE, 0x77, 0xC8, 0x8F, 0x75, 0x05,
0x7B, 0xD6, 0xA2, 0xE5, 0x0C, 0x47, 0x24,
/*1C00:*/ 0x26, 0xAF, 0x6F, 0xC0, 0x1B, 0xBE, 0xBA, 0x74, 0xE6,
0xB0, 0xC1, 0x6B, 0x8C, 0x7F, 0x26, 0x4A,
/*1C10:*/ 0x9E, 0xBD, 0x9F, 0x16, 0x01, 0xD7, 0x5F, 0xC0, 0x5B,
0x25, 0xF1, 0x26, 0x53, 0xE5, 0xF5, 0x0D,
/*1C20:*/ 0xE0, 0x58, 0x7A, 0x6F, 0x22, 0xE4, 0x16, 0xC9, 0x32,
0x39, 0x17, 0x0C, 0x67, 0x0B, 0x9D, 0x26,
/*1C30:*/ 0x34, 0x2E, 0x7D, 0x84, 0xDD, 0x61, 0xC6, 0xD6, 0x58,
0xF3, 0x41, 0xBF, 0x38, 0x3C, 0x18, 0x72,
/*1C40:*/ 0xF3, 0xF7, 0xB5, 0x7E, 0x12, 0xD1, 0x59, 0x1B, 0xB4,
0x69, 0x9F, 0x38, 0x3D, 0x76, 0x22, 0xB1,
/*1C50:*/ 0x25, 0x32, 0x51, 0x73, 0x91, 0xB3, 0x84, 0xAB, 0x92,
0x26, 0x3D, 0x12, 0x22, 0xA2, 0x25, 0xBA,
/*1C60:*/ 0xE6, 0x05, 0x3C, 0xDF, 0xAE, 0x63, 0x36, 0xE6, 0xAD,
0x45, 0x93, 0x08, 0x3B, 0xFB, 0x95, 0x81,
/*1C70:*/ 0x52, 0x14, 0xC3, 0xE8, 0xC3, 0x68, 0x4D, 0xA0, 0x27,
0xC0, 0xD0, 0x4C, 0xDD, 0x61, 0xC6, 0xD6,
/*1C80:*/ 0xF9, 0xA8, 0xD0, 0xE2, 0xD8, 0x99, 0x2C, 0xB7, 0xE6,
0x1C, 0xB6, 0x87, 0x72, 0xCE, 0xA1, 0xE6,
/*1C90:*/ 0x22, 0xA2, 0x25, 0xBA, 0xA3, 0xBD, 0x45, 0x9C, 0xFD,
0x68, 0x02, 0x03, 0x60, 0x43, 0xCC, 0x0E,
/*1CA0:*/ 0x9E, 0xE0, 0xC6, 0x8C, 0x61, 0xE8, 0x92, 0xBB, 0x60,
0x43, 0xCC, 0x0E, 0xB5, 0x99, 0x8F, 0xB7,
/*1CB0:*/ 0x72, 0xCE, 0xA1, 0xE6, 0x69, 0x5E, 0x78, 0x07, 0x22,
0xA2, 0x25, 0xBA, 0xA1, 0xDC, 0x17, 0xE0,
/*1CC0:*/ 0x88, 0x5D, 0xD3, 0xB7, 0xD7, 0xE3, 0xE9, 0x28, 0x0B,
0x94, 0xDD, 0x44, 0x72, 0x5F, 0x47, 0x67,
/*1CD0:*/ 0x3E, 0xCA, 0xD9, 0x17, 0x8E, 0xB3, 0xD6, 0xCA, 0x94,
0xAC, 0xDE, 0xF1, 0x0B, 0x94, 0xDD, 0x44,
/*1CE0:*/ 0xC9, 0x9A, 0xC6, 0xDD, 0x5A, 0x6C, 0x52, 0x04, 0xEF,
0x02, 0xF1, 0x1F, 0xA1, 0x44, 0x3D, 0x80,
/*1CF0:*/ 0xD2, 0x93, 0xE6, 0x24, 0xFC, 0xA4, 0x4A, 0xBD, 0x9E,
0xBD, 0x9F, 0x16, 0x3E, 0x54, 0xED, 0x1F,
/*1D00:*/ 0x65, 0xF8, 0xDE, 0x27, 0xA0, 0x29, 0x6B, 0x19, 0xD2,
0xA0, 0x78, 0x54, 0x1A, 0x22, 0xCB, 0x38,
/*1D10:*/ 0x0D, 0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*1D20:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B6D2_A[] = {
/*2448:*/ 0xE5, 0x0C, 0x47, 0x24, 0xDF, 0xC7, 0x1C, 0x7D,
/*2450:*/ 0x4D, 0x1E, 0x4D, 0x3C, 0xC8, 0xC4, 0xD0, 0xA3, 0x4C,
0x79, 0xC2, 0xEB, 0x55, 0xAF, 0xCB, 0x7D,
/*2460:*/ 0x08, 0xBD, 0xC5, 0x11, 0x77, 0xC8, 0x8F, 0x75, 0xEB,
0x76, 0x06, 0x7B, 0x76, 0x0E, 0x7F, 0x97,
/*2470:*/ 0xFA, 0xCE, 0xAF, 0xA5, 0x70, 0xF3, 0x21, 0x92, 0xA0,
0x3E, 0xD4, 0xB4, 0x0F, 0x10, 0x4F, 0x74,
/*2480:*/ 0xB9, 0xB0, 0x7A, 0x35, 0xA7, 0x54, 0x4C, 0xAD, 0xD2,
0x93, 0xE6, 0x24, 0xD3, 0x81, 0x65, 0xE4,
/*2490:*/ 0x3F, 0x86, 0x27, 0xB9, 0x52, 0x1B, 0x4B, 0xEE, 0x28,
0x4C, 0x01, 0x90, 0x87, 0xA3, 0x0D, 0x5B,
/*24A0:*/ 0x16, 0x31, 0xE8, 0x05, 0xD8, 0x99, 0x2C, 0xB7, 0x53,
0xE5, 0xF5, 0x0D, 0x77, 0xF6, 0x5E, 0xC9,
/*24B0:*/ 0x77, 0xA4, 0xCE, 0x28, 0x62, 0x0B, 0x86, 0x44, 0x1B,
0xA6, 0x0E, 0x00, 0xEB, 0x76, 0x06, 0x7B,
/*24C0:*/ 0xF2, 0xD1, 0x99, 0xBC, 0x44, 0x09, 0xC7, 0xED, 0x6E,
0x35, 0xFA, 0x9F, 0x43, 0xDF, 0xC5, 0x2E,
/*24D0:*/ 0xE1, 0x24, 0x92, 0x9E, 0xD7, 0xE3, 0xE9, 0x28, 0x0B,
0x91, 0x90, 0x54, 0x48, 0xE6, 0x76, 0x2D,
/*24E0:*/ 0x91, 0xB3, 0x84, 0xAB, 0x31, 0xEA, 0x4D, 0xAD, 0x6F,
0xEE, 0xE7, 0x2D, 0x12, 0x7A, 0x2C, 0xF2,
/*24F0:*/ 0x53, 0xE5, 0xF5, 0x0D, 0x01, 0x62, 0x83, 0xE7, 0x2D,
0x75, 0xCC, 0x80, 0xF4, 0xE3, 0x73, 0xBE,
/*2500:*/ 0x72, 0xCE, 0xA1, 0xE6, 0x3B, 0x22, 0x07, 0x87, 0x2D,
0x75, 0xCC, 0x80, 0xCC, 0x79, 0x89, 0x67,
/*2510:*/ 0x88, 0x8D, 0xF5, 0x4B, 0x2D, 0xEA, 0x0D, 0x1D, 0x76,
0x0E, 0x7F, 0x97, 0xA4, 0xF8, 0x7B, 0x5B,
/*2520:*/ 0x43, 0x6C, 0xEF, 0x92, 0x75, 0x80, 0x94, 0x83, 0x95,
0x37, 0x3A, 0x30, 0xF3, 0x37, 0xA7, 0x4A,
/*2530:*/ 0x87, 0xA3, 0x0D, 0x5B, 0x11, 0x68, 0xF0, 0x4A, 0xAD,
0xBE, 0xFA, 0xE0, 0x8E, 0xB3, 0xD6, 0xCA,
/*2540:*/ 0x5D, 0x65, 0xB5, 0x0A, 0x72, 0x38, 0xA6, 0x78, 0xFA,
0xCE, 0xAF, 0xA5, 0x58, 0x1F, 0xAC, 0x46,
/*2550:*/ 0x84, 0x8E, 0xCA, 0xB6, 0x3E, 0xF1, 0xB5, 0x10, 0x43,
0xDF, 0xC5, 0x2E, 0x8C, 0x7F, 0x26, 0x4A,
/*2560:*/ 0x62, 0x0B, 0x86, 0x44, 0x70, 0x92, 0x08, 0xC8, 0x1B,
0xBE, 0xBA, 0x74, 0x55, 0xAF, 0xCB, 0x7D,
/*2570:*/ 0xC8, 0xC4, 0xD0, 0xA3, 0x09, 0x0D, 0x2C, 0x7E, 0x40,
0xB3, 0x1B, 0x9E, 0x38, 0x8D, 0x3D, 0x7F,
/*2580:*/ 0xDC, 0xBD, 0x76, 0x19, 0xC8, 0xF7, 0x44, 0xFE, 0x44,
0x4C, 0x5D, 0xB4, 0xAD, 0x45, 0x93, 0x08,
/*2590:*/ 0xEC, 0xC6, 0xB5, 0x4C, 0xAE, 0xBA, 0x7D, 0xFF, 0x40,
0xB3, 0x1B, 0x9E, 0x19, 0x76, 0xC1, 0xDE,
/*25A0:*/ 0x6C, 0xD3, 0xE5, 0x55, 0x38, 0x3C, 0x18, 0x72, 0x6C,
0x4E, 0xC0, 0x23, 0x0C, 0x31, 0xB7, 0x7D,
/*25B0:*/ 0x16, 0x42, 0x66, 0x51, 0xE5, 0x26, 0x45, 0x81, 0xA4,
0xCC, 0xA3, 0x0A, 0x5B, 0x1A, 0x7E, 0x64,
/*25C0:*/ 0x53, 0xE5, 0xF5, 0x0D, 0x3F, 0xF5, 0x3C, 0x3E, 0xF0,
0x25, 0xD0, 0x02, 0x68, 0xA7, 0x03, 0x45,
/*25D0:*/ 0x4D, 0x1E, 0x4D, 0x3C, 0xA8, 0xB8, 0x38, 0x0A, 0x89,
0xBB, 0xCE, 0xE7, 0x4C, 0x79, 0xC2, 0xEB,
/*25E0:*/ 0xE2, 0x9F, 0xD6, 0x27, 0x7B, 0x07, 0x82, 0x94, 0x53,
0xE5, 0xF5, 0x0D, 0x26, 0xAD, 0xDB, 0x83,
/*25F0:*/ 0x60, 0x43, 0xCC, 0x0E, 0x53, 0xF6, 0x3D, 0xAE, 0x8E,
0x92, 0x63, 0xB0, 0x8F, 0x08, 0x7C, 0x84,
/*2600:*/ 0x33, 0xD9, 0x7D, 0xD3, 0x32, 0x39, 0x17, 0x0C, 0x05,
0xAD, 0xAB, 0x34, 0x8C, 0x7F, 0x26, 0x4A,
/*2610:*/ 0x46, 0xC2, 0x68, 0x74, 0x34, 0x2E, 0x7D, 0x84, 0x70,
0x35, 0xAC, 0xDE, 0x6F, 0x0B, 0xAA, 0x0C,
/*2620:*/ 0xFC, 0x2C, 0x50, 0x09, 0xA4, 0xCC, 0xA3, 0x0A, 0xC8,
0xC4, 0xD0, 0xA3, 0x55, 0xA1, 0xE1, 0x6B,
/*2630:*/ 0x75, 0x18, 0x6F, 0xB8, 0xE6, 0xD3, 0x65, 0xF8, 0x26,
0xAF, 0x6F, 0xC0, 0x88, 0x5D, 0xD3, 0xB7,
/*2640:*/ 0xC4, 0x68, 0xAC, 0x83, 0x72, 0xCE, 0xA1, 0xE6, 0x7E,
0x03, 0x73, 0x1D, 0x7F, 0x7E, 0x86, 0xF1,
/*2650:*/ 0x18, 0x3A, 0x8E, 0x81, 0x22, 0xA7, 0xE7, 0x6E, 0x9D,
0x62, 0x7B, 0x14, 0x41, 0xAB, 0xEA, 0x23,
/*2660:*/ 0xD8, 0x99, 0x2C, 0xB7, 0xCA, 0x62, 0xA6, 0x46, 0xFC,
0x8A, 0x20, 0xF0, 0xE3, 0xA9, 0xC2, 0xC3,
/*2670:*/ 0x88, 0x8D, 0xF5, 0x4B, 0xD2, 0xC8, 0x3B, 0x42, 0x4C,
0x2C, 0x53, 0x9D, 0x32, 0x39, 0x17, 0x0C,
/*2680:*/ 0x5D, 0x20, 0x4E, 0x4E, 0xA8, 0xB8, 0x38, 0x0A, 0xC0,
0x00, 0x3D, 0xCA, 0x22, 0xE4, 0x16, 0xC9,
/*2690:*/ 0xBF, 0xB4, 0xD2, 0x8F, 0x91, 0x24, 0x1D, 0x9F, 0x95,
0xCB, 0x9B, 0xC3, 0xE0, 0x58, 0x7A, 0x6F,
/*26A0:*/ 0xC6, 0xB3, 0x24, 0x7D, 0x8E, 0x92, 0x63, 0xB0, 0xA0,
0x4F, 0x6B, 0xA5, 0xD8, 0x7C, 0x3C, 0x08,
/*26B0:*/ 0x6D, 0x08, 0x98, 0x24, 0x8E, 0xB3, 0xD6, 0xCA, 0xE0,
0x9C, 0x6D, 0x48, 0x7A, 0x97, 0x8F, 0xDF,
/*26C0:*/ 0x57, 0x11, 0x3A, 0x5F, 0x05, 0xAD, 0xAB, 0x34, 0x9C,
0x9C, 0x45, 0xA8, 0x33, 0x94, 0x16, 0xD9,
/*26D0:*/ 0x77, 0xC8, 0x8F, 0x75, 0xF4, 0xE3, 0x73, 0xBE, 0xDF,
0xC7, 0x1C, 0x7D, 0xE6, 0xD3, 0x65, 0xF8,
/*26E0:*/ 0x6A, 0x19, 0xF9, 0x2F, 0x5D, 0x65, 0xB5, 0x0A, 0x0C,
0x48, 0x3A, 0x3A, 0xC8, 0xC4, 0xD0, 0xA3,
/*26F0:*/ 0x3F, 0x71, 0x9A, 0x6A, 0xB5, 0x99, 0x8F, 0xB7, 0xE0,
0x9C, 0x6D, 0x48, 0x30, 0x52, 0xB4, 0xEB,
/*2700:*/ 0xE0, 0x58, 0x7A, 0x6F, 0xAD, 0x45, 0x93, 0x08, 0x59,
0x03, 0xA8, 0x1A, 0x01, 0xD7, 0x5F, 0xC0,
/*2710:*/ 0x44, 0x09, 0xC7, 0xED, 0x43, 0x6C, 0xEF, 0x92, 0xA7,
0x54, 0x4C, 0xAD, 0x52, 0x1B, 0x4B, 0xEE,
/*2720:*/ 0x8C, 0x7F, 0x26, 0x4A, 0xC9, 0x9A, 0xC6, 0xDD, 0xEB,
0x76, 0x06, 0x7B, 0x0D, 0xF0, 0xAD, 0xBA,
/*2730:*/ 0x0D, 0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*2740:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B6D2_B[] = {
/*2DB8:*/ 0x10, 0x20, 0x84, 0x97, 0xFE, 0x05, 0x3F, 0x81,
/*2DC0:*/ 0x83, 0x55, 0x2F, 0xE2, 0x73, 0x06, 0x71, 0x53, 0x4D,
0x1E, 0x4D, 0x3C, 0x9C, 0x26, 0xF2, 0x99,
/*2DD0:*/ 0xE1, 0x32, 0x9B, 0x77, 0x70, 0x6E, 0x67, 0xDF, 0x6C,
0xD3, 0xE5, 0x55, 0xC3, 0x68, 0x4D, 0xA0,
/*2DE0:*/ 0x35, 0x64, 0xCB, 0xAB, 0x5B, 0x25, 0xF1, 0x26, 0xC3,
0x37, 0xAD, 0x8F, 0xA3, 0xBD, 0x45, 0x9C,
/*2DF0:*/ 0xB5, 0x99, 0x8F, 0xB7, 0x88, 0x5D, 0xD3, 0xB7, 0xBF,
0xB4, 0xD2, 0x8F, 0x55, 0xAF, 0xCB, 0x7D,
/*2E00:*/ 0xC9, 0x9A, 0xC6, 0xDD, 0x00, 0x31, 0x68, 0x11, 0x2D,
0xC1, 0xCA, 0x89, 0xA1, 0xDC, 0x17, 0xE0,
/*2E10:*/ 0x4B, 0xFA, 0xF6, 0xE0, 0xA4, 0xCC, 0xA3, 0x0A, 0x88,
0x8D, 0xF5, 0x4B, 0x3B, 0x5C, 0x56, 0x54,
/*2E20:*/ 0xF9, 0xA8, 0xD0, 0xE2, 0x51, 0x07, 0x91, 0xBA, 0x3C,
0x6C, 0xB3, 0x5F, 0xC3, 0x37, 0xAD, 0x8F,
/*2E30:*/ 0x68, 0x36, 0xD9, 0x72, 0x0B, 0x91, 0x90, 0x54, 0x24,
0x1A, 0xBF, 0x03, 0x44, 0x09, 0xC7, 0xED,
/*2E40:*/ 0xC8, 0xF7, 0x44, 0xFE, 0x01, 0x62, 0x83, 0xE7, 0x3D,
0x72, 0xD9, 0x4A, 0x0B, 0x91, 0x90, 0x54,
/*2E50:*/ 0x30, 0xD9, 0x62, 0x05, 0x48, 0xE6, 0x76, 0x2D, 0x72,
0x5F, 0x47, 0x67, 0xC6, 0x4E, 0x14, 0x97,
/*2E60:*/ 0xD8, 0x99, 0x2C, 0xB7, 0xD7, 0xAB, 0xF5, 0x95, 0x15,
0x5D, 0x3E, 0xDE, 0xAB, 0xD1, 0xD3, 0x0E,
/*2E70:*/ 0x6E, 0x35, 0xFA, 0x9F, 0x7E, 0x03, 0x73, 0x1D, 0xF4,
0x2A, 0x00, 0x0E, 0x8D, 0xCA, 0x5F, 0x62,
/*2E80:*/ 0x33, 0x94, 0x16, 0xD9, 0x14, 0xED, 0x2F, 0x1A, 0x53,
0xE5, 0xF5, 0x0D, 0xB4, 0x1E, 0x49, 0xD5,
/*2E90:*/ 0xD3, 0x37, 0xFF, 0xEE, 0xE5, 0x0C, 0x47, 0x24, 0x92,
0x26, 0x3D, 0x12, 0x32, 0x39, 0x17, 0x0C,
/*2EA0:*/ 0x35, 0x64, 0xCB, 0xAB, 0x53, 0xE5, 0xF5, 0x0D, 0x46,
0x4E, 0xB0, 0x73, 0x6C, 0x4E, 0xC0, 0x23,
/*2EB0:*/ 0x9E, 0x3A, 0xCF, 0x5B, 0x0B, 0x94, 0xDD, 0x44, 0x17,
0x08, 0x0D, 0x3F, 0x2F, 0x67, 0xDA, 0xAC,
/*2EC0:*/ 0x63, 0xAE, 0xE3, 0x1C, 0xFC, 0x8A, 0x20, 0xF0, 0xDF,
0xC7, 0x1C, 0x7D, 0x1B, 0xBE, 0xBA, 0x74,
/*2ED0:*/ 0xF3, 0xF7, 0xB5, 0x7E, 0xB9, 0xD7, 0x98, 0x99, 0x8E,
0xB3, 0xD6, 0xCA, 0xDC, 0xBD, 0x76, 0x19,
/*2EE0:*/ 0x17, 0x08, 0x0D, 0x3F, 0xD8, 0x2F, 0x7E, 0xCA, 0x01,
0xD7, 0x5F, 0xC0, 0x6C, 0x4E, 0xC0, 0x23,
/*2EF0:*/ 0x48, 0xE6, 0x76, 0x2D, 0x6E, 0xA5, 0x40, 0xAE, 0x5D,
0x65, 0xB5, 0x0A, 0xE6, 0xD3, 0x65, 0xF8,
/*2F00:*/ 0x7E, 0x03, 0x73, 0x1D, 0xC8, 0xF7, 0x44, 0xFE, 0x55,
0xA1, 0xE1, 0x6B, 0x93, 0x60, 0x00, 0x4C,
/*2F10:*/ 0x3B, 0x22, 0x07, 0x87, 0xF3, 0xF7, 0xB5, 0x7E, 0x42,
0xFA, 0x83, 0x36, 0xC8, 0xC4, 0xD0, 0xA3,
/*2F20:*/ 0x11, 0x43, 0xBF, 0x52, 0x28, 0x4C, 0x01, 0x90, 0xE0,
0x32, 0x37, 0xB3, 0x33, 0xD9, 0x7D, 0xD3,
/*2F30:*/ 0x23, 0xEB, 0x14, 0x0E, 0x35, 0x64, 0xCB, 0xAB, 0xC9,
0x17, 0xFF, 0x0E, 0xD1, 0x40, 0xB6, 0x6C,
/*2F40:*/ 0x2D, 0x4B, 0xA5, 0x82, 0xAD, 0x45, 0x93, 0x08, 0x1B,
0xA6, 0x0E, 0x00, 0x8E, 0x11, 0x1D, 0xA3,
/*2F50:*/ 0x11, 0x68, 0xF0, 0x4A, 0x7E, 0x03, 0x73, 0x1D, 0x11,
0x68, 0xF0, 0x4A, 0xC8, 0xF7, 0x44, 0xFE,
/*2F60:*/ 0x15, 0x04, 0x21, 0x64, 0xE6, 0xD3, 0x65, 0xF8, 0x55,
0xAF, 0xCB, 0x7D, 0xC9, 0x9A, 0xC6, 0xDD,
/*2F70:*/ 0x86, 0xBD, 0xE9, 0x78, 0x59, 0x72, 0x6E, 0xA4, 0x0B,
0xD9, 0x3A, 0x41, 0x24, 0x1A, 0xBF, 0x03,
/*2F80:*/ 0x4C, 0x79, 0xC2, 0xEB, 0x3B, 0x22, 0x07, 0x87, 0xC8,
0xF7, 0x44, 0xFE, 0xC8, 0xF7, 0x44, 0xFE,
/*2F90:*/ 0xA9, 0x20, 0xF4, 0x24, 0x23, 0xA4, 0x9B, 0x3E, 0x30,
0x52, 0xB4, 0xEB, 0x12, 0xBC, 0x59, 0xD4,
/*2FA0:*/ 0xFC, 0xA4, 0x4A, 0xBD, 0x0B, 0x91, 0x90, 0x54, 0x2D,
0x75, 0xCC, 0x80, 0xF3, 0xF7, 0xB5, 0x7E,
/*2FB0:*/ 0x30, 0xC5, 0x0D, 0x6C, 0xCC, 0x79, 0x89, 0x67, 0x45,
0xCF, 0x2B, 0x5E, 0x75, 0x80, 0x94, 0x83,
/*2FC0:*/ 0xD7, 0x9F, 0x82, 0x50, 0x5E, 0xAC, 0xB2, 0x86, 0x0C,
0xA4, 0x6C, 0xFC, 0x78, 0xBE, 0x58, 0x61,
/*2FD0:*/ 0x16, 0xCE, 0xEE, 0xEB, 0x5D, 0x65, 0xB5, 0x0A, 0x0C,
0x48, 0x3A, 0x3A, 0xF4, 0x2A, 0x00, 0x0E,
/*2FE0:*/ 0x68, 0x36, 0xD9, 0x72, 0x4B, 0x91, 0x59, 0xAA, 0x10,
0x20, 0x84, 0x97, 0xC4, 0x28, 0x18, 0xE6,
/*2FF0:*/ 0x8E, 0x92, 0x63, 0xB0, 0x00, 0xBC, 0x1A, 0xC2, 0xEB,
0x76, 0x06, 0x7B, 0xEC, 0xC6, 0xB5, 0x4C,
/*3000:*/ 0xA0, 0xAF, 0xBB, 0xFC, 0x44, 0x09, 0xC7, 0xED, 0xA3,
0x49, 0x13, 0x9C, 0xF7, 0x95, 0x36, 0x9E,
/*3010:*/ 0x10, 0x20, 0x84, 0x97, 0x88, 0x8D, 0xF5, 0x4B, 0x7D,
0x8C, 0xFD, 0x17, 0xF0, 0x25, 0xD0, 0x02,
/*3020:*/ 0x87, 0xA3, 0x0D, 0x5B, 0x41, 0x0F, 0x82, 0xFB, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*3030:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B6D2_C[] = {
/*35F8:*/ 0x95, 0xCB, 0x9B, 0xC3, 0x96, 0x90, 0xD1, 0xAC,
/*3600:*/ 0x45, 0xCF, 0x2B, 0x5E, 0xC6, 0x71, 0xAD, 0x6F, 0x5B,
0x1A, 0x7E, 0x64, 0x6C, 0x4E, 0xC0, 0x23,
/*3610:*/ 0xD2, 0x93, 0xE6, 0x24, 0x3E, 0x77, 0xA8, 0x9D, 0xD9,
0x27, 0xDA, 0xA5, 0x32, 0x39, 0x17, 0x0C,
/*3620:*/ 0x7B, 0x07, 0x82, 0x94, 0xAB, 0xBE, 0x44, 0x55, 0xEB,
0x76, 0x06, 0x7B, 0x88, 0x59, 0xC4, 0x4D,
/*3630:*/ 0x2D, 0x13, 0x24, 0x84, 0x55, 0xA1, 0xE1, 0x6B, 0x9A,
0x45, 0xD2, 0xAA, 0x44, 0x4C, 0x5D, 0xB4,
/*3640:*/ 0xCA, 0xE9, 0xF0, 0x51, 0x1C, 0x53, 0x14, 0x15, 0x5E,
0x8C, 0x7B, 0xD8, 0x3B, 0x22, 0x07, 0x87,
/*3650:*/ 0xAB, 0xD1, 0xD3, 0x0E, 0x3D, 0x72, 0xD9, 0x4A, 0x57,
0x89, 0xDA, 0xB2, 0x16, 0x42, 0x66, 0x51,
/*3660:*/ 0x79, 0xE7, 0xE8, 0x15, 0xB6, 0x45, 0x9D, 0x23, 0x3E,
0x77, 0xA8, 0x9D, 0xB9, 0xB0, 0x7A, 0x35,
/*3670:*/ 0x3E, 0x77, 0xA8, 0x9D, 0x0C, 0x48, 0x3A, 0x3A, 0xA1,
0x44, 0x3D, 0x80, 0x48, 0xF6, 0x2D, 0x5F,
/*3680:*/ 0x5F, 0xF2, 0xFA, 0xF6, 0xFC, 0x8A, 0x20, 0xF0, 0x7A,
0x1A, 0xB6, 0x8B, 0x01, 0x62, 0x83, 0xE7,
/*3690:*/ 0x6A, 0xAB, 0x24, 0xB1, 0x59, 0x72, 0x6E, 0xA4, 0x7C,
0x77, 0x4B, 0xD7, 0xCF, 0x7E, 0x80, 0xB6,
/*36A0:*/ 0x59, 0x6B, 0x25, 0xE6, 0x87, 0xA3, 0x0D, 0x5B, 0x84,
0xE6, 0x91, 0xA6, 0x12, 0xE9, 0xB6, 0xBE,
/*36B0:*/ 0x88, 0x8D, 0xF5, 0x4B, 0x8E, 0xB3, 0xD6, 0xCA, 0x2F,
0x67, 0xDA, 0xAC, 0xC8, 0xF7, 0x44, 0xFE,
/*36C0:*/ 0x41, 0x0F, 0x82, 0xFB, 0xC1, 0xF6, 0x2F, 0x66, 0xFD,
0x68, 0x02, 0x03, 0xDE, 0x02, 0x1C, 0xBD,
/*36D0:*/ 0xA4, 0xCC, 0xA3, 0x0A, 0xE8, 0x83, 0x83, 0x7C, 0x31,
0xDB, 0xF4, 0x11, 0xED, 0x06, 0x1B, 0xEA,
/*36E0:*/ 0x27, 0xC0, 0xD0, 0x4C, 0x78, 0xBE, 0x58, 0x61, 0x84,
0x8E, 0xCA, 0xB6, 0x0C, 0x48, 0x3A, 0x3A,
/*36F0:*/ 0x30, 0xD9, 0x62, 0x05, 0xAC, 0xD9, 0x82, 0x35, 0xBC,
0x9B, 0x2F, 0xC1, 0x59, 0x03, 0xA8, 0x1A,
/*3700:*/ 0x72, 0xCE, 0xA1, 0xE6, 0xFD, 0x70, 0xEB, 0xC1, 0x52,
0x1B, 0x4B, 0xEE, 0x0B, 0x94, 0x45, 0xA7,
/*3710:*/ 0x08, 0xBD, 0xC5, 0x11, 0xFC, 0xB8, 0x3C, 0xEB, 0xD5,
0x80, 0xA1, 0x41, 0xA2, 0x61, 0x35, 0x41,
/*3720:*/ 0x3B, 0x5C, 0x56, 0x54, 0xAD, 0xBE, 0xFA, 0xE0, 0x77,
0xC8, 0x8F, 0x75, 0xC3, 0x79, 0x85, 0xAB,
/*3730:*/ 0x3B, 0xFB, 0x95, 0x81, 0x2D, 0x4B, 0xA5, 0x82, 0x84,
0x8E, 0xCA, 0xB6, 0x65, 0xF8, 0xDE, 0x27,
/*3740:*/ 0xC0, 0x18, 0x1D, 0xAE, 0x41, 0x0F, 0x82, 0xFB, 0x3F,
0x86, 0x27, 0xB9, 0xE2, 0x6D, 0x41, 0xB7,
/*3750:*/ 0x28, 0x4C, 0x01, 0x90, 0xA7, 0x54, 0x4C, 0xAD, 0x03,
0x02, 0xB4, 0x2D, 0xCF, 0xC9, 0xD5, 0x98,
/*3760:*/ 0x8C, 0x7F, 0x26, 0x4A, 0x50, 0x45, 0x07, 0xF0, 0xFC,
0xB8, 0x3C, 0xEB, 0x6C, 0x4E, 0xC0, 0x23,
/*3770:*/ 0x9C, 0x26, 0xF2, 0x99, 0x69, 0x5E, 0x78, 0x07, 0x32,
0x39, 0x17, 0x0C, 0xD9, 0x27, 0xDA, 0xA5,
/*3780:*/ 0xAD, 0xBE, 0xFA, 0xE0, 0xBC, 0x9B, 0x2F, 0xC1, 0x89,
0xBB, 0xCE, 0xE7, 0x04, 0x47, 0x46, 0x11,
/*3790:*/ 0xDB, 0x90, 0x37, 0x22, 0xB9, 0xB0, 0x7A, 0x35, 0x65,
0xF8, 0xDE, 0x27, 0xBB, 0x4B, 0x29, 0x9F,
/*37A0:*/ 0xD3, 0x37, 0xFF, 0xEE, 0x18, 0x3A, 0x8E, 0x81, 0x87,
0xA3, 0x0D, 0x5B, 0x3B, 0x5C, 0x56, 0x54,
/*37B0:*/ 0x3E, 0x77, 0xA8, 0x9D, 0xA8, 0x21, 0x62, 0x44, 0x55,
0xAF, 0xCB, 0x7D, 0x8D, 0xCA, 0x5F, 0x62,
/*37C0:*/ 0x0B, 0x91, 0x90, 0x54, 0xD9, 0xB8, 0x25, 0xCB, 0x04,
0x47, 0x46, 0x11, 0xCA, 0x62, 0xA6, 0x46,
/*37D0:*/ 0x76, 0x0E, 0x7F, 0x97, 0x5F, 0xF2, 0xFA, 0xF6, 0xC3,
0x79, 0x85, 0xAB, 0x93, 0x60, 0x00, 0x4C,
/*37E0:*/ 0x30, 0x52, 0xB4, 0xEB, 0x95, 0xCB, 0x9B, 0xC3, 0x86,
0xBD, 0xE9, 0x78, 0x49, 0xE4, 0x72, 0xD0,
/*37F0:*/ 0x43, 0x6C, 0xEF, 0x92, 0xBD, 0x41, 0xBE, 0x36, 0x4D,
0x74, 0xF8, 0x85, 0x91, 0xB3, 0x84, 0xAB,
/*3800:*/ 0x6C, 0xD3, 0xE5, 0x55, 0xE6, 0x05, 0x3C, 0xDF, 0x9C,
0x26, 0xF2, 0x99, 0xD4, 0x56, 0x59, 0x6D,
/*3810:*/ 0xE6, 0x1C, 0xB6, 0x87, 0x72, 0x38, 0xA6, 0x78, 0xD2,
0xA0, 0x78, 0x54, 0xDC, 0xBD, 0x76, 0x19,
/*3820:*/ 0x99, 0x7B, 0x6F, 0xC5, 0x8E, 0xB3, 0xD6, 0xCA, 0x16,
0xCE, 0xEE, 0xEB, 0x0B, 0x91, 0x90, 0x54,
/*3830:*/ 0xD8, 0x7C, 0x3C, 0x08, 0xA1, 0xDC, 0x17, 0xE0, 0x6D,
0x08, 0x98, 0x24, 0xE6, 0x05, 0x3C, 0xDF,
/*3840:*/ 0x5F, 0xF2, 0xFA, 0xF6, 0x0B, 0x91, 0x90, 0x54, 0x95,
0xCB, 0x9B, 0xC3, 0x2F, 0x1F, 0xC8, 0xD8,
/*3850:*/ 0x0D, 0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*3860:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B6D2_D[] = {
/*3E28:*/ 0x59, 0x6B, 0x25, 0xE6, 0x0B, 0x91, 0x90, 0x54,
/*3E30:*/ 0x04, 0xDE, 0x4C, 0x7C, 0x14, 0xD0, 0xDE, 0x5E, 0xE4,
0x4E, 0x30, 0x3A, 0x19, 0x1C, 0x75, 0x90,
/*3E40:*/ 0x3D, 0x72, 0xD9, 0x4A, 0x53, 0xE5, 0xF5, 0x0D, 0x55,
0xA1, 0xE1, 0x6B, 0x9E, 0xE0, 0xC6, 0x8C,
/*3E50:*/ 0x99, 0x7B, 0x6F, 0xC5, 0x04, 0x47, 0x46, 0x11, 0xD2,
0xA0, 0x78, 0x54, 0xCA, 0x62, 0xA6, 0x46,
/*3E60:*/ 0x87, 0xA3, 0x0D, 0x5B, 0x8D, 0x6C, 0x70, 0x47, 0x63,
0xD9, 0xBE, 0x5C, 0x88, 0x5D, 0xD3, 0xB7,
/*3E70:*/ 0x04, 0xEA, 0xA7, 0x28, 0x72, 0x5F, 0x47, 0x67, 0x0C,
0x36, 0xC2, 0xE3, 0xFC, 0x8A, 0x20, 0xF0,
/*3E80:*/ 0xB9, 0xB0, 0x7A, 0x35, 0xFC, 0x2C, 0x50, 0x09, 0x95,
0xCB, 0x9B, 0xC3, 0x77, 0xF6, 0x5E, 0xC9,
/*3E90:*/ 0xE6, 0xD3, 0x65, 0xF8, 0x00, 0x31, 0x68, 0x11, 0x7E,
0xB6, 0xC1, 0x8E, 0x06, 0x8C, 0x28, 0x9D,
/*3EA0:*/ 0x30, 0xD9, 0x62, 0x05, 0x8E, 0xB3, 0xD6, 0xCA, 0x3E,
0x2C, 0x92, 0xD5, 0xC1, 0xF6, 0x2F, 0x66,
/*3EB0:*/ 0x8C, 0x7F, 0x26, 0x4A, 0x4A, 0xDF, 0x4D, 0xF0, 0x05,
0x6E, 0x54, 0x33, 0x55, 0xA1, 0xE1, 0x6B,
/*3EC0:*/ 0x04, 0xEA, 0xA7, 0x28, 0xE6, 0x05, 0x3C, 0xDF, 0x41,
0x0F, 0x82, 0xFB, 0x2B, 0xCE, 0x76, 0xE4,
/*3ED0:*/ 0x01, 0x30, 0x5A, 0xF7, 0x34, 0x2E, 0x7D, 0x84, 0x8C,
0x6D, 0x9F, 0x7A, 0x3B, 0x22, 0x07, 0x87,
/*3EE0:*/ 0xA3, 0xAE, 0xC3, 0xFA, 0xC0, 0xA2, 0x3F, 0xEA, 0xE6,
0xD3, 0x65, 0xF8, 0xCB, 0x53, 0xD3, 0xFF,
/*3EF0:*/ 0xCC, 0x79, 0x89, 0x67, 0x85, 0x9D, 0xF0, 0x35, 0xBC,
0x9B, 0x2F, 0xC1, 0x91, 0x24, 0x1D, 0x9F,
/*3F00:*/ 0xE1, 0x24, 0x92, 0x9E, 0xE1, 0x24, 0x92, 0x9E, 0x6C,
0x3C, 0x44, 0x49, 0xC0, 0x00, 0x3D, 0xCA,
/*3F10:*/ 0xB2, 0x0E, 0xC0, 0x25, 0x50, 0x45, 0x07, 0xF0, 0x16,
0x42, 0x66, 0x51, 0x93, 0x60, 0x00, 0x4C,
/*3F20:*/ 0x4D, 0x01, 0x6B, 0x7A, 0x01, 0x42, 0xCD, 0xC9, 0x72,
0x38, 0xA6, 0x78, 0x33, 0x94, 0x16, 0xD9,
/*3F30:*/ 0xCF, 0x7E, 0x80, 0xB6, 0xFC, 0xB8, 0x3C, 0xEB, 0xD9,
0xB8, 0x25, 0xCB, 0xE0, 0x58, 0x7A, 0x6F,
/*3F40:*/ 0xB9, 0xB0, 0x7A, 0x35, 0xA7, 0x54, 0x4C, 0xAD, 0x58,
0xF3, 0x41, 0xBF, 0xAB, 0xD1, 0xD3, 0x0E,
/*3F50:*/ 0xEC, 0x88, 0xC7, 0xDB, 0xA0, 0x18, 0x0B, 0x31, 0x3E,
0xF1, 0xB5, 0x10, 0x35, 0xFF, 0xB0, 0x1A,
/*3F60:*/ 0x70, 0x6E, 0x67, 0xDF, 0xFC, 0x8A, 0x20, 0xF0, 0x01,
0x62, 0x83, 0xE7, 0xFC, 0x8A, 0x20, 0xF0,
/*3F70:*/ 0xA4, 0xCC, 0xA3, 0x0A, 0xB0, 0x93, 0x08, 0x65, 0x87,
0xA3, 0x0D, 0x5B, 0x0B, 0x91, 0x90, 0x54,
/*3F80:*/ 0x84, 0xE6, 0x91, 0xA6, 0x38, 0x3C, 0x18, 0x72, 0xD7,
0xE3, 0xE9, 0x28, 0x12, 0xE9, 0xB6, 0xBE,
/*3F90:*/ 0x25, 0x32, 0x51, 0x73, 0x4F, 0xBE, 0x7E, 0xBA, 0xDC,
0xBD, 0x76, 0x19, 0x27, 0xC0, 0xD0, 0x4C,
/*3FA0:*/ 0x30, 0xD9, 0x62, 0x05, 0x44, 0x09, 0xC7, 0xED, 0x48,
0xE6, 0x76, 0x2D, 0x3B, 0xE7, 0x29, 0xC7,
/*3FB0:*/ 0x3E, 0xCA, 0xD9, 0x17, 0x01, 0x62, 0x83, 0xE7, 0xD7,
0x9F, 0x82, 0x50, 0x8E, 0x92, 0x63, 0xB0,
/*3FC0:*/ 0x31, 0xEA, 0x4D, 0xAD, 0x3F, 0x86, 0x27, 0xB9, 0xC5,
0x0F, 0xC0, 0xC6, 0xFD, 0x72, 0xC3, 0xA8,
/*3FD0:*/ 0xC0, 0x00, 0x3D, 0xCA, 0x72, 0x38, 0xA6, 0x78, 0x29,
0x78, 0x60, 0x9B, 0xAD, 0x45, 0x93, 0x08,
/*3FE0:*/ 0xB5, 0x99, 0x8F, 0xB7, 0x88, 0x5D, 0xD3, 0xB7, 0x4B,
0x91, 0x59, 0xAA, 0xC0, 0x94, 0x54, 0xC9,
/*3FF0:*/ 0x70, 0x6E, 0x67, 0xDF, 0xAB, 0xD1, 0xD3, 0x0E, 0x61,
0xE8, 0x92, 0xBB, 0x02, 0x44, 0x8C, 0xB9,
/*4000:*/ 0xFD, 0x68, 0x02, 0x03, 0xFD, 0x68, 0x02, 0x03, 0x9E,
0xE0, 0xC6, 0x8C, 0xCC, 0x79, 0x89, 0x67,
/*4010:*/ 0x91, 0x24, 0x1D, 0x9F, 0xA4, 0xF8, 0x7B, 0x5B, 0xA8,
0xB8, 0x38, 0x0A, 0xEC, 0xC6, 0xB5, 0x4C,
/*4020:*/ 0xA9, 0x20, 0xF4, 0x24, 0xA8, 0x21, 0x62, 0x44, 0x33,
0x94, 0x16, 0xD9, 0xDD, 0xB8, 0xF4, 0x8F,
/*4030:*/ 0xDB, 0x90, 0x37, 0x22, 0xC0, 0xA2, 0x3F, 0xEA, 0x8E,
0x92, 0x63, 0xB0, 0xF4, 0x2A, 0x00, 0x0E,
/*4040:*/ 0xC3, 0x79, 0x85, 0xAB, 0x84, 0x8E, 0xCA, 0xB6, 0x12,
0xBC, 0x59, 0xD4, 0x17, 0x08, 0x0D, 0x3F,
/*4050:*/ 0x9A, 0x45, 0xD2, 0xAA, 0xD2, 0x1D, 0x28, 0xFD, 0xC6,
0x71, 0xAD, 0x6F, 0x12, 0xF1, 0xCE, 0xFD,
/*4060:*/ 0xFC, 0x8A, 0x20, 0xF0, 0xDF, 0xC7, 0x1C, 0x7D, 0x1A,
0x22, 0xCB, 0x38, 0x4F, 0xA6, 0xC1, 0x11,
/*4070:*/ 0xAD, 0xBE, 0xFA, 0xE0, 0x52, 0xE5, 0x37, 0xD4, 0x3E,
0x54, 0xED, 0x1F, 0x55, 0xAF, 0xCB, 0x7D,
/*4080:*/ 0x5D, 0x65, 0xB5, 0x0A, 0x2E, 0x7B, 0x7B, 0xBD, 0x31,
0xAF, 0x65, 0x25, 0xC0, 0x00, 0x3D, 0xCA,
/*4090:*/ 0x63, 0xD9, 0xBE, 0x5C, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*40A0:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B6D2_E[] = {
/*4698:*/ 0x68, 0x36, 0xD9, 0x72, 0x87, 0xA3, 0x0D, 0x5B,
/*46A0:*/ 0x0B, 0x91, 0x90, 0x54, 0x44, 0x09, 0xC7, 0xED, 0xDD,
0xB8, 0xF4, 0x8F, 0xE6, 0x1C, 0xB6, 0x87,
/*46B0:*/ 0xC3, 0x79, 0x85, 0xAB, 0x24, 0x1A, 0xBF, 0x03, 0x59,
0x72, 0x6E, 0xA4, 0x30, 0x52, 0xB4, 0xEB,
/*46C0:*/ 0x1A, 0x22, 0xCB, 0x38, 0xE6, 0x1C, 0xB6, 0x87, 0x53,
0xE5, 0xF5, 0x0D, 0x30, 0xC5, 0x0D, 0x6C,
/*46D0:*/ 0xF4, 0x2A, 0x00, 0x0E, 0xC5, 0x21, 0xEF, 0xFE, 0x45,
0xCF, 0xD3, 0x8B, 0xC3, 0x37, 0xAD, 0x8F,
/*46E0:*/ 0x78, 0xBE, 0x58, 0x61, 0x51, 0x07, 0x91, 0xBA, 0x2D,
0xC1, 0xCA, 0x89, 0x53, 0xE5, 0xF5, 0x0D,
/*46F0:*/ 0x63, 0xD9, 0xBE, 0x5C, 0xB4, 0x1E, 0x49, 0xD5, 0x4F,
0xA6, 0xC1, 0x11, 0x87, 0xA3, 0x0D, 0x5B,
/*4700:*/ 0x7C, 0x77, 0x4B, 0xD7, 0xE3, 0xA6, 0xD3, 0x18, 0x9D,
0x62, 0x7B, 0x14, 0x1B, 0xA6, 0x0E, 0x00,
/*4710:*/ 0x8E, 0x92, 0x63, 0xB0, 0xC8, 0xF7, 0x44, 0xFE, 0xA1,
0x44, 0x3D, 0x80, 0xFE, 0x94, 0x13, 0x1E,
/*4720:*/ 0x41, 0x0F, 0x82, 0xFB, 0x26, 0xAF, 0x6F, 0xC0, 0x5A,
0x6C, 0x52, 0x04, 0x4B, 0x91, 0x59, 0xAA,
/*4730:*/ 0x0E, 0xE5, 0xEF, 0x22, 0xC9, 0x17, 0xFF, 0x0E, 0x03,
0x02, 0xB4, 0x2D, 0x43, 0xDF, 0xC5, 0x2E,
/*4740:*/ 0xA4, 0xCC, 0xA3, 0x0A, 0x12, 0xD1, 0x59, 0x1B, 0x22,
0xA2, 0x25, 0xBA, 0x6D, 0x08, 0x98, 0x24,
/*4750:*/ 0xA4, 0xCC, 0xA3, 0x0A, 0x93, 0x60, 0x00, 0x4C, 0x03,
0x59, 0x9A, 0x4E, 0x06, 0xFF, 0xC3, 0x32,
/*4760:*/ 0x84, 0xE6, 0x91, 0xA6, 0x5B, 0x7C, 0xE7, 0xB9, 0xAD,
0xBE, 0xFA, 0xE0, 0x72, 0x38, 0xA6, 0x78,
/*4770:*/ 0x88, 0x8D, 0xF5, 0x4B, 0x1A, 0x22, 0xCB, 0x38, 0x0B,
0x91, 0x90, 0x54, 0xC0, 0x00, 0x3D, 0xCA,
/*4780:*/ 0x1B, 0xA6, 0x0E, 0x00, 0x2D, 0xEA, 0x0D, 0x1D, 0x86,
0xBD, 0xE9, 0x78, 0xA3, 0xBD, 0x45, 0x9C,
/*4790:*/ 0xE6, 0xD3, 0x65, 0xF8, 0xC4, 0x68, 0xAC, 0x83, 0x95,
0xCB, 0x9B, 0xC3, 0x63, 0xD9, 0xBE, 0x5C,
/*47A0:*/ 0xAD, 0xBE, 0xFA, 0xE0, 0x4D, 0x74, 0xF8, 0x85, 0x02,
0x44, 0x8C, 0xB9, 0xE2, 0x99, 0xC6, 0x51,
/*47B0:*/ 0x9A, 0x45, 0xD2, 0xAA, 0x12, 0x7A, 0x2C, 0xF2, 0x00,
0x31, 0x68, 0x11, 0x04, 0xEA, 0xA7, 0x28,
/*47C0:*/ 0xD8, 0x99, 0x2C, 0xB7, 0x12, 0x7A, 0x2C, 0xF2, 0xE5,
0x0C, 0x47, 0x24, 0xC9, 0x17, 0xFF, 0x0E,
/*47D0:*/ 0xE6, 0xD3, 0x65, 0xF8, 0x05, 0xAD, 0xAB, 0x34, 0xDD,
0xB8, 0xF4, 0x8F, 0xB9, 0x17, 0x54, 0x7B,
/*47E0:*/ 0x4F, 0xBE, 0x7E, 0xBA, 0x75, 0x80, 0x94, 0x83, 0x9E,
0x43, 0xAF, 0x3B, 0x01, 0x62, 0x83, 0xE7,
/*47F0:*/ 0x38, 0x8D, 0x3D, 0x7F, 0xB9, 0xB0, 0x7A, 0x35, 0xB9,
0xB0, 0x7A, 0x35, 0xDF, 0xC7, 0x1C, 0x7D,
/*4800:*/ 0x44, 0x09, 0xC7, 0xED, 0xEF, 0x02, 0xF1, 0x1F, 0x30,
0xC5, 0x0D, 0x6C, 0x91, 0x24, 0x1D, 0x9F,
/*4810:*/ 0xC2, 0xEF, 0x17, 0x8C, 0x78, 0xBE, 0x58, 0x61, 0x49,
0xE4, 0x72, 0xD0, 0x72, 0x38, 0xA6, 0x78,
/*4820:*/ 0xC5, 0x21, 0xEF, 0xFE, 0xCF, 0x7E, 0x80, 0xB6, 0x1A,
0xD8, 0xBC, 0x1F, 0xF9, 0xA8, 0xD0, 0xE2,
/*4830:*/ 0xC3, 0x68, 0x4D, 0xA0, 0xEC, 0x88, 0xC7, 0xDB, 0x31,
0xAF, 0x65, 0x25, 0x23, 0xA4, 0x9B, 0x3E,
/*4840:*/ 0x38, 0x3C, 0x18, 0x72, 0x65, 0x1C, 0x74, 0x7F, 0x45,
0xCF, 0x2B, 0x5E, 0x55, 0x59, 0xA8, 0xF2,
/*4850:*/ 0xA4, 0xCC, 0xA3, 0x0A, 0xC3, 0x68, 0x4D, 0xA0, 0x04,
0x47, 0x46, 0x11, 0x2D, 0xEA, 0x0D, 0x1D,
/*4860:*/ 0x03, 0x02, 0xB4, 0x2D, 0xC3, 0x79, 0x85, 0xAB, 0x02,
0x6F, 0xC8, 0xEA, 0x28, 0x4C, 0x01, 0x90,
/*4870:*/ 0xCB, 0xCC, 0xC1, 0xC4, 0x08, 0xBD, 0xC5, 0x11, 0xCE,
0x1C, 0x58, 0x49, 0x0B, 0x91, 0x90, 0x54,
/*4880:*/ 0x89, 0xBB, 0xCE, 0xE7, 0x8E, 0x92, 0x63, 0xB0, 0x04,
0x47, 0x46, 0x11, 0xDF, 0xC7, 0x1C, 0x7D,
/*4890:*/ 0x03, 0x02, 0xB4, 0x2D, 0x48, 0xE6, 0x76, 0x2D, 0x55,
0xAF, 0xCB, 0x7D, 0xC3, 0x37, 0xAD, 0x8F,
/*48A0:*/ 0x43, 0xAA, 0xB9, 0x19, 0xA3, 0xAE, 0xC3, 0xFA, 0xB9,
0xB0, 0x7A, 0x35, 0x1B, 0xA6, 0x0E, 0x00,
/*48B0:*/ 0x1A, 0x22, 0xCB, 0x38, 0x32, 0x39, 0x17, 0x0C, 0x1E,
0xAA, 0x3D, 0x3F, 0xD8, 0x7C, 0x3C, 0x08,
/*48C0:*/ 0x3E, 0x54, 0xED, 0x1F, 0x45, 0xCF, 0x2B, 0x5E, 0x3E,
0xCA, 0xD9, 0x17, 0x15, 0x04, 0x21, 0x64,
/*48D0:*/ 0x04, 0x47, 0x46, 0x11, 0x92, 0x82, 0xF5, 0xB9, 0xE4,
0x4E, 0x30, 0x3A, 0x11, 0x43, 0xBF, 0x52,
/*48E0:*/ 0x30, 0x52, 0xB4, 0xEB, 0x01, 0xD7, 0x5F, 0xC0, 0x9A,
0x45, 0xD2, 0xAA, 0xEB, 0x76, 0x06, 0x7B,
/*48F0:*/ 0x8E, 0x92, 0x63, 0xB0, 0x12, 0x7A, 0x2C, 0xF2, 0x5B,
0x25, 0xF1, 0x26, 0x8E, 0xB3, 0xD6, 0xCA,
/*4900:*/ 0xBC, 0x9B, 0x2F, 0xC1, 0xC6, 0x71, 0xAD, 0x6F, 0x8E,
0xB3, 0xD6, 0xCA, 0x4B, 0x91, 0x59, 0xAA,
/*4910:*/ 0x19, 0x1A, 0x36, 0xD0, 0x15, 0x5D, 0x3E, 0xDE, 0x8E,
0x23, 0x09, 0xF3, 0x45, 0xCF, 0xD3, 0x8B,
/*4920:*/ 0x29, 0x78, 0x60, 0x9B, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*4930:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B6D2_F[] = {
/*4F78:*/ 0x53, 0xE5, 0xF5, 0x0D, 0x76, 0x0E, 0x7F, 0x97,
/*4F80:*/ 0xFD, 0x70, 0xEB, 0xC1, 0x69, 0x28, 0xA5, 0xF2, 0x8E,
0x11, 0x1D, 0xA3, 0xA9, 0x86, 0xA4, 0x12,
/*4F90:*/ 0x59, 0x72, 0x6E, 0xA4, 0x77, 0xC8, 0x8F, 0x75, 0xFC,
0xB8, 0x3C, 0xEB, 0x3A, 0x72, 0xF6, 0x53,
/*4FA0:*/ 0xF6, 0x3F, 0xAB, 0xDB, 0x58, 0xF3, 0x41, 0xBF, 0x5F,
0xF2, 0xFA, 0xF6, 0x4F, 0xBE, 0x7E, 0xBA,
/*4FB0:*/ 0x93, 0x60, 0x00, 0x4C, 0x2F, 0x1F, 0xC8, 0xD8, 0x41,
0x0F, 0x82, 0xFB, 0x3B, 0xE7, 0x29, 0xC7,
/*4FC0:*/ 0x3D, 0x72, 0xD9, 0x4A, 0x6E, 0x35, 0xFA, 0x9F, 0xB4,
0x69, 0x9F, 0x38, 0x48, 0xF6, 0x2D, 0x5F,
/*4FD0:*/ 0x84, 0xE6, 0x91, 0xA6, 0xF7, 0x33, 0x41, 0x1A, 0x5D,
0x20, 0x4E, 0x4E, 0xEC, 0x76, 0x94, 0x37,
/*4FE0:*/ 0xB4, 0x69, 0x9F, 0x38, 0x6D, 0x08, 0x98, 0x24, 0xFC,
0xA4, 0x4A, 0xBD, 0x5E, 0xAC, 0xB2, 0x86,
/*4FF0:*/ 0xA4, 0xCC, 0xA3, 0x0A, 0x77, 0xA4, 0xCE, 0x28, 0x64,
0xA6, 0xD4, 0xF0, 0x84, 0xE6, 0x91, 0xA6,
/*5000:*/ 0xA0, 0x18, 0x0B, 0x31, 0x6C, 0xD3, 0xE5, 0x55, 0x00,
0xBC, 0x1A, 0xC2, 0xC0, 0x94, 0x54, 0xC9,
/*5010:*/ 0xF2, 0xD1, 0x99, 0xBC, 0x05, 0xB0, 0x7F, 0x45, 0x5B,
0x1A, 0x7E, 0x64, 0x0C, 0x31, 0xB7, 0x7D,
/*5020:*/ 0x8E, 0xB3, 0xD6, 0xCA, 0x41, 0x0F, 0x82, 0xFB, 0x9C,
0x26, 0xF2, 0x99, 0x38, 0x6D, 0xFD, 0x2D,
/*5030:*/ 0x03, 0x59, 0x9A, 0x4E, 0x30, 0xC5, 0x0D, 0x6C, 0xAB,
0xD1, 0xD3, 0x0E, 0x6F, 0xEE, 0xE7, 0x2D,
/*5040:*/ 0xDF, 0xC7, 0x1C, 0x7D, 0xD9, 0xB8, 0x25, 0xCB, 0x67,
0x0B, 0x9D, 0x26, 0x94, 0xE8, 0xA4, 0xD1,
/*5050:*/ 0x3E, 0x77, 0xA8, 0x9D, 0xD2, 0x1D, 0x28, 0xFD, 0xD4,
0x56, 0x59, 0x6D, 0xE6, 0x05, 0x3C, 0xDF,
/*5060:*/ 0x3E, 0x77, 0xA8, 0x9D, 0x58, 0xF3, 0x41, 0xBF, 0x43,
0xDF, 0xC5, 0x2E, 0x04, 0xDE, 0x4C, 0x7C,
/*5070:*/ 0x3B, 0xFB, 0x95, 0x81, 0x38, 0x6D, 0xFD, 0x2D, 0x02,
0x6F, 0xC8, 0xEA, 0x43, 0xDF, 0xC5, 0x2E,
/*5080:*/ 0x7B, 0x07, 0x82, 0x94, 0xE6, 0xD3, 0x65, 0xF8, 0x30,
0xC5, 0x0D, 0x6C, 0x1A, 0x22, 0xCB, 0x38,
/*5090:*/ 0x1B, 0xBE, 0xBA, 0x74, 0x3E, 0x77, 0xA8, 0x9D, 0x7F,
0x7E, 0x86, 0xF1, 0xD2, 0xC8, 0x3B, 0x42,
/*50A0:*/ 0xD8, 0x7C, 0x3C, 0x08, 0x15, 0x5D, 0x3E, 0xDE, 0xE0,
0x9C, 0x6D, 0x48, 0x99, 0x7B, 0x6F, 0xC5,
/*50B0:*/ 0x4F, 0xA6, 0xC1, 0x11, 0x0C, 0x48, 0x3A, 0x3A, 0x10,
0x20, 0x84, 0x97, 0xB4, 0x1E, 0x49, 0xD5,
/*50C0:*/ 0x57, 0x89, 0xDA, 0xB2, 0xCB, 0xCC, 0xC1, 0xC4, 0x05,
0x0A, 0xA5, 0xE5, 0xAC, 0xD9, 0x82, 0x35,
/*50D0:*/ 0x5F, 0xF2, 0xFA, 0xF6, 0x15, 0x5D, 0x3E, 0xDE, 0xDD,
0x61, 0xC6, 0xD6, 0xF7, 0x33, 0x41, 0x1A,
/*50E0:*/ 0xA7, 0x54, 0x4C, 0xAD, 0xFC, 0xB8, 0x3C, 0xEB, 0x1E,
0xE6, 0x3B, 0xD2, 0xA2, 0x54, 0x67, 0x83,
/*50F0:*/ 0x8E, 0xB3, 0xD6, 0xCA, 0xF7, 0x33, 0x41, 0x1A, 0x52,
0xE5, 0x37, 0xD4, 0x1D, 0x84, 0xD9, 0x14,
/*5100:*/ 0x43, 0x6C, 0xEF, 0x92, 0xA3, 0xAE, 0xC3, 0xFA, 0x57,
0x89, 0xDA, 0xB2, 0x6A, 0x19, 0xF9, 0x2F,
/*5110:*/ 0x34, 0x2E, 0x7D, 0x84, 0x10, 0x20, 0x84, 0x97, 0xA8,
0x5F, 0xB1, 0x36, 0x59, 0x72, 0x6E, 0xA4,
/*5120:*/ 0x74, 0x2F, 0xB8, 0xE3, 0x72, 0x38, 0xA6, 0x78, 0x79,
0xE7, 0x73, 0x3D, 0x88, 0x5D, 0xD3, 0xB7,
/*5130:*/ 0x12, 0xD1, 0x59, 0x1B, 0xA4, 0xCC, 0xA3, 0x0A, 0x7E,
0xB6, 0xC1, 0x8E, 0x05, 0x7B, 0xD6, 0xA2,
/*5140:*/ 0x3D, 0x76, 0x22, 0xB1, 0xAC, 0xD9, 0x82, 0x35, 0x53,
0xE5, 0xF5, 0x0D, 0x12, 0xBC, 0x59, 0xD4,
/*5150:*/ 0x59, 0x72, 0x6E, 0xA4, 0x0B, 0x91, 0x90, 0x54, 0xE6,
0x05, 0x3C, 0xDF, 0xDC, 0xBD, 0x76, 0x19,
/*5160:*/ 0xA2, 0x54, 0x67, 0x83, 0xEC, 0x88, 0xC7, 0xDB, 0x12,
0x7A, 0x2C, 0xF2, 0xC0, 0x00, 0x3D, 0xCA,
/*5170:*/ 0xF1, 0xC8, 0xF9, 0x48, 0xD3, 0x69, 0xC4, 0x75, 0x8F,
0x08, 0x7C, 0x84, 0x16, 0x31, 0xE8, 0x05,
/*5180:*/ 0xF2, 0xD1, 0x99, 0xBC, 0xE3, 0xA9, 0xC2, 0xC3, 0x57,
0x89, 0xDA, 0xB2, 0x3F, 0x71, 0x9A, 0x6A,
/*5190:*/ 0x84, 0x8E, 0xCA, 0xB6, 0x01, 0xD7, 0x5F, 0xC0, 0xF6,
0x90, 0xE9, 0xDC, 0x02, 0x6D, 0x90, 0x88,
/*51A0:*/ 0x48, 0xE6, 0x76, 0x2D, 0x12, 0xE9, 0xB6, 0xBE, 0x72,
0x38, 0xA6, 0x78, 0xA4, 0xCC, 0xA3, 0x0A,
/*51B0:*/ 0x7D, 0x8C, 0xFD, 0x17, 0x59, 0x72, 0x6E, 0xA4, 0xE6,
0xD3, 0x65, 0xF8, 0x12, 0xD1, 0x59, 0x1B,
/*51C0:*/ 0xD7, 0xE3, 0xE9, 0x28, 0x01, 0x62, 0x83, 0xE7, 0xE7,
0x01, 0x84, 0x50, 0x29, 0x78, 0x60, 0x9B,
/*51D0:*/ 0xD3, 0x37, 0xFF, 0xEE, 0x3F, 0x71, 0x9A, 0x6A, 0x25,
0x32, 0x51, 0x73, 0x87, 0x35, 0xDF, 0x32,
/*51E0:*/ 0xA9, 0x86, 0xA4, 0x12, 0x57, 0x89, 0xDA, 0xB2, 0x11,
0x43, 0xBF, 0x52, 0x00, 0x27, 0x01, 0xB8,
/*51F0:*/ 0x48, 0xF6, 0x2D, 0x5F, 0x12, 0xD1, 0x59, 0x1B, 0x9E,
0xBD, 0x9F, 0x16, 0x0A, 0x4A, 0x75, 0x7B,
/*5200:*/ 0x6C, 0x4E, 0xC0, 0x23, 0xD2, 0xA0, 0x78, 0x54, 0x84,
0xE6, 0x91, 0xA6, 0xC8, 0xF7, 0x44, 0xFE,
/*5210:*/ 0x85, 0x9D, 0xF0, 0x35, 0x9C, 0x26, 0xF2, 0x99, 0x7E,
0x03, 0x73, 0x1D, 0x0C, 0x48, 0x3A, 0x3A,
/*5220:*/ 0x41, 0x0F, 0x82, 0xFB, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*5230:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

void* Block6Data1[] = {
     B6D2_0,
     B6D2_1,
     B6D2_2,
     B6D2_3,
     B6D2_4,
     B6D2_5,
     B6D2_6,
     B6D2_7,
     B6D2_8,
     B6D2_9,
     B6D2_A,
     B6D2_B,
     B6D2_C,
     B6D2_D,
     B6D2_E,
     B6D2_F
};

void* Block6Funcs[] = {
     sub_48DC2D,
     sub_48DD35,
     sub_48DE3D,
     sub_48DF45,
     sub_48E04D,
     sub_48E155,
     sub_48E25D,
     sub_48E364,
     sub_48E46C,
     sub_48E574,
     sub_48E67C,
     sub_48E783,
     sub_48E88B,
     sub_48E993,
     sub_48EA9B,
     sub_48EBA3,
     sub_48ECAB,
     sub_48EDB3,
     sub_48EEBB,
     sub_48EFC3,
     sub_48F0CB,
     sub_48F1D3,
     sub_48F2DB,
     sub_48F3E3,
     sub_48F4EB,
     sub_48F5F3,
     sub_48F6FB,
     sub_48F803,
     sub_48F90B,
     sub_48FA13,
     sub_48FB1B,
     sub_48FC20,
     sub_48FD28,
     sub_48FE2F,
     sub_48FF37,
     sub_49003F,
     sub_490147,
     sub_49024F,
     sub_490356,
     sub_49045E,
     sub_490566,
     sub_49066E,
     sub_490776,
     sub_49087E,
     sub_490983,
     sub_490A8B,
     sub_490B93,
     sub_490C9B,
     sub_490DA3,
     sub_490EAB,
     sub_490FB3,
     sub_4910BB,
     sub_4911C3,
     sub_4912CB,
     sub_4913D3,
     sub_4914DB,
     sub_4915E3,
     sub_4916EB,
     sub_4917F3,
     sub_4918FB,
     sub_491A03,
     sub_491B0B,
     sub_491C13,
     sub_491D1B,
     sub_491E20,
     sub_491F28,
     sub_492030,
     sub_492138,
     sub_492240,
     sub_492345,
     sub_49244D,
     sub_492555,
     sub_49265C,
     sub_492764,
     sub_49286C,
     sub_492974,
     sub_492A7C,
     sub_492B81,
     sub_492C89,
     sub_492D91,
     sub_492E99,
     sub_492FA1,
     sub_4930A9,
     sub_4931B1,
     sub_4932B9,
     sub_4933C1,
     sub_4934C9,
     sub_4935D1,
     sub_4936D9,
     sub_4937E1,
     sub_4938E9,
     sub_4939F1,
     sub_493AF9,
     sub_493C01,
     sub_493D09,
     sub_493E11,
     sub_493F19,
     sub_494021,
     sub_494129,
     sub_494231,
     sub_494339,
     sub_494441,
     sub_494549,
     sub_494651,
     sub_494759,
     sub_494861,
     sub_494969,
     sub_494A71,
     sub_494B79,
     sub_494C81,
     sub_494D89,
     sub_494E91,
     sub_494F99,
     sub_4950A1,
     sub_4951A9,
     sub_4952B1,
     sub_4953B9,
     sub_4954BE,
     sub_4955C6,
     sub_4956CE,
     sub_4957D6,
     sub_4958DE,
     sub_4959E6,
     sub_495AEE,
     sub_495BF6,
     sub_495CFE,
     sub_495E06,
     sub_495F0E,
     sub_496016,
     sub_49611E,
     sub_496226,
     sub_49632E,
     sub_496436,
     sub_49653E,
     sub_496645,
     sub_49674D,
     sub_496855,
     sub_49695C,
     sub_496A64,
     sub_496B6C,
     sub_496C74,
     sub_496D7C,
     sub_496E84,
     sub_496F8C,
     sub_497094,
     sub_49719C,
     sub_4972A4,
     sub_4973AC,
     sub_4974B4,
     sub_4975BC,
     sub_4976C1,
     sub_4977C9,
     sub_4978D1,
     sub_4979D9,
     sub_497AE1,
     sub_497BE9,
     sub_497CF1,
     sub_497DF9,
     sub_497F01,
     sub_498009,
     sub_498111,
     sub_498219,
     sub_498321,
     sub_498429,
     sub_498531,
     sub_498639,
     sub_498740,
     sub_498848,
     sub_498950,
     sub_498A58,
     sub_498B5D,
     sub_498C65,
     sub_498D6D,
     sub_498E75,
     sub_498F7D,
     sub_499085,
     sub_49918D,
     sub_499295,
     sub_49939D,
     sub_4994A5,
     sub_4995AA,
     sub_4996B2,
     sub_4997BA,
     sub_4998C2,
     sub_4999CA,
     sub_499AD2,
     sub_499BDA,
     sub_499CDF,
     sub_499DE7,
     sub_499EEF,
     sub_499FF7,
     sub_49A0FF,
     sub_49A207,
     sub_49A30F,
     sub_49A417,
     sub_49A51F,
     sub_49A627,
     sub_49A72F,
     sub_49A836,
     sub_49A93E,
     sub_49AA46,
     sub_49AB4E,
     sub_49AC56,
     sub_49AD5E,
     sub_49AE66,
     sub_49AF6E,
     sub_49B076,
     sub_49B17E,
     sub_49B286,
     sub_49B38E,
     sub_49B496,
     sub_49B59E,
     sub_49B6A6,
     sub_49B7AE,
     sub_49B8B6,
     sub_49B9BE,
     sub_49BAC6,
     sub_49BBCE,
     sub_49BCD6,
     sub_49BDDE,
     sub_49BEE6,
     sub_49BFEE,
     sub_49C0F6,
     sub_49C1FB,
     sub_49C303,
     sub_49C40B,
     sub_49C512,
     sub_49C61A,
     sub_49C722,
     sub_49C82A,
     sub_49C932,
     sub_49CA3A,
     sub_49CB42,
     sub_49CC4A,
     sub_49CD52,
     sub_49CE5A,
     sub_49CF62,
     sub_49D06A,
     sub_49D172,
     sub_49D27A,
     sub_49D382,
     sub_49D48A,
     sub_49D592,
     sub_49D69A,
     sub_49D7A2,
     sub_49D8AA,
     sub_49D9B2,
     sub_49DABA,
     sub_49DBC2,
     sub_49DCCA,
     sub_49DDD2,
     sub_49DED7,
     sub_49DFDF,
     sub_49E0E7,
     sub_49E1EF,
     sub_49E2F7
};


BYTE dword_4DF3C0[] = {
/*0000:*/ 0x6B, 0x5F, 0xAD, 0x79, 0x21, 0x59, 0xAD, 0x79, 0x03,
0x67, 0x7B, 0xB2, 0xD9, 0xC2, 0xD3, 0x06,
/*0010:*/ 0xD8, 0x07, 0x4F, 0xCB, 0xC9, 0x00, 0xF9, 0x22, 0x73,
0xF0, 0x84, 0xC3, 0xE3, 0xDA, 0xFC, 0xB7,
/*0020:*/ 0xA9, 0x4E, 0x96, 0xDF, 0x6A, 0x56, 0xAC, 0x09, 0xD3,
0x55, 0x42, 0x46, 0x7E, 0xFB, 0x08, 0x94,
/*0030:*/ 0xC1, 0xBD, 0x62, 0xE3, 0x30, 0x2D, 0x00, 0xE4, 0x32,
0xC6, 0xAE, 0x0B, 0x73, 0xE2, 0x7D, 0x3A,
/*0040:*/ 0xB8, 0x0C, 0xAD, 0x79, 0x95, 0x1C, 0x9E, 0xCB, 0x4D,
0xAE, 0x9E, 0xCB, 0x19, 0xB8, 0x36, 0x7F,
/*0050:*/ 0x08, 0x96, 0xAA, 0xB2, 0x32, 0x94, 0x1C, 0x5B, 0x50,
0x12, 0x6E, 0xBA, 0x39, 0xE6, 0x19, 0xCE,
/*0060:*/ 0xA5, 0xC8, 0x73, 0xA6, 0x93, 0xFA, 0x49, 0x70, 0x5B,
0x43, 0xA7, 0x3F, 0x20, 0x93, 0xED, 0xED,
/*0070:*/ 0x32, 0xCE, 0x87, 0x9A, 0xA7, 0x46, 0xE5, 0x9D, 0x1A,
0xA6, 0x4B, 0x72, 0x79, 0xFA, 0x98, 0x43,
/*0080:*/ 0xCC, 0x3E, 0x7B, 0xB2, 0x8D, 0x72, 0x9E, 0xCB, 0x49,
0x75, 0xE0, 0xB4, 0xF1, 0xD2, 0xEF, 0xB4,
/*0090:*/ 0xF7, 0xAB, 0x7C, 0x79, 0xD9, 0xB3, 0xCA, 0x90, 0xD9,
0x01, 0xB8, 0x71, 0x4D, 0x37, 0xCF, 0x05,
/*00A0:*/ 0x35, 0x8B, 0xA5, 0x6D, 0xA5, 0x33, 0x9F, 0xBB, 0x04,
0xC9, 0x71, 0xF4, 0x68, 0x94, 0x3B, 0x26,
/*00B0:*/ 0xD0, 0xAA, 0x51, 0x51, 0xD4, 0xB8, 0x33, 0x56, 0xF6,
0xB9, 0x9D, 0xB9, 0x8D, 0x4D, 0x4E, 0x88,
/*00C0:*/ 0xCB, 0x53, 0xD3, 0x06, 0x82, 0x5F, 0x36, 0x7F, 0x14,
0xA6, 0xE0, 0xB4, 0x84, 0x0D, 0xD4, 0xCD,
/*00D0:*/ 0x41, 0xED, 0xD4, 0xCD, 0xA3, 0xD8, 0x62, 0x24, 0xDF,
0x54, 0x10, 0xC5, 0x55, 0xFB, 0x67, 0xB1,
/*00E0:*/ 0x83, 0xDE, 0x0D, 0xD9, 0x20, 0x25, 0x37, 0x0F, 0x4D,
0x49, 0xD9, 0x40, 0x2F, 0x47, 0x93, 0x92,
/*00F0:*/ 0x5C, 0x4A, 0xF6, 0xE5, 0x6F, 0x42, 0x9B, 0xE2, 0xCA,
0x23, 0x35, 0x0D, 0xC0, 0xBF, 0xE6, 0x3C,
/*0100:*/ 0x61, 0xD3, 0x4F, 0xCB, 0x2F, 0xA6, 0xAA, 0xB2, 0x9F,
0xA0, 0x7C, 0x79, 0xD4, 0xBC, 0xD4, 0xCD,
/*0110:*/ 0x29, 0x36, 0xFE, 0xE9, 0xB5, 0xD2, 0xFE, 0xE9, 0x26,
0x6B, 0x8C, 0x08, 0xFC, 0x9C, 0xFB, 0x7C,
/*0120:*/ 0xDF, 0xAC, 0x9E, 0x14, 0x90, 0x2F, 0xAB, 0xC2, 0x41,
0x9E, 0x45, 0x8D, 0xDE, 0x74, 0x0F, 0x5F,
/*0130:*/ 0xA4, 0x21, 0x65, 0x28, 0x78, 0x60, 0x07, 0x2F, 0xD6,
0x5B, 0xA9, 0xC0, 0x80, 0x8C, 0x7A, 0xF1,
/*0140:*/ 0xDA, 0x17, 0xF9, 0x22, 0xE4, 0xBD, 0x1C, 0x5B, 0xE1,
0xAC, 0xCA, 0x90, 0xD6, 0x88, 0x62, 0x24,
/*0150:*/ 0xBF, 0xB4, 0xFE, 0xE9, 0x83, 0xC8, 0x3A, 0xE1, 0x58,
0x82, 0x3A, 0xE1, 0xF0, 0xE0, 0x4D, 0x95,
/*0160:*/ 0xFC, 0xED, 0x27, 0xFD, 0xE6, 0x00, 0x1D, 0x2B, 0xAE,
0x0E, 0xF3, 0x64, 0x54, 0xD7, 0xB9, 0xB6,
/*0170:*/ 0x01, 0x60, 0xD3, 0xC1, 0x99, 0xC1, 0xB1, 0xC6, 0xCD,
0x13, 0x1F, 0x29, 0xDF, 0x9C, 0xCC, 0x18,
/*0180:*/ 0x80, 0x21, 0x8B, 0xC3, 0x88, 0x10, 0x6E, 0xBA, 0x5A,
0x25, 0xB8, 0x71, 0xEA, 0x0B, 0x10, 0xC5,
/*0190:*/ 0xF7, 0x70, 0x8C, 0x08, 0x2A, 0x8F, 0x3A, 0xE1, 0x74,
0xC9, 0x3F, 0x74, 0x3C, 0x9F, 0x3F, 0x74,
/*01A0:*/ 0x24, 0xEC, 0x55, 0x1C, 0x17, 0xA5, 0x6F, 0xCA, 0x20,
0xA3, 0x81, 0x85, 0xDE, 0xA4, 0xCB, 0x57,
/*01B0:*/ 0x01, 0x44, 0xA1, 0x20, 0x48, 0x5A, 0xC3, 0x27, 0x39,
0x1C, 0x6D, 0xC8, 0x86, 0xE6, 0xBE, 0xF9,
/*01C0:*/ 0x21, 0x7C, 0xFC, 0xB7, 0x51, 0xDD, 0x19, 0xCE, 0x92,
0x1D, 0xCF, 0x05, 0x2A, 0xCE, 0x67, 0xB1,
/*01D0:*/ 0xDC, 0x97, 0xFB, 0x7C, 0x43, 0xFF, 0x4D, 0x95, 0xAE,
0x32, 0x3F, 0x74, 0xD7, 0xE7, 0x22, 0x68,
/*01E0:*/ 0x24, 0x88, 0x22, 0x68, 0x7B, 0xB3, 0x18, 0xBE, 0xBA,
0x4E, 0xF6, 0xF1, 0x4E, 0x34, 0xBC, 0x23,
/*01F0:*/ 0x6F, 0xFC, 0xD6, 0x54, 0x94, 0x59, 0xB4, 0x53, 0x63,
0x98, 0x15, 0xBC, 0x22, 0xAA, 0xC9, 0x8D,
/*0200:*/ 0x28, 0xB1, 0x96, 0xDF, 0x97, 0xBC, 0x73, 0xA6, 0x89,
0xF6, 0xA5, 0x6D, 0x63, 0xC9, 0x0D, 0xD9,
/*0210:*/ 0xE7, 0x23, 0x91, 0x14, 0xD2, 0x70, 0x27, 0xFD, 0x74,
0x20, 0x55, 0x1C, 0x0A, 0x39, 0x22, 0x68,
/*0220:*/ 0x11, 0x8F, 0x72, 0xD6, 0xEA, 0xFF, 0x72, 0xD6, 0x8D,
0xA4, 0x9C, 0x99, 0x32, 0x04, 0xD6, 0x4B,
/*0230:*/ 0xB8, 0x0C, 0xBC, 0x3C, 0xA3, 0x16, 0xDE, 0x3B, 0x33,
0xF4, 0x70, 0xD4, 0x78, 0x67, 0xA3, 0xE5,
/*0240:*/ 0x98, 0x28, 0xAC, 0x09, 0xB1, 0x83, 0x49, 0x70, 0xE0,
0x63, 0x9F, 0xBB, 0x13, 0x00, 0x37, 0x0F,
/*0250:*/ 0x5C, 0xDA, 0xAB, 0xC2, 0xBA, 0x23, 0x1D, 0x2B, 0x1B,
0x8F, 0x6F, 0xCA, 0x85, 0x5C, 0x18, 0xBE,
/*0260:*/ 0x39, 0xE2, 0x72, 0xD6, 0xC9, 0x48, 0xA6, 0x4F, 0x01,
0xFC, 0xA6, 0x4F, 0xC9, 0x09, 0xEC, 0x9D,
/*0270:*/ 0xBC, 0x73, 0x86, 0xEA, 0x69, 0x72, 0xE4, 0xED, 0xED,
0x6F, 0x4A, 0x02, 0xC1, 0x0C, 0x99, 0x33,
/*0280:*/ 0xE2, 0x8B, 0x42, 0x46, 0x4C, 0xF9, 0xA7, 0x3F, 0xF5,
0x16, 0x71, 0xF4, 0x9E, 0x56, 0xD9, 0x40,
/*0290:*/ 0xF8, 0x68, 0x45, 0x8D, 0x79, 0x1D, 0xF3, 0x64, 0x3B,
0x86, 0x81, 0x85, 0xD1, 0xE9, 0xF6, 0xF1,
/*02A0:*/ 0xDB, 0x74, 0x9C, 0x99, 0x1F, 0xBB, 0xA6, 0x4F, 0x2F,
0x03, 0x02, 0xD2, 0xA5, 0x18, 0x02, 0xD2,
/*02B0:*/ 0xE0, 0x5B, 0x68, 0xA5, 0x2B, 0x62, 0x05, 0xA2, 0x14,
0x2B, 0xA4, 0x4D, 0xDD, 0xC9, 0x77, 0x7C,
/*02C0:*/ 0x49, 0xC0, 0x08, 0x94, 0x2A, 0x22, 0xED, 0xED, 0x73,
0x18, 0x3B, 0x26, 0xF1, 0xF9, 0x9C, 0x92,
/*02D0:*/ 0x4F, 0x90, 0x0F, 0x5F, 0x33, 0x10, 0xB9, 0xB6, 0x8F,
0xF8, 0xCB, 0x57, 0xFB, 0xD8, 0xBC, 0x23,
/*02E0:*/ 0x5D, 0x0E, 0xD6, 0x4B, 0x41, 0x71, 0xEC, 0x9D, 0x2B,
0xF6, 0x02, 0xD2, 0x50, 0xC1, 0x22, 0x77,
/*02F0:*/ 0x66, 0x7C, 0x22, 0x77, 0xAE, 0x75, 0x40, 0x70, 0x6E,
0xB6, 0xEE, 0x9F, 0x1F, 0x40, 0x3D, 0xAE,
/*0300:*/ 0xE0, 0x72, 0x62, 0xE3, 0x56, 0x99, 0x87, 0x9A, 0x4D,
0xA1, 0x51, 0x51, 0x41, 0x18, 0xF9, 0xE5,
/*0310:*/ 0x16, 0x55, 0x65, 0x28, 0x75, 0xA8, 0xD3, 0xC1, 0x0E,
0x4C, 0xA1, 0x20, 0xB0, 0x68, 0xD6, 0x54,
/*0320:*/ 0x48, 0xEF, 0xBC, 0x3C, 0x1A, 0xC0, 0x86, 0xEA, 0x49,
0x8D, 0x68, 0xA5, 0x8D, 0x65, 0x22, 0x77,
/*0330:*/ 0x1F, 0x5D, 0x25, 0x07, 0x28, 0xEE, 0x2A, 0x07, 0x28,
0x08, 0x84, 0xE8, 0x2B, 0xF1, 0x57, 0xD9,
/*0340:*/ 0x4C, 0xBA, 0x00, 0xE4, 0x0A, 0x03, 0xEA, 0x9D, 0xB0,
0x8A, 0x33, 0x56, 0xCF, 0x28, 0x9B, 0xE2,
/*0350:*/ 0x9F, 0x4D, 0x07, 0x2F, 0xC9, 0x52, 0xB1, 0xC6, 0x7D,
0xF8, 0xC3, 0x27, 0x2A, 0x5C, 0xB4, 0x53,
/*0360:*/ 0x47, 0xC4, 0xDE, 0x3B, 0xEA, 0xAE, 0xEB, 0xED, 0xD4,
0xE3, 0x0A, 0xA2, 0x07, 0x64, 0x40, 0x70,
/*0370:*/ 0x66, 0x67, 0x2A, 0x07, 0x83, 0xC2, 0xE6, 0xEF, 0x83,
0x61, 0xE6, 0xEF, 0xEE, 0xB4, 0x35, 0xDE,
/*0380:*/ 0xE2, 0xA5, 0xAE, 0x0B, 0x0E, 0x38, 0x4B, 0x72, 0x2F,
0x92, 0x9D, 0xB9, 0xE9, 0xC5, 0x35, 0x0D,
/*0390:*/ 0xF5, 0x9A, 0xA9, 0xC0, 0x0A, 0x1A, 0x1F, 0x29, 0x4E,
0xD0, 0x6D, 0xC8, 0x50, 0xC8, 0x1A, 0xBC,
/*03A0:*/ 0x8F, 0xEA, 0x70, 0xD4, 0x8E, 0x44, 0x4A, 0x02, 0x49,
0x01, 0xA4, 0x4D, 0xC9, 0x0E, 0xEE, 0x9F,
/*03B0:*/ 0x24, 0xB0, 0x84, 0xE8, 0x23, 0x57, 0xE6, 0xEF, 0x03,
0x88, 0x9B, 0x31, 0x1A, 0x6C, 0x9B, 0x31,
/*03C0:*/ 0xF2, 0x6F, 0x7D, 0x3A, 0x64, 0x79, 0x98, 0x43, 0x95,
0xEC, 0x4E, 0x88, 0x8B, 0x81, 0xE6, 0x3C,
/*03D0:*/ 0xFB, 0x33, 0x7A, 0xF1, 0xC1, 0x52, 0xCC, 0x18, 0x3F,
0x26, 0xBE, 0xF9, 0xAE, 0x84, 0xC9, 0x8D,
/*03E0:*/ 0x1C, 0x22, 0xA3, 0xE5, 0x06, 0xAA, 0x99, 0x33, 0x06,
0xBB, 0x78, 0x7C, 0x0C, 0x5B, 0x3D, 0xAE,
/*03F0:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};


BYTE dword_4D92CC[] = {
0x8E, 0x8C, 0xDF, 0x65, 0x0B, 0x07, 0x3A, 0x1C, 0xD9, 0xCD, 0xEC,
0xD7, 0xC1, 0xE8, 0x44, 0x63,
0xBD, 0x33, 0xD8, 0xAE, 0x8F, 0x38, 0x6E, 0x47, 0xE1, 0x8B, 0x1C,
0xA6, 0x06, 0x8C, 0x6B, 0xD2,
0x6D, 0x61, 0x01, 0xBA, 0x69, 0xAD, 0x3B, 0x6C, 0xF5, 0x65, 0xD5,
0x23, 0xC8, 0xEA, 0x9F, 0xF1,
0xDD, 0x5F, 0xF5, 0x86, 0x6C, 0xFB, 0x97, 0x81, 0x1A, 0xEE, 0x39,
0x6E, 0xDD, 0x25, 0xEA, 0x5F
};
```

```
void* off_4DDCDC[] = {
          SixBlock0,
          SixBlock1,
          SixBlock2,
          SixBlock3,
          SixBlock4,
          SixBlock5,
          SixBlock6,
          SixBlock7,
          SixBlock8,
          SixBlock9,
          SixBlockA,
          SixBlockB,
          SixBlockB,
          SixBlockD,
          SixBlockE,
          SixBlockF
};

BYTE byte_4DDBA0[] = {
0x00, 0x03, 0x0F, 0x0E, 0x02, 0x0D, 0x06, 0x0D, 0x03, 0x09, 0x03,
0x0B, 0x0C, 0x0C, 0x06, 0x01,
0x0D, 0x07, 0x04, 0x0D, 0x08, 0x0A, 0x00, 0x0B, 0x05, 0x05, 0x08,
0x0A, 0x09, 0x05, 0x0D, 0x08,
0x05, 0x0F, 0x01, 0x0C, 0x0A, 0x09, 0x04, 0x02, 0x0E, 0x06, 0x08,
0x05, 0x08, 0x07, 0x0B, 0x09,
0x05, 0x0A, 0x03, 0x00, 0x0E, 0x0C, 0x0C, 0x0D, 0x0A, 0x09, 0x08,
0x09, 0x05, 0x0D, 0x0F, 0x0C,
0x08, 0x08, 0x04, 0x0A, 0x08, 0x0F, 0x0B, 0x08, 0x05, 0x00, 0x0B,
0x0A, 0x08, 0x00, 0x01, 0x0D,
0x00, 0x04, 0x0E, 0x06, 0x0A, 0x01, 0x02, 0x04, 0x09, 0x0D, 0x03,
0x0D, 0x0C, 0x08, 0x0A, 0x0C,
0x0D, 0x0B, 0x03, 0x09, 0x09, 0x0C, 0x04, 0x0E, 0x05, 0x0C, 0x0E,
0x0B, 0x09, 0x0A, 0x05, 0x02,
0x04, 0x0A, 0x0A, 0x08, 0x09, 0x0B, 0x09, 0x09, 0x07, 0x0E, 0x00,
0x0D, 0x09, 0x05, 0x08, 0x02,
0x07, 0x05, 0x03, 0x01, 0x07, 0x00, 0x06, 0x0C, 0x0A, 0x04, 0x09,
0x0E, 0x05, 0x03, 0x01, 0x0C,
0x0E, 0x0C, 0x0C, 0x07, 0x07, 0x02, 0x0C, 0x0A, 0x08, 0x09, 0x02,
0x04, 0x0C, 0x07, 0x0A, 0x08,
0x06, 0x00, 0x03, 0x04, 0x07, 0x00, 0x0A, 0x02, 0x01, 0x05, 0x0E,
0x03, 0x00, 0x0A, 0x06, 0x03,
0x08, 0x08, 0x0F, 0x08, 0x06, 0x00, 0x0C, 0x00, 0x02, 0x00, 0x0F,
0x02, 0x01, 0x05, 0x09, 0x07,
0x00, 0x06, 0x0E, 0x0A, 0x0B, 0x09, 0x02, 0x07, 0x04, 0x0C, 0x0C,
0x0F, 0x07, 0x02, 0x0B, 0x0A,
0x0B, 0x0D, 0x0F, 0x0E, 0x06, 0x03, 0x0A, 0x04, 0x07, 0x05, 0x0C,
0x0F, 0x05, 0x0A, 0x0A, 0x06,
0x0A, 0x08, 0x05, 0x09, 0x03, 0x02, 0x04, 0x00, 0x0B, 0x02, 0x06,
0x02, 0x05, 0x0B, 0x01, 0x04,
0x09, 0x06, 0x0C, 0x01, 0x03, 0x00, 0x0C, 0x07, 0x0D, 0x03, 0x0E,
0x07,
};

BYTE dword_552CA9[] = {0x0E, 0x2A, 0x85, 0x64};


//block 6 dynamic functions
void* Block6DynamicFuncs[] = {
sub_48D4EE,
sub_48D2A4,
sub_482654,
sub_48A696,
sub_48B8EB,
sub_48B4C8,
sub_47F71C,
sub_48DA6B,
sub_48A34A,
sub_48778D,
sub_48BCA8,
sub_489D38,
sub_486E92,
sub_485AD2,
sub_48A4A6,
sub_484B20,
sub_48873D,
sub_48D647,
sub_48649F,
sub_4857D3,
sub_48A2BE,
sub_48ABB6,
sub_489EBA,
sub_486D34,
sub_48AEC3,
sub_4850F1,
sub_4821A5,
sub_487EE3,
sub_4896E4,
sub_48BAC0,
sub_484F0B,
sub_48D8AF,
sub_487F9B,
sub_48B85F,
sub_485051,
sub_47F7E9,
sub_485593,
sub_48468F,
sub_4847E1,
sub_487692,
sub_482781,
sub_485315,
sub_486128,
sub_48B379,
sub_4838D4,
sub_488E61,
sub_489A35,
sub_48A589,
sub_483784,
sub_48B048,
sub_48830C,
sub_48D6F8,
sub_48363D,
sub_4808ED,
sub_4837FF,
sub_489F92,
sub_48572F,
sub_486088,
sub_48C479,
sub_484526,
sub_47FD40,
sub_4851C2,
sub_482511,
sub_4872DC,
sub_486C52,
sub_489299,
sub_485EFB,
sub_4867A8,
sub_483D1B,
sub_48D987,
sub_48D37A,
sub_482347,
sub_47FE0F,
sub_48B144,
sub_48C809,
sub_48ADAB,
sub_484DC4,
sub_48A8A9,
sub_488671,
sub_489AE0,
sub_48A3DB,
sub_488260,
sub_4859B7,
sub_485898,
sub_48BF8D,
sub_487BED,
sub_483136,
sub_485479,
sub_48B41E,
sub_489500,
sub_4853D4,
sub_480513,
sub_480753,
sub_48027A,
sub_48C558,
sub_48818D,
sub_4826EF,
sub_489C62,
sub_486362,
sub_4868CA,
sub_48C8AB,
sub_483C44,
sub_48CE93,
sub_4898DB,
sub_4806A8,
sub_48839F,
sub_484D34,
sub_48C5F7,
sub_48903D,
sub_482AC5,
sub_4879C2,
sub_4848BA,
sub_487CA9,
sub_48565C,
sub_485C4D,
sub_48AAED,
sub_482867,
sub_484BCA,
sub_483549,
sub_480ABC,
sub_48654F,
sub_489214,
sub_48A749,
sub_485280,
sub_482FB4,
sub_482EFE,
sub_47FA7F,
sub_4803F9,
sub_485CCB,
sub_48DAF1,
sub_485A3D,
sub_4840CF,
sub_487137,
sub_482930,
sub_48CAF8,
sub_48D461,
sub_484315,
sub_4833EE,
sub_48A015,
sub_488F97,
sub_483208,
sub_488CA2,
sub_487B44,
sub_4823C8,
sub_48097F,
sub_4829D3,
sub_480350,
sub_4845BB,
sub_484488,
sub_48B65C,
sub_48A993,
sub_487DEA,
sub_482E3D,
sub_488055,
sub_48349D,
sub_484E68,
sub_488108,
sub_48246C,
sub_482C9E,
sub_488475,
sub_486299,
sub_489BB2,
sub_48BED9,
sub_48DBAA,
sub_483EB0,
sub_484003,
sub_48682F,
sub_480022,
sub_487043,
sub_487383,
sub_488C12,
sub_489798,
sub_4861C8,
sub_47FCB2,
sub_48A0FB,
sub_4889F5,
sub_48A60F,
sub_48CFE9,
sub_483F62,
sub_47FBF8,
sub_48493A,
sub_48C274,
sub_4899A6,
sub_48BE35,
sub_4885F8,
sub_4836E0,
sub_487916,
sub_487445,
sub_48C973,
sub_48640A,
sub_48B2BC,
sub_488F0A,
sub_48A1B3,
sub_48C180,
sub_483349,
sub_48AF5D,
sub_483976,
sub_48CF27,
sub_489832,
sub_48BB6B,
sub_48D1F8,
sub_4832AE,
sub_48B761,
sub_48D098,
sub_47F9AE,
sub_484A99,
sub_48B9EF,
sub_488B2B,
sub_48CDAE,
sub_47FF6D,
sub_48BC05,
sub_483B62,
sub_48854E,
sub_48912A,
sub_4888F0,
sub_482B40,
sub_485E46,
sub_47F8EF,
sub_487D4D,
sub_4875A3,
sub_48C3D7,
sub_48D7F5,
sub_4874F5,
sub_486A5F,
sub_48C776,
sub_48D11F,
sub_48B1EC,
sub_48C332,
sub_484752,
sub_48CC9F,
sub_48B5B5,
sub_48AA4C,
sub_4865F8,
sub_4807FD,
sub_488AA6,
sub_480A1B,
sub_4801E3,
sub_484255,
sub_4843C4,
sub_48A7DD,
sub_48C6A1,
sub_485BB2,
sub_480491,
sub_484C97,
sub_48259B,
sub_484F93,
sub_488803,
sub_482D75,
sub_4866AC,
sub_4822B2,
sub_47FB2E,
sub_489419
};
```

```
//block 6 dynamic functions sub data
DWORD dword_4D9374 = 0xBD3CAC82;
DWORD dword_4D9370 = 0xBA510D2C;
DWORD dword_4D937C = 0xDDC1D886;
DWORD dword_4D9380 = 0xB7740927;
DWORD dword_4D93AC = 0x338F1DEE;
DWORD dword_4D93B0 = 0x182A7565;
DWORD dword_4D93A8 = 0xE4505EBF;
DWORD dword_4D9378 = 0xB8EA996B;
DWORD dword_4D93A4 = 0xE994CC65;
DWORD dword_4D9388 = 0xA7E50904;
DWORD dword_4D938C = 0xB9E1AC81;
DWORD dword_4D9394 = 0x202FC031;
DWORD dword_4D9398 = 0xD70E4E5A;
DWORD dword_4D939C = 0xE1B4BEFB;
DWORD dword_4D93A0 = 0x20CC263B;
DWORD dword_4D9390 = 0xD072C8D7;
DWORD dword_4D9384 = 0x6A073847;

DWORD unk_552CA9 = 0x64852A0E;


//block 6 dynamic functions sub functions layer 1

void* off_4DDC9C = &sub_47CFB0;
void* off_4DDCA0 = &sub_47D0EF;
void* off_4DDCA4 = &sub_47D2B5;
void* off_4DDCA8 = &sub_47D4F9;
void* off_4DDCAC = &sub_47D6CE;
void* off_4DDCB0 = &sub_47D9DB;
void* off_4DDCB4 = &sub_47DCDA;
void* off_4DDCB8 = &sub_47DE96;
void* off_4DDCBC = &sub_47E17A;
void* off_4DDCC0 = &sub_47E2FC;
void* off_4DDCC4 = &sub_47E508;
void* off_4DDCC8 = &sub_47E746;
void* off_4DDCCC = &sub_47E9B4;
void* off_4DDCD0 = &sub_47EB90;
void* off_4DDCD4 = &sub_47EE6B;
void* off_4DDCD8 = &sub_47F09F;
void* off_4DDCDC_2 = &sub_47D04F;
void* off_4DDCE0 = &sub_47D1D2;
void* off_4DDCE4 = &sub_47D3D7;
void* off_4DDCE8 = &sub_47D5E3;
void* off_4DDCEC = &sub_47D854;
void* off_4DDCF0 = &sub_47DB59;
void* off_4DDCF4 = &sub_47DDB8;
void* off_4DDCF8 = &sub_47E008;
void* off_4DDCFC = &sub_47E23B;
void* off_4DDD00 = &sub_47E402;
```

```
void* off_4DDD04 = &sub_47E627;
void* off_4DDD08 = &sub_47E87C;
void* off_4DDD0C = &sub_47EAA2;
void* off_4DDD10 = &sub_47ECFE;
void* off_4DDD14 = &sub_47EF85;
void* off_4DDD18 = &sub_47F22B;


//block 6 original dynamic call addresses

DWORD B6OriginalFuncs[] =
{
0x48D4EE,
0x48D2A4,
0x482654,
0x48A696,
0x48B8EB,
0x48B4C8,
0x47F71C,
0x48DA6B,
0x48A34A,
0x48778D,
0x48BCA8,
0x489D38,
0x486E92,
0x485AD2,
0x48A4A6,
0x484B20,
0x48873D,
0x48D647,
0x48649F,
0x4857D3,
0x48A2BE,
0x48ABB6,
0x489EBA,
0x486D34,
0x48AEC3,
0x4850F1,
0x4821A5,
0x487EE3,
0x4896E4,
0x48BAC0,
0x484F0B,
0x48D8AF,
0x487F9B,
0x48B85F,
0x485051,
0x47F7E9,
0x485593,
0x48468F,
0x4847E1,
0x487692,
0x482781,
0x485315,
0x486128,
0x48B379,
0x4838D4,
0x488E61,
0x489A35,
0x48A589,
0x483784,
0x48B048,
0x48830C,
0x48D6F8,
0x48363D,
0x4808ED,
0x4837FF,
0x489F92,
0x48572F,
0x486088,
0x48C479,
0x484526,
0x47FD40,
0x4851C2,
0x482511,
0x4872DC,
0x486C52,
0x489299,
0x485EFB,
0x4867A8,
0x483D1B,
0x48D987,
0x48D37A,
0x482347,
0x47FE0F,
0x48B144,
0x48C809,
0x48ADAB,
0x484DC4,
0x48A8A9,
0x488671,
0x489AE0,
0x48A3DB,
0x488260,
0x4859B7,
0x485898,
0x48BF8D,
0x487BED,
0x483136,
0x485479,
0x48B41E,
0x489500,
0x4853D4,
0x480513,
0x480753,
0x48027A,
0x48C558,
0x48818D,
0x4826EF,
0x489C62,
0x486362,
0x4868CA,
0x48C8AB,
0x483C44,
0x48CE93,
0x4898DB,
0x4806A8,
0x48839F,
0x484D34,
0x48C5F7,
0x48903D,
0x482AC5,
0x4879C2,
0x4848BA,
0x487CA9,
0x48565C,
0x485C4D,
0x48AAED,
0x482867,
0x484BCA,
0x483549,
0x480ABC,
0x48654F,
0x489214,
0x48A749,
0x485280,
0x482FB4,
0x482EFE,
0x47FA7F,
0x4803F9,
0x485CCB,
0x48DAF1,
0x485A3D,
0x4840CF,
0x487137,
0x482930,
0x48CAF8,
0x48D461,
0x484315,
0x4833EE,
0x48A015,
0x488F97,
0x483208,
0x488CA2,
0x487B44,
0x4823C8,
0x48097F,
0x4829D3,
0x480350,
0x4845BB,
0x484488,
0x48B65C,
0x48A993,
0x487DEA,
0x482E3D,
0x488055,
0x48349D,
0x484E68,
0x488108,
0x48246C,
0x482C9E,
0x488475,
0x486299,
0x489BB2,
0x48BED9,
0x48DBAA,
0x483EB0,
0x484003,
0x48682F,
0x480022,
0x487043,
0x487383,
0x488C12,
0x489798,
0x4861C8,
0x47FCB2,
0x48A0FB,
0x4889F5,
0x48A60F,
0x48CFE9,
0x483F62,
0x47FBF8,
0x48493A,
0x48C274,
0x4899A6,
0x48BE35,
0x4885F8,
0x4836E0,
0x487916,
0x487445,
0x48C973,
0x48640A,
0x48B2BC,
0x488F0A,
0x48A1B3,
0x48C180,
0x483349,
0x48AF5D,
0x483976,
0x48CF27,
0x489832,
0x48BB6B,
0x48D1F8,
0x4832AE,
0x48B761,
0x48D098,
0x47F9AE,
0x484A99,
0x48B9EF,
0x488B2B,
0x48CDAE,
0x47FF6D,
0x48BC05,
0x483B62,
0x48854E,
0x48912A,
0x4888F0,
0x482B40,
0x485E46,
0x47F8EF,
0x487D4D,
0x4875A3,
0x48C3D7,
0x48D7F5,
0x4874F5,
0x486A5F,
0x48C776,
0x48D11F,
0x48B1EC,
0x48C332,
0x484752,
0x48CC9F,
0x48B5B5,
0x48AA4C,
0x4865F8,
0x4807FD,
0x488AA6,
0x480A1B,
0x4801E3,
0x484255,
0x4843C4,
0x48A7DD,
0x48C6A1,
0x485BB2,
0x480491,
0x484C97,
0x48259B,
0x484F93,
0x488803,
0x482D75,
0x4866AC,
0x4822B2,
0x47FB2E,
0x489419
};



/////////////////////////////
// block 7
/////////////////////////////

BYTE B7D0[] = {
/*00CF80:*/ 0x84, 0x39, 0x50, 0xC7, 0x2E, 0x48, 0xD1, 0x0C, 0x5B,
0xF9, 0xF1, 0x79, 0x3A, 0xAE, 0xF3, 0xB1,
/*00CF90:*/ 0xF8, 0x55, 0x25, 0x38, 0xDB, 0xAA, 0x73, 0x23, 0x05,
0x28, 0xA5, 0x40, 0xBD, 0xC5, 0x30, 0xDA,
/*00CFA0:*/ 0xC5, 0xA6, 0x1F, 0x24, 0x44, 0x10, 0xA9, 0x33, 0xCA,
0x82, 0x1C, 0x53, 0x9C, 0xD9, 0x61, 0x70,
/*00CFB0:*/ 0xF0, 0xE3, 0xB9, 0x2B, 0x47, 0xF3, 0x13, 0xEA, 0x22,
0x09, 0xCD, 0x6A, 0x8C, 0x97, 0x6F, 0xA7,
/*00CFC0:*/ 0x6D, 0x38, 0x50, 0xC7, 0x2F, 0x48, 0xD1, 0x0C, 0x5A,
0xF9, 0xF1, 0x79, 0x3D, 0xAE, 0xF3, 0xB1,
/*00CFD0:*/ 0xF8, 0x55, 0x25, 0x38, 0xD0, 0xAA, 0x73, 0x23, 0x03,
0x28, 0xA5, 0x40, 0xFD, 0xC2, 0x30, 0xDA,
/*00CFE0:*/ 0xE5, 0xA6, 0x1F, 0x24, 0x58, 0x10, 0xA9, 0x33, 0x55,
0x82, 0x1C, 0x53, 0xBD, 0xD9, 0x61, 0x70,
/*00CFF0:*/ 0xFC, 0xE3, 0xB9, 0x2B, 0x85, 0xFF, 0x13, 0xEA, 0x3D,
0x09, 0xCD, 0x6A, 0xCC, 0x97, 0x6F, 0xA7,
/*00D000:*/ 0xB0, 0x39, 0x50, 0xC7, 0x3C, 0x48, 0xD1, 0x0C, 0x5A,
0xF9, 0xF1, 0x79, 0x87, 0xAE, 0xF3, 0xB1,
/*00D010:*/ 0xB1, 0x55, 0x25, 0x38, 0x99, 0xAA, 0x73, 0x23, 0x3F,
0x28, 0xA5, 0x40, 0x93, 0xC5, 0x30, 0xDA,
/*00D020:*/ 0xE9, 0xA6, 0x1F, 0x24, 0x15, 0x10, 0xA9, 0x33, 0xE7,
0x82, 0x1C, 0x53, 0xA5, 0xD9, 0x61, 0x70,
/*00D030:*/ 0x2E, 0xE2, 0xB9, 0x2B, 0xAA, 0xF2, 0x13, 0xEA, 0x5C,
0x09, 0xCD, 0x6A, 0x94, 0x95, 0x6F, 0xA7,
/*00D040:*/ 0xAF, 0x39, 0x50, 0xC7, 0x19, 0x48, 0xD1, 0x0C, 0x12,
0xF9, 0xF1, 0x79, 0x38, 0xAE, 0xF3, 0xB1,
/*00D050:*/ 0x2D, 0x54, 0x25, 0x38, 0xDE, 0xAA, 0x73, 0x23, 0x7D,
0x28, 0xA5, 0x40, 0xCD, 0xC5, 0x30, 0xDA,
/*00D060:*/ 0xBE, 0xA6, 0x1F, 0x24, 0x2B, 0x11, 0xA9, 0x33, 0x8D,
0x82, 0x1C, 0x53, 0x93, 0xD9, 0x61, 0x70,
/*00D070:*/ 0xFE, 0xE3, 0xB9, 0x2B, 0x8B, 0xF2, 0x13, 0xEA, 0x2B,
0x09, 0xCD, 0x6A, 0xCD, 0x97, 0x6F, 0xA7,
/*00D080:*/ 0xA0, 0x39, 0x50, 0xC7, 0x2E, 0x48, 0xD1, 0x0C, 0x68,
0xF9, 0xF1, 0x79, 0x19, 0xAE, 0xF3, 0xB1,
/*00D090:*/ 0xEB, 0x55, 0x25, 0x38, 0xAC, 0xA8, 0x73, 0x23, 0x28,
0x28, 0xA5, 0x40, 0x18, 0xC5, 0x30, 0xDA,
/*00D0A0:*/ 0xA3, 0xA6, 0x1F, 0x24, 0x4B, 0x10, 0xA9, 0x33, 0x14,
0x83, 0x1C, 0x53, 0xA7, 0xD8, 0x61, 0x70,
/*00D0B0:*/ 0xED, 0xE3, 0xB9, 0x2B, 0x85, 0xF2, 0x13, 0xEA, 0x3F,
0x09, 0xCD, 0x6A, 0xE2, 0x97, 0x6F, 0xA7,
/*00D0C0:*/ 0xD2, 0x39, 0x50, 0xC7, 0x0A, 0x48, 0xD1, 0x0C, 0x6B,
0xF9, 0xF1, 0x79, 0x38, 0xAE, 0xF3, 0xB1,
/*00D0D0:*/ 0x68, 0x54, 0x25, 0x38, 0xD0, 0xAA, 0x73, 0x23, 0x0F,
0x28, 0xA5, 0x40, 0xAC, 0xC5, 0x30, 0xDA,
/*00D0E0:*/ 0xFA, 0xA6, 0x1F, 0x24, 0x1D, 0x12, 0xA9, 0x33, 0xDC,
0x82, 0x1C, 0x53, 0x8C, 0xD9, 0x61, 0x70,
/*00D0F0:*/ 0xFC, 0xE3, 0xB9, 0x2B, 0x88, 0xF3, 0x13, 0xEA, 0x36,
0x09, 0xCD, 0x6A, 0x75, 0x97, 0x6F, 0xA7,
/*00D100:*/ 0xA2, 0x39, 0x50, 0xC7, 0x20, 0x48, 0xD1, 0x0C, 0x40,
0xF9, 0xF1, 0x79, 0x0F, 0xAC, 0xF3, 0xB1,
/*00D110:*/ 0xD7, 0x55, 0x25, 0x38, 0xCF, 0xAA, 0x73, 0x23, 0x05,
0x28, 0xA5, 0x40, 0x8E, 0xC5, 0x30, 0xDA,
/*00D120:*/ 0xD7, 0xA6, 0x1F, 0x24, 0x4E, 0x10, 0xA9, 0x33, 0xCA,
0x82, 0x1C, 0x53, 0x8B, 0xD9, 0x61, 0x70,
/*00D130:*/ 0xFD, 0xE3, 0xB9, 0x2B, 0x90, 0xF2, 0x13, 0xEA, 0x3F,
0x09, 0xCD, 0x6A, 0xA1, 0x97, 0x6F, 0xA7,
/*00D140:*/ 0x4A, 0x39, 0x50, 0xC7, 0x3F, 0x48, 0xD1, 0x0C, 0x55,
0xF9, 0xF1, 0x79, 0x4D, 0xAC, 0xF3, 0xB1,
/*00D150:*/ 0xBB, 0x55, 0x25, 0x38, 0xDE, 0xAA, 0x73, 0x23, 0xE6,
0x28, 0xA5, 0x40, 0xF3, 0xCD, 0x30, 0xDA,
/*00D160:*/ 0xF1, 0xA6, 0x1F, 0x24, 0xD8, 0x11, 0xA9, 0x33, 0xC1,
0x82, 0x1C, 0x53, 0x82, 0xD9, 0x61, 0x70,
/*00D170:*/ 0x72, 0xE3, 0xB9, 0x2B, 0xF8, 0xF2, 0x13, 0xEA, 0x81,
0xF6, 0x32, 0x95, 0xC0, 0x96, 0x6F, 0xA7,
/*00D180:*/ 0xB7, 0x39, 0x50, 0xC7, 0xB6, 0x48, 0xD1, 0x0C, 0x40,
0xF9, 0xF1, 0x79, 0xCB, 0x51, 0x0C, 0x4E,
/*00D190:*/ 0x6B, 0x54, 0x25, 0x38, 0xCF, 0xAA, 0x73, 0x23, 0x17,
0x28, 0xA5, 0x40, 0xD5, 0xC5, 0x30, 0xDA,
/*00D1A0:*/ 0xF3, 0xA6, 0x1F, 0x24, 0x7D, 0x10, 0xA9, 0x33, 0xCA,
0x82, 0x1C, 0x53, 0x97, 0x26, 0x9E, 0x8F,
/*00D1B0:*/ 0xD2, 0xE3, 0xB9, 0x2B, 0x85, 0xF2, 0x13, 0xEA, 0x3E,
0x09, 0xCD, 0x6A, 0xC6, 0x97, 0x6F, 0xA7,
/*00D1C0:*/ 0xB7, 0x39, 0x50, 0xC7, 0x0D, 0xF0, 0xAD, 0xBA, 0x0D,
0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*00D1D0:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};

BYTE B7D1[] = {
/*00D8D0:*/ 0x73, 0xC7, 0xAF, 0x38, 0xFA, 0x48, 0xD1, 0x0C, 0x0E,
0xF9, 0xF1, 0x79, 0x37, 0xAE, 0xF3, 0xB1,
/*00D8E0:*/ 0xDF, 0x55, 0x25, 0x38, 0x90, 0xAA, 0x73, 0x23, 0x8B,
0x28, 0xA5, 0x40, 0xB9, 0xC5, 0x30, 0xDA,
/*00D8F0:*/ 0xB7, 0xA6, 0x1F, 0x24, 0x11, 0x10, 0xA9, 0x33, 0xB6,
0x80, 0x1C, 0x53, 0xA3, 0xD9, 0x61, 0x70,
/*00D900:*/ 0xF2, 0xE3, 0xB9, 0x2B, 0x2E, 0xF3, 0x13, 0xEA, 0x66,
0x09, 0xCD, 0x6A, 0xD8, 0x97, 0x6F, 0xA7,
/*00D910:*/ 0xD4, 0x39, 0x50, 0xC7, 0x37, 0x48, 0xD1, 0x0C, 0xBF,
0x06, 0x0E, 0x86, 0x43, 0xAE, 0xF3, 0xB1,
/*00D920:*/ 0xD2, 0x55, 0x25, 0x38, 0x69, 0xA8, 0x73, 0x23, 0x2C,
0x28, 0xA5, 0x40, 0xB6, 0xC5, 0x30, 0xDA,
/*00D930:*/ 0x38, 0xA6, 0x1F, 0x24, 0x99, 0x10, 0xA9, 0x33, 0xE2,
0x80, 0x1C, 0x53, 0xAA, 0xD9, 0x61, 0x70,
/*00D940:*/ 0xF3, 0xE3, 0xB9, 0x2B, 0x04, 0xF2, 0x13, 0xEA, 0x2F,
0x09, 0xCD, 0x6A, 0xD9, 0x97, 0x6F, 0xA7,
/*00D950:*/ 0xA0, 0x39, 0x50, 0xC7, 0x26, 0x48, 0xD1, 0x0C, 0x02,
0xF9, 0xF1, 0x79, 0x17, 0xAE, 0xF3, 0xB1,
/*00D960:*/ 0xEB, 0x54, 0x25, 0x38, 0xC6, 0xA2, 0x73, 0x23, 0x4C,
0x28, 0xA5, 0x40, 0x73, 0xC5, 0x30, 0xDA,
/*00D970:*/ 0xEB, 0xA6, 0x1F, 0x24, 0x03, 0x10, 0xA9, 0x33, 0x2D,
0x82, 0x1C, 0x53, 0xD5, 0xD9, 0x61, 0x70,
/*00D980:*/ 0xF7, 0xE3, 0xB9, 0x2B, 0x9E, 0xF2, 0x13, 0xEA, 0xC3,
0x09, 0xCD, 0x6A, 0xD2, 0x97, 0x6F, 0xA7,
/*00D990:*/ 0x3A, 0x39, 0x50, 0xC7, 0x20, 0x48, 0xD1, 0x0C, 0x5B,
0xF9, 0xF1, 0x79, 0x3D, 0xAE, 0xF3, 0xB1,
/*00D9A0:*/ 0xEB, 0x55, 0x25, 0x38, 0xD0, 0xAA, 0x73, 0x23, 0x04,
0x28, 0xA5, 0x40, 0xB6, 0xC5, 0x30, 0xDA,
/*00D9B0:*/ 0xB6, 0xA6, 0x1F, 0x24, 0x02, 0x10, 0xA9, 0x33, 0xD4,
0x82, 0x1C, 0x53, 0x35, 0xD9, 0x61, 0x70,
/*00D9C0:*/ 0xC2, 0xE3, 0xB9, 0x2B, 0x45, 0xF2, 0x13, 0xEA, 0x2B,
0x09, 0xCD, 0x6A, 0x99, 0x97, 0x6F, 0xA7,
/*00D9D0:*/ 0xEA, 0x39, 0x50, 0xC7, 0x27, 0x48, 0xD1, 0x0C, 0x5B,
0xF9, 0xF1, 0x79, 0x71, 0xAE, 0xF3, 0xB1,
/*00D9E0:*/ 0xEA, 0x55, 0x25, 0x38, 0xEA, 0xAA, 0x73, 0x23, 0x20,
0x28, 0xA5, 0x40, 0x8E, 0xC5, 0x30, 0xDA,
/*00D9F0:*/ 0xCA, 0xA6, 0x1F, 0x24, 0xBD, 0x10, 0xA9, 0x33, 0xC8,
0x82, 0x1C, 0x53, 0x95, 0xD9, 0x61, 0x70,
/*00DA00:*/ 0xCE, 0xE3, 0xB9, 0x2B, 0x84, 0xF2, 0x13, 0xEA, 0x11,
0x09, 0xCD, 0x6A, 0xA7, 0x97, 0x6F, 0xA7,
/*00DA10:*/ 0xA0, 0x39, 0x50, 0xC7, 0x3D, 0x48, 0xD1, 0x0C, 0x6E,
0xF9, 0xF1, 0x79, 0x0D, 0xAE, 0xF3, 0xB1,
/*00DA20:*/ 0xF9, 0x55, 0x25, 0x38, 0xD9, 0xAA, 0x73, 0x23, 0x08,
0x28, 0xA5, 0x40, 0xA6, 0xC5, 0x30, 0xDA,
/*00DA30:*/ 0x22, 0x59, 0xE0, 0xDB, 0x5B, 0x10, 0xA9, 0x33, 0xEF,
0x82, 0x1C, 0x53, 0x3B, 0x26, 0x9E, 0x8F,
/*00DA40:*/ 0xBB, 0xE3, 0xB9, 0x2B, 0x99, 0xF2, 0x13, 0xEA, 0x30,
0x09, 0xCD, 0x6A, 0x95, 0x96, 0x6F, 0xA7,
/*00DA50:*/ 0xB4, 0x39, 0x50, 0xC7, 0x91, 0xB7, 0x2E, 0xF3, 0x88,
0x06, 0x0E, 0x86, 0x39, 0xAE, 0xF3, 0xB1,
/*00DA60:*/ 0xF4, 0x55, 0x25, 0x38, 0xD9, 0xAA, 0x73, 0x23, 0x14,
0x28, 0xA5, 0x40, 0xCF, 0xC5, 0x30, 0xDA,
/*00DA70:*/ 0x90, 0xA7, 0x1F, 0x24, 0x3B, 0x10, 0xA9, 0x33, 0x18,
0x82, 0x1C, 0x53, 0xEB, 0xD9, 0x61, 0x70,
/*00DA80:*/ 0xE5, 0xE2, 0xB9, 0x2B, 0xD4, 0xF2, 0x13, 0xEA, 0x16,
0x0B, 0xCD, 0x6A, 0xD7, 0x97, 0x6F, 0xA7,
/*00DA90:*/ 0xF0, 0x39, 0x50, 0xC7, 0x32, 0x49, 0xD1, 0x0C, 0xCB,
0xF9, 0xF1, 0x79, 0x0B, 0xAE, 0xF3, 0xB1,
/*00DAA0:*/ 0xF4, 0x55, 0x25, 0x38, 0xD7, 0xAA, 0x73, 0x23, 0x0E,
0x28, 0xA5, 0x40, 0x5D, 0xC5, 0x30, 0xDA,
/*00DAB0:*/ 0xE0, 0xA6, 0x1F, 0x24, 0xE5, 0xEF, 0x56, 0xCC, 0xF5,
0x82, 0x1C, 0x53, 0x8C, 0xD9, 0x61, 0x70,
/*00DAC0:*/ 0xFC, 0xE3, 0xB9, 0x2B, 0x8A, 0xF2, 0x13, 0xEA, 0x32,
0x09, 0xCD, 0x6A, 0xFB, 0x95, 0x6F, 0xA7,
/*00DAD0:*/ 0xA3, 0x39, 0x50, 0xC7, 0x32, 0x48, 0xD1, 0x0C, 0x84,
0xF9, 0xF1, 0x79, 0x08, 0xAE, 0xF3, 0xB1,
/*00DAE0:*/ 0xF9, 0x55, 0x25, 0x38, 0xDE, 0xAA, 0x73, 0x23, 0xD1,
0x28, 0xA5, 0x40, 0xBA, 0xC5, 0x30, 0xDA,
/*00DAF0:*/ 0xE7, 0xA6, 0x1F, 0x24, 0x24, 0x11, 0xA9, 0x33, 0xCB,
0x82, 0x1C, 0x53, 0x88, 0xD9, 0x61, 0x70,
/*00DB00:*/ 0xEA, 0xE3, 0xB9, 0x2B, 0x85, 0xF2, 0x13, 0xEA, 0x23,
0x09, 0xCD, 0x6A, 0xD9, 0x95, 0x6F, 0xA7,
/*00DB10:*/ 0xA2, 0x39, 0x50, 0xC7, 0xEC, 0xB7, 0x2E, 0xF3, 0x2E,
0xF9, 0xF1, 0x79, 0x2E, 0xAE, 0xF3, 0xB1,
/*00DB20:*/ 0xDD, 0x55, 0x25, 0x38, 0xD2, 0xAA, 0x73, 0x23, 0x12,
0x28, 0xA5, 0x40, 0xA2, 0xC5, 0x30, 0xDA,
/*00DB30:*/ 0xF4, 0xA6, 0x1F, 0x24, 0x02, 0x10, 0xA9, 0x33, 0x7C,
0x82, 0x1C, 0x53, 0xB4, 0xD9, 0x61, 0x70,
/*00DB40:*/ 0x68, 0xE3, 0xB9, 0x2B, 0x5D, 0xF2, 0x13, 0xEA, 0x3F,
0x09, 0xCD, 0x6A, 0xF8, 0x97, 0x6F, 0xA7,
/*00DB50:*/ 0xBE, 0x39, 0x50, 0xC7, 0x22, 0x48, 0xD1, 0x0C, 0x6D,
0xF9, 0xF1, 0x79, 0x29, 0xAE, 0xF3, 0xB1,
/*00DB60:*/ 0xF9, 0x55, 0x25, 0x38, 0xC0, 0xAA, 0x73, 0x23, 0x0C,
0x28, 0xA5, 0x40, 0x97, 0xC5, 0x30, 0xDA,
/*00DB70:*/ 0x3C, 0xA6, 0x1F, 0x24, 0x04, 0x10, 0xA9, 0x33, 0x8D,
0x82, 0x1C, 0x53, 0x83, 0xD9, 0x61, 0x70,
/*00DB80:*/ 0xF4, 0xE3, 0xB9, 0x2B, 0x56, 0x0D, 0xEC, 0x15, 0x3D,
0x09, 0xCD, 0x6A, 0xD8, 0x97, 0x6F, 0xA7,
/*00DB90:*/ 0x8C, 0x39, 0x50, 0xC7, 0x2E, 0x48, 0xD1, 0x0C, 0x5A,
0xF9, 0xF1, 0x79, 0x3A, 0xAE, 0xF3, 0xB1,
/*00DBA0:*/ 0xEB, 0x55, 0x25, 0x38, 0xD8, 0xAA, 0x73, 0x23, 0x85,
0xD6, 0x5A, 0xBF, 0x94, 0xC5, 0x30, 0xDA,
/*00DBB0:*/ 0xE6, 0xA6, 0x1F, 0x24, 0x0D, 0xF0, 0xAD, 0xBA, 0x0D,
0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*00DBC0:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};

BYTE B7D2[] = {
/*00E1F0:*/ 0xA0, 0x39, 0x50, 0xC7, 0x3F, 0x48, 0xD1, 0x0C, 0x79,
0xF9, 0xF1, 0x79, 0x70, 0xAE, 0xF3, 0xB1,
/*00E200:*/ 0x8B, 0x55, 0x25, 0x38, 0x8B, 0xA8, 0x73, 0x23, 0x78,
0xD6, 0x5A, 0xBF, 0x66, 0xC4, 0x30, 0xDA,
/*00E210:*/ 0xB4, 0xA6, 0x1F, 0x24, 0x22, 0x10, 0xA9, 0x33, 0x4B,
0x81, 0x1C, 0x53, 0x97, 0xD9, 0x61, 0x70,
/*00E220:*/ 0x1F, 0xE3, 0xB9, 0x2B, 0x9B, 0xF2, 0x13, 0xEA, 0x28,
0x09, 0xCD, 0x6A, 0xEC, 0x97, 0x6F, 0xA7,
/*00E230:*/ 0xA0, 0x39, 0x50, 0xC7, 0x6A, 0x48, 0xD1, 0x0C, 0x4F,
0xF9, 0xF1, 0x79, 0x3A, 0xAE, 0xF3, 0xB1,
/*00E240:*/ 0xEC, 0x55, 0x25, 0x38, 0xF0, 0xAB, 0x73, 0x23, 0x0A,
0x28, 0xA5, 0x40, 0x87, 0xC5, 0x30, 0xDA,
/*00E250:*/ 0xF3, 0xA6, 0x1F, 0x24, 0x6D, 0x10, 0xA9, 0x33, 0x27,
0x7D, 0xE3, 0xAC, 0x98, 0xD9, 0x61, 0x70,
/*00E260:*/ 0xE7, 0xE2, 0xB9, 0x2B, 0xA2, 0xF0, 0x13, 0xEA, 0xB5,
0x08, 0xCD, 0x6A, 0x0D, 0x97, 0x6F, 0xA7,
/*00E270:*/ 0xA7, 0x39, 0x50, 0xC7, 0x32, 0x48, 0xD1, 0x0C, 0x67,
0xFB, 0xF1, 0x79, 0x33, 0xAE, 0xF3, 0xB1,
/*00E280:*/ 0xAB, 0x55, 0x25, 0x38, 0xCA, 0xAA, 0x73, 0x23, 0x0F,
0x28, 0xA5, 0x40, 0x8D, 0xC5, 0x30, 0xDA,
/*00E290:*/ 0xEA, 0xA6, 0x1F, 0x24, 0x49, 0x10, 0xA9, 0x33, 0x50,
0x82, 0x1C, 0x53, 0x82, 0xD9, 0x61, 0x70,
/*00E2A0:*/ 0xC1, 0xE2, 0xB9, 0x2B, 0x8B, 0xF2, 0x13, 0xEA, 0x3F,
0x09, 0xCD, 0x6A, 0xC9, 0x97, 0x6F, 0xA7,
/*00E2B0:*/ 0xA2, 0x39, 0x50, 0xC7, 0x2F, 0x48, 0xD1, 0x0C, 0x5A,
0xF9, 0xF1, 0x79, 0x38, 0xAE, 0xF3, 0xB1,
/*00E2C0:*/ 0xBF, 0x55, 0x25, 0x38, 0xCB, 0xAA, 0x73, 0x23, 0x04,
0x28, 0xA5, 0x40, 0xB2, 0xC5, 0x30, 0xDA,
/*00E2D0:*/ 0xAA, 0xA6, 0x1F, 0x24, 0x54, 0x10, 0xA9, 0x33, 0xC8,
0x82, 0x1C, 0x53, 0x99, 0xD9, 0x61, 0x70,
/*00E2E0:*/ 0xEC, 0xE3, 0xB9, 0x2B, 0x8A, 0xF2, 0x13, 0xEA, 0x2E,
0x09, 0xCD, 0x6A, 0xCA, 0x97, 0x6F, 0xA7,
/*00E2F0:*/ 0xB1, 0x39, 0x50, 0xC7, 0xEB, 0x48, 0xD1, 0x0C, 0x5B,
0xF9, 0xF1, 0x79, 0x19, 0xAE, 0xF3, 0xB1,
/*00E300:*/ 0x37, 0xAA, 0xDA, 0xC7, 0xEF, 0xAA, 0x73, 0x23, 0x79,
0x29, 0xA5, 0x40, 0x2A, 0xC5, 0x30, 0xDA,
/*00E310:*/ 0xC5, 0xA6, 0x1F, 0x24, 0x32, 0x10, 0xA9, 0x33, 0x96,
0x83, 0x1C, 0x53, 0x8E, 0xD9, 0x61, 0x70,
/*00E320:*/ 0xF1, 0xE3, 0xB9, 0x2B, 0x8A, 0xF2, 0x13, 0xEA, 0x3C,
0x09, 0xCD, 0x6A, 0xD9, 0x97, 0x6F, 0xA7,
/*00E330:*/ 0xA8, 0x39, 0x50, 0xC7, 0x5B, 0x48, 0xD1, 0x0C, 0x5A,
0xF9, 0xF1, 0x79, 0x28, 0xAE, 0xF3, 0xB1,
/*00E340:*/ 0xEC, 0x55, 0x25, 0x38, 0xCE, 0xAA, 0x73, 0x23, 0x04,
0x28, 0xA5, 0x40, 0x06, 0xC5, 0x30, 0xDA,
/*00E350:*/ 0xE7, 0xA6, 0x1F, 0x24, 0x89, 0x10, 0xA9, 0x33, 0xCA,
0x82, 0x1C, 0x53, 0xEA, 0xD9, 0x61, 0x70,
/*00E360:*/ 0xD8, 0xE3, 0xB9, 0x2B, 0x9E, 0xF2, 0x13, 0xEA, 0x28,
0x09, 0xCD, 0x6A, 0xD2, 0x97, 0x6F, 0xA7,
/*00E370:*/ 0xDA, 0x39, 0x50, 0xC7, 0x1C, 0x48, 0xD1, 0x0C, 0x74,
0xF9, 0xF1, 0x79, 0x67, 0xAE, 0xF3, 0xB1,
/*00E380:*/ 0xB7, 0x54, 0x25, 0x38, 0x27, 0x54, 0x8C, 0xDC, 0x04,
0x28, 0xA5, 0x40, 0xBA, 0xC5, 0x30, 0xDA,
/*00E390:*/ 0xEB, 0xA6, 0x1F, 0x24, 0x75, 0x10, 0xA9, 0x33, 0xC8,
0x82, 0x1C, 0x53, 0xC3, 0xD9, 0x61, 0x70,
/*00E3A0:*/ 0x12, 0xE3, 0xB9, 0x2B, 0x8A, 0xF2, 0x13, 0xEA, 0x1B,
0x08, 0xCD, 0x6A, 0xD9, 0x97, 0x6F, 0xA7,
/*00E3B0:*/ 0xAE, 0x39, 0x50, 0xC7, 0x77, 0x48, 0xD1, 0x0C, 0x24,
0xF9, 0xF1, 0x79, 0x07, 0xAE, 0xF3, 0xB1,
/*00E3C0:*/ 0xF8, 0x55, 0x25, 0x38, 0xCD, 0xAA, 0x73, 0x23, 0x11,
0x28, 0xA5, 0x40, 0xB7, 0xC5, 0x30, 0xDA,
/*00E3D0:*/ 0x5A, 0xA6, 0x1F, 0x24, 0x03, 0x10, 0xA9, 0x33, 0xF9,
0x82, 0x1C, 0x53, 0x83, 0xD9, 0x61, 0x70,
/*00E3E0:*/ 0xB2, 0xE3, 0xB9, 0x2B, 0x8B, 0xF2, 0x13, 0xEA, 0x28,
0x09, 0xCD, 0x6A, 0xCF, 0x97, 0x6F, 0xA7,
/*00E3F0:*/ 0x4B, 0x3E, 0x50, 0xC7, 0x1F, 0x48, 0xD1, 0x0C, 0x49,
0xF9, 0xF1, 0x79, 0x41, 0xAF, 0xF3, 0xB1,
/*00E400:*/ 0xA2, 0xAA, 0xDA, 0xC7, 0xD9, 0xAA, 0x73, 0x23, 0x06,
0x28, 0xA5, 0x40, 0xA8, 0xC5, 0x30, 0xDA,
/*00E410:*/ 0xF3, 0xA6, 0x1F, 0x24, 0x0A, 0x10, 0xA9, 0x33, 0xC4,
0x83, 0x1C, 0x53, 0x9F, 0xD9, 0x61, 0x70,
/*00E420:*/ 0xFD, 0xE3, 0xB9, 0x2B, 0x89, 0xF2, 0x13, 0xEA, 0xDD,
0xF6, 0x32, 0x95, 0xEB, 0x97, 0x6F, 0xA7,
/*00E430:*/ 0xB8, 0x39, 0x50, 0xC7, 0xBA, 0x48, 0xD1, 0x0C, 0x7E,
0xF8, 0xF1, 0x79, 0x39, 0xAE, 0xF3, 0xB1,
/*00E440:*/ 0xDC, 0x4C, 0x25, 0x38, 0x14, 0xB2, 0x73, 0x23, 0x23,
0x28, 0xA5, 0x40, 0x18, 0x3A, 0xCF, 0x25,
/*00E450:*/ 0xE5, 0xA6, 0x1F, 0x24, 0x04, 0x10, 0xA9, 0x33, 0xCB,
0x82, 0x1C, 0x53, 0xB5, 0xD9, 0x61, 0x70,
/*00E460:*/ 0x71, 0xE3, 0xB9, 0x2B, 0x8A, 0xF2, 0x13, 0xEA, 0x1B,
0x09, 0xCD, 0x6A, 0xDA, 0x97, 0x6F, 0xA7,
/*00E470:*/ 0xAB, 0x39, 0x50, 0xC7, 0xFD, 0x48, 0xD1, 0x0C, 0x60,
0xF9, 0xF1, 0x79, 0x0D, 0xF0, 0xAD, 0xBA,
/*00E480:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};

BYTE B7D3[] = {
/*00EA80:*/ 0xAF, 0x39, 0x50, 0xC7, 0x21, 0x48, 0xD1, 0x0C, 0x4B,
0xF9, 0xF1, 0x79, 0x0C, 0xAE, 0xF3, 0xB1,
/*00EA90:*/ 0xF7, 0x55, 0x25, 0x38, 0x0E, 0xAB, 0x73, 0x23, 0xA0,
0x28, 0xA5, 0x40, 0xAC, 0xC5, 0x30, 0xDA,
/*00EAA0:*/ 0xF8, 0xA6, 0x1F, 0x24, 0xAF, 0x11, 0xA9, 0x33, 0x99,
0x81, 0x1C, 0x53, 0x0B, 0xD8, 0x61, 0x70,
/*00EAB0:*/ 0xFC, 0xE3, 0xB9, 0x2B, 0x8C, 0xF2, 0x13, 0xEA, 0xF5,
0x08, 0xCD, 0x6A, 0xD8, 0x97, 0x6F, 0xA7,
/*00EAC0:*/ 0x21, 0x39, 0x50, 0xC7, 0x79, 0x4A, 0xD1, 0x0C, 0x5A,
0xF9, 0xF1, 0x79, 0x18, 0xAE, 0xF3, 0xB1,
/*00EAD0:*/ 0xF9, 0x55, 0x25, 0x38, 0xE3, 0xAA, 0x73, 0x23, 0x04,
0x28, 0xA5, 0x40, 0xF1, 0xC7, 0x30, 0xDA,
/*00EAE0:*/ 0xEA, 0xA6, 0x1F, 0x24, 0x0F, 0x10, 0xA9, 0x33, 0x52,
0x7D, 0xE3, 0xAC, 0x8B, 0xD9, 0x61, 0x70,
/*00EAF0:*/ 0x1A, 0x1C, 0x46, 0xD4, 0xA4, 0xF2, 0x13, 0xEA, 0x3F,
0x09, 0xCD, 0x6A, 0xCA, 0x97, 0x6F, 0xA7,
/*00EB00:*/ 0xB1, 0x39, 0x50, 0xC7, 0x39, 0x48, 0xD1, 0x0C, 0x71,
0xF9, 0xF1, 0x79, 0x38, 0xAE, 0xF3, 0xB1,
/*00EB10:*/ 0xF1, 0x55, 0x25, 0x38, 0xCD, 0xAA, 0x73, 0x23, 0x0C,
0x28, 0xA5, 0x40, 0xA9, 0x3A, 0xCF, 0x25,
/*00EB20:*/ 0xF4, 0xA6, 0x1F, 0x24, 0x12, 0x10, 0xA9, 0x33, 0xCB,
0x82, 0x1C, 0x53, 0xE1, 0xD9, 0x61, 0x70,
/*00EB30:*/ 0xFD, 0xE3, 0xB9, 0x2B, 0xF0, 0xF2, 0x13, 0xEA, 0x29,
0x09, 0xCD, 0x6A, 0xDB, 0x97, 0x6F, 0xA7,
/*00EB40:*/ 0xA0, 0x39, 0x50, 0xC7, 0x7F, 0x48, 0xD1, 0x0C, 0x6D,
0xF9, 0xF1, 0x79, 0x31, 0xAE, 0xF3, 0xB1,
/*00EB50:*/ 0x94, 0x55, 0x25, 0x38, 0xC3, 0xAA, 0x73, 0x23, 0x05,
0x28, 0xA5, 0x40, 0x15, 0x3A, 0xCF, 0x25,
/*00EB60:*/ 0x92, 0xA6, 0x1F, 0x24, 0xF2, 0xEF, 0x56, 0xCC, 0x2C,
0x7D, 0xE3, 0xAC, 0x92, 0xD9, 0x61, 0x70,
/*00EB70:*/ 0xE6, 0xE3, 0xB9, 0x2B, 0xCA, 0xF2, 0x13, 0xEA, 0x3F,
0x09, 0xCD, 0x6A, 0xD8, 0x97, 0x6F, 0xA7,
/*00EB80:*/ 0x59, 0x39, 0x50, 0xC7, 0x2E, 0x48, 0xD1, 0x0C, 0x53,
0xF9, 0xF1, 0x79, 0x29, 0xAE, 0xF3, 0xB1,
/*00EB90:*/ 0xF8, 0x55, 0x25, 0x38, 0xD1, 0xAA, 0x73, 0x23, 0x17,
0x28, 0xA5, 0x40, 0xF3, 0x3A, 0xCF, 0x25,
/*00EBA0:*/ 0xE7, 0xA6, 0x1F, 0x24, 0x7F, 0x10, 0xA9, 0x33, 0xCA,
0x82, 0x1C, 0x53, 0x44, 0xD9, 0x61, 0x70,
/*00EBB0:*/ 0xEA, 0xE3, 0xB9, 0x2B, 0x86, 0xF2, 0x13, 0xEA, 0x07,
0x09, 0xCD, 0x6A, 0xCC, 0x97, 0x6F, 0xA7,
/*00EBC0:*/ 0xA1, 0x39, 0x50, 0xC7, 0x31, 0x48, 0xD1, 0x0C, 0x3A,
0xFB, 0xF1, 0x79, 0x2D, 0xAE, 0xF3, 0xB1,
/*00EBD0:*/ 0xC6, 0x55, 0x25, 0x38, 0xD9, 0xAA, 0x73, 0x23, 0xBB,
0xD7, 0x5A, 0xBF, 0xAE, 0xC5, 0x30, 0xDA,
/*00EBE0:*/ 0x1D, 0xA6, 0x1F, 0x24, 0x0C, 0x10, 0xA9, 0x33, 0xCB,
0x82, 0x1C, 0x53, 0xE2, 0xD9, 0x61, 0x70,
/*00EBF0:*/ 0xB9, 0xE3, 0xB9, 0x2B, 0x60, 0x0D, 0xEC, 0x15, 0x04,
0x09, 0xCD, 0x6A, 0xD8, 0x97, 0x6F, 0xA7,
/*00EC00:*/ 0xC3, 0x38, 0x50, 0xC7, 0x18, 0x48, 0xD1, 0x0C, 0x5A,
0xF9, 0xF1, 0x79, 0x26, 0xAF, 0xF3, 0xB1,
/*00EC10:*/ 0xE0, 0x55, 0x25, 0x38, 0xD9, 0xAA, 0x73, 0x23, 0x02,
0x28, 0xA5, 0x40, 0xBA, 0xC5, 0x30, 0xDA,
/*00EC20:*/ 0x87, 0xA4, 0x1F, 0x24, 0x9B, 0x10, 0xA9, 0x33, 0xCB,
0x82, 0x1C, 0x53, 0x25, 0x26, 0x9E, 0x8F,
/*00EC30:*/ 0xFC, 0xE3, 0xB9, 0x2B, 0x9D, 0xF2, 0x13, 0xEA, 0x91,
0x08, 0xCD, 0x6A, 0xD0, 0x97, 0x6F, 0xA7,
/*00EC40:*/ 0x80, 0x38, 0x50, 0xC7, 0x73, 0x48, 0xD1, 0x0C, 0x1C,
0xF9, 0xF1, 0x79, 0x36, 0xAE, 0xF3, 0xB1,
/*00EC50:*/ 0xB7, 0x55, 0x25, 0x38, 0xB2, 0xAA, 0x73, 0x23, 0x05,
0x28, 0xA5, 0x40, 0xA8, 0xC5, 0x30, 0xDA,
/*00EC60:*/ 0x1F, 0xA7, 0x1F, 0x24, 0x18, 0x10, 0xA9, 0x33, 0x83,
0x82, 0x1C, 0x53, 0x19, 0xD9, 0x61, 0x70,
/*00EC70:*/ 0xFF, 0xE3, 0xB9, 0x2B, 0x3E, 0xF2, 0x13, 0xEA, 0x2D,
0x09, 0xCD, 0x6A, 0xE0, 0x97, 0x6F, 0xA7,
/*00EC80:*/ 0xB7, 0x39, 0x50, 0xC7, 0x13, 0x49, 0xD1, 0x0C, 0x6C,
0xF9, 0xF1, 0x79, 0x4A, 0xAE, 0xF3, 0xB1,
/*00EC90:*/ 0xF9, 0x55, 0x25, 0x38, 0xC8, 0xAA, 0x73, 0x23, 0x71,
0x28, 0xA5, 0x40, 0x83, 0xC5, 0x30, 0xDA,
/*00ECA0:*/ 0xE8, 0xA6, 0x1F, 0x24, 0x27, 0x10, 0xA9, 0x33, 0x7D,
0x82, 0x1C, 0x53, 0x83, 0xD9, 0x61, 0x70,
/*00ECB0:*/ 0xF5, 0xE3, 0xB9, 0x2B, 0x95, 0xF2, 0x13, 0xEA, 0x1B,
0x09, 0xCD, 0x6A, 0xEA, 0x97, 0x6F, 0xA7,
/*00ECC0:*/ 0xA2, 0x39, 0x50, 0xC7, 0x94, 0x49, 0xD1, 0x0C, 0x47,
0xF9, 0xF1, 0x79, 0x3B, 0xAE, 0xF3, 0xB1,
/*00ECD0:*/ 0xE6, 0x55, 0x25, 0x38, 0xDB, 0xAA, 0x73, 0x23, 0x83,
0xD4, 0x5A, 0xBF, 0xBD, 0xC5, 0x30, 0xDA,
/*00ECE0:*/ 0xF0, 0xA6, 0x1F, 0x24, 0x76, 0x10, 0xA9, 0x33, 0x8C,
0x82, 0x1C, 0x53, 0xD7, 0xD9, 0x61, 0x70,
/*00ECF0:*/ 0xA2, 0xE3, 0xB9, 0x2B, 0x0D, 0xF0, 0xAD, 0xBA, 0x0D,
0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*00ED00:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
```

```
};

BYTE B7D4[] = {
/*00F330:*/ 0xA9, 0x39, 0x50, 0xC7, 0xDA, 0xB7, 0x2E, 0xF3, 0xCA,
0xF9, 0xF1, 0x79, 0x3A, 0xAE, 0xF3, 0xB1,
/*00F340:*/ 0x5E, 0x55, 0x25, 0x38, 0xD6, 0xAA, 0x73, 0x23, 0x5D,
0x28, 0xA5, 0x40, 0xB1, 0xC5, 0x30, 0xDA,
/*00F350:*/ 0x22, 0xA6, 0x1F, 0x24, 0x03, 0x10, 0xA9, 0x33, 0xFC,
0x82, 0x1C, 0x53, 0x1D, 0xD9, 0x61, 0x70,
/*00F360:*/ 0xFE, 0xE3, 0xB9, 0x2B, 0x77, 0xF3, 0x13, 0xEA, 0x3F,
0x09, 0xCD, 0x6A, 0x6D, 0x9B, 0x6F, 0xA7,
/*00F370:*/ 0xA0, 0x39, 0x50, 0xC7, 0x2E, 0x48, 0xD1, 0x0C, 0xA9,
0xFA, 0xF1, 0x79, 0x23, 0xAE, 0xF3, 0xB1,
/*00F380:*/ 0x8C, 0x55, 0x25, 0x38, 0xDE, 0xAA, 0x73, 0x23, 0x04,
0x28, 0xA5, 0x40, 0xC5, 0xC5, 0x30, 0xDA,
/*00F390:*/ 0xC0, 0xA6, 0x1F, 0x24, 0x24, 0x10, 0xA9, 0x33, 0x92,
0x82, 0x1C, 0x53, 0x91, 0xD9, 0x61, 0x70,
/*00F3A0:*/ 0xF2, 0xE3, 0xB9, 0x2B, 0x81, 0xF2, 0x13, 0xEA, 0x23,
0x09, 0xCD, 0x6A, 0xD0, 0x97, 0x6F, 0xA7,
/*00F3B0:*/ 0x91, 0x39, 0x50, 0xC7, 0x2F, 0x48, 0xD1, 0x0C, 0x44,
0xF8, 0xF1, 0x79, 0x64, 0xAE, 0xF3, 0xB1,
/*00F3C0:*/ 0x85, 0x55, 0x25, 0x38, 0x83, 0xAA, 0x73, 0x23, 0x03,
0x28, 0xA5, 0x40, 0xBB, 0xC5, 0x30, 0xDA,
/*00F3D0:*/ 0x0D, 0xA6, 0x1F, 0x24, 0x08, 0x10, 0xA9, 0x33, 0xD5,
0x82, 0x1C, 0x53, 0x9F, 0x26, 0x9E, 0x8F,
/*00F3E0:*/ 0xE2, 0xE3, 0xB9, 0x2B, 0x81, 0xF2, 0x13, 0xEA, 0xD9,
0x09, 0xCD, 0x6A, 0xD6, 0x97, 0x6F, 0xA7,
/*00F3F0:*/ 0xB2, 0x39, 0x50, 0xC7, 0x26, 0x48, 0xD1, 0x0C, 0xCB,
0xF9, 0xF1, 0x79, 0xF9, 0xAE, 0xF3, 0xB1,
/*00F400:*/ 0x0A, 0x55, 0x25, 0x38, 0xD2, 0xAA, 0x73, 0x23, 0x04,
0x28, 0xA5, 0x40, 0x8D, 0xC5, 0x30, 0xDA,
/*00F410:*/ 0xF2, 0xA6, 0x1F, 0x24, 0x14, 0x10, 0xA9, 0x33, 0xED,
0x82, 0x1C, 0x53, 0xBF, 0xD9, 0x61, 0x70,
/*00F420:*/ 0xC6, 0xE2, 0xB9, 0x2B, 0x97, 0xF2, 0x13, 0xEA, 0x14,
0x0A, 0xCD, 0x6A, 0xDB, 0x97, 0x6F, 0xA7,
/*00F430:*/ 0xE3, 0x39, 0x50, 0xC7, 0x2F, 0x48, 0xD1, 0x0C, 0x5A,
0xF9, 0xF1, 0x79, 0x6E, 0xAE, 0xF3, 0xB1,
/*00F440:*/ 0xB5, 0x55, 0x25, 0x38, 0xD9, 0xAA, 0x73, 0x23, 0x4B,
0x28, 0xA5, 0x40, 0xB5, 0xC5, 0x30, 0xDA,
/*00F450:*/ 0xEC, 0xA6, 0x1F, 0x24, 0x08, 0x10, 0xA9, 0x33, 0xC3,
0x82, 0x1C, 0x53, 0x81, 0xD9, 0x61, 0x70,
/*00F460:*/ 0xFD, 0xE3, 0xB9, 0x2B, 0x40, 0xF2, 0x13, 0xEA, 0x33,
0x09, 0xCD, 0x6A, 0xD9, 0x97, 0x6F, 0xA7,
/*00F470:*/ 0xA8, 0x39, 0x50, 0xC7, 0x2F, 0x48, 0xD1, 0x0C, 0x1E,
0x06, 0x0E, 0x86, 0x28, 0xAE, 0xF3, 0xB1,
/*00F480:*/ 0xF0, 0x55, 0x25, 0x38, 0xD0, 0xAA, 0x73, 0x23, 0x1E,
0x28, 0xA5, 0x40, 0xA7, 0xC5, 0x30, 0xDA,
/*00F490:*/ 0xE7, 0xA6, 0x1F, 0x24, 0x37, 0x10, 0xA9, 0x33, 0xCC,
0x82, 0x1C, 0x53, 0x83, 0xD9, 0x61, 0x70,
/*00F4A0:*/ 0xC8, 0xE3, 0xB9, 0x2B, 0xA3, 0xF2, 0x13, 0xEA, 0x15,
0x0B, 0xCD, 0x6A, 0x84, 0x97, 0x6F, 0xA7,
/*00F4B0:*/ 0x9D, 0x38, 0x50, 0xC7, 0x04, 0x48, 0xD1, 0x0C, 0x5A,
0xF9, 0xF1, 0x79, 0x3A, 0xAE, 0xF3, 0xB1,
/*00F4C0:*/ 0xEE, 0x55, 0x25, 0x38, 0x8F, 0xAA, 0x73, 0x23, 0x52,
0x28, 0xA5, 0x40, 0xBA, 0xC5, 0x30, 0xDA,
/*00F4D0:*/ 0xE7, 0xA6, 0x1F, 0x24, 0x6B, 0xEF, 0x56, 0xCC, 0xE4,
0x82, 0x1C, 0x53, 0xF1, 0xD9, 0x61, 0x70,
/*00F4E0:*/ 0x62, 0xE2, 0xB9, 0x2B, 0x92, 0xF2, 0x13, 0xEA, 0x2D,
0x09, 0xCD, 0x6A, 0xDE, 0x97, 0x6F, 0xA7,
/*00F4F0:*/ 0xE2, 0x39, 0x50, 0xC7, 0x7C, 0x48, 0xD1, 0x0C, 0x40,
0xF9, 0xF1, 0x79, 0x37, 0xAE, 0xF3, 0xB1,
/*00F500:*/ 0xFE, 0x55, 0x25, 0x38, 0xFB, 0xAA, 0x73, 0x23, 0x1E,
0x28, 0xA5, 0x40, 0xBA, 0xC5, 0x30, 0xDA,
/*00F510:*/ 0xB9, 0xA6, 0x1F, 0x24, 0x6A, 0x10, 0xA9, 0x33, 0xDF,
0x82, 0x1C, 0x53, 0x81, 0xD9, 0x61, 0x70,
/*00F520:*/ 0xBA, 0xE3, 0xB9, 0x2B, 0xA1, 0xF2, 0x13, 0xEA, 0xFB,
0x08, 0xCD, 0x6A, 0xD9, 0x97, 0x6F, 0xA7,
/*00F530:*/ 0x75, 0xC4, 0xAF, 0x38, 0x14, 0x48, 0xD1, 0x0C, 0x59,
0xF9, 0xF1, 0x79, 0x38, 0xAE, 0xF3, 0xB1,
/*00F540:*/ 0x19, 0x55, 0x25, 0x38, 0x12, 0xA9, 0x73, 0x23, 0x59,
0x29, 0xA5, 0x40, 0xBB, 0xC5, 0x30, 0xDA,
/*00F550:*/ 0xB8, 0xA6, 0x1F, 0x24, 0x13, 0x10, 0xA9, 0x33, 0xC8,
0x82, 0x1C, 0x53, 0xD6, 0xD9, 0x61, 0x70,
/*00F560:*/ 0xE0, 0xE3, 0xB9, 0x2B, 0xFE, 0xF3, 0x13, 0xEA, 0x36,
0x09, 0xCD, 0x6A, 0x9E, 0x97, 0x6F, 0xA7,
/*00F570:*/ 0xA1, 0x39, 0x50, 0xC7, 0x4F, 0x49, 0xD1, 0x0C, 0x45,
0xF9, 0xF1, 0x79, 0x09, 0xAE, 0xF3, 0xB1,
/*00F580:*/ 0xF6, 0x55, 0x25, 0x38, 0xCB, 0xAA, 0x73, 0x23, 0x4E,
0x28, 0xA5, 0x40, 0xC0, 0xC5, 0x30, 0xDA,
/*00F590:*/ 0x6E, 0xA6, 0x1F, 0x24, 0x0C, 0x10, 0xA9, 0x33, 0xA9,
0x82, 0x1C, 0x53, 0xAB, 0xD8, 0x61, 0x70,
/*00F5A0:*/ 0x66, 0xE3, 0xB9, 0x2B, 0xC8, 0xF0, 0x13, 0xEA, 0x34,
0x09, 0xCD, 0x6A, 0x88, 0x97, 0x6F, 0xA7,
/*00F5B0:*/ 0xAF, 0x39, 0x50, 0xC7, 0xFA, 0xB7, 0x2E, 0xF3, 0x7F,
0xF9, 0xF1, 0x79, 0x2F, 0xAE, 0xF3, 0xB1,
/*00F5C0:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};

BYTE B7D5[] = {
/*00FB60:*/ 0x15, 0x39, 0x50, 0xC7, 0xDD, 0x48, 0xD1, 0x0C, 0x22,
0xF9, 0xF1, 0x79, 0x1E, 0xAE, 0xF3, 0xB1,
/*00FB70:*/ 0xF9, 0x55, 0x25, 0x38, 0xD9, 0xAA, 0x73, 0x23, 0xF9,
0x28, 0xA5, 0x40, 0xBB, 0xC5, 0x30, 0xDA,
/*00FB80:*/ 0xE5, 0xA6, 0x1F, 0x24, 0x45, 0x10, 0xA9, 0x33, 0xD1,
0x82, 0x1C, 0x53, 0x24, 0xD9, 0x61, 0x70,
/*00FB90:*/ 0xA7, 0xE3, 0xB9, 0x2B, 0xBA, 0xF2, 0x13, 0xEA, 0x29,
0xF7, 0x32, 0x95, 0xD2, 0x97, 0x6F, 0xA7,
/*00FBA0:*/ 0xB6, 0x39, 0x50, 0xC7, 0x4E, 0xB5, 0x2E, 0xF3, 0x5D,
0xF9, 0xF1, 0x79, 0x11, 0xAE, 0xF3, 0xB1,
/*00FBB0:*/ 0x6E, 0xAA, 0xDA, 0xC7, 0xCB, 0xAA, 0x73, 0x23, 0x05,
0x28, 0xA5, 0x40, 0xB8, 0xC5, 0x30, 0xDA,
/*00FBC0:*/ 0x89, 0xA4, 0x1F, 0x24, 0x15, 0x10, 0xA9, 0x33, 0xCB,
0x82, 0x1C, 0x53, 0x83, 0xD9, 0x61, 0x70,
/*00FBD0:*/ 0xFD, 0xE3, 0xB9, 0x2B, 0x85, 0xF2, 0x13, 0xEA, 0x36,
0x09, 0xCD, 0x6A, 0xD0, 0x97, 0x6F, 0xA7,
/*00FBE0:*/ 0xB2, 0x39, 0x50, 0xC7, 0x2F, 0x48, 0xD1, 0x0C, 0x52,
0xF9, 0xF1, 0x79, 0x26, 0xAE, 0xF3, 0xB1,
/*00FBF0:*/ 0xC2, 0x55, 0x25, 0x38, 0xD6, 0xAA, 0x73, 0x23, 0x04,
0x28, 0xA5, 0x40, 0x99, 0xC5, 0x30, 0xDA,
/*00FC00:*/ 0x89, 0xA6, 0x1F, 0x24, 0x36, 0x10, 0xA9, 0x33, 0x79,
0x82, 0x1C, 0x53, 0xBD, 0xD9, 0x61, 0x70,
/*00FC10:*/ 0xA1, 0xE3, 0xB9, 0x2B, 0xE6, 0x0C, 0xEC, 0x15, 0x37,
0x09, 0xCD, 0x6A, 0xD7, 0x97, 0x6F, 0xA7,
/*00FC20:*/ 0xB3, 0x39, 0x50, 0xC7, 0xC4, 0xB7, 0x2E, 0xF3, 0x5D,
0xF9, 0xF1, 0x79, 0x38, 0xAE, 0xF3, 0xB1,
/*00FC30:*/ 0xD4, 0x55, 0x25, 0x38, 0xD9, 0xAA, 0x73, 0x23, 0x05,
0x28, 0xA5, 0x40, 0xBC, 0xC4, 0x30, 0xDA,
/*00FC40:*/ 0xF5, 0xA6, 0x1F, 0x24, 0xDF, 0xEF, 0x56, 0xCC, 0xC0,
0x82, 0x1C, 0x53, 0xC6, 0xD9, 0x61, 0x70,
/*00FC50:*/ 0x10, 0x1C, 0x46, 0xD4, 0xD0, 0xF2, 0x13, 0xEA, 0x31,
0x09, 0xCD, 0x6A, 0x55, 0x97, 0x6F, 0xA7,
/*00FC60:*/ 0xF4, 0x39, 0x50, 0xC7, 0x07, 0x48, 0xD1, 0x0C, 0x54,
0xF9, 0xF1, 0x79, 0x3D, 0xAE, 0xF3, 0xB1,
/*00FC70:*/ 0xE1, 0x54, 0x25, 0x38, 0x9A, 0xAA, 0x73, 0x23, 0x05,
0x28, 0xA5, 0x40, 0xBA, 0xC5, 0x30, 0xDA,
/*00FC80:*/ 0x31, 0xA7, 0x1F, 0x24, 0x31, 0x10, 0xA9, 0x33, 0x21,
0x7D, 0xE3, 0xAC, 0x14, 0xD9, 0x61, 0x70,
/*00FC90:*/ 0xFD, 0xE3, 0xB9, 0x2B, 0x9E, 0xF2, 0x13, 0xEA, 0x0A,
0x09, 0xCD, 0x6A, 0xD1, 0x97, 0x6F, 0xA7,
/*00FCA0:*/ 0xB4, 0x39, 0x50, 0xC7, 0x2E, 0x48, 0xD1, 0x0C, 0x4A,
0xF9, 0xF1, 0x79, 0xB5, 0xAE, 0xF3, 0xB1,
/*00FCB0:*/ 0xC2, 0x58, 0x25, 0x38, 0xCB, 0xAA, 0x73, 0x23, 0xDD,
0x28, 0xA5, 0x40, 0xA8, 0xC5, 0x30, 0xDA,
/*00FCC0:*/ 0x1D, 0xA7, 0x1F, 0x24, 0xA3, 0xED, 0x56, 0xCC, 0xDA,
0x82, 0x1C, 0x53, 0x84, 0xD9, 0x61, 0x70,
/*00FCD0:*/ 0xF2, 0xE3, 0xB9, 0x2B, 0x99, 0xF2, 0x13, 0xEA, 0x3F,
0x08, 0xCD, 0x6A, 0xCE, 0x97, 0x6F, 0xA7,
/*00FCE0:*/ 0x51, 0xC6, 0xAF, 0x38, 0xC4, 0x48, 0xD1, 0x0C, 0x74,
0xF9, 0xF1, 0x79, 0x55, 0xAE, 0xF3, 0xB1,
/*00FCF0:*/ 0xF9, 0x55, 0x25, 0x38, 0xDE, 0xAA, 0x73, 0x23, 0x73,
0x28, 0xA5, 0x40, 0x35, 0xC5, 0x30, 0xDA,
/*00FD00:*/ 0xF5, 0xA6, 0x1F, 0x24, 0xEB, 0x11, 0xA9, 0x33, 0xCB,
0x82, 0x1C, 0x53, 0x81, 0xD9, 0x61, 0x70,
/*00FD10:*/ 0x25, 0xE2, 0xB9, 0x2B, 0x86, 0xF2, 0x13, 0xEA, 0x3F,
0x09, 0xCD, 0x6A, 0xEE, 0x97, 0x6F, 0xA7,
/*00FD20:*/ 0xB6, 0x39, 0x50, 0xC7, 0x0F, 0x48, 0xD1, 0x0C, 0x58,
0xF9, 0xF1, 0x79, 0xC7, 0xAE, 0xF3, 0xB1,
/*00FD30:*/ 0x07, 0x54, 0x25, 0x38, 0x54, 0xAA, 0x73, 0x23, 0x14,
0x28, 0xA5, 0x40, 0xA4, 0xC5, 0x30, 0xDA,
/*00FD40:*/ 0xD4, 0xA4, 0x1F, 0x24, 0x1F, 0x10, 0xA9, 0x33, 0xCA,
0x82, 0x1C, 0x53, 0xDE, 0xDB, 0x61, 0x70,
/*00FD50:*/ 0xFC, 0xE3, 0xB9, 0x2B, 0x05, 0x0D, 0xEC, 0x15, 0x3E,
0x09, 0xCD, 0x6A, 0xF5, 0x97, 0x6F, 0xA7,
/*00FD60:*/ 0xB1, 0x3B, 0x50, 0xC7, 0x16, 0x48, 0xD1, 0x0C, 0xD2,
0xF9, 0xF1, 0x79, 0x4A, 0x51, 0x0C, 0x4E,
/*00FD70:*/ 0xB4, 0x55, 0x25, 0x38, 0x00, 0x55, 0x8C, 0xDC, 0x0A,
0x28, 0xA5, 0x40, 0x92, 0xC5, 0x30, 0xDA,
/*00FD80:*/ 0xF3, 0xA6, 0x1F, 0x24, 0x56, 0x10, 0xA9, 0x33, 0xC8,
0x82, 0x1C, 0x53, 0x26, 0x26, 0x9E, 0x8F,
/*00FD90:*/ 0x3E, 0x1C, 0x46, 0xD4, 0xA1, 0xF2, 0x13, 0xEA, 0x34,
0xF7, 0x32, 0x95, 0xD9, 0x97, 0x6F, 0xA7,
/*00FDA0:*/ 0x04, 0x39, 0x50, 0xC7, 0x0D, 0xF0, 0xAD, 0xBA, 0x0D,
0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*00FDB0:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};

BYTE B7D6[] = {
/*04D8:*/ 0xA8, 0xC4, 0xAF, 0x38, 0x57, 0x48, 0xD1, 0x0C,
/*04E0:*/ 0x5D, 0xF9, 0xF1, 0x79, 0xD0, 0xAC, 0xF3, 0xB1, 0x87,
0x55, 0x25, 0x38, 0xD9, 0xAA, 0x73, 0x23,
/*04F0:*/ 0x46, 0x28, 0xA5, 0x40, 0xB3, 0xC5, 0x30, 0xDA, 0xEC,
0xA6, 0x1F, 0x24, 0x04, 0x10, 0xA9, 0x33,
/*0500:*/ 0x94, 0x82, 0x1C, 0x53, 0x80, 0xD9, 0x61, 0x70, 0x82,
0xE3, 0xB9, 0x2B, 0x8B, 0xF2, 0x13, 0xEA,
/*0510:*/ 0x3E, 0x09, 0xCD, 0x6A, 0xD0, 0x97, 0x6F, 0xA7, 0xCD,
0x39, 0x50, 0xC7, 0x02, 0x48, 0xD1, 0x0C,
/*0520:*/ 0x2D, 0xFB, 0xF1, 0x79, 0x28, 0xAE, 0xF3, 0xB1, 0xE5,
0x55, 0x25, 0x38, 0x82, 0xAA, 0x73, 0x23,
/*0530:*/ 0x08, 0x28, 0xA5, 0x40, 0xB7, 0xC5, 0x30, 0xDA, 0xEF,
0xA6, 0x1F, 0x24, 0x25, 0x10, 0xA9, 0x33,
/*0540:*/ 0xDD, 0x82, 0x1C, 0x53, 0xB0, 0xD9, 0x61, 0x70, 0x61,
0xE3, 0xB9, 0x2B, 0xCA, 0xF2, 0x13, 0xEA,
/*0550:*/ 0x3E, 0x09, 0xCD, 0x6A, 0xD7, 0x95, 0x6F, 0xA7, 0xE7,
0x39, 0x50, 0xC7, 0x2E, 0x48, 0xD1, 0x0C,
/*0560:*/ 0x1F, 0xF9, 0xF1, 0x79, 0x38, 0xAE, 0xF3, 0xB1, 0xF5,
0x55, 0x25, 0x38, 0xC8, 0xAA, 0x73, 0x23,
/*0570:*/ 0x7B, 0x28, 0xA5, 0x40, 0xDF, 0xC5, 0x30, 0xDA, 0xED,
0xA6, 0x1F, 0x24, 0x3F, 0x10, 0xA9, 0x33,
/*0580:*/ 0xF6, 0x80, 0x1C, 0x53, 0xA0, 0xD9, 0x61, 0x70, 0xA6,
0xE3, 0xB9, 0x2B, 0x1B, 0xF3, 0x13, 0xEA,
/*0590:*/ 0x28, 0x09, 0xCD, 0x6A, 0xBC, 0x97, 0x6F, 0xA7, 0xFE,
0x39, 0x50, 0xC7, 0x2F, 0x48, 0xD1, 0x0C,
/*05A0:*/ 0xD1, 0xF9, 0xF1, 0x79, 0xBB, 0x50, 0x0C, 0x4E, 0x87,
0x55, 0x25, 0x38, 0xFE, 0xAA, 0x73, 0x23,
/*05B0:*/ 0x04, 0x28, 0xA5, 0x40, 0xDF, 0x3A, 0xCF, 0x25, 0xD2,
0xA6, 0x1F, 0x24, 0x0C, 0x10, 0xA9, 0x33,
/*05C0:*/ 0xCA, 0x82, 0x1C, 0x53, 0xE0, 0xD9, 0x61, 0x70, 0x8B,
0xE3, 0xB9, 0x2B, 0xAA, 0xF2, 0x13, 0xEA,
/*05D0:*/ 0x37, 0x09, 0xCD, 0x6A, 0xF5, 0x97, 0x6F, 0xA7, 0xA1,
0x39, 0x50, 0xC7, 0x2F, 0x48, 0xD1, 0x0C,
/*05E0:*/ 0x5A, 0xF9, 0xF1, 0x79, 0x36, 0xAE, 0xF3, 0xB1, 0xF8,
0x55, 0x25, 0x38, 0x80, 0xAA, 0x73, 0x23,
/*05F0:*/ 0x08, 0x2A, 0xA5, 0x40, 0xBB, 0xC5, 0x30, 0xDA, 0xF1,
0xA6, 0x1F, 0x24, 0x2D, 0x10, 0xA9, 0x33,
/*0600:*/ 0xCB, 0x82, 0x1C, 0x53, 0x9E, 0xD9, 0x61, 0x70, 0xFB,
0xE3, 0xB9, 0x2B, 0xB6, 0xF2, 0x13, 0xEA,
/*0610:*/ 0x29, 0x09, 0xCD, 0x6A, 0xCB, 0x97, 0x6F, 0xA7, 0xA8,
0x39, 0x50, 0xC7, 0xF5, 0x48, 0xD1, 0x0C,
/*0620:*/ 0x5A, 0xF9, 0xF1, 0x79, 0x34, 0xAE, 0xF3, 0xB1, 0xF1,
0x55, 0x25, 0x38, 0xD8, 0xAA, 0x73, 0x23,
/*0630:*/ 0x04, 0x28, 0xA5, 0x40, 0xBB, 0xC5, 0x30, 0xDA, 0xBE,
0xBF, 0x1F, 0x24, 0x2E, 0x10, 0xA9, 0x33,
/*0640:*/ 0xC7, 0x82, 0x1C, 0x53, 0x45, 0x26, 0x9E, 0x8F, 0xA2,
0xE3, 0xB9, 0x2B, 0x96, 0xF2, 0x13, 0xEA,
/*0650:*/ 0x79, 0x09, 0xCD, 0x6A, 0x9F, 0x97, 0x6F, 0xA7, 0xE0,
0x39, 0x50, 0xC7, 0x2E, 0x48, 0xD1, 0x0C,
/*0660:*/ 0x3E, 0xF9, 0xF1, 0x79, 0x29, 0xAE, 0xF3, 0xB1, 0xD4,
0x55, 0x25, 0x38, 0xFE, 0xAA, 0x73, 0x23,
/*0670:*/ 0x04, 0x28, 0xA5, 0x40, 0xA0, 0xC5, 0x30, 0xDA, 0xE6,
0xA6, 0x1F, 0x24, 0x12, 0x10, 0xA9, 0x33,
/*0680:*/ 0x7F, 0x82, 0x1C, 0x53, 0x89, 0xD9, 0x61, 0x70, 0xA8,
0xE3, 0xB9, 0x2B, 0xDD, 0xF2, 0x13, 0xEA,
/*0690:*/ 0x33, 0x09, 0xCD, 0x6A, 0xF4, 0x97, 0x6F, 0xA7, 0xA1,
0x39, 0x50, 0xC7, 0x11, 0x48, 0xD1, 0x0C,
/*06A0:*/ 0x40, 0xF9, 0xF1, 0x79, 0x38, 0xAE, 0xF3, 0xB1, 0xFE,
0x55, 0x25, 0x38, 0xFD, 0xAA, 0x73, 0x23,
/*06B0:*/ 0x37, 0x29, 0xA5, 0x40, 0x9E, 0xC5, 0x30, 0xDA, 0xF5,
0xA6, 0x1F, 0x24, 0x1F, 0x10, 0xA9, 0x33,
/*06C0:*/ 0xC2, 0x82, 0x1C, 0x53, 0x96, 0xD9, 0x61, 0x70, 0xC2,
0xE3, 0xB9, 0x2B, 0xF7, 0xF2, 0x13, 0xEA,
/*06D0:*/ 0x3E, 0x09, 0xCD, 0x6A, 0x72, 0x96, 0x6F, 0xA7, 0xB1,
0x39, 0x50, 0xC7, 0xAE, 0x48, 0xD1, 0x0C,
/*06E0:*/ 0xD5, 0xF9, 0xF1, 0x79, 0x3D, 0xAE, 0xF3, 0xB1, 0xF8,
0x55, 0x25, 0x38, 0xD9, 0xAA, 0x73, 0x23,
/*06F0:*/ 0x30, 0x28, 0xA5, 0x40, 0xFF, 0xC5, 0x30, 0xDA, 0xD5,
0xA6, 0x1F, 0x24, 0x1F, 0x10, 0xA9, 0x33,
/*0700:*/ 0xAA, 0x82, 0x1C, 0x53, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*0710:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};

BYTE B7D7[] = {
/*0CD0:*/ 0x9B, 0x39, 0x50, 0xC7, 0xC6, 0x49, 0xD1, 0x0C, 0x70,
0xF9, 0xF1, 0x79, 0xF4, 0x50, 0x0C, 0x4E,
/*0CE0:*/ 0xDE, 0x55, 0x25, 0x38, 0x99, 0xAA, 0x73, 0x23, 0x13,
0x28, 0xA5, 0x40, 0xF6, 0xC5, 0x30, 0xDA,
/*0CF0:*/ 0xBC, 0xA6, 0x1F, 0x24, 0x01, 0x04, 0xA9, 0x33, 0xE8,
0x82, 0x1C, 0x53, 0x82, 0xD9, 0x61, 0x70,
/*0D00:*/ 0xE0, 0xE3, 0xB9, 0x2B, 0x87, 0xF4, 0x13, 0xEA, 0x24,
0x09, 0xCD, 0x6A, 0xE7, 0x97, 0x6F, 0xA7,
/*0D10:*/ 0xA0, 0x39, 0x50, 0xC7, 0x1F, 0x48, 0xD1, 0x0C, 0x48,
0xF9, 0xF1, 0x79, 0x3B, 0xAE, 0xF3, 0xB1,
/*0D20:*/ 0xD4, 0x55, 0x25, 0x38, 0xD9, 0xAA, 0x73, 0x23, 0x12,
0x28, 0xA5, 0x40, 0x79, 0x3A, 0xCF, 0x25,
/*0D30:*/ 0xCB, 0xA6, 0x1F, 0x24, 0x46, 0x10, 0xA9, 0x33, 0xCA,
0x82, 0x1C, 0x53, 0x92, 0xD9, 0x61, 0x70,
/*0D40:*/ 0xFD, 0xE3, 0xB9, 0x2B, 0x81, 0xF2, 0x13, 0xEA, 0x2C,
0x09, 0xCD, 0x6A, 0xCB, 0x97, 0x6F, 0xA7,
/*0D50:*/ 0xAA, 0x39, 0x50, 0xC7, 0x3D, 0x48, 0xD1, 0x0C, 0x5B,
0xF9, 0xF1, 0x79, 0x28, 0xAE, 0xF3, 0xB1,
/*0D60:*/ 0x3F, 0x55, 0x25, 0x38, 0xB2, 0xAA, 0x73, 0x23, 0xA4,
0x28, 0xA5, 0x40, 0xFD, 0xC5, 0x30, 0xDA,
/*0D70:*/ 0xE7, 0xA6, 0x1F, 0x24, 0xB3, 0x10, 0xA9, 0x33, 0xCC,
0x82, 0x1C, 0x53, 0x81, 0xD9, 0x61, 0x70,
/*0D80:*/ 0xFC, 0xE3, 0xB9, 0x2B, 0xAB, 0xF2, 0x13, 0xEA, 0x3F,
0x09, 0xCD, 0x6A, 0xD9, 0x97, 0x6F, 0xA7,
/*0D90:*/ 0x40, 0xC6, 0xAF, 0x38, 0x34, 0x48, 0xD1, 0x0C, 0x6E,
0xF9, 0xF1, 0x79, 0x26, 0xAC, 0xF3, 0xB1,
/*0DA0:*/ 0xF9, 0x55, 0x25, 0x38, 0xAC, 0x55, 0x8C, 0xDC, 0x42,
0x28, 0xA5, 0x40, 0xBA, 0xC5, 0x30, 0xDA,
/*0DB0:*/ 0xE5, 0xA6, 0x1F, 0x24, 0x60, 0x10, 0xA9, 0x33, 0xC5,
0x82, 0x1C, 0x53, 0x82, 0xD9, 0x61, 0x70,
/*0DC0:*/ 0xE7, 0xE3, 0xB9, 0x2B, 0x93, 0xF2, 0x13, 0xEA, 0x31,
0x09, 0xCD, 0x6A, 0xFE, 0x97, 0x6F, 0xA7,
/*0DD0:*/ 0xE4, 0x39, 0x50, 0xC7, 0xDF, 0x48, 0xD1, 0x0C, 0x62,
0xF9, 0xF1, 0x79, 0x0B, 0xAC, 0xF3, 0xB1,
/*0DE0:*/ 0x5E, 0x55, 0x25, 0x38, 0x85, 0xAA, 0x73, 0x23, 0x5F,
0x29, 0xA5, 0x40, 0x57, 0x3B, 0xCF, 0x25,
/*0DF0:*/ 0xC0, 0xA6, 0x1F, 0x24, 0x17, 0x10, 0xA9, 0x33, 0xCB,
0x82, 0x1C, 0x53, 0x69, 0xD9, 0x61, 0x70,
/*0E00:*/ 0x39, 0xE3, 0xB9, 0x2B, 0xBD, 0xF3, 0x13, 0xEA, 0xD4,
0x09, 0xCD, 0x6A, 0xF9, 0x95, 0x6F, 0xA7,
/*0E10:*/ 0x8F, 0x39, 0x50, 0xC7, 0x0C, 0x48, 0xD1, 0x0C, 0x39,
0xF9, 0xF1, 0x79, 0x01, 0xAC, 0xF3, 0xB1,
/*0E20:*/ 0xC5, 0x55, 0x25, 0x38, 0x75, 0x55, 0x8C, 0xDC, 0x13,
0x28, 0xA5, 0x40, 0x10, 0x3A, 0xCF, 0x25,
/*0E30:*/ 0x5E, 0xA6, 0x1F, 0x24, 0x41, 0x11, 0xA9, 0x33, 0xB8,
0x82, 0x1C, 0x53, 0x5A, 0x24, 0x9E, 0x8F,
/*0E40:*/ 0xF4, 0xE2, 0xB9, 0x2B, 0x8A, 0xF2, 0x13, 0xEA, 0x57,
0x08, 0xCD, 0x6A, 0xC7, 0x97, 0x6F, 0xA7,
/*0E50:*/ 0xCF, 0x39, 0x50, 0xC7, 0x2F, 0x48, 0xD1, 0x0C, 0x23,
0xF8, 0xF1, 0x79, 0x7A, 0xAE, 0xF3, 0xB1,
/*0E60:*/ 0xF8, 0x55, 0x25, 0x38, 0x05, 0x55, 0x8C, 0xDC, 0x05,
0x28, 0xA5, 0x40, 0xAA, 0xC5, 0x30, 0xDA,
/*0E70:*/ 0xEC, 0xA6, 0x1F, 0x24, 0x0A, 0x10, 0xA9, 0x33, 0xC0,
0x82, 0x1C, 0x53, 0xA4, 0xD9, 0x61, 0x70,
/*0E80:*/ 0xFD, 0xE3, 0xB9, 0x2B, 0x81, 0xF2, 0x13, 0xEA, 0x54,
0x09, 0xCD, 0x6A, 0xDE, 0x97, 0x6F, 0xA7,
/*0E90:*/ 0xA1, 0x39, 0x50, 0xC7, 0x38, 0x48, 0xD1, 0x0C, 0x5A,
0xF9, 0xF1, 0x79, 0x29, 0xAE, 0xF3, 0xB1,
/*0EA0:*/ 0xF5, 0x55, 0x25, 0x38, 0xD5, 0xAA, 0x73, 0x23, 0x1B,
0x28, 0xA5, 0x40, 0x37, 0xC5, 0x30, 0xDA,
/*0EB0:*/ 0xD8, 0xAB, 0x1F, 0x24, 0x94, 0xEE, 0x56, 0xCC, 0x07,
0x82, 0x1C, 0x53, 0x8F, 0xD9, 0x61, 0x70,
/*0EC0:*/ 0xD6, 0xE3, 0xB9, 0x2B, 0x8D, 0xF2, 0x13, 0xEA, 0x22,
0x09, 0xCD, 0x6A, 0xD9, 0x97, 0x6F, 0xA7,
/*0ED0:*/ 0xB5, 0x39, 0x50, 0xC7, 0x28, 0x48, 0xD1, 0x0C, 0x57,
0xF9, 0xF1, 0x79, 0x1B, 0xAE, 0xF3, 0xB1,
/*0EE0:*/ 0xC5, 0x55, 0x25, 0x38, 0x93, 0xAA, 0x73, 0x23, 0xDD,
0x28, 0xA5, 0x40, 0x89, 0xC5, 0x30, 0xDA,
/*0EF0:*/ 0xE5, 0xA6, 0x1F, 0x24, 0x2B, 0x10, 0xA9, 0x33, 0xC3,
0x82, 0x1C, 0x53, 0x8A, 0xD9, 0x61, 0x70,
/*0F00:*/ 0x90, 0xE0, 0xB9, 0x2B, 0x8A, 0xF2, 0x13, 0xEA, 0x3E,
0x09, 0xCD, 0x6A, 0x0C, 0x97, 0x6F, 0xA7,
/*0F10:*/ 0xAF, 0x39, 0x50, 0xC7, 0x2F, 0x48, 0xD1, 0x0C, 0xCD,
0x06, 0x0E, 0x86, 0x14, 0xAE, 0xF3, 0xB1,
/*0F20:*/ 0xD0, 0x5E, 0x25, 0x38, 0xC6, 0xAA, 0x73, 0x23, 0xB0,
0xD7, 0x5A, 0xBF, 0x68, 0xC4, 0x30, 0xDA,
/*0F30:*/ 0xE7, 0xA6, 0x1F, 0x24, 0x03, 0x10, 0xA9, 0x33, 0xEF,
0x82, 0x1C, 0x53, 0x3D, 0xD8, 0x61, 0x70,
/*0F40:*/ 0x03, 0xE2, 0xB9, 0x2B, 0xD0, 0xF2, 0x13, 0xEA, 0x0A,
0x09, 0xCD, 0x6A, 0xD9, 0x97, 0x6F, 0xA7,
/*0F50:*/ 0xCE, 0x39, 0x50, 0xC7, 0xFB, 0x48, 0xD1, 0x0C, 0x49,
0xF9, 0xF1, 0x79, 0x3A, 0xAE, 0xF3, 0xB1,
/*0F60:*/ 0xF4, 0x55, 0x25, 0x38, 0xB4, 0xAA, 0x73, 0x23, 0x61,
0x2A, 0xA5, 0x40, 0xFD, 0xC5, 0x30, 0xDA,
/*0F70:*/ 0xBD, 0xA7, 0x1F, 0x24, 0x14, 0x10, 0xA9, 0x33, 0xAB,
0x82, 0x1C, 0x53, 0xC9, 0xD9, 0x61, 0x70,
/*0F80:*/ 0x8E, 0xE3, 0xB9, 0x2B, 0x0D, 0xF0, 0xAD, 0xBA, 0x0D,
0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*0F90:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};

BYTE B7D8[] = {
/*1478:*/ 0x0E, 0x39, 0x50, 0xC7, 0x17, 0x48, 0xD1, 0x0C,
/*1480:*/ 0x56, 0xF9, 0xF1, 0x79, 0x54, 0xAE, 0xF3, 0xB1, 0xF9,
0x55, 0x25, 0x38, 0x67, 0xAA, 0x73, 0x23,
/*1490:*/ 0x0C, 0x28, 0xA5, 0x40, 0xBD, 0xC5, 0x30, 0xDA, 0x72,
0xA7, 0x1F, 0x24, 0x28, 0x11, 0xA9, 0x33,
/*14A0:*/ 0x53, 0x82, 0x1C, 0x53, 0x8B, 0xD9, 0x61, 0x70, 0xFC,
0xE3, 0xB9, 0x2B, 0x5F, 0xF0, 0x13, 0xEA,
/*14B0:*/ 0x3C, 0x09, 0xCD, 0x6A, 0xD8, 0x97, 0x6F, 0xA7, 0xAD,
0x39, 0x50, 0xC7, 0xE6, 0xB7, 0x2E, 0xF3,
/*14C0:*/ 0xF4, 0xF9, 0xF1, 0x79, 0xA5, 0xAF, 0xF3, 0xB1, 0xE8,
0x55, 0x25, 0x38, 0x8B, 0xA0, 0x73, 0x23,
/*14D0:*/ 0x0B, 0x28, 0xA5, 0x40, 0xBB, 0xC5, 0x30, 0xDA, 0xF1,
0xA6, 0x1F, 0x24, 0x01, 0x10, 0xA9, 0x33,
/*14E0:*/ 0x79, 0x83, 0x1C, 0x53, 0x5A, 0x26, 0x9E, 0x8F, 0xD7,
0xE3, 0xB9, 0x2B, 0x8D, 0xF2, 0x13, 0xEA,
/*14F0:*/ 0x3F, 0x09, 0xCD, 0x6A, 0xCF, 0x97, 0x6F, 0xA7, 0x69,
0x38, 0x50, 0xC7, 0xF1, 0xB6, 0x2E, 0xF3,
/*1500:*/ 0x53, 0xF9, 0xF1, 0x79, 0x31, 0xAE, 0xF3, 0xB1, 0x30,
0x55, 0x25, 0x38, 0xD9, 0xAA, 0x73, 0x23,
/*1510:*/ 0xF6, 0x28, 0xA5, 0x40, 0xB0, 0xC5, 0x30, 0xDA, 0x56,
0xA6, 0x1F, 0x24, 0x12, 0x10, 0xA9, 0x33,
/*1520:*/ 0xFA, 0x82, 0x1C, 0x53, 0xC0, 0xD9, 0x61, 0x70, 0xEC,
0xE3, 0xB9, 0x2B, 0xA8, 0xF3, 0x13, 0xEA,
/*1530:*/ 0x33, 0x09, 0xCD, 0x6A, 0x8F, 0x97, 0x6F, 0xA7, 0xAD,
0x39, 0x50, 0xC7, 0x7B, 0xB6, 0x2E, 0xF3,
/*1540:*/ 0x5D, 0xF9, 0xF1, 0x79, 0x36, 0xAE, 0xF3, 0xB1, 0xF9,
0x55, 0x25, 0x38, 0x9D, 0xAA, 0x73, 0x23,
/*1550:*/ 0xAD, 0xD5, 0x5A, 0xBF, 0xC8, 0xC5, 0x30, 0xDA, 0xAF,
0xA6, 0x1F, 0x24, 0x61, 0x10, 0xA9, 0x33,
/*1560:*/ 0xCB, 0x82, 0x1C, 0x53, 0x85, 0xD9, 0x61, 0x70, 0xEB,
0xE3, 0xB9, 0x2B, 0xAA, 0xF2, 0x13, 0xEA,
/*1570:*/ 0x31, 0x09, 0xCD, 0x6A, 0xD2, 0x97, 0x6F, 0xA7, 0x18,
0x39, 0x50, 0xC7, 0x65, 0x49, 0xD1, 0x0C,
/*1580:*/ 0x5B, 0xF9, 0xF1, 0x79, 0x33, 0xAE, 0xF3, 0xB1, 0x86,
0x55, 0x25, 0x38, 0x8E, 0xAA, 0x73, 0x23,
/*1590:*/ 0x0B, 0x28, 0xA5, 0x40, 0x91, 0xC5, 0x30, 0xDA, 0xE0,
0xA6, 0x1F, 0x24, 0x04, 0x10, 0xA9, 0x33,
/*15A0:*/ 0x21, 0x82, 0x1C, 0x53, 0x9B, 0xD9, 0x61, 0x70, 0x52,
0xE2, 0xB9, 0x2B, 0xFB, 0xF3, 0x13, 0xEA,
/*15B0:*/ 0x3F, 0x09, 0xCD, 0x6A, 0xD2, 0x97, 0x6F, 0xA7, 0xA1,
0x39, 0x50, 0xC7, 0x25, 0x48, 0xD1, 0x0C,
/*15C0:*/ 0x66, 0xF9, 0xF1, 0x79, 0x3A, 0xAE, 0xF3, 0xB1, 0x0F,
0xAA, 0xDA, 0xC7, 0xFA, 0xAA, 0x73, 0x23,
/*15D0:*/ 0x1E, 0x28, 0xA5, 0x40, 0x61, 0x3A, 0xCF, 0x25, 0xC9,
0xA5, 0x1F, 0x24, 0x02, 0x10, 0xA9, 0x33,
/*15E0:*/ 0xCB, 0x82, 0x1C, 0x53, 0xD7, 0xD9, 0x61, 0x70, 0xF2,
0xE3, 0xB9, 0x2B, 0x7A, 0xF3, 0x13, 0xEA,
/*15F0:*/ 0x0C, 0x0B, 0xCD, 0x6A, 0x28, 0x68, 0x90, 0x58, 0xCC,
0x39, 0x50, 0xC7, 0x2E, 0x48, 0xD1, 0x0C,
/*1600:*/ 0xAF, 0xF9, 0xF1, 0x79, 0x28, 0xAE, 0xF3, 0xB1, 0xD4,
0x55, 0x25, 0x38, 0xE9, 0x54, 0x8C, 0xDC,
/*1610:*/ 0x05, 0x28, 0xA5, 0x40, 0xA3, 0xC5, 0x30, 0xDA, 0x5A,
0xA7, 0x1F, 0x24, 0x15, 0x10, 0xA9, 0x33,
/*1620:*/ 0x61, 0x82, 0x1C, 0x53, 0x83, 0xD9, 0x61, 0x70, 0xEC,
0xE3, 0xB9, 0x2B, 0x3D, 0x0D, 0xEC, 0x15,
/*1630:*/ 0x3E, 0x09, 0xCD, 0x6A, 0xD9, 0x97, 0x6F, 0xA7, 0xB1,
0x39, 0x50, 0xC7, 0x08, 0x41, 0xD1, 0x0C,
/*1640:*/ 0x4E, 0xF9, 0xF1, 0x79, 0x2B, 0xAE, 0xF3, 0xB1, 0xE3,
0x55, 0x25, 0x38, 0xC2, 0xAA, 0x73, 0x23,
/*1650:*/ 0x0B, 0x28, 0xA5, 0x40, 0xB9, 0xC5, 0x30, 0xDA, 0x00,
0x59, 0xE0, 0xDB, 0x03, 0x10, 0xA9, 0x33,
/*1660:*/ 0xFD, 0x82, 0x1C, 0x53, 0x82, 0xD9, 0x61, 0x70, 0xD5,
0xE3, 0xB9, 0x2B, 0x48, 0xF3, 0x13, 0xEA,
/*1670:*/ 0x2D, 0x09, 0xCD, 0x6A, 0xD1, 0x97, 0x6F, 0xA7, 0xAB,
0x39, 0x50, 0xC7, 0x2E, 0x48, 0xD1, 0x0C,
/*1680:*/ 0x55, 0xF9, 0xF1, 0x79, 0xA8, 0xAE, 0xF3, 0xB1, 0xF7,
0x55, 0x25, 0x38, 0xC9, 0xAB, 0x73, 0x23,
/*1690:*/ 0x15, 0x28, 0xA5, 0x40, 0xA7, 0xC5, 0x30, 0xDA, 0xF8,
0xA6, 0x1F, 0x24, 0x11, 0x10, 0xA9, 0x33,
/*16A0:*/ 0xD7, 0x82, 0x1C, 0x53, 0x80, 0xD9, 0x61, 0x70, 0xF4,
0xE3, 0xB9, 0x2B, 0xD3, 0xF2, 0x13, 0xEA,
/*16B0:*/ 0xA1, 0x09, 0xCD, 0x6A, 0xD4, 0x97, 0x6F, 0xA7, 0xC2,
0x38, 0x50, 0xC7, 0x2E, 0x49, 0xD1, 0x0C,
/*16C0:*/ 0x52, 0xF9, 0xF1, 0x79, 0x0B, 0xAE, 0xF3, 0xB1, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*16D0:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};
```

```
BYTE B7D9[] = {
/*1E18:*/ 0xF0, 0xC6, 0xAF, 0x38, 0xB1, 0x49, 0xD1, 0x0C,
/*1E20:*/ 0x47, 0xF9, 0xF1, 0x79, 0x39, 0xAE, 0xF3, 0xB1, 0xE5,
0x55, 0x25, 0x38, 0xC5, 0xAA, 0x73, 0x23,
/*1E30:*/ 0x66, 0x28, 0xA5, 0x40, 0xED, 0xC5, 0x30, 0xDA, 0x14,
0xA7, 0x1F, 0x24, 0x02, 0x10, 0xA9, 0x33,
/*1E40:*/ 0x95, 0x82, 0x1C, 0x53, 0x82, 0xD9, 0x61, 0x70, 0xFB,
0xE3, 0xB9, 0x2B, 0xF1, 0xF3, 0x13, 0xEA,
/*1E50:*/ 0x2B, 0x09, 0xCD, 0x6A, 0xE8, 0x97, 0x6F, 0xA7, 0xA7,
0x39, 0x50, 0xC7, 0x04, 0x48, 0xD1, 0x0C,
/*1E60:*/ 0x46, 0xF9, 0xF1, 0x79, 0x2B, 0xAE, 0xF3, 0xB1, 0x53,
0x54, 0x25, 0x38, 0x9B, 0x54, 0x8C, 0xDC,
/*1E70:*/ 0x19, 0x29, 0xA5, 0x40, 0x62, 0xC5, 0x30, 0xDA, 0x8C,
0xA6, 0x1F, 0x24, 0x10, 0x10, 0xA9, 0x33,
/*1E80:*/ 0x58, 0x82, 0x1C, 0x53, 0x12, 0xD9, 0x61, 0x70, 0xCB,
0xE3, 0xB9, 0x2B, 0x12, 0xF2, 0x13, 0xEA,
/*1E90:*/ 0x6C, 0x09, 0xCD, 0x6A, 0xC3, 0x97, 0x6F, 0xA7, 0x5C,
0x38, 0x50, 0xC7, 0x9B, 0x48, 0xD1, 0x0C,
/*1EA0:*/ 0xE8, 0x06, 0x0E, 0x86, 0xAC, 0xAF, 0xF3, 0xB1, 0xB8,
0x55, 0x25, 0x38, 0xD8, 0xAA, 0x73, 0x23,
/*1EB0:*/ 0x37, 0x28, 0xA5, 0x40, 0xBA, 0xC5, 0x30, 0xDA, 0xAB,
0xA5, 0x1F, 0x24, 0x03, 0x10, 0xA9, 0x33,
/*1EC0:*/ 0xDC, 0x82, 0x1C, 0x53, 0x8A, 0xD9, 0x61, 0x70, 0xB3,
0xE3, 0xB9, 0x2B, 0x7F, 0x0D, 0xEC, 0x15,
/*1ED0:*/ 0xCA, 0x09, 0xCD, 0x6A, 0xC1, 0x97, 0x6F, 0xA7, 0x74,
0x38, 0x50, 0xC7, 0x34, 0x48, 0xD1, 0x0C,
/*1EE0:*/ 0x8D, 0xF9, 0xF1, 0x79, 0x09, 0xAE, 0xF3, 0xB1, 0x2A,
0x55, 0x25, 0x38, 0xD8, 0xAA, 0x73, 0x23,
/*1EF0:*/ 0x05, 0x28, 0xA5, 0x40, 0xBA, 0xC5, 0x30, 0xDA, 0xEA,
0xA6, 0x1F, 0x24, 0x1F, 0x10, 0xA9, 0x33,
/*1F00:*/ 0xCA, 0x82, 0x1C, 0x53, 0x89, 0xD9, 0x61, 0x70, 0xDE,
0xE1, 0xB9, 0x2B, 0x6F, 0xF2, 0x13, 0xEA,
/*1F10:*/ 0x2E, 0x09, 0xCD, 0x6A, 0xCF, 0x97, 0x6F, 0xA7, 0x80,
0x39, 0x50, 0xC7, 0x3F, 0x48, 0xD1, 0x0C,
/*1F20:*/ 0x5A, 0xF9, 0xF1, 0x79, 0x2E, 0xAE, 0xF3, 0xB1, 0xEB,
0x55, 0x25, 0x38, 0x76, 0xAB, 0x73, 0x23,
/*1F30:*/ 0x2F, 0xD6, 0x5A, 0xBF, 0xBA, 0xC5, 0x30, 0xDA, 0xE7,
0xA6, 0x1F, 0x24, 0x19, 0x10, 0xA9, 0x33,
/*1F40:*/ 0xC3, 0x82, 0x1C, 0x53, 0x83, 0xD9, 0x61, 0x70, 0xFD,
0xE3, 0xB9, 0x2B, 0xA0, 0xF2, 0x13, 0xEA,
/*1F50:*/ 0x33, 0x09, 0xCD, 0x6A, 0xD8, 0x97, 0x6F, 0xA7, 0xAB,
0x39, 0x50, 0xC7, 0xF0, 0x48, 0xD1, 0x0C,
/*1F60:*/ 0xAD, 0x06, 0x0E, 0x86, 0x3B, 0xAE, 0xF3, 0xB1, 0xDE,
0x55, 0x25, 0x38, 0xD8, 0xAA, 0x73, 0x23,
/*1F70:*/ 0x05, 0x28, 0xA5, 0x40, 0x56, 0xC5, 0x30, 0xDA, 0xA3,
0xA6, 0x1F, 0x24, 0x03, 0x10, 0xA9, 0x33,
/*1F80:*/ 0xC8, 0x82, 0x1C, 0x53, 0xE3, 0xD9, 0x61, 0x70, 0x56,
0xE3, 0xB9, 0x2B, 0x81, 0xF2, 0x13, 0xEA,
/*1F90:*/ 0x38, 0x09, 0xCD, 0x6A, 0xD1, 0x97, 0x6F, 0xA7, 0xAB,
0x39, 0x50, 0xC7, 0x2F, 0x48, 0xD1, 0x0C,
/*1FA0:*/ 0x57, 0xF8, 0xF1, 0x79, 0x3A, 0xAE, 0xF3, 0xB1, 0x92,
0x54, 0x25, 0x38, 0xEB, 0xAB, 0x73, 0x23,
/*1FB0:*/ 0xD7, 0x28, 0xA5, 0x40, 0x94, 0xC4, 0x30, 0xDA, 0xA5,
0x59, 0xE0, 0xDB, 0x5C, 0x10, 0xA9, 0x33,
/*1FC0:*/ 0xCB, 0x82, 0x1C, 0x53, 0x26, 0xD8, 0x61, 0x70, 0xB5,
0xE3, 0xB9, 0x2B, 0x8A, 0xF2, 0x13, 0xEA,
/*1FD0:*/ 0x3F, 0x09, 0xCD, 0x6A, 0xD9, 0x97, 0x6F, 0xA7, 0x40,
0xC6, 0xAF, 0x38, 0x85, 0xB7, 0x2E, 0xF3,
/*1FE0:*/ 0x5A, 0xF9, 0xF1, 0x79, 0x36, 0xAE, 0xF3, 0xB1, 0xBD,
0x55, 0x25, 0x38, 0xDB, 0xAA, 0x73, 0x23,
/*1FF0:*/ 0x2E, 0x29, 0xA5, 0x40, 0xC5, 0xC4, 0x30, 0xDA, 0xB9,
0xA7, 0x1F, 0x24, 0x13, 0x10, 0xA9, 0x33,
/*2000:*/ 0x4F, 0x83, 0x1C, 0x53, 0x82, 0xD9, 0x61, 0x70, 0xFD,
0xE3, 0xB9, 0x2B, 0x8B, 0xF2, 0x13, 0xEA,
/*2010:*/ 0xE9, 0xF6, 0x32, 0x95, 0x89, 0x97, 0x6F, 0xA7, 0x44,
0xC6, 0xAF, 0x38, 0x2D, 0x48, 0xD1, 0x0C,
/*2020:*/ 0x1C, 0xF9, 0xF1, 0x79, 0xD9, 0x50, 0x0C, 0x4E, 0xF8,
0x55, 0x25, 0x38, 0x6F, 0x53, 0x8C, 0xDC,
/*2030:*/ 0xEF, 0x28, 0xA5, 0x40, 0xBD, 0xC5, 0x30, 0xDA, 0xE6,
0xA6, 0x1F, 0x24, 0x27, 0xEF, 0x56, 0xCC,
/*2040:*/ 0xE3, 0x82, 0x1C, 0x53, 0xCE, 0xD9, 0x61, 0x70, 0x90,
0xE3, 0xB9, 0x2B, 0x8B, 0xF2, 0x13, 0xEA,
/*2050:*/ 0x11, 0x09, 0xCD, 0x6A, 0xF7, 0x97, 0x6F, 0xA7, 0xA2,
0x39, 0x50, 0xC7, 0x31, 0x49, 0xD1, 0x0C,
/*2060:*/ 0x45, 0xF9, 0xF1, 0x79, 0x38, 0xAE, 0xF3, 0xB1, 0xF4,
0x55, 0x25, 0x38, 0xFA, 0x55, 0x8C, 0xDC,
/*2070:*/ 0x62, 0x28, 0xA5, 0x40, 0x95, 0xC5, 0x30, 0xDA, 0xC1,
0xA6, 0x1F, 0x24, 0x2D, 0x11, 0xA9, 0x33,
/*2080:*/ 0x10, 0x83, 0x1C, 0x53, 0x92, 0xD9, 0x61, 0x70, 0xC6,
0x1C, 0x46, 0xD4, 0x8A, 0xF2, 0x13, 0xEA,
/*2090:*/ 0x2E, 0x09, 0xCD, 0x6A, 0xC7, 0x97, 0x6F, 0xA7, 0xDD,
0xC6, 0xAF, 0x38, 0x21, 0x48, 0xD1, 0x0C,
/*20A0:*/ 0x4E, 0xF9, 0xF1, 0x79, 0x33, 0xAE, 0xF3, 0xB1, 0x05,
0xA8, 0xDA, 0xC7, 0xD0, 0xAA, 0x73, 0x23,
/*20B0:*/ 0x0F, 0x28, 0xA5, 0x40, 0xB6, 0xC5, 0x30, 0xDA, 0xEC,
0xA6, 0x1F, 0x24, 0x8A, 0x10, 0xA9, 0x33,
/*20C0:*/ 0xDD, 0x82, 0x1C, 0x53, 0x95, 0xD9, 0x61, 0x70, 0x0D,
0xE3, 0xB9, 0x2B, 0x8A, 0xF2, 0x13, 0xEA,
/*20D0:*/ 0x32, 0x09, 0xCD, 0x6A, 0xC4, 0x97, 0x6F, 0xA7, 0xB3,
0x39, 0x50, 0xC7, 0x22, 0x48, 0xD1, 0x0C,
/*20E0:*/ 0x5A, 0xF9, 0xF1, 0x79, 0x63, 0xAC, 0xF3, 0xB1, 0xFE,
0x55, 0x25, 0x38, 0xD9, 0xAA, 0x73, 0x23,
/*20F0:*/ 0x0A, 0x28, 0xA5, 0x40, 0x81, 0xC5, 0x30, 0xDA, 0x6A,
0xA6, 0x1F, 0x24, 0x16, 0x10, 0xA9, 0x33,
/*2100:*/ 0xC4, 0x82, 0x1C, 0x53, 0x83, 0xD9, 0x61, 0x70, 0xEC,
0xE3, 0xB9, 0x2B, 0x85, 0xF2, 0x13, 0xEA,
/*2110:*/ 0x3F, 0x09, 0xCD, 0x6A, 0xDB, 0x97, 0x6F, 0xA7, 0xA7,
0x39, 0x50, 0xC7, 0x3F, 0x48, 0xD1, 0x0C,
/*2120:*/ 0x0D, 0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*2130:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};
```

```
BYTE B7DA[] = {
/*2828:*/ 0xB4, 0x39, 0x50, 0xC7, 0xDE, 0x49, 0xD1, 0x0C,
/*2830:*/ 0x5A, 0xF9, 0xF1, 0x79, 0x36, 0xAE, 0xF3, 0xB1, 0xEB,
0x55, 0x25, 0x38, 0xFF, 0xAA, 0x73, 0x23,
/*2840:*/ 0x15, 0x28, 0xA5, 0x40, 0x11, 0xC5, 0x30, 0xDA, 0xE4,
0xA6, 0x1F, 0x24, 0x01, 0x10, 0xA9, 0x33,
/*2850:*/ 0xC4, 0x82, 0x1C, 0x53, 0x9A, 0xD9, 0x61, 0x70, 0xF4,
0xE1, 0xB9, 0x2B, 0x8B, 0xF2, 0x13, 0xEA,
/*2860:*/ 0x0E, 0x09, 0xCD, 0x6A, 0x8E, 0x97, 0x6F, 0xA7, 0xA0,
0x39, 0x50, 0xC7, 0x2F, 0x48, 0xD1, 0x0C,
/*2870:*/ 0x71, 0xF9, 0xF1, 0x79, 0x33, 0xAE, 0xF3, 0xB1, 0xF9,
0x55, 0x25, 0x38, 0x98, 0xAA, 0x73, 0x23,
/*2880:*/ 0x04, 0x28, 0xA5, 0x40, 0x5E, 0xC5, 0x30, 0xDA, 0xE7,
0xA6, 0x1F, 0x24, 0x0D, 0x10, 0xA9, 0x33,
/*2890:*/ 0xEA, 0x81, 0x1C, 0x53, 0xBD, 0xD9, 0x61, 0x70, 0x3F,
0x1D, 0x46, 0xD4, 0xB1, 0xEB, 0x13, 0xEA,
/*28A0:*/ 0x2E, 0x09, 0xCD, 0x6A, 0xC5, 0x95, 0x6F, 0xA7, 0xA0,
0x39, 0x50, 0xC7, 0x23, 0x48, 0xD1, 0x0C,
/*28B0:*/ 0x43, 0xF9, 0xF1, 0x79, 0x35, 0xAE, 0xF3, 0xB1, 0xEA,
0x55, 0x25, 0x38, 0xE3, 0xAA, 0x73, 0x23,
/*28C0:*/ 0x8B, 0xD7, 0x5A, 0xBF, 0xB2, 0xC5, 0x30, 0xDA, 0xFF,
0xA6, 0x1F, 0x24, 0x31, 0x10, 0xA9, 0x33,
/*28D0:*/ 0xDF, 0x82, 0x1C, 0x53, 0x8A, 0xD9, 0x61, 0x70, 0xF3,
0xE3, 0xB9, 0x2B, 0x84, 0xF2, 0x13, 0xEA,
/*28E0:*/ 0x1F, 0x09, 0xCD, 0x6A, 0xCB, 0x97, 0x6F, 0xA7, 0xB8,
0x39, 0x50, 0xC7, 0x36, 0x48, 0xD1, 0x0C,
/*28F0:*/ 0x4B, 0xF9, 0xF1, 0x79, 0x3A, 0xAE, 0xF3, 0xB1, 0xF8,
0x55, 0x25, 0x38, 0xC8, 0xAA, 0x73, 0x23,
/*2900:*/ 0x25, 0x28, 0xA5, 0x40, 0x70, 0xC5, 0x30, 0xDA, 0xE7,
0xA6, 0x1F, 0x24, 0x1B, 0x10, 0xA9, 0x33,
/*2910:*/ 0xF8, 0x82, 0x1C, 0x53, 0xA6, 0xD9, 0x61, 0x70, 0x89,
0xE3, 0xB9, 0x2B, 0x17, 0x0C, 0xEC, 0x15,
/*2920:*/ 0x04, 0x08, 0xCD, 0x6A, 0x9F, 0x97, 0x6F, 0xA7, 0x8A,
0x39, 0x50, 0xC7, 0x68, 0x48, 0xD1, 0x0C,
/*2930:*/ 0x5B, 0xF9, 0xF1, 0x79, 0xCB, 0x51, 0x0C, 0x4E, 0xF8,
0x55, 0x25, 0x38, 0xC7, 0xAA, 0x73, 0x23,
/*2940:*/ 0x6C, 0x28, 0xA5, 0x40, 0xAF, 0xC5, 0x30, 0xDA, 0x49,
0xA6, 0x1F, 0x24, 0x00, 0x10, 0xA9, 0x33,
/*2950:*/ 0x8B, 0x83, 0x1C, 0x53, 0x65, 0x26, 0x9E, 0x8F, 0xE5,
0xE3, 0xB9, 0x2B, 0x93, 0xF2, 0x13, 0xEA,
/*2960:*/ 0x9D, 0xF6, 0x32, 0x95, 0xE2, 0x97, 0x6F, 0xA7, 0xAF,
0x39, 0x50, 0xC7, 0x30, 0x48, 0xD1, 0x0C,
/*2970:*/ 0x5A, 0xF9, 0xF1, 0x79, 0x32, 0xAE, 0xF3, 0xB1, 0xFA,
0x55, 0x25, 0x38, 0x60, 0xAA, 0x73, 0x23,
/*2980:*/ 0x45, 0x28, 0xA5, 0x40, 0xBB, 0xC5, 0x30, 0xDA, 0xD5,
0xA6, 0x1F, 0x24, 0x33, 0x10, 0xA9, 0x33,
/*2990:*/ 0xEC, 0x82, 0x1C, 0x53, 0x82, 0xD9, 0x61, 0x70, 0xED,
0xE3, 0xB9, 0x2B, 0xC0, 0xF2, 0x13, 0xEA,
/*29A0:*/ 0x32, 0x09, 0xCD, 0x6A, 0xCB, 0x97, 0x6F, 0xA7, 0xB6,
0x39, 0x50, 0xC7, 0x39, 0x48, 0xD1, 0x0C,
/*29B0:*/ 0x46, 0xF9, 0xF1, 0x79, 0x5B, 0xAE, 0xF3, 0xB1, 0xF9,
0x55, 0x25, 0x38, 0xB9, 0xAA, 0x73, 0x23,
/*29C0:*/ 0x0A, 0x28, 0xA5, 0x40, 0xB7, 0xC5, 0x30, 0xDA, 0xCE,
0xA6, 0x1F, 0x24, 0x0D, 0x10, 0xA9, 0x33,
/*29D0:*/ 0xDD, 0x82, 0x1C, 0x53, 0x8F, 0xD9, 0x61, 0x70, 0x97,
0xE3, 0xB9, 0x2B, 0x8D, 0xF2, 0x13, 0xEA,
/*29E0:*/ 0x38, 0x09, 0xCD, 0x6A, 0xD8, 0x97, 0x6F, 0xA7, 0xA0,
0x39, 0x50, 0xC7, 0x2C, 0x48, 0xD1, 0x0C,
/*29F0:*/ 0x4E, 0xF9, 0xF1, 0x79, 0x38, 0xAE, 0xF3, 0xB1, 0xE2,
0x55, 0x25, 0x38, 0xD8, 0xAA, 0x73, 0x23,
/*2A00:*/ 0x16, 0x28, 0xA5, 0x40, 0xB1, 0xC5, 0x30, 0xDA, 0xE7,
0xA6, 0x1F, 0x24, 0x39, 0x10, 0xA9, 0x33,
/*2A10:*/ 0xFB, 0x83, 0x1C, 0x53, 0x2C, 0xD1, 0x61, 0x70, 0xFC,
0xE3, 0xB9, 0x2B, 0xE4, 0xF2, 0x13, 0xEA,
/*2A20:*/ 0x68, 0x09, 0xCD, 0x6A, 0x3F, 0x68, 0x90, 0x58, 0xBC,
0x38, 0x50, 0xC7, 0x25, 0x48, 0xD1, 0x0C,
/*2A30:*/ 0x28, 0xF9, 0xF1, 0x79, 0x1D, 0xAE, 0xF3, 0xB1, 0xCC,
0x55, 0x25, 0x38, 0x35, 0x55, 0x8C, 0xDC,
/*2A40:*/ 0x10, 0x28, 0xA5, 0x40, 0xB5, 0xC5, 0x30, 0xDA, 0x8B,
0xA6, 0x1F, 0x24, 0x0C, 0x10, 0xA9, 0x33,
/*2A50:*/ 0x1B, 0x82, 0x1C, 0x53, 0x83, 0xD9, 0x61, 0x70, 0x8A,
0xE3, 0xB9, 0x2B, 0x8B, 0xF2, 0x13, 0xEA,
/*2A60:*/ 0x1A, 0x0B, 0xCD, 0x6A, 0xEB, 0x81, 0x6F, 0xA7, 0xBA,
0x39, 0x50, 0xC7, 0x77, 0x48, 0xD1, 0x0C,
/*2A70:*/ 0x70, 0xF9, 0xF1, 0x79, 0x27, 0xAE, 0xF3, 0xB1, 0xBF,
0x54, 0x25, 0x38, 0xC3, 0xAA, 0x73, 0x23,
/*2A80:*/ 0x0A, 0x28, 0xA5, 0x40, 0xB7, 0xC5, 0x30, 0xDA, 0xE6,
0xA6, 0x1F, 0x24, 0x11, 0x10, 0xA9, 0x33,
/*2A90:*/ 0xCA, 0x82, 0x1C, 0x53, 0x95, 0xD9, 0x61, 0x70, 0xFF,
0xE3, 0xB9, 0x2B, 0x8B, 0xF2, 0x13, 0xEA,
/*2AA0:*/ 0x23, 0x09, 0xCD, 0x6A, 0xB9, 0x97, 0x6F, 0xA7, 0xAE,
0x39, 0x50, 0xC7, 0x70, 0x45, 0xD1, 0x0C,
/*2AB0:*/ 0x5A, 0xF9, 0xF1, 0x79, 0x35, 0xAE, 0xF3, 0xB1, 0x98,
0x55, 0x25, 0x38, 0x1A, 0x55, 0x8C, 0xDC,
/*2AC0:*/ 0x25, 0x28, 0xA5, 0x40, 0x83, 0xC5, 0x30, 0xDA, 0xBD,
0xA6, 0x1F, 0x24, 0x15, 0x10, 0xA9, 0x33,
/*2AD0:*/ 0xC7, 0x82, 0x1C, 0x53, 0xE7, 0xD9, 0x61, 0x70, 0xFD,
0xE3, 0xB9, 0x2B, 0xE3, 0xF0, 0x13, 0xEA,
/*2AE0:*/ 0x00, 0x0B, 0xCD, 0x6A, 0x0D, 0x96, 0x6F, 0xA7, 0xB4,
0x39, 0x50, 0xC7, 0x3B, 0x48, 0xD1, 0x0C,
/*2AF0:*/ 0x19, 0xF9, 0xF1, 0x79, 0x35, 0xAE, 0xF3, 0xB1, 0xE4,
0x55, 0x25, 0x38, 0xC3, 0xAA, 0x73, 0x23,
/*2B00:*/ 0x04, 0x28, 0xA5, 0x40, 0xBA, 0xC5, 0x30, 0xDA, 0xE6,
0xA6, 0x1F, 0x24, 0x0D, 0xF0, 0xAD, 0xBA,
/*2B10:*/ 0x0D, 0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*2B20:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B7DB[] = {
/*30F8:*/ 0xAC, 0x39, 0x50, 0xC7, 0x78, 0x48, 0xD1, 0x0C,
/*3100:*/ 0x58, 0xF9, 0xF1, 0x79, 0xA8, 0xAE, 0xF3, 0xB1, 0x35,
0x55, 0x25, 0x38, 0xE5, 0xA8, 0x73, 0x23,
/*3110:*/ 0x71, 0xD7, 0x5A, 0xBF, 0xB5, 0xC5, 0x30, 0xDA, 0xEA,
0xA6, 0x1F, 0x24, 0x14, 0x10, 0xA9, 0x33,
/*3120:*/ 0x11, 0x7D, 0xE3, 0xAC, 0x90, 0xD9, 0x61, 0x70, 0xFC,
0xE3, 0xB9, 0x2B, 0xE4, 0xF2, 0x13, 0xEA,
/*3130:*/ 0x2C, 0x09, 0xCD, 0x6A, 0x8B, 0x97, 0x6F, 0xA7, 0xB5,
0x39, 0x50, 0xC7, 0x22, 0x48, 0xD1, 0x0C,
/*3140:*/ 0x48, 0xF8, 0xF1, 0x79, 0x2E, 0xAE, 0xF3, 0xB1, 0xBC,
0x55, 0x25, 0x38, 0xD9, 0xAA, 0x73, 0x23,
/*3150:*/ 0xD2, 0xD7, 0x5A, 0xBF, 0xB4, 0xC5, 0x30, 0xDA, 0xF3,
0xA6, 0x1F, 0x24, 0x03, 0x10, 0xA9, 0x33,
/*3160:*/ 0xD3, 0x82, 0x1C, 0x53, 0x83, 0xD9, 0x61, 0x70, 0xFC,
0xE3, 0xB9, 0x2B, 0x88, 0xF2, 0x13, 0xEA,
/*3170:*/ 0x6B, 0x09, 0xCD, 0x6A, 0x92, 0x97, 0x6F, 0xA7, 0xBD,
0x39, 0x50, 0xC7, 0x3B, 0x48, 0xD1, 0x0C,
/*3180:*/ 0x25, 0xF9, 0xF1, 0x79, 0x93, 0xAF, 0xF3, 0xB1, 0xF5,
0x55, 0x25, 0x38, 0xEF, 0xA8, 0x73, 0x23,
/*3190:*/ 0x08, 0x28, 0xA5, 0x40, 0x70, 0x3A, 0xCF, 0x25, 0xFF,
0xA6, 0x1F, 0x24, 0x35, 0x12, 0xA9, 0x33,
/*31A0:*/ 0xC8, 0x82, 0x1C, 0x53, 0x94, 0xD9, 0x61, 0x70, 0x05,
0xE3, 0xB9, 0x2B, 0x80, 0xF2, 0x13, 0xEA,
/*31B0:*/ 0x3F, 0x09, 0xCD, 0x6A, 0xC1, 0x68, 0x90, 0x58, 0xA1,
0x39, 0x50, 0xC7, 0x2F, 0x48, 0xD1, 0x0C,
/*31C0:*/ 0x5B, 0xF9, 0xF1, 0x79, 0xCE, 0xAF, 0xF3, 0xB1, 0xED,
0x55, 0x25, 0x38, 0xCF, 0xAA, 0x73, 0x23,
/*31D0:*/ 0x4B, 0x2B, 0xA5, 0x40, 0xBB, 0xC5, 0x30, 0xDA, 0x4A,
0xA7, 0x1F, 0x24, 0x19, 0x10, 0xA9, 0x33,
/*31E0:*/ 0xD5, 0x82, 0x1C, 0x53, 0x92, 0xD9, 0x61, 0x70, 0x3A,
0x1D, 0x46, 0xD4, 0xAD, 0xF2, 0x13, 0xEA,
/*31F0:*/ 0x39, 0xF6, 0x32, 0x95, 0xE4, 0x97, 0x6F, 0xA7, 0xE0,
0xC4, 0xAF, 0x38, 0x3B, 0x48, 0xD1, 0x0C,
/*3200:*/ 0x5A, 0xF9, 0xF1, 0x79, 0x43, 0xAE, 0xF3, 0xB1, 0xD2,
0x55, 0x25, 0x38, 0xED, 0xAA, 0x73, 0x23,
/*3210:*/ 0x4D, 0x28, 0xA5, 0x40, 0xB4, 0xC5, 0x30, 0xDA, 0xE8,
0xA6, 0x1F, 0x24, 0x0D, 0x10, 0xA9, 0x33,
/*3220:*/ 0xE5, 0x82, 0x1C, 0x53, 0x3F, 0xDF, 0x61, 0x70, 0x95,
0xE3, 0xB9, 0x2B, 0x88, 0xF3, 0x13, 0xEA,
/*3230:*/ 0x30, 0x09, 0xCD, 0x6A, 0xCF, 0x97, 0x6F, 0xA7, 0xD9,
0x39, 0x50, 0xC7, 0x30, 0x48, 0xD1, 0x0C,
/*3240:*/ 0x3D, 0xF9, 0xF1, 0x79, 0x74, 0xAE, 0xF3, 0xB1, 0xC7,
0x55, 0x25, 0x38, 0x9C, 0xAA, 0x73, 0x23,
/*3250:*/ 0x05, 0x28, 0xA5, 0x40, 0x94, 0xC5, 0x30, 0xDA, 0x21,
0xA4, 0x1F, 0x24, 0x42, 0x10, 0xA9, 0x33,
/*3260:*/ 0xE9, 0x80, 0x1C, 0x53, 0x88, 0xD9, 0x61, 0x70, 0xFD,
0xE3, 0xB9, 0x2B, 0x8A, 0xF2, 0x13, 0xEA,
/*3270:*/ 0x0F, 0x09, 0xCD, 0x6A, 0xD8, 0x97, 0x6F, 0xA7, 0xA2,
0x39, 0x50, 0xC7, 0xB6, 0xB5, 0x2E, 0xF3,
/*3280:*/ 0x53, 0xF9, 0xF1, 0x79, 0x34, 0xAE, 0xF3, 0xB1, 0xF2,
0x55, 0x25, 0x38, 0xDE, 0xAA, 0x73, 0x23,
/*3290:*/ 0x5D, 0x28, 0xA5, 0x40, 0xCC, 0xC5, 0x30, 0xDA, 0xD5,
0xA6, 0x1F, 0x24, 0xAB, 0x11, 0xA9, 0x33,
/*32A0:*/ 0x39, 0x86, 0x1C, 0x53, 0x6B, 0xDC, 0x61, 0x70, 0xEF,
0xE3, 0xB9, 0x2B, 0x8B, 0xF2, 0x13, 0xEA,
/*32B0:*/ 0x38, 0x09, 0xCD, 0x6A, 0xFD, 0x97, 0x6F, 0xA7, 0x4F,
0x39, 0x50, 0xC7, 0x11, 0x44, 0xD1, 0x0C,
/*32C0:*/ 0x25, 0xF9, 0xF1, 0x79, 0x39, 0xAE, 0xF3, 0xB1, 0xB7,
0x57, 0x25, 0x38, 0xFF, 0xAA, 0x73, 0x23,
/*32D0:*/ 0xBE, 0xD7, 0x5A, 0xBF, 0xC9, 0xC5, 0x30, 0xDA, 0xEF,
0xA7, 0x1F, 0x24, 0x0F, 0x10, 0xA9, 0x33,
/*32E0:*/ 0xE2, 0x82, 0x1C, 0x53, 0xFE, 0xD9, 0x61, 0x70, 0x99,
0xE1, 0xB9, 0x2B, 0x8B, 0xF2, 0x13, 0xEA,
/*32F0:*/ 0x29, 0x09, 0xCD, 0x6A, 0x37, 0x97, 0x6F, 0xA7, 0xA0,
0x39, 0x50, 0xC7, 0x21, 0x48, 0xD1, 0x0C,
/*3300:*/ 0x56, 0xF9, 0xF1, 0x79, 0x3A, 0xAE, 0xF3, 0xB1, 0x24,
0xAA, 0xDA, 0xC7, 0xD6, 0xAA, 0x73, 0x23,
/*3310:*/ 0x04, 0x28, 0xA5, 0x40, 0xB1, 0xC5, 0x30, 0xDA, 0xC0,
0xA6, 0x1F, 0x24, 0x15, 0x10, 0xA9, 0x33,
/*3320:*/ 0xDA, 0x82, 0x1C, 0x53, 0x82, 0xD9, 0x61, 0x70, 0xFE,
0xE3, 0xB9, 0x2B, 0x97, 0xF2, 0x13, 0xEA,
/*3330:*/ 0x47, 0x09, 0xCD, 0x6A, 0xD9, 0x97, 0x6F, 0xA7, 0xA0,
0x39, 0x50, 0xC7, 0x25, 0x48, 0xD1, 0x0C,
/*3340:*/ 0xFE, 0x07, 0x0E, 0x86, 0x73, 0xAE, 0xF3, 0xB1, 0xF4,
0x55, 0x25, 0x38, 0xCD, 0xAA, 0x73, 0x23,
/*3350:*/ 0x06, 0x28, 0xA5, 0x40, 0x9B, 0xC5, 0x30, 0xDA, 0xA7,
0xA6, 0x1F, 0x24, 0x0E, 0x10, 0xA9, 0x33,
/*3360:*/ 0x98, 0x80, 0x1C, 0x53, 0x24, 0xD8, 0x61, 0x70, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*3370:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B7DC[] = {
/*3928:*/ 0xB9, 0x39, 0x50, 0xC7, 0x0C, 0x48, 0xD1, 0x0C,
/*3930:*/ 0xBB, 0x07, 0x0E, 0x86, 0x32, 0xAE, 0xF3, 0xB1, 0xDF,
0x55, 0x25, 0x38, 0xC7, 0xAA, 0x73, 0x23,
/*3940:*/ 0xBA, 0xD7, 0x5A, 0xBF, 0x0D, 0xC5, 0x30, 0xDA, 0x77,
0x59, 0xE0, 0xDB, 0x13, 0x10, 0xA9, 0x33,
/*3950:*/ 0x93, 0x82, 0x1C, 0x53, 0x83, 0xD9, 0x61, 0x70, 0xE1,
0xE0, 0xB9, 0x2B, 0xF2, 0xF2, 0x13, 0xEA,
/*3960:*/ 0x1F, 0x09, 0xCD, 0x6A, 0xB5, 0x97, 0x6F, 0xA7, 0xA0,
0x39, 0x50, 0xC7, 0x68, 0x48, 0xD1, 0x0C,
/*3970:*/ 0x13, 0xF9, 0xF1, 0x79, 0x38, 0xAE, 0xF3, 0xB1, 0x14,
0xA9, 0xDA, 0xC7, 0xD8, 0xAA, 0x73, 0x23,
/*3980:*/ 0x76, 0x28, 0xA5, 0x40, 0xBA, 0xC5, 0x30, 0xDA, 0xE5,
0xA6, 0x1F, 0x24, 0x03, 0x10, 0xA9, 0x33,
/*3990:*/ 0xFF, 0x82, 0x1C, 0x53, 0x93, 0xD9, 0x61, 0x70, 0x8B,
0xE3, 0xB9, 0x2B, 0x33, 0xF2, 0x13, 0xEA,
/*39A0:*/ 0x18, 0x09, 0xCD, 0x6A, 0xDE, 0x97, 0x6F, 0xA7, 0xAD,
0x39, 0x50, 0xC7, 0xA4, 0xB7, 0x2E, 0xF3,
/*39B0:*/ 0xD5, 0xFA, 0xF1, 0x79, 0x57, 0xAE, 0xF3, 0xB1, 0xCE,
0x54, 0x25, 0x38, 0x83, 0xAB, 0x73, 0x23,
/*39C0:*/ 0xB0, 0x28, 0xA5, 0x40, 0xAB, 0xC5, 0x30, 0xDA, 0xEC,
0xA6, 0x1F, 0x24, 0x03, 0x10, 0xA9, 0x33,
/*39D0:*/ 0xC2, 0x82, 0x1C, 0x53, 0xC1, 0xD9, 0x61, 0x70, 0xFC,
0xE3, 0xB9, 0x2B, 0x88, 0xF2, 0x13, 0xEA,
/*39E0:*/ 0x3F, 0x09, 0xCD, 0x6A, 0xE9, 0x97, 0x6F, 0xA7, 0x7D,
0x39, 0x50, 0xC7, 0x48, 0x48, 0xD1, 0x0C,
/*39F0:*/ 0x83, 0xF9, 0xF1, 0x79, 0x86, 0xAC, 0xF3, 0xB1, 0xE0,
0xAA, 0xDA, 0xC7, 0xD3, 0xAA, 0x73, 0x23,
/*3A00:*/ 0x10, 0x28, 0xA5, 0x40, 0xAF, 0xC5, 0x30, 0xDA, 0xB5,
0xA6, 0x1F, 0x24, 0x27, 0x10, 0xA9, 0x33,
/*3A10:*/ 0x83, 0x7C, 0xE3, 0xAC, 0x9E, 0xD9, 0x61, 0x70, 0xFD,
0xE3, 0xB9, 0x2B, 0x8A, 0xF2, 0x13, 0xEA,
/*3A20:*/ 0x30, 0x09, 0xCD, 0x6A, 0xF2, 0x96, 0x6F, 0xA7, 0xAB,
0x39, 0x50, 0xC7, 0xEE, 0x48, 0xD1, 0x0C,
/*3A30:*/ 0xB9, 0xF9, 0xF1, 0x79, 0x3A, 0xAE, 0xF3, 0xB1, 0xF9,
0x55, 0x25, 0x38, 0xD6, 0xAA, 0x73, 0x23,
/*3A40:*/ 0xD1, 0x28, 0xA5, 0x40, 0x12, 0xC4, 0x30, 0xDA, 0x2A,
0xA7, 0x1F, 0x24, 0xDD, 0xEF, 0x56, 0xCC,
/*3A50:*/ 0x6B, 0x82, 0x1C, 0x53, 0xA5, 0xD9, 0x61, 0x70, 0x1A,
0x1C, 0x46, 0xD4, 0xA4, 0xF2, 0x13, 0xEA,
/*3A60:*/ 0x0A, 0x09, 0xCD, 0x6A, 0xD9, 0x97, 0x6F, 0xA7, 0xA2,
0x39, 0x50, 0xC7, 0xD9, 0x48, 0xD1, 0x0C,
/*3A70:*/ 0x49, 0xF9, 0xF1, 0x79, 0x8F, 0xAE, 0xF3, 0xB1, 0xC0,
0x55, 0x25, 0x38, 0xB4, 0xAA, 0x73, 0x23,
/*3A80:*/ 0x04, 0x28, 0xA5, 0x40, 0xB9, 0xC5, 0x30, 0xDA, 0xAA,
0xA6, 0x1F, 0x24, 0x0B, 0x10, 0xA9, 0x33,
/*3A90:*/ 0xC0, 0x82, 0x1C, 0x53, 0x83, 0xD9, 0x61, 0x70, 0xB5,
0xE3, 0xB9, 0x2B, 0x06, 0xF2, 0x13, 0xEA,
/*3AA0:*/ 0x07, 0x09, 0xCD, 0x6A, 0xCD, 0x97, 0x6F, 0xA7, 0xA1,
0x39, 0x50, 0xC7, 0x2F, 0x48, 0xD1, 0x0C,
/*3AB0:*/ 0x25, 0xF9, 0xF1, 0x79, 0x15, 0xAE, 0xF3, 0xB1, 0xE1,
0x55, 0x25, 0x38, 0xEE, 0xAA, 0x73, 0x23,
/*3AC0:*/ 0x55, 0x28, 0xA5, 0x40, 0x88, 0xC7, 0x30, 0xDA, 0xEE,
0xA6, 0x1F, 0x24, 0xCF, 0x10, 0xA9, 0x33,
/*3AD0:*/ 0x9F, 0x88, 0x1C, 0x53, 0x8F, 0xD9, 0x61, 0x70, 0x0D,
0xE2, 0xB9, 0x2B, 0x8A, 0xF2, 0x13, 0xEA,
/*3AE0:*/ 0x1C, 0x09, 0xCD, 0x6A, 0xA9, 0x95, 0x6F, 0xA7, 0xA0,
0x39, 0x50, 0xC7, 0x2C, 0x48, 0xD1, 0x0C,
/*3AF0:*/ 0xE6, 0x06, 0x0E, 0x86, 0x68, 0xAE, 0xF3, 0xB1, 0xFB,
0x55, 0x25, 0x38, 0xCB, 0xAA, 0x73, 0x23,
/*3B00:*/ 0x0F, 0x28, 0xA5, 0x40, 0xE4, 0xC5, 0x30, 0xDA, 0xE6,
0xA6, 0x1F, 0x24, 0x59, 0x10, 0xA9, 0x33,
/*3B10:*/ 0xD3, 0x82, 0x1C, 0x53, 0xE9, 0xD9, 0x61, 0x70, 0xE5,
0xE3, 0xB9, 0x2B, 0xD7, 0xF2, 0x13, 0xEA,
/*3B20:*/ 0x39, 0x09, 0xCD, 0x6A, 0x26, 0x96, 0x6F, 0xA7, 0x83,
0x2F, 0x50, 0xC7, 0x6A, 0x48, 0xD1, 0x0C,
/*3B30:*/ 0xC3, 0xF9, 0xF1, 0x79, 0x55, 0xAE, 0xF3, 0xB1, 0xAE,
0x54, 0x25, 0x38, 0xD8, 0xAA, 0x73, 0x23,
/*3B40:*/ 0x04, 0x28, 0xA5, 0x40, 0xA9, 0xC5, 0x30, 0xDA, 0xE0,
0xA6, 0x1F, 0x24, 0xD1, 0x11, 0xA9, 0x33,
/*3B50:*/ 0xDF, 0x82, 0x1C, 0x53, 0x8E, 0xD9, 0x61, 0x70, 0x22,
0xF8, 0xB9, 0x2B, 0x72, 0xF2, 0x13, 0xEA,
/*3B60:*/ 0x3E, 0x09, 0xCD, 0x6A, 0x30, 0x97, 0x6F, 0xA7, 0xB2,
0x39, 0x50, 0xC7, 0x1F, 0x4A, 0xD1, 0x0C,
/*3B70:*/ 0x4E, 0xF9, 0xF1, 0x79, 0x81, 0xAF, 0xF3, 0xB1, 0xE1,
0x55, 0x25, 0x38, 0x55, 0xAA, 0x73, 0x23,
/*3B80:*/ 0x0D, 0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*3B90:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B7DD[] = {
/*4168:*/ 0xCC, 0x39, 0x50, 0xC7, 0xDF, 0x48, 0xD1, 0x0C,
/*4170:*/ 0x5B, 0xF9, 0xF1, 0x79, 0xF4, 0x51, 0x0C, 0x4E, 0xAA,
0x55, 0x25, 0x38, 0x7E, 0xAA, 0x73, 0x23,
/*4180:*/ 0x0E, 0xD7, 0x5A, 0xBF, 0x83, 0xC5, 0x30, 0xDA, 0x79,
0xA7, 0x1F, 0x24, 0x03, 0x10, 0xA9, 0x33,
/*4190:*/ 0xDF, 0x82, 0x1C, 0x53, 0x92, 0xD9, 0x61, 0x70, 0xED,
0x1F, 0x46, 0xD4, 0x85, 0xF2, 0x13, 0xEA,
/*41A0:*/ 0x82, 0x09, 0xCD, 0x6A, 0xC9, 0x97, 0x6F, 0xA7, 0x8F,
0x39, 0x50, 0xC7, 0x38, 0x48, 0xD1, 0x0C,
/*41B0:*/ 0x5B, 0xF9, 0xF1, 0x79, 0x2E, 0xAC, 0xF3, 0xB1, 0xA6,
0x55, 0x25, 0x38, 0xFA, 0xAA, 0x73, 0x23,
/*41C0:*/ 0x37, 0x28, 0xA5, 0x40, 0xAB, 0xC5, 0x30, 0xDA, 0x19,
0x58, 0xE0, 0xDB, 0x1E, 0x10, 0xA9, 0x33,
/*41D0:*/ 0xDE, 0x82, 0x1C, 0x53, 0x83, 0xD9, 0x61, 0x70, 0xD4,
0xE3, 0xB9, 0x2B, 0x87, 0xF2, 0x13, 0xEA,
/*41E0:*/ 0x34, 0x09, 0xCD, 0x6A, 0xFE, 0x97, 0x6F, 0xA7, 0x98,
0x39, 0x50, 0xC7, 0x36, 0x48, 0xD1, 0x0C,
/*41F0:*/ 0x58, 0xF9, 0xF1, 0x79, 0x38, 0xAE, 0xF3, 0xB1, 0x75,
0x55, 0x25, 0x38, 0x89, 0xAA, 0x73, 0x23,
/*4200:*/ 0x07, 0x2A, 0xA5, 0x40, 0xC5, 0xC7, 0x30, 0xDA, 0xF5,
0xA6, 0x1F, 0x24, 0x68, 0x12, 0xA9, 0x33,
/*4210:*/ 0xB0, 0x82, 0x1C, 0x53, 0x88, 0xD9, 0x61, 0x70, 0xD1,
0xE3, 0xB9, 0x2B, 0x89, 0xF2, 0x13, 0xEA,
/*4220:*/ 0xD3, 0xF7, 0x32, 0x95, 0xD6, 0x97, 0x6F, 0xA7, 0x2E,
0x20, 0x50, 0xC7, 0x2E, 0x48, 0xD1, 0x0C,
/*4230:*/ 0x57, 0xF9, 0xF1, 0x79, 0x31, 0xAE, 0xF3, 0xB1, 0xF6,
0x55, 0x25, 0x38, 0xD7, 0xAA, 0x73, 0x23,
/*4240:*/ 0x04, 0x28, 0xA5, 0x40, 0xB2, 0xC5, 0x30, 0xDA, 0x63,
0x5B, 0xE0, 0xDB, 0xFA, 0x10, 0xA9, 0x33,
/*4250:*/ 0xCB, 0x82, 0x1C, 0x53, 0xC7, 0xD9, 0x61, 0x70, 0x10,
0x1C, 0x46, 0xD4, 0xA1, 0xF3, 0x13, 0xEA,
/*4260:*/ 0x77, 0x09, 0xCD, 0x6A, 0xD8, 0x97, 0x6F, 0xA7, 0x8E,
0x39, 0x50, 0xC7, 0x34, 0x48, 0xD1, 0x0C,
/*4270:*/ 0x5A, 0xF9, 0xF1, 0x79, 0x03, 0xAF, 0xF3, 0xB1, 0xFE,
0x55, 0x25, 0x38, 0x3C, 0xAA, 0x73, 0x23,
/*4280:*/ 0x78, 0x2A, 0xA5, 0x40, 0xED, 0xC5, 0x30, 0xDA, 0xE4,
0xA6, 0x1F, 0x24, 0x0A, 0x10, 0xA9, 0x33,
/*4290:*/ 0x17, 0x83, 0x1C, 0x53, 0x82, 0xD9, 0x61, 0x70, 0xEB,
0xE3, 0xB9, 0x2B, 0x84, 0xF2, 0x13, 0xEA,
/*42A0:*/ 0x3B, 0x08, 0xCD, 0x6A, 0x85, 0x96, 0x6F, 0xA7, 0xBA,
0x39, 0x50, 0xC7, 0x1A, 0x48, 0xD1, 0x0C,
/*42B0:*/ 0x5B, 0xF9, 0xF1, 0x79, 0x3A, 0xAE, 0xF3, 0xB1, 0x81,
0x55, 0x25, 0x38, 0x5F, 0xAB, 0x73, 0x23,
/*42C0:*/ 0x9D, 0x2A, 0xA5, 0x40, 0xD3, 0xCF, 0x30, 0xDA, 0xE7,
0xA6, 0x1F, 0x24, 0x32, 0x10, 0xA9, 0x33,
/*42D0:*/ 0xEA, 0x82, 0x1C, 0x53, 0x82, 0xD9, 0x61, 0x70, 0xA2,
0xE2, 0xB9, 0x2B, 0xA6, 0xF2, 0x13, 0xEA,
/*42E0:*/ 0x32, 0x09, 0xCD, 0x6A, 0x8F, 0x97, 0x6F, 0xA7, 0xA1,
0x39, 0x50, 0xC7, 0x46, 0x48, 0xD1, 0x0C,
/*42F0:*/ 0xE3, 0xF8, 0xF1, 0x79, 0x6F, 0xAE, 0xF3, 0xB1, 0xF6,
0x55, 0x25, 0x38, 0xCC, 0xAA, 0x73, 0x23,
/*4300:*/ 0x08, 0x28, 0xA5, 0x40, 0xBB, 0xC5, 0x30, 0xDA, 0xEB,
0xA6, 0x1F, 0x24, 0x02, 0x10, 0xA9, 0x33,
/*4310:*/ 0xCA, 0x82, 0x1C, 0x53, 0x84, 0xD9, 0x61, 0x70, 0xFB,
0xE3, 0xB9, 0x2B, 0x26, 0xF2, 0x13, 0xEA,
/*4320:*/ 0x3E, 0x09, 0xCD, 0x6A, 0xDB, 0x97, 0x6F, 0xA7, 0xAB,
0x39, 0x50, 0xC7, 0x0F, 0x48, 0xD1, 0x0C,
/*4330:*/ 0x9A, 0x06, 0x0E, 0x86, 0x36, 0xAE, 0xF3, 0xB1, 0xF8,
0x55, 0x25, 0x38, 0x86, 0xAA, 0x73, 0x23,
/*4340:*/ 0x03, 0x28, 0xA5, 0x40, 0xCC, 0xC5, 0x30, 0xDA, 0xF2,
0xA6, 0x1F, 0x24, 0x16, 0x10, 0xA9, 0x33,
/*4350:*/ 0xEA, 0x82, 0x1C, 0x53, 0xAB, 0xD9, 0x61, 0x70, 0xF3,
0xE3, 0xB9, 0x2B, 0x9E, 0xF2, 0x13, 0xEA,
/*4360:*/ 0x2E, 0x01, 0xCD, 0x6A, 0xA1, 0x97, 0x6F, 0xA7, 0x8B,
0x39, 0x50, 0xC7, 0x27, 0x48, 0xD1, 0x0C,
/*4370:*/ 0x52, 0xF9, 0xF1, 0x79, 0x3A, 0xAE, 0xF3, 0xB1, 0xFA,
0x55, 0x25, 0x38, 0x9B, 0xAA, 0x73, 0x23,
/*4380:*/ 0x05, 0x28, 0xA5, 0x40, 0xAC, 0xC5, 0x30, 0xDA, 0xE6,
0xA6, 0x1F, 0x24, 0x24, 0x10, 0xA9, 0x33,
/*4390:*/ 0xDE, 0x82, 0x1C, 0x53, 0x90, 0xD9, 0x61, 0x70, 0xFD,
0xE3, 0xB9, 0x2B, 0x9C, 0xF2, 0x13, 0xEA,
/*43A0:*/ 0xBB, 0x08, 0xCD, 0x6A, 0x6C, 0x97, 0x6F, 0xA7, 0x8B,
0x39, 0x50, 0xC7, 0x27, 0x48, 0xD1, 0x0C,
/*43B0:*/ 0x41, 0xF9, 0xF1, 0x79, 0x40, 0x51, 0x0C, 0x4E, 0xFE,
0x55, 0x25, 0x38, 0xD8, 0xAA, 0x73, 0x23,
/*43C0:*/ 0x0F, 0x28, 0xA5, 0x40, 0x4C, 0xC4, 0x30, 0xDA, 0xF0,
0xA6, 0x1F, 0x24, 0x00, 0x10, 0xA9, 0x33,
/*43D0:*/ 0x04, 0x7D, 0xE3, 0xAC, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*43E0:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};

BYTE B7DE[] = {
/*4A08:*/ 0xA9, 0x39, 0x50, 0xC7, 0x53, 0x48, 0xD1, 0x0C,
/*4A10:*/ 0x66, 0xF9, 0xF1, 0x79, 0x68, 0xAE, 0xF3, 0xB1, 0x98,
0x55, 0x25, 0x38, 0xD9, 0xAA, 0x73, 0x23,
/*4A20:*/ 0x03, 0x28, 0xA5, 0x40, 0xEA, 0xC5, 0x30, 0xDA, 0xD5,
0xA6, 0x1F, 0x24, 0x08, 0x10, 0xA9, 0x33,
/*4A30:*/ 0x32, 0x81, 0x1C, 0x53, 0x84, 0xD9, 0x61, 0x70, 0xEA,
0xE3, 0xB9, 0x2B, 0x93, 0xF2, 0x13, 0xEA,
/*4A40:*/ 0x05, 0x09, 0xCD, 0x6A, 0xD9, 0x97, 0x6F, 0xA7, 0xEC,
0x39, 0x50, 0xC7, 0x27, 0x48, 0xD1, 0x0C,
/*4A50:*/ 0x76, 0xF9, 0xF1, 0x79, 0x3A, 0xAE, 0xF3, 0xB1, 0xE8,
0x55, 0x25, 0x38, 0xA3, 0xAB, 0x73, 0x23,
/*4A60:*/ 0x09, 0x28, 0xA5, 0x40, 0x0C, 0xC4, 0x30, 0xDA, 0xEB,
0xA4, 0x1F, 0x24, 0x37, 0x12, 0xA9, 0x33,
/*4A70:*/ 0xC7, 0x82, 0x1C, 0x53, 0xA6, 0xD9, 0x61, 0x70, 0x07,
0xE3, 0xB9, 0x2B, 0x9E, 0xF2, 0x13, 0xEA,
/*4A80:*/ 0x25, 0x09, 0xCD, 0x6A, 0x9B, 0x69, 0x90, 0x58, 0xDF,
0x39, 0x50, 0xC7, 0xC4, 0xB7, 0x2E, 0xF3,
/*4A90:*/ 0x23, 0xF9, 0xF1, 0x79, 0x79, 0xAE, 0xF3, 0xB1, 0xD7,
0x55, 0x25, 0x38, 0xD8, 0xAA, 0x73, 0x23,
/*4AA0:*/ 0x73, 0x28, 0xA5, 0x40, 0xB4, 0xC5, 0x30, 0xDA, 0xE6,
0xA6, 0x1F, 0x24, 0x02, 0x10, 0xA9, 0x33,
/*4AB0:*/ 0xF8, 0x82, 0x1C, 0x53, 0x8A, 0xD9, 0x61, 0x70, 0x27,
0xE3, 0xB9, 0x2B, 0x04, 0xF2, 0x13, 0xEA,
/*4AC0:*/ 0x4F, 0x0B, 0xCD, 0x6A, 0x80, 0x97, 0x6F, 0xA7, 0x80,
0x39, 0x50, 0xC7, 0x36, 0x48, 0xD1, 0x0C,
/*4AD0:*/ 0x5A, 0xF9, 0xF1, 0x79, 0x6D, 0xAE, 0xF3, 0xB1, 0x4C,
0x55, 0x25, 0x38, 0xDE, 0xAA, 0x73, 0x23,
/*4AE0:*/ 0x28, 0x28, 0xA5, 0x40, 0xBA, 0xC5, 0x30, 0xDA, 0x91,
0xA7, 0x1F, 0x24, 0x02, 0x10, 0xA9, 0x33,
/*4AF0:*/ 0xE6, 0x83, 0x1C, 0x53, 0x48, 0x26, 0x9E, 0x8F, 0x46,
0xE3, 0xB9, 0x2B, 0x87, 0xF2, 0x13, 0xEA,
/*4B00:*/ 0xD6, 0xF6, 0x32, 0x95, 0xC0, 0x97, 0x6F, 0xA7, 0xF0,
0x38, 0x50, 0xC7, 0x20, 0x48, 0xD1, 0x0C,
/*4B10:*/ 0xED, 0xF8, 0xF1, 0x79, 0x3A, 0xAE, 0xF3, 0xB1, 0xF1,
0x55, 0x25, 0x38, 0xD9, 0xAA, 0x73, 0x23,
/*4B20:*/ 0x0B, 0x28, 0xA5, 0x40, 0x89, 0xC5, 0x30, 0xDA, 0xE7,
0xA6, 0x1F, 0x24, 0x00, 0x10, 0xA9, 0x33,
/*4B30:*/ 0xC8, 0x82, 0x1C, 0x53, 0x9C, 0xD9, 0x61, 0x70, 0xDC,
0xE3, 0xB9, 0x2B, 0x5E, 0x0D, 0xEC, 0x15,
/*4B40:*/ 0xFB, 0x09, 0xCD, 0x6A, 0xD9, 0x97, 0x6F, 0xA7, 0x8A,
0x39, 0x50, 0xC7, 0x06, 0x48, 0xD1, 0x0C,
/*4B50:*/ 0x5A, 0xF9, 0xF1, 0x79, 0x2F, 0x44, 0x0C, 0x4E, 0x2A,
0xAA, 0xDA, 0xC7, 0x7B, 0xAB, 0x73, 0x23,
/*4B60:*/ 0x15, 0x28, 0xA5, 0x40, 0x80, 0xC4, 0x30, 0xDA, 0xF3,
0xA4, 0x1F, 0x24, 0x6A, 0x10, 0xA9, 0x33,
/*4B70:*/ 0x81, 0x82, 0x1C, 0x53, 0x84, 0xD9, 0x61, 0x70, 0x29,
0x1C, 0x46, 0xD4, 0x89, 0xF2, 0x13, 0xEA,
/*4B80:*/ 0x3F, 0x09, 0xCD, 0x6A, 0x8C, 0x97, 0x6F, 0xA7, 0xAE,
0x39, 0x50, 0xC7, 0x34, 0x48, 0xD1, 0x0C,
/*4B90:*/ 0x57, 0xF9, 0xF1, 0x79, 0x3A, 0xAE, 0xF3, 0xB1, 0xE8,
0x55, 0x25, 0x38, 0xC1, 0xAA, 0x73, 0x23,
/*4BA0:*/ 0x46, 0x28, 0xA5, 0x40, 0xAA, 0xC5, 0x30, 0xDA, 0xE7,
0xA6, 0x1F, 0x24, 0x1C, 0x10, 0xA9, 0x33,
/*4BB0:*/ 0xB8, 0x82, 0x1C, 0x53, 0x8F, 0xD9, 0x61, 0x70, 0x74,
0xE0, 0xB9, 0x2B, 0x83, 0xF2, 0x13, 0xEA,
/*4BC0:*/ 0x2A, 0x09, 0xCD, 0x6A, 0x4C, 0x96, 0x6F, 0xA7, 0xA9,
0x39, 0x50, 0xC7, 0xB1, 0x48, 0xD1, 0x0C,
/*4BD0:*/ 0x58, 0xF9, 0xF1, 0x79, 0x1C, 0xAE, 0xF3, 0xB1, 0xFB,
0x55, 0x25, 0x38, 0x56, 0xAA, 0x73, 0x23,
/*4BE0:*/ 0x0F, 0x28, 0xA5, 0x40, 0xB4, 0xC4, 0x30, 0xDA, 0xEE,
0xA6, 0x1F, 0x24, 0x1F, 0x10, 0xA9, 0x33,
/*4BF0:*/ 0xDD, 0x82, 0x1C, 0x53, 0xDA, 0xD9, 0x61, 0x70, 0xF0,
0xE3, 0xB9, 0x2B, 0x87, 0xF2, 0x13, 0xEA,
/*4C00:*/ 0xFF, 0xF6, 0x32, 0x95, 0x36, 0x96, 0x6F, 0xA7, 0xAF,
0x39, 0x50, 0xC7, 0x3F, 0x48, 0xD1, 0x0C,
/*4C10:*/ 0x36, 0xF9, 0xF1, 0x79, 0x11, 0xAF, 0xF3, 0xB1, 0x33,
0x55, 0x25, 0x38, 0xBC, 0xAA, 0x73, 0x23,
/*4C20:*/ 0x04, 0x28, 0xA5, 0x40, 0x9A, 0xC5, 0x30, 0xDA, 0xE6,
0xA6, 0x1F, 0x24, 0x01, 0x10, 0xA9, 0x33,
/*4C30:*/ 0xBF, 0x82, 0x1C, 0x53, 0x83, 0xD9, 0x61, 0x70, 0xFC,
0xE3, 0xB9, 0x2B, 0xFC, 0xF2, 0x13, 0xEA,
/*4C40:*/ 0x32, 0x09, 0xCD, 0x6A, 0xD8, 0x97, 0x6F, 0xA7, 0xCC,
0x3B, 0x50, 0xC7, 0x2E, 0x48, 0xD1, 0x0C,
/*4C50:*/ 0x4E, 0xF9, 0xF1, 0x79, 0x35, 0xAE, 0xF3, 0xB1, 0xFA,
0x55, 0x25, 0x38, 0xD9, 0xAA, 0x73, 0x23,
/*4C60:*/ 0x04, 0x28, 0xA5, 0x40, 0xAF, 0xC5, 0x30, 0xDA, 0xCB,
0xA6, 0x1F, 0x24, 0x08, 0x10, 0xA9, 0x33,
/*4C70:*/ 0xC4, 0x82, 0x1C, 0x53, 0x1F, 0xD9, 0x61, 0x70, 0xC1,
0xE3, 0xB9, 0x2B, 0xBB, 0xF2, 0x13, 0xEA,
/*4C80:*/ 0x3C, 0x09, 0xCD, 0x6A, 0xD4, 0x97, 0x6F, 0xA7, 0xB3,
0x39, 0x50, 0xC7, 0x09, 0x48, 0xD1, 0x0C,
/*4C90:*/ 0x5D, 0xF9, 0xF1, 0x79, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*4CA0:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B7DF[] = {
/*5308:*/ 0xB9, 0x39, 0x50, 0xC7, 0x42, 0xB7, 0x2E, 0xF3,
/*5310:*/ 0x5A, 0xF9, 0xF1, 0x79, 0x15, 0xAE, 0xF3, 0xB1, 0xF9,
0x55, 0x25, 0x38, 0x5B, 0xAB, 0x73, 0x23,
/*5320:*/ 0x43, 0x28, 0xA5, 0x40, 0xB4, 0xC5, 0x30, 0xDA, 0x94,
0xA6, 0x1F, 0x24, 0x39, 0x12, 0xA9, 0x33,
/*5330:*/ 0xC2, 0x82, 0x1C, 0x53, 0x8E, 0xD9, 0x61, 0x70, 0xA2,
0xE3, 0xB9, 0x2B, 0xEB, 0xF2, 0x13, 0xEA,
/*5340:*/ 0x55, 0x09, 0xCD, 0x6A, 0xBE, 0x97, 0x6F, 0xA7, 0xB7,
0x39, 0x50, 0xC7, 0x68, 0x48, 0xD1, 0x0C,
/*5350:*/ 0xD9, 0xF9, 0xF1, 0x79, 0x5D, 0xAF, 0xF3, 0xB1, 0x2F,
0x55, 0x25, 0x38, 0x9A, 0xAA, 0x73, 0x23,
/*5360:*/ 0xD8, 0xD7, 0x5A, 0xBF, 0xBB, 0xC5, 0x30, 0xDA, 0xE9,
0xA6, 0x1F, 0x24, 0x1A, 0x12, 0xA9, 0x33,
/*5370:*/ 0xC9, 0x82, 0x1C, 0x53, 0x82, 0xD9, 0x61, 0x70, 0xFC,
0xE3, 0xB9, 0x2B, 0x47, 0xF2, 0x13, 0xEA,
/*5380:*/ 0xA5, 0x09, 0xCD, 0x6A, 0xD9, 0x97, 0x6F, 0xA7, 0xF9,
0xC6, 0xAF, 0x38, 0x23, 0x48, 0xD1, 0x0C,
/*5390:*/ 0x5B, 0xF9, 0xF1, 0x79, 0x3B, 0xAE, 0xF3, 0xB1, 0xF6,
0x55, 0x25, 0x38, 0xEA, 0xAA, 0x73, 0x23,
/*53A0:*/ 0x19, 0x28, 0xA5, 0x40, 0xF1, 0xC4, 0x30, 0xDA, 0xB9,
0xA6, 0x1F, 0x24, 0x0F, 0x10, 0xA9, 0x33,
/*53B0:*/ 0x9B, 0x81, 0x1C, 0x53, 0xE7, 0xD9, 0x61, 0x70, 0x89,
0xE3, 0xB9, 0x2B, 0x79, 0xF2, 0x13, 0xEA,
/*53C0:*/ 0x03, 0x09, 0xCD, 0x6A, 0xD9, 0x97, 0x6F, 0xA7, 0xB1,
0x39, 0x50, 0xC7, 0x74, 0x48, 0xD1, 0x0C,
/*53D0:*/ 0x4B, 0xF8, 0xF1, 0x79, 0x3A, 0xAE, 0xF3, 0xB1, 0xF1,
0x55, 0x25, 0x38, 0xD9, 0xAA, 0x73, 0x23,
/*53E0:*/ 0x73, 0x29, 0xA5, 0x40, 0xBB, 0xC5, 0x30, 0xDA, 0x9C,
0xA6, 0x1F, 0x24, 0x57, 0x10, 0xA9, 0x33,
/*53F0:*/ 0xF2, 0x82, 0x1C, 0x53, 0x94, 0xD9, 0x61, 0x70, 0x50,
0xE0, 0xB9, 0x2B, 0x15, 0xF2, 0x13, 0xEA,
/*5400:*/ 0x45, 0x09, 0xCD, 0x6A, 0x8E, 0x97, 0x6F, 0xA7, 0xAD,
0x39, 0x50, 0xC7, 0x3C, 0x48, 0xD1, 0x0C,
/*5410:*/ 0x4C, 0xF9, 0xF1, 0x79, 0x3A, 0xAE, 0xF3, 0xB1, 0x9D,
0x55, 0x25, 0x38, 0xD9, 0xAA, 0x73, 0x23,
/*5420:*/ 0x20, 0x28, 0xA5, 0x40, 0xB8, 0xC4, 0x30, 0xDA, 0xE6,
0xA6, 0x1F, 0x24, 0xCB, 0x12, 0xA9, 0x33,
/*5430:*/ 0xF8, 0x82, 0x1C, 0x53, 0x5E, 0x26, 0x9E, 0x8F, 0xFD,
0xE3, 0xB9, 0x2B, 0xF2, 0xF0, 0x13, 0xEA,
/*5440:*/ 0x69, 0x09, 0xCD, 0x6A, 0x0D, 0x6A, 0x90, 0x58, 0xA9,
0x39, 0x50, 0xC7, 0x3B, 0x48, 0xD1, 0x0C,
/*5450:*/ 0x4C, 0xF9, 0xF1, 0x79, 0x2C, 0xAE, 0xF3, 0xB1, 0xB3,
0x55, 0x25, 0x38, 0xD8, 0xAA, 0x73, 0x23,
/*5460:*/ 0x23, 0x28, 0xA5, 0x40, 0xB5, 0xC5, 0x30, 0xDA, 0xBF,
0xA6, 0x1F, 0x24, 0x0F, 0x10, 0xA9, 0x33,
/*5470:*/ 0xC2, 0x82, 0x1C, 0x53, 0x5E, 0xD8, 0x61, 0x70, 0x67,
0x1C, 0x46, 0xD4, 0xB3, 0xF2, 0x13, 0xEA,
/*5480:*/ 0x32, 0x08, 0xCD, 0x6A, 0xCF, 0x97, 0x6F, 0xA7, 0xAC,
0x39, 0x50, 0xC7, 0x2D, 0x48, 0xD1, 0x0C,
/*5490:*/ 0x5D, 0xF9, 0xF1, 0x79, 0x4D, 0xAC, 0xF3, 0xB1, 0x88,
0x54, 0x25, 0x38, 0x8E, 0xAA, 0x73, 0x23,
/*54A0:*/ 0x10, 0x28, 0xA5, 0x40, 0x30, 0xC5, 0x30, 0xDA, 0xCC,
0xA6, 0x1F, 0x24, 0x19, 0x10, 0xA9, 0x33,
/*54B0:*/ 0x93, 0x82, 0x1C, 0x53, 0xDE, 0xDA, 0x61, 0x70, 0x2A,
0xE1, 0xB9, 0x2B, 0x8B, 0xF2, 0x13, 0xEA,
/*54C0:*/ 0x3C, 0x09, 0xCD, 0x6A, 0xEF, 0x97, 0x6F, 0xA7, 0xA3,
0x39, 0x50, 0xC7, 0x6E, 0x48, 0xD1, 0x0C,
/*54D0:*/ 0x57, 0xF9, 0xF1, 0x79, 0x09, 0xAE, 0xF3, 0xB1, 0xE5,
0x55, 0x25, 0x38, 0x85, 0xAA, 0x73, 0x23,
/*54E0:*/ 0x10, 0x28, 0xA5, 0x40, 0xC9, 0xC5, 0x30, 0xDA, 0xDB,
0xA6, 0x1F, 0x24, 0x4E, 0x10, 0xA9, 0x33,
/*54F0:*/ 0xC2, 0x82, 0x1C, 0x53, 0x19, 0xD9, 0x61, 0x70, 0x0F,
0xE3, 0xB9, 0x2B, 0xBC, 0xF2, 0x13, 0xEA,
/*5500:*/ 0x0B, 0x09, 0xCD, 0x6A, 0x9E, 0x96, 0x6F, 0xA7, 0xA7,
0x39, 0x50, 0xC7, 0x27, 0x48, 0xD1, 0x0C,
/*5510:*/ 0x4C, 0xF9, 0xF1, 0x79, 0x24, 0xAE, 0xF3, 0xB1, 0xA7,
0x55, 0x25, 0x38, 0x6C, 0xAA, 0x73, 0x23,
/*5520:*/ 0x16, 0x28, 0xA5, 0x40, 0x22, 0xC4, 0x30, 0xDA, 0xF3,
0xA6, 0x1F, 0x24, 0x02, 0x10, 0xA9, 0x33,
/*5530:*/ 0xCA, 0x82, 0x1C, 0x53, 0x8A, 0xD9, 0x61, 0x70, 0xFC,
0xE3, 0xB9, 0x2B, 0x81, 0xF2, 0x13, 0xEA,
/*5540:*/ 0x30, 0x09, 0xCD, 0x6A, 0x07, 0x93, 0x6F, 0xA7, 0x24,
0x39, 0x50, 0xC7, 0x3A, 0x48, 0xD1, 0x0C,
/*5550:*/ 0xBF, 0xF9, 0xF1, 0x79, 0xE3, 0x51, 0x0C, 0x4E, 0xE9,
0x55, 0x25, 0x38, 0xAA, 0xAA, 0x73, 0x23,
/*5560:*/ 0x5F, 0x28, 0xA5, 0x40, 0xEE, 0xC5, 0x30, 0xDA, 0xE6,
0xA6, 0x1F, 0x24, 0x17, 0x10, 0xA9, 0x33,
/*5570:*/ 0xC6, 0x82, 0x1C, 0x53, 0x94, 0xD9, 0x61, 0x70, 0xFC,
0xE3, 0xB9, 0x2B, 0x7F, 0x0D, 0xEC, 0x15,
/*5580:*/ 0x3F, 0x09, 0xCD, 0x6A, 0xD2, 0x97, 0x6F, 0xA7, 0xA9,
0x39, 0x50, 0xC7, 0x2F, 0x48, 0xD1, 0x0C,
/*5590:*/ 0x0E, 0xF9, 0xF1, 0x79, 0x06, 0xAE, 0xF3, 0xB1, 0xE9,
0x55, 0x25, 0x38, 0xBD, 0x54, 0x8C, 0xDC,
/*55A0:*/ 0xA3, 0x28, 0xA5, 0x40, 0xC8, 0xC5, 0x30, 0xDA, 0xEB,
0x58, 0xE0, 0xDB, 0xAA, 0xEF, 0x56, 0xCC,
/*55B0:*/ 0xD8, 0x82, 0x1C, 0x53, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*55C0:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

void* B7Array[] = {
     B7D0,
     B7D1,
     B7D2,
     B7D3,
     B7D4,
     B7D5,
     B7D6,
     B7D7,
     B7D8,
     B7D9,
     B7DA,
     B7DB,
     B7DC,
     B7DD,
     B7DE,
     B7DF
};

BYTE B7StackArray[] = {
0xA4, 0x39, 0x50, 0xC7, 0x2A, 0x48, 0xD1, 0x0C, 0x5E, 0xF9, 0xF1,
0x79, 0x3E, 0xAE, 0xF3, 0xB1,
0xFD, 0x55, 0x25, 0x38, 0xDD, 0xAA, 0x73, 0x23, 0x00, 0x28, 0xA5,
0x40, 0xBE, 0xC5, 0x30, 0xDA,
0xE3, 0xA6, 0x1F, 0x24, 0x07, 0x10, 0xA9, 0x33, 0xCF, 0x82, 0x1C,
0x53, 0x87, 0xD9, 0x61, 0x70,
0xF8, 0xE3, 0xB9, 0x2B, 0x8E, 0xF2, 0x13, 0xEA, 0x3B, 0x09, 0xCD,
0x6A, 0xDD, 0x97, 0x6F, 0xA7
};


/////////////////////////
// block 8
/////////////////////////

BYTE B8D0[] = {/*00CEC8:*/ 0x01, 0x01, 0x01, 0x04, 0x01, 0x01,
0x01, 0x01,
/*00CED0:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x04, 0x01, 0x01, 0x05,
0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x05,
/*00CEE0:*/ 0x01, 0x01, 0x05, 0x01, 0x04, 0x05, 0x01, 0x01, 0x01,
0x01, 0x04, 0x05, 0x01, 0x01, 0x01, 0x01,
/*00CEF0:*/ 0x01, 0x01, 0x01, 0x01, 0x04, 0x01, 0x01, 0x05, 0x01,
0x01, 0x01, 0x01, 0x05, 0x01, 0x01, 0x01,
/*00CF00:*/ 0x01, 0x05, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x05, 0x01, 0x05,
/*00CF10:*/ 0x01, 0x01, 0x05, 0x05, 0x01, 0x01, 0x04, 0x01, 0x01,
0x01, 0x01, 0x01, 0x05, 0x01, 0x01, 0x01,
/*00CF20:*/ 0x01, 0x05, 0x01, 0x01, 0x04, 0x04, 0x01, 0x04, 0x01,
0x01, 0x01, 0x05, 0x01, 0x01, 0x01, 0x01,
/*00CF30:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x04, 0x01, 0x04,
0x01, 0x01, 0x05, 0x01, 0x01, 0x05, 0x05,
/*00CF40:*/ 0x01, 0x05, 0x01, 0x01, 0x04, 0x01, 0x01, 0x05, 0x01,
0x04, 0x01, 0x01, 0x05, 0x01, 0x01, 0x01,
/*00CF50:*/ 0x01, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01,
0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*00CF60:*/ 0x0D, 0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*00CF70:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B8D1[] = {
/*00D7F8:*/ 0x05, 0x05, 0x01, 0x01, 0x01, 0x01, 0x05, 0x01,
/*00D800:*/ 0x01, 0x01, 0x05, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x05, 0x01, 0x01,
/*00D810:*/ 0x05, 0x05, 0x05, 0x01, 0x01, 0x05, 0x01, 0x01, 0x04,
0x01, 0x01, 0x01, 0x05, 0x05, 0x01, 0x05,
/*00D820:*/ 0x01, 0x04, 0x05, 0x01, 0x01, 0x01, 0x04, 0x01, 0x04,
0x01, 0x01, 0x01, 0x01, 0x01, 0x04, 0x01,
/*00D830:*/ 0x01, 0x01, 0x01, 0x05, 0x01, 0x04, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00D840:*/ 0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x04,
0x01, 0x01, 0x01, 0x04, 0x04, 0x01, 0x01,
/*00D850:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x05, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00D860:*/ 0x04, 0x01, 0x05, 0x01, 0x05, 0x01, 0x05, 0x01, 0x01,
0x05, 0x05, 0x01, 0x01, 0x01, 0x01, 0x05,
/*00D870:*/ 0x01, 0x01, 0x01, 0x01, 0x04, 0x04, 0x01, 0x05, 0x01,
0x01, 0x05, 0x01, 0x04, 0x01, 0x04, 0x04,
/*00D880:*/ 0x04, 0x05, 0x04, 0x01, 0x01, 0x01, 0x01, 0x05, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00D890:*/ 0x01, 0x01, 0x05, 0x01, 0x05, 0x05, 0x04, 0x01, 0x01,
0x01, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01,
/*00D8A0:*/ 0x05, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01,
0x04, 0x04, 0x04, 0x01, 0x01, 0x05, 0x01,
/*00D8B0:*/ 0x01, 0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*00D8C0:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B8D2[] = {
/*00E128:*/ 0x04, 0x01, 0x01, 0x01, 0x01, 0x05, 0x04, 0x05,
/*00E130:*/ 0x01, 0x01, 0x04, 0x01, 0x05, 0x01, 0x01, 0x01, 0x04,
0x01, 0x01, 0x04, 0x01, 0x05, 0x01, 0x01,
/*00E140:*/ 0x01, 0x01, 0x01, 0x01, 0x05, 0x04, 0x05, 0x04, 0x01,
0x01, 0x05, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00E150:*/ 0x01, 0x01, 0x05, 0x01, 0x05, 0x01, 0x04, 0x01, 0x01,
0x01, 0x04, 0x01, 0x01, 0x01, 0x04, 0x01,
/*00E160:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01,
0x05, 0x01, 0x01, 0x01, 0x01, 0x05, 0x05,
/*00E170:*/ 0x01, 0x01, 0x05, 0x01, 0x01, 0x04, 0x01, 0x04, 0x01,
0x01, 0x04, 0x01, 0x01, 0x01, 0x04, 0x05,
/*00E180:*/ 0x04, 0x05, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x04, 0x04, 0x04, 0x04,
/*00E190:*/ 0x01, 0x01, 0x01, 0x01, 0x04, 0x04, 0x05, 0x04, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00E1A0:*/ 0x05, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x04,
0x01, 0x01, 0x04, 0x04, 0x01, 0x01, 0x01,
/*00E1B0:*/ 0x01, 0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x04, 0x05, 0x01, 0x04, 0x04, 0x01, 0x01,
/*00E1C0:*/ 0x01, 0x01, 0x04, 0x01, 0x04, 0x04, 0x01, 0x01, 0x01,
0x05, 0x01, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*00E1D0:*/ 0x0D, 0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*00E1E0:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B8D3[] = {
/*00E9C8:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x05, 0x05, 0x01,
/*00E9D0:*/ 0x01, 0x04, 0x04, 0x04, 0x04, 0x01, 0x05, 0x01, 0x04,
0x04, 0x04, 0x01, 0x01, 0x01, 0x01, 0x04,
/*00E9E0:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x05,
/*00E9F0:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x04,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00EA00:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x04, 0x01, 0x05,
0x04, 0x01, 0x01, 0x01, 0x01, 0x01, 0x04,
/*00EA10:*/ 0x04, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x05, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00EA20:*/ 0x04, 0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01, 0x05,
0x01, 0x01, 0x05, 0x01, 0x04, 0x01, 0x04,
/*00EA30:*/ 0x04, 0x05, 0x04, 0x01, 0x04, 0x01, 0x05, 0x01, 0x05,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00EA40:*/ 0x04, 0x01, 0x01, 0x04, 0x01, 0x05, 0x01, 0x01, 0x01,
0x05, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00EA50:*/ 0x01, 0x01, 0x04, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01,
0x05, 0x01, 0x01, 0x01, 0x01, 0x04, 0x01,
/*00EA60:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*00EA70:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};

BYTE B8D4[] = {
/*00F268:*/ 0x01, 0x01, 0x04, 0x04, 0x05, 0x01, 0x01, 0x01,
/*00F270:*/ 0x04, 0x04, 0x01, 0x04, 0x01, 0x05, 0x01, 0x04, 0x04,
0x04, 0x04, 0x01, 0x01, 0x01, 0x04, 0x01,
/*00F280:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00F290:*/ 0x05, 0x01, 0x01, 0x04, 0x01, 0x01, 0x04, 0x01, 0x01,
0x01, 0x05, 0x05, 0x04, 0x01, 0x01, 0x01,
/*00F2A0:*/ 0x01, 0x01, 0x01, 0x01, 0x05, 0x01, 0x04, 0x01, 0x01,
0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00F2B0:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x05, 0x01, 0x04, 0x01,
0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00F2C0:*/ 0x04, 0x01, 0x01, 0x04, 0x01, 0x01, 0x05, 0x01, 0x05,
0x01, 0x04, 0x04, 0x01, 0x01, 0x01, 0x04,
/*00F2D0:*/ 0x04, 0x05, 0x01, 0x01, 0x05, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00F2E0:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x04, 0x04, 0x05,
0x01, 0x01, 0x01, 0x05, 0x04, 0x05, 0x01,
/*00F2F0:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x05, 0x01, 0x01, 0x01,
0x05, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00F300:*/ 0x04, 0x01, 0x01, 0x05, 0x05, 0x04, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x0D, 0xF0, 0xAD, 0xBA,
/*00F310:*/ 0x0D, 0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*00F320:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B8D5[] = {
/*00FAA8:*/ 0x05, 0x04, 0x01, 0x01, 0x04, 0x04, 0x04, 0x01,
/*00FAB0:*/ 0x01, 0x01, 0x01, 0x05, 0x01, 0x01, 0x04, 0x01, 0x01,
0x04, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00FAC0:*/ 0x05, 0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00FAD0:*/ 0x01, 0x01, 0x05, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x04, 0x01, 0x04,
/*00FAE0:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x04, 0x01,
0x01, 0x01, 0x01, 0x05, 0x01, 0x01, 0x01,
/*00FAF0:*/ 0x05, 0x01, 0x01, 0x05, 0x01, 0x01, 0x01, 0x01, 0x01,
0x04, 0x01, 0x05, 0x05, 0x01, 0x04, 0x01,
/*00FB00:*/ 0x05, 0x05, 0x01, 0x01, 0x01, 0x01, 0x05, 0x01, 0x01,
0x04, 0x01, 0x01, 0x04, 0x01, 0x01, 0x05,
/*00FB10:*/ 0x01, 0x05, 0x04, 0x01, 0x05, 0x01, 0x04, 0x01, 0x01,
0x01, 0x01, 0x05, 0x05, 0x05, 0x01, 0x01,
/*00FB20:*/ 0x05, 0x01, 0x01, 0x05, 0x04, 0x01, 0x01, 0x01, 0x05,
0x01, 0x05, 0x05, 0x01, 0x01, 0x01, 0x01,
/*00FB30:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x05, 0x01, 0x04,
0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*00FB40:*/ 0x0D, 0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*00FB50:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B8D6[] = {
/*00FDC8:*/ 0x04, 0x01, 0x01, 0x04, 0x01, 0x04, 0x01, 0x01,
/*00FDD0:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x05, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00FDE0:*/ 0x01, 0x01, 0x01, 0x01, 0x04, 0x01, 0x01, 0x05, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00FDF0:*/ 0x01, 0x01, 0x05, 0x01, 0x01, 0x05, 0x01, 0x01, 0x01,
0x01, 0x05, 0x04, 0x01, 0x01, 0x04, 0x04,
/*00FE00:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x04, 0x01, 0x01, 0x01, 0x04, 0x01,
/*00FE10:*/ 0x01, 0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x05, 0x04, 0x01, 0x01, 0x01, 0x04, 0x01,
/*00FE20:*/ 0x04, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x04, 0x01, 0x01, 0x01, 0x01, 0x04, 0x01,
/*00FE30:*/ 0x01, 0x01, 0x05, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x05, 0x01,
/*00FE40:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x05, 0x01,
0x04, 0x05, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00FE50:*/ 0x01, 0x01, 0x01, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*00FE60:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B8D7[] = {
/*00FE70:*/ 0x01, 0x04, 0x01, 0x05, 0x01, 0x01, 0x01, 0x01, 0x01,
0x05, 0x01, 0x01, 0x01, 0x05, 0x01, 0x01,
/*00FE80:*/ 0x04, 0x01, 0x01, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00FE90:*/ 0x01, 0x01, 0x01, 0x01, 0x05, 0x01, 0x04, 0x01, 0x04,
0x04, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01,
/*00FEA0:*/ 0x01, 0x01, 0x01, 0x05, 0x01, 0x04, 0x01, 0x04, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00FEB0:*/ 0x01, 0x04, 0x01, 0x05, 0x05, 0x01, 0x05, 0x05, 0x01,
0x01, 0x01, 0x04, 0x05, 0x05, 0x05, 0x05,
/*00FEC0:*/ 0x01, 0x01, 0x01, 0x05, 0x01, 0x01, 0x01, 0x01, 0x05,
0x05, 0x01, 0x04, 0x05, 0x04, 0x05, 0x01,
/*00FED0:*/ 0x01, 0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00FEE0:*/ 0x01, 0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x05, 0x04,
0x05, 0x05, 0x01, 0x01, 0x01, 0x01, 0x04,
/*00FEF0:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x04, 0x01, 0x01,
0x01, 0x01, 0x01, 0x05, 0x04, 0x01, 0x05,
/*00FF00:*/ 0x01, 0x01, 0x01, 0x01, 0x04, 0x01, 0x01, 0x04, 0x01,
0x04, 0x01, 0x05, 0x05, 0x01, 0x01, 0x04,
/*00FF10:*/ 0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x05, 0x01, 0x05,
0x01, 0x01, 0x01, 0x01, 0xF0, 0xAD, 0xBA,
/*00FF20:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B8D8[] = {
/*00FF38:*/ 0x05, 0x01, 0x01, 0x01, 0x01, 0x05, 0x01, 0x01,
/*00FF40:*/ 0x04, 0x05, 0x04, 0x01, 0x04, 0x04, 0x01, 0x01, 0x01,
0x01, 0x05, 0x04, 0x01, 0x05, 0x01, 0x01,
/*00FF50:*/ 0x01, 0x01, 0x05, 0x01, 0x01, 0x01, 0x04, 0x01, 0x04,
0x05, 0x01, 0x01, 0x04, 0x04, 0x05, 0x01,
/*00FF60:*/ 0x05, 0x01, 0x01, 0x01, 0x01, 0x05, 0x01, 0x01, 0x01,
0x05, 0x01, 0x01, 0x04, 0x01, 0x05, 0x01,
/*00FF70:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x05,
0x05, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00FF80:*/ 0x01, 0x01, 0x05, 0x01, 0x05, 0x05, 0x01, 0x01, 0x01,
0x01, 0x01, 0x04, 0x01, 0x01, 0x01, 0x01,
/*00FF90:*/ 0x04, 0x01, 0x04, 0x01, 0x01, 0x04, 0x04, 0x01, 0x01,
0x01, 0x05, 0x01, 0x01, 0x05, 0x01, 0x01,
/*00FFA0:*/ 0x05, 0x01, 0x04, 0x04, 0x01, 0x01, 0x01, 0x04, 0x01,
0x05, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*00FFB0:*/ 0x01, 0x04, 0x01, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01,
0x04, 0x01, 0x04, 0x01, 0x05, 0x01, 0x01,
/*00FFC0:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x05, 0x01, 0x05,
0x05, 0x01, 0x01, 0x0D, 0xF0, 0xAD, 0xBA,
/*00FFD0:*/ 0x0D, 0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA, 0xAB,
0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB,
/*00FFE0:*/ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B8D9[] = {
/*1D30:*/ 0x04, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x04,
0x01, 0x01, 0x01, 0x01, 0x05, 0x01, 0x01,
/*1D40:*/ 0x01, 0x01, 0x01, 0x01, 0x05, 0x04, 0x05, 0x04, 0x01,
0x01, 0x05, 0x05, 0x01, 0x04, 0x01, 0x01,
/*1D50:*/ 0x05, 0x05, 0x01, 0x05, 0x01, 0x01, 0x01, 0x01, 0x05,
0x01, 0x01, 0x01, 0x01, 0x01, 0x05, 0x01,
/*1D60:*/ 0x04, 0x01, 0x05, 0x01, 0x05, 0x01, 0x01, 0x04, 0x01,
0x01, 0x01, 0x01, 0x04, 0x04, 0x01, 0x01,
/*1D70:*/ 0x01, 0x01, 0x04, 0x01, 0x01, 0x04, 0x04, 0x01, 0x01,
0x01, 0x01, 0x04, 0x01, 0x01, 0x01, 0x01,
/*1D80:*/ 0x01, 0x05, 0x01, 0x01, 0x01, 0x01, 0x01, 0x04, 0x01,
0x04, 0x01, 0x01, 0x05, 0x01, 0x01, 0x01,
/*1D90:*/ 0x01, 0x01, 0x05, 0x04, 0x05, 0x05, 0x05, 0x04, 0x04,
0x01, 0x04, 0x05, 0x01, 0x04, 0x04, 0x04,
/*1DA0:*/ 0x01, 0x01, 0x04, 0x01, 0x01, 0x01, 0x05, 0x05, 0x05,
0x01, 0x05, 0x01, 0x01, 0x01, 0x01, 0x01,
/*1DB0:*/ 0x01, 0x01, 0x01, 0x05, 0x01, 0x05, 0x04, 0x01, 0x01,
0x05, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*1DC0:*/ 0x01, 0x05, 0x01, 0x01, 0x01, 0x05, 0x01, 0x01, 0x01,
0x05, 0x04, 0x01, 0x04, 0x01, 0x01, 0x01,
/*1DD0:*/ 0x04, 0x01, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01, 0x01,
0x05, 0x01, 0x01, 0x05, 0x04, 0x01, 0x01,
/*1DE0:*/ 0x01, 0x01, 0x01, 0x05, 0x01, 0x04, 0x01, 0x01, 0x04,
0x01, 0x01, 0x04, 0x01, 0x01, 0x04, 0x01,
/*1DF0:*/ 0x01, 0x01, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA, 0x0D,
0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*1E00:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B8DA[] = {
```

```
/*2750:*/ 0x01, 0x05, 0x01, 0x01, 0x01, 0x01, 0x01, 0x04, 0x01,
0x01, 0x01, 0x01, 0x05, 0x01, 0x01, 0x01,
/*2760:*/ 0x04, 0x01, 0x01, 0x01, 0x04, 0x01, 0x04, 0x05, 0x01,
0x01, 0x05, 0x01, 0x05, 0x04, 0x01, 0x05,
/*2770:*/ 0x04, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*2780:*/ 0x01, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01, 0x04, 0x04,
0x01, 0x01, 0x01, 0x01, 0x05, 0x05, 0x01,
/*2790:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x05,
0x01, 0x05, 0x01, 0x01, 0x01, 0x01, 0x01,
/*27A0:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*27B0:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*27C0:*/ 0x04, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x05, 0x04, 0x04, 0x01, 0x01, 0x01,
/*27D0:*/ 0x04, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x05, 0x04, 0x01, 0x01, 0x05, 0x05,
/*27E0:*/ 0x01, 0x01, 0x01, 0x01, 0x05, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*27F0:*/ 0x01, 0x05, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x05, 0x04, 0x05,
/*2800:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*2810:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B8DB[] = {
/*3040:*/ 0x01, 0x01, 0x01, 0x05, 0x05, 0x05, 0x05, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*3050:*/ 0x01, 0x01, 0x05, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01,
0x04, 0x01, 0x04, 0x04, 0x01, 0x01, 0x01,
/*3060:*/ 0x01, 0x01, 0x01, 0x05, 0x01, 0x05, 0x01, 0x01, 0x01,
0x05, 0x01, 0x01, 0x04, 0x01, 0x04, 0x04,
/*3070:*/ 0x01, 0x01, 0x01, 0x05, 0x01, 0x01, 0x04, 0x01, 0x05,
0x01, 0x01, 0x01, 0x05, 0x01, 0x05, 0x01,
/*3080:*/ 0x04, 0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x05, 0x01, 0x04, 0x01, 0x01,
/*3090:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x05,
0x01, 0x04, 0x01, 0x01, 0x04, 0x01, 0x01,
/*30A0:*/ 0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x05, 0x04, 0x04, 0x01, 0x01, 0x01, 0x01,
/*30B0:*/ 0x05, 0x05, 0x01, 0x01, 0x05, 0x01, 0x01, 0x01, 0x05,
0x01, 0x01, 0x01, 0x04, 0x01, 0x01, 0x05,
/*30C0:*/ 0x04, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x04,
/*30D0:*/ 0x04, 0x01, 0x05, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x05, 0x04, 0x0D, 0xF0, 0xAD, 0xBA,
/*30E0:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};
```

```
BYTE B8DC[] = {
/*3870:*/ 0x01, 0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x05, 0x01,
0x01, 0x01, 0x04, 0x04, 0x01, 0x01, 0x01,
/*3880:*/ 0x04, 0x01, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01, 0x01,
0x04, 0x01, 0x01, 0x01, 0x05, 0x01, 0x01,
/*3890:*/ 0x01, 0x01, 0x04, 0x01, 0x05, 0x05, 0x04, 0x01, 0x01,
0x04, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01,
/*38A0:*/ 0x05, 0x01, 0x04, 0x04, 0x04, 0x01, 0x01, 0x01, 0x01,
0x01, 0x04, 0x01, 0x01, 0x04, 0x01, 0x05,
/*38B0:*/ 0x01, 0x04, 0x05, 0x04, 0x04, 0x01, 0x05, 0x05, 0x04,
0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x04,
/*38C0:*/ 0x01, 0x04, 0x01, 0x04, 0x01, 0x01, 0x04, 0x01, 0x01,
0x01, 0x01, 0x04, 0x01, 0x04, 0x01, 0x01,
/*38D0:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x05, 0x01,
0x05, 0x04, 0x01, 0x05, 0x04, 0x01, 0x05,
/*38E0:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x05,
/*38F0:*/ 0x04, 0x01, 0x05, 0x01, 0x05, 0x01, 0x04, 0x01, 0x01,
0x04, 0x01, 0x01, 0x04, 0x05, 0x01, 0x04,
/*3900:*/ 0x01, 0x05, 0x01, 0x05, 0x01, 0x05, 0xAD, 0xBA, 0x0D,
0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*3910:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B8DD[] = {
/*40B0:*/ 0x01, 0x05, 0x01, 0x01, 0x01, 0x05, 0x05, 0x01, 0x04,
0x01, 0x01, 0x01, 0x04, 0x01, 0x05, 0x01,
/*40C0:*/ 0x01, 0x01, 0x01, 0x05, 0x01, 0x01, 0x01, 0x01, 0x05,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*40D0:*/ 0x01, 0x01, 0x01, 0x01, 0x05, 0x01, 0x05, 0x05, 0x01,
0x05, 0x01, 0x01, 0x01, 0x01, 0x05, 0x01,
/*40E0:*/ 0x04, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x04,
0x05, 0x04, 0x01, 0x01, 0x05, 0x01, 0x01,
/*40F0:*/ 0x01, 0x01, 0x04, 0x05, 0x01, 0x04, 0x05, 0x01, 0x01,
0x01, 0x05, 0x01, 0x01, 0x01, 0x05, 0x05,
/*4100:*/ 0x01, 0x01, 0x01, 0x04, 0x01, 0x05, 0x04, 0x05, 0x01,
0x01, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01,
/*4110:*/ 0x01, 0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x05, 0x01, 0x01,
/*4120:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x05, 0x01,
/*4130:*/ 0x01, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x05, 0x05,
/*4140:*/ 0x01, 0x01, 0x01, 0x05, 0x01, 0x01, 0x01, 0x05, 0x01,
0x01, 0x01, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*4150:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B8DE[] = {
```

```
/*4940:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01,
0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01,
/*4950:*/ 0x01, 0x01, 0x01, 0x04, 0x01, 0x05, 0x01, 0x05, 0x05,
0x05, 0x01, 0x01, 0x05, 0x01, 0x01, 0x05,
/*4960:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x05, 0x05, 0x05, 0x01,
/*4970:*/ 0x01, 0x01, 0x04, 0x01, 0x05, 0x01, 0x01, 0x04, 0x05,
0x01, 0x05, 0x01, 0x04, 0x01, 0x01, 0x01,
/*4980:*/ 0x05, 0x01, 0x05, 0x04, 0x01, 0x04, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x04, 0x04,
/*4990:*/ 0x01, 0x01, 0x04, 0x04, 0x01, 0x05, 0x01, 0x05, 0x05,
0x01, 0x01, 0x01, 0x01, 0x01, 0x04, 0x01,
/*49A0:*/ 0x01, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x04, 0x01, 0x01, 0x05,
/*49B0:*/ 0x01, 0x04, 0x01, 0x01, 0x01, 0x04, 0x01, 0x05, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x05,
/*49C0:*/ 0x01, 0x01, 0x01, 0x05, 0x05, 0x01, 0x04, 0x01, 0x01,
0x01, 0x01, 0x04, 0x04, 0x01, 0x01, 0x01,
/*49D0:*/ 0x05, 0x04, 0x01, 0x01, 0x01, 0x04, 0x01, 0x01, 0x01,
0x01, 0x01, 0x04, 0x01, 0x01, 0x01, 0x01,
/*49E0:*/ 0x01, 0x01, 0x01, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA, 0x0D,
0xF0, 0xAD, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*49F0:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};

BYTE B8DF[] = {
/*5240:*/ 0x01, 0x05, 0x04, 0x01, 0x04, 0x05, 0x01, 0x01, 0x01,
0x05, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
/*5250:*/ 0x01, 0x01, 0x05, 0x04, 0x05, 0x01, 0x01, 0x01, 0x01,
0x05, 0x01, 0x01, 0x04, 0x05, 0x05, 0x01,
/*5260:*/ 0x05, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x05, 0x01,
0x01, 0x05, 0x01, 0x01, 0x04, 0x01, 0x04,
/*5270:*/ 0x01, 0x01, 0x05, 0x01, 0x01, 0x01, 0x05, 0x01, 0x01,
0x01, 0x01, 0x01, 0x05, 0x05, 0x01, 0x01,
/*5280:*/ 0x01, 0x01, 0x01, 0x04, 0x01, 0x04, 0x01, 0x05, 0x01,
0x04, 0x01, 0x01, 0x01, 0x05, 0x01, 0x04,
/*5290:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x05, 0x01, 0x01, 0x05, 0x01,
/*52A0:*/ 0x01, 0x01, 0x01, 0x05, 0x05, 0x01, 0x01, 0x05, 0x01,
0x01, 0x01, 0x04, 0x04, 0x01, 0x01, 0x01,
/*52B0:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x05, 0x05, 0x01, 0x01, 0x05,
/*52C0:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x05, 0x01, 0x05, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x05,
/*52D0:*/ 0x01, 0x01, 0x04, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x04, 0x01, 0x04, 0x01,
/*52E0:*/ 0x01, 0x01, 0x01, 0x01, 0x01, 0x05, 0x04, 0x01, 0x04,
0x01, 0x01, 0xBA, 0x0D, 0xF0, 0xAD, 0xBA,
/*52F0:*/ 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0xAB, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
};
```

```
void* B8Array[] = {
     B8D0,
     B8D1,
     B8D2,
     B8D3,
     B8D4,
     B8D5,
     B8D6,
     B8D7,
     B8D8,
     B8D9,
     B8DA,
     B8DB,
     B8DC,
     B8DD,
     B8DE,
     B8DF
};
-------------------------------------------------------------------
--------------------------------


------------- file core.cpp ---------------------------------------
--------------------------------
///////////////////////////////////////////////////
//     NANOMULATOR
// armadillo 4.20 nanomites core emulator
// written by andreageddon [RET/UIC]
//
// data is in coredata.cpp
///////////////////////////////////////////////////
#include "core.h"

///////////////////////////
// BLOCK 2
//////////////////////////
__declspec(naked) void Block2Func2(void)
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 cmp     dword ptr [ebp+0Ch], 0
                 jnz     short ProcessAddress
                 xor     eax, eax
                 jmp     ZeroQuit
 ProcessAddress:
                 mov     eax, [ebp+8]
                 xor     eax, 0FFFFFFFFh
                 mov     [ebp+8], eax
```

```
CalculusLoop:
                cmp     dword ptr [ebp+10h], 8
                jb      EightBytesAddress
                mov     ecx, [ebp+0Ch]
                xor     edx, edx
                mov     dl, [ecx]
                mov     eax, [ebp+8]
                xor     eax, edx
                and     eax, 0FFh
                mov     ecx, [ebp+8]
                shr     ecx, 8
                mov     edx, dword ptr Block2Func2Data1[eax*4]
                xor     edx, ecx
                mov     [ebp+8], edx
                mov     eax, [ebp+0Ch]
                add     eax, 1
                mov     [ebp+0Ch], eax
                mov     ecx, [ebp+0Ch]
                xor     edx, edx
                mov     dl, [ecx]
                mov     eax, [ebp+8]
                xor     eax, edx
                and     eax, 0FFh
                mov     ecx, [ebp+8]
                shr     ecx, 8
                mov     edx, dword ptr Block2Func2Data1[eax*4]
                xor     edx, ecx
                mov     [ebp+8], edx
                mov     eax, [ebp+0Ch]
                add     eax, 1
                mov     [ebp+0Ch], eax
                mov     ecx, [ebp+0Ch]
                xor     edx, edx
                mov     dl, [ecx]
                mov     eax, [ebp+8]
                xor     eax, edx
                and     eax, 0FFh
                mov     ecx, [ebp+8]
                shr     ecx, 8
                mov     edx, dword ptr Block2Func2Data1[eax*4]
                xor     edx, ecx
                mov     [ebp+8], edx
                mov     eax, [ebp+0Ch]
                add     eax, 1
                mov     [ebp+0Ch], eax
                mov     ecx, [ebp+0Ch]
                xor     edx, edx
                mov     dl, [ecx]
                mov     eax, [ebp+8]
                xor     eax, edx
                and     eax, 0FFh
```

```
mov     ecx, [ebp+8]
shr     ecx, 8
mov     edx, dword ptr Block2Func2Data1[eax*4]
xor     edx, ecx
mov     [ebp+8], edx
mov     eax, [ebp+0Ch]
add     eax, 1
mov     [ebp+0Ch], eax
mov     ecx, [ebp+0Ch]
xor     edx, edx
mov     dl, [ecx]
mov     eax, [ebp+8]
xor     eax, edx
and     eax, 0FFh
mov     ecx, [ebp+8]
shr     ecx, 8
mov     edx, dword ptr Block2Func2Data1[eax*4]
xor     edx, ecx
mov     [ebp+8], edx
mov     eax, [ebp+0Ch]
add     eax, 1
mov     [ebp+0Ch], eax
mov     ecx, [ebp+0Ch]
xor     edx, edx
mov     dl, [ecx]
mov     eax, [ebp+8]
xor     eax, edx
and     eax, 0FFh
mov     ecx, [ebp+8]
shr     ecx, 8
mov     edx, dword ptr Block2Func2Data1[eax*4]
xor     edx, ecx
mov     [ebp+8], edx
mov     eax, [ebp+0Ch]
add     eax, 1
mov     [ebp+0Ch], eax
mov     ecx, [ebp+0Ch]
xor     edx, edx
mov     dl, [ecx]
mov     eax, [ebp+8]
xor     eax, edx
and     eax, 0FFh
mov     ecx, [ebp+8]
shr     ecx, 8
mov     edx, dword ptr Block2Func2Data1[eax*4]
xor     edx, ecx
mov     [ebp+8], edx
mov     eax, [ebp+0Ch]
add     eax, 1
mov     [ebp+0Ch], eax
mov     ecx, [ebp+0Ch]
xor     edx, edx
```

```
                mov     dl, [ecx]
                mov     eax, [ebp+8]
                xor     eax, edx
                and     eax, 0FFh
                mov     ecx, [ebp+8]
                shr     ecx, 8
                mov     edx, dword ptr Block2Func2Data1[eax*4]
                xor     edx, ecx
                mov     [ebp+8], edx
                mov     eax, [ebp+0Ch]
                add     eax, 1
                mov     [ebp+0Ch], eax
                mov     ecx, [ebp+10h]
                sub     ecx, 8
                mov     [ebp+10h], ecx
                jmp     CalculusLoop
 EightBytesAddress:
                cmp     dword ptr [ebp+10h], 0
                jz      short ZeroByteAddr

 FourByteAddress:
                mov     edx, [ebp+0Ch]
                xor     eax, eax
                mov     al, [edx]
                mov     ecx, [ebp+8]
                xor     ecx, eax
                and     ecx, 0FFh
                mov     edx, [ebp+8]
                shr     edx, 8
                mov     eax, dword ptr Block2Func2Data1[ecx*4]
                xor     eax, edx
                mov     [ebp+8], eax
                mov     ecx, [ebp+0Ch]
                add     ecx, 1
                mov     [ebp+0Ch], ecx
                mov     edx, [ebp+10h]
                sub     edx, 1
                mov     [ebp+10h], edx
                cmp     dword ptr [ebp+10h], 0
                jnz     short FourByteAddress
 ZeroByteAddr:
                mov     eax, [ebp+8]
                xor     eax, 0FFFFFFFFh
 ZeroQuit:
                pop     ebp
                retn
    }
}

__declspec(naked) DWORD Block2Func1(DWORD *Address, DWORD Param1,
DWORD Param2)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp+8]
                push    ecx
                mov     edx, [ebp+10h]
                xor     edx, 0FFFFFFFFh
                push    edx
                call    Block2Func2
                add     esp, 0Ch
                xor     eax, 0FFFFFFFFh
                pop     ebp
                retn
    }
}

//////////////////////////
// block 3
//////////////////////////

__declspec(naked) void SubFunc0(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 14h
                mov     eax, 1
                mov     ecx, [ebp+0Ch]
                shl     eax, cl
                sub     eax, 1
                mov     ecx, [ebp+10h]
                shl     eax, cl
                mov     dword ptr [ebp-10h], eax
                mov     ecx, [ebp-10h]
                not     ecx
                mov     edx, [ebp+8]
                and     edx, ecx
                mov     [ebp-4], edx
                mov     eax, 1
                mov     ecx, [ebp+10h]
                shl     eax, cl
                mov     [ebp-0Ch], eax
                mov     ecx, [ebp+0Ch]
                mov     edx, [ebp+10h]
                lea     ecx, [edx+ecx-1]
                mov     eax, 1
                shl     eax, cl
                mov     dword ptr [ebp-14h], eax
```

```
                mov     dword ptr [ebp-8], 0
                jmp     short SubFunc0_1

SubFunc0_2:
                mov     ecx, [ebp-8]
                add     ecx, 1
                mov     [ebp-8], ecx

SubFunc0_1:
                mov     edx, [ebp-8]
                cmp     edx, [ebp+0Ch]
                jge     short SubFunc0_4
                mov     eax, [ebp+8]
                and     eax, [ebp-0Ch]
                test    eax, eax
                jz      short SubFunc0_3
                mov     ecx, [ebp-4]
                or      ecx, [ebp-14h]
                mov     [ebp-4], ecx

SubFunc0_3:
                mov     edx, [ebp-0Ch]
                shl     edx, 1
                mov     [ebp-0Ch], edx
                mov     eax, [ebp-14h]
                shr     eax, 1
                mov     dword ptr [ebp-14h], eax
                jmp     short SubFunc0_2

SubFunc0_4:
                mov     eax, [ebp-4]
                mov     esp, ebp
                pop     ebp
                retn
    }
}

__declspec(naked) void SubFunc1(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 10h
                mov     eax, 1
                mov     ecx, [ebp+0Ch]
                shl     eax, cl
                sub     eax, 1
                mov     [ebp-4], eax
                mov     edx, [ebp-4]
                mov     ecx, [ebp+10h]
                shl     edx, cl
                mov     [ebp-0Ch], edx
                mov     eax, [ebp+8]
                and     eax, [ebp-4]
                mov     dword ptr [ebp-10h], eax
                mov     edx, [ebp+8]
                and     edx, [ebp-0Ch]
                mov     ecx, [ebp+10h]
                shr     edx, cl
                mov     [ebp-8], edx
                mov     eax, [ebp-4]
                or      eax, [ebp-0Ch]
                not     eax
                mov     ecx, [ebp+8]
                and     ecx, eax
                mov     [ebp+8], ecx
                mov     edx, [ebp-10h]
                mov     ecx, [ebp+10h]
                shl     edx, cl
                or      edx, [ebp-8]
                mov     eax, [ebp+8]
                or      eax, edx
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                mov     esp, ebp
                pop     ebp
                retn
     }
}

__declspec(naked) void SubFunc2(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 8
                cmp     dword ptr [ebp+14h], 0
                jge     short SubFunc2_1
                mov     eax, [ebp+14h]
                add     eax, [ebp+0Ch]
                mov     [ebp+14h], eax
 SubFunc2_1:
                mov     edx, 1
                mov     ecx, [ebp+0Ch]
                shl     edx, cl
                sub     edx, 1
                mov     [ebp-8], edx
                mov     eax, [ebp+8]
                mov     ecx, [ebp+10h]
                shr     eax, cl
                and     eax, [ebp-8]
                mov     [ebp-4], eax
                mov     edx, [ebp-4]
                mov     ecx, [ebp+14h]
                shr     edx, cl
                mov     ecx, [ebp+0Ch]
                sub     ecx, [ebp+14h]
                mov     eax, [ebp-4]
                shl     eax, cl
                or      edx, eax
                and     edx, [ebp-8]
                mov     [ebp-4], edx
                mov     edx, [ebp-8]
                mov     ecx, [ebp+10h]
                shl     edx, cl
                not     edx
                mov     eax, [ebp+8]
                and     eax, edx
                mov     [ebp+8], eax
                mov     edx, [ebp-4]
                mov     ecx, [ebp+10h]
                shl     edx, cl
                mov     eax, [ebp+8]
                or      eax, edx
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                mov     esp, ebp
                pop     ebp
                retn
    }
}

__declspec(naked) void SubFunc3(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                cmp     dword ptr [ebp+0Ch], 0
                jge     short SubFunc3_1
                mov     eax, [ebp+0Ch]
                add     eax, 20h
                mov     [ebp+0Ch], eax

 SubFunc3_1:
                mov     eax, [ebp+8]
                mov     ecx, [ebp+0Ch]
                shr     eax, cl
                mov     ecx, 20h
                sub     ecx, [ebp+0Ch]
                mov     edx, [ebp+8]
                shl     edx, cl
                or      eax, edx
                pop     ebp
```

```
            retn
    }
}

__declspec(naked) void Block3Func0(void)
{
    __asm
    {
            push    ebp
            mov     ebp, esp
            mov     eax, [ebp+8]
            xor     eax, 6E4957A8h
            mov     [ebp+8], eax
            push    5
            push    15h
            mov     ecx, [ebp+8]
            push    ecx
            call    SubFunc0
            add     esp, 0Ch
            mov     [ebp+8], eax
            push    0Dh
            push    2
            mov     edx, [ebp+8]
            push    edx
            call    SubFunc1
            add     esp, 0Ch
            mov     [ebp+8], eax
            push    0Ch
            push    9
            mov     eax, [ebp+8]
            push    eax
            call    SubFunc0
            add     esp, 0Ch
            mov     [ebp+8], eax
            mov     ecx, [ebp+8]
            xor     ecx, 4B487412h
            mov     [ebp+8], ecx
            push    5
            push    0Eh
            push    7
            mov     edx, [ebp+8]
            push    edx
            call    SubFunc2
            add     esp, 10h
            mov     [ebp+8], eax
            push    13h
            push    8
            mov     eax, [ebp+8]
            push    eax
            call    SubFunc1
            add     esp, 0Ch
            mov     [ebp+8], eax
```

```
                        mov     ecx, [ebp+8]
                        xor     ecx, 0D972B853h
                        mov     [ebp+8], ecx
                        push    7
                        push    16h
                        mov     edx, [ebp+8]
                        push    edx
                        call    SubFunc0
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        mov     eax, [ebp+8]
                        pop     ebp
                        retn
            }
        }

        __declspec(naked) void  Block3Func1(void)
        {
            __asm
            {
                        push    ebp
                        mov     ebp, esp
                        mov     eax, [ebp+8]
                        xor     eax, 0FB1E52AFh
                        mov     [ebp+8], eax
                        push    1Ah
                        push    5
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    SubFunc0
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    16h
                        push    0
                        push    1Ch
                        mov     edx, [ebp+8]
                        push    edx
                        call    SubFunc2
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    0Ah
                        push    0Ah
                        mov     eax, [ebp+8]
                        push    eax
                        call    SubFunc0
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    0Dh
                        push    2
                        push    14h
                        mov     ecx, [ebp+8]
                        push    ecx
```

```
                    call    SubFunc2
                    add     esp, 10h
                    mov     [ebp+8], eax
                    mov     edx, [ebp+8]
                    xor     edx, 8B9D36E9h
                    mov     [ebp+8], edx
                    push    1Ah
                    mov     eax, [ebp+8]
                    push    eax
                    call    SubFunc3
                    add     esp, 8
                    mov     [ebp+8], eax
                    push    7
                    push    14h
                    push    0Bh
                    mov     ecx, [ebp+8]
                    push    ecx
                    call    SubFunc2
                    add     esp, 10h
                    mov     [ebp+8], eax
                    mov     edx, [ebp+8]
                    xor     edx, 0A52B3D68h
                    mov     [ebp+8], edx
                    push    14h
                    push    1
                    mov     eax, [ebp+8]
                    push    eax
                    call    SubFunc1
                    add     esp, 0Ch
                    mov     [ebp+8], eax
                    push    0
                    push    1Eh
                    mov     ecx, [ebp+8]
                    push    ecx
                    call    SubFunc0
                    add     esp, 0Ch
                    mov     [ebp+8], eax
                    push    0Ch
                    mov     edx, [ebp+8]
                    push    edx
                    call    SubFunc3
                    add     esp, 8
                    mov     [ebp+8], eax
                    mov     eax, [ebp+8]
                    xor     eax, 40174D5Bh
                    mov     [ebp+8], eax
                    mov     eax, [ebp+8]
                    pop     ebp
                    retn
        }
    }
```

```
__declspec(naked) void  Block3Func2(void)
{
     __asm
     {
                  push    ebp
                  mov     ebp, esp
                  mov     eax, [ebp+8]
                  xor     eax, 0FBBD38E7h
                  mov     [ebp+8], eax
                  push    17h
                  push    4
                  mov     ecx, [ebp+8]
                  push    ecx
                  call    SubFunc1
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    11h
                  mov     edx, [ebp+8]
                  push    edx
                  call    SubFunc3
                  add     esp, 8
                  mov     [ebp+8], eax
                  push    6
                  push    9
                  push    0Eh
                  mov     eax, [ebp+8]
                  push    eax
                  call    SubFunc2
                  add     esp, 10h
                  mov     [ebp+8], eax
                  push    1Ah
                  mov     ecx, [ebp+8]
                  push    ecx
                  call    SubFunc3
                  add     esp, 8
                  mov     [ebp+8], eax
                  mov     edx, [ebp+8]
                  xor     edx, 81A5E699h
                  mov     [ebp+8], edx
                  push    0Bh
                  push    0
                  push    18h
                  mov     eax, [ebp+8]
                  push    eax
                  call    SubFunc2
                  add     esp, 10h
                  mov     [ebp+8], eax
                  push    0
                  push    11h
                  mov     ecx, [ebp+8]
                  push    ecx
                  call    SubFunc0
```

```
add     esp, 0Ch
mov     [ebp+8], eax
push    0Eh
push    8
mov     edx, [ebp+8]
push    edx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    9
push    1
push    1Eh
mov     eax, [ebp+8]
push    eax
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    9
push    2
mov     ecx, [ebp+8]
push    ecx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    4
push    17h
mov     edx, [ebp+8]
push    edx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    0Fh
mov     eax, [ebp+8]
push    eax
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    2
push    0Fh
push    9
mov     ecx, [ebp+8]
push    ecx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 0F185A47Ch
mov     [ebp+8], edx
push    0Fh
mov     eax, [ebp+8]
push    eax
call    SubFunc3
```

```
                add     esp, 8
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                pop     ebp
                retn
    }
}

__declspec(naked) void  Block3Func3(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                mov     eax, [ebp+8]
                xor     eax, 5C2ACD4Dh
                mov     [ebp+8], eax
                push    0Ah
                push    3
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    6
                push    0Ah
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc0
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    5
                push    7
                push    11h
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc2
                add     esp, 10h
                mov     [ebp+8], eax
                push    0Bh
                push    7
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    7
                push    8
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc0
                add     esp, 0Ch
```

```
                        mov     [ebp+8], eax
                        push    7
                        push    0Ah
                        push    0Bh
                        mov     eax, [ebp+8]
                        push    eax
                        call    SubFunc2
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    16h
                        push    2
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    SubFunc1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    5
                        push    7
                        push    8
                        mov     edx, [ebp+8]
                        push    edx
                        call    SubFunc2
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    0Fh
                        push    5
                        mov     eax, [ebp+8]
                        push    eax
                        call    SubFunc1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    12h
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    SubFunc3
                        add     esp, 8
                        mov     [ebp+8], eax
                        push    6
                        push    3
                        push    0Dh
                        mov     edx, [ebp+8]
                        push    edx
                        call    SubFunc2
                        add     esp, 10h
                        mov     [ebp+8], eax
                        mov     eax, [ebp+8]
                        pop     ebp
                        retn
    }
}

__declspec(naked) void  Block3Func4(void)
```

```
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                mov     eax, [ebp+8]
                xor     eax, 47D8FFBDh
                mov     [ebp+8], eax
                push    4
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc3
                add     esp, 8
                mov     [ebp+8], eax
                push    17h
                push    0
                push    1Ch
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc2
                add     esp, 10h
                mov     [ebp+8], eax
                push    7
                push    0Fh
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc0
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    13h
                push    5
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    1Bh
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc3
                add     esp, 8
                mov     [ebp+8], eax
                push    3
                push    0Dh
                push    8
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc2
                add     esp, 10h
                mov     [ebp+8], eax
                push    0Ah
                push    7
```

```
mov     ecx, [ebp+8]
push    ecx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    0
push    1Fh
mov     edx, [ebp+8]
push    edx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    14h
push    1
mov     eax, [ebp+8]
push    eax
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    6
push    12h
mov     ecx, [ebp+8]
push    ecx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    12h
push    5
mov     edx, [ebp+8]
push    edx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    15h
mov     eax, [ebp+8]
push    eax
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    15h
push    4
mov     ecx, [ebp+8]
push    ecx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    3
push    0Fh
mov     edx, [ebp+8]
push    edx
call    SubFunc0
add     esp, 0Ch
```

```
                        mov     [ebp+8], eax
                        push    9
                        push    8
                        mov     eax, [ebp+8]
                        push    eax
                        call    SubFunc1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        mov     ecx, [ebp+8]
                        xor     ecx, 89C7C9A6h
                        mov     [ebp+8], ecx
                        push    0Eh
                        push    1
                        mov     edx, [ebp+8]
                        push    edx
                        call    SubFunc1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    0Ch
                        push    0Fh
                        mov     eax, [ebp+8]
                        push    eax
                        call    SubFunc0
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    13h
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    SubFunc3
                        add     esp, 8
                        mov     [ebp+8], eax
                        push    5
                        push    7
                        push    17h
                        mov     edx, [ebp+8]
                        push    edx
                        call    SubFunc2
                        add     esp, 10h
                        mov     [ebp+8], eax
                        mov     eax, [ebp+8]
                        pop     ebp
                        retn
     }
}

__declspec(naked) void  Block3Func5(void)
{
     __asm
     {
                        push    ebp
                        mov     ebp, esp
                        mov     eax, [ebp+8]
```

```
xor     eax, 24A4A3B5h
mov     [ebp+8], eax
push    17h
push    8
mov     ecx, [ebp+8]
push    ecx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    10h
push    2
mov     edx, [ebp+8]
push    edx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    13h
push    0
push    1Fh
mov     eax, [ebp+8]
push    eax
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 8B465658h
mov     [ebp+8], ecx
push    0Eh
mov     edx, [ebp+8]
push    edx
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
mov     eax, [ebp+8]
xor     eax, 5BFB6B28h
mov     [ebp+8], eax
push    0Eh
push    1
mov     ecx, [ebp+8]
push    ecx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    5
push    1
push    1Eh
mov     edx, [ebp+8]
push    edx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    5
```

```
mov     eax, [ebp+8]
push    eax
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    1Bh
push    0
push    1Ch
mov     ecx, [ebp+8]
push    ecx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    0Bh
push    1
mov     edx, [ebp+8]
push    edx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    9
push    0
push    0Dh
mov     eax, [ebp+8]
push    eax
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    14h
push    4
mov     ecx, [ebp+8]
push    ecx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    15h
mov     edx, [ebp+8]
push    edx
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
mov     eax, [ebp+8]
xor     eax, 7C3547F7h
mov     [ebp+8], eax
push    0Bh
push    4
mov     ecx, [ebp+8]
push    ecx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    4
```

```
                push    0Bh
                push    6
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc2
                add     esp, 10h
                mov     [ebp+8], eax
                push    0Ch
                push    1
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    6
                push    3
                push    16h
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc2
                add     esp, 10h
                mov     [ebp+8], eax
                push    0
                push    1Fh
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc0
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                pop     ebp
                retn
     }
}

__declspec(naked) void  Block3Func6(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                mov     eax, [ebp+8]
                xor     eax, 34473F96h
                mov     [ebp+8], eax
                push    11h
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc3
                add     esp, 8
                mov     [ebp+8], eax
                mov     edx, [ebp+8]
                xor     edx, 5ED8936Ch
```

```
mov     [ebp+8], edx
push    0Ch
push    8
mov     eax, [ebp+8]
push    eax
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    0Ah
push    12h
mov     ecx, [ebp+8]
push    ecx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 6BA330BFh
mov     [ebp+8], edx
push    4
push    8
push    13h
mov     eax, [ebp+8]
push    eax
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    0Eh
mov     ecx, [ebp+8]
push    ecx
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    0Bh
push    4
push    19h
mov     edx, [ebp+8]
push    edx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    3
mov     eax, [ebp+8]
push    eax
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 251AFCFEh
mov     [ebp+8], ecx
push    0Fh
mov     edx, [ebp+8]
push    edx
```

```
                call    SubFunc3
                add     esp, 8
                mov     [ebp+8], eax
                push    0Dh
                push    0Bh
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc0
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                pop     ebp
                retn
     }
}

__declspec(naked) void  Block3Func7(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                mov     eax, [ebp+8]
                xor     eax, 5A07B2A1h
                mov     [ebp+8], eax
                push    10h
                push    0Bh
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc0
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0Bh
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc3
                add     esp, 8
                mov     [ebp+8], eax
                push    0Dh
                push    6
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc1
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     ecx, [ebp+8]
                xor     ecx, 70648B0Dh
                mov     [ebp+8], ecx
                push    3
                push    11h
                mov     edx, [ebp+8]
                push    edx
```

```
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
mov     eax, [ebp+8]
xor     eax, 3773D297h
mov     [ebp+8], eax
push    0
push    1Eh
mov     ecx, [ebp+8]
push    ecx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 49253F31h
mov     [ebp+8], edx
push    0Fh
push    2
mov     eax, [ebp+8]
push    eax
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    10h
push    0
push    1Fh
mov     ecx, [ebp+8]
push    ecx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    11h
push    5
mov     edx, [ebp+8]
push    edx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    11h
push    9
mov     eax, [ebp+8]
push    eax
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 0ABA4CE9Ah
mov     [ebp+8], ecx
push    0Dh
mov     edx, [ebp+8]
push    edx
call    SubFunc3
```

```
                  add     esp, 8
                  mov     [ebp+8], eax
                  mov     eax, [ebp+8]
                  xor     eax, 0F6547803h
                  mov     [ebp+8], eax
                  push    13h
                  push    4
                  mov     ecx, [ebp+8]
                  push    ecx
                  call    SubFunc0
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  mov     edx, [ebp+8]
                  xor     edx, 3A22CD09h
                  mov     [ebp+8], edx
                  push    4
                  mov     eax, [ebp+8]
                  push    eax
                  call    SubFunc3
                  add     esp, 8
                  mov     [ebp+8], eax
                  push    5
                  push    0Ah
                  push    0Fh
                  mov     ecx, [ebp+8]
                  push    ecx
                  call    SubFunc2
                  add     esp, 10h
                  mov     [ebp+8], eax
                  push    1Fh
                  mov     edx, [ebp+8]
                  push    edx
                  call    SubFunc3
                  add     esp, 8
                  mov     [ebp+8], eax
                  push    0
                  push    7
                  mov     eax, [ebp+8]
                  push    eax
                  call    SubFunc0
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  mov     eax, [ebp+8]
                  pop     ebp
                  retn
     }
}

__declspec(naked) void  Block3Func8(void)
{
     __asm
     {
```

```
push    ebp
mov     ebp, esp
mov     eax, [ebp+8]
xor     eax, 949D26F0h
mov     [ebp+8], eax
push    7
push    9
push    10h
mov     ecx, [ebp+8]
push    ecx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 58887477h
mov     [ebp+8], edx
push    5
push    5
mov     eax, [ebp+8]
push    eax
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    1Dh
mov     ecx, [ebp+8]
push    ecx
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    0Ch
push    6
push    0Fh
mov     edx, [ebp+8]
push    edx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    15h
mov     eax, [ebp+8]
push    eax
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 0A8DB534Eh
mov     [ebp+8], ecx
push    10h
push    5
mov     edx, [ebp+8]
push    edx
call    SubFunc1
add     esp, 0Ch
```

```
                        mov     [ebp+8], eax
                        push    5
                        push    19h
                        mov     eax, [ebp+8]
                        push    eax
                        call    SubFunc0
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    0Bh
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    SubFunc3
                        add     esp, 8
                        mov     [ebp+8], eax
                        mov     eax, [ebp+8]
                        pop     ebp
                        retn

    }
}

__declspec(naked) void  Block3Func9(void)
{
    __asm
    {
                        push    ebp
                        mov     ebp, esp
                        mov     eax, [ebp+8]
                        xor     eax, 0EECED199h
                        mov     [ebp+8], eax
                        push    6
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    SubFunc3
                        add     esp, 8
                        mov     [ebp+8], eax
                        push    2
                        push    17h
                        push    3
                        mov     edx, [ebp+8]
                        push    edx
                        call    SubFunc2
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    0Eh
                        push    0Fh
                        mov     eax, [ebp+8]
                        push    eax
                        call    SubFunc0
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    0Eh
                        push    2
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    SubFunc1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        mov     edx, [ebp+8]
                        xor     edx, 0F097965Dh
                        mov     [ebp+8], edx
                        push    15h
                        mov     eax, [ebp+8]
                        push    eax
                        call    SubFunc3
                        add     esp, 8
                        mov     [ebp+8], eax
                        mov     ecx, [ebp+8]
                        xor     ecx, 247C11F6h
                        mov     [ebp+8], ecx
                        push    14h
                        push    1
                        mov     edx, [ebp+8]
                        push    edx
                        call    SubFunc1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    0Ah
                        mov     eax, [ebp+8]
                        push    eax
                        call    SubFunc3
                        add     esp, 8
                        mov     [ebp+8], eax
                        push    19h
                        push    1
                        push    1Eh
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    SubFunc2
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    0Bh
                        push    8
                        mov     edx, [ebp+8]
                        push    edx
                        call    SubFunc1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    2
                        push    1Dh
                        mov     eax, [ebp+8]
                        push    eax
                        call    SubFunc0
                        add     esp, 0Ch
```

```
                mov     [ebp+8], eax
                mov     ecx, [ebp+8]
                xor     ecx, 71B455A8h
                mov     [ebp+8], ecx
                push    0Eh
                push    2
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc1
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                pop     ebp
                retn
    }
}

__declspec(naked) void  Block3FuncA(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                mov     eax, [ebp+8]
                xor     eax, 0FDEB38C7h
                mov     [ebp+8], eax
                push    1
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc3
                add     esp, 8
                mov     [ebp+8], eax
                push    2
                push    8
                push    4
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc2
                add     esp, 10h
                mov     [ebp+8], eax
                push    0Ch
                push    7
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    10h
                push    7
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc0
```

```
add     esp, 0Ch
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 0DCA20F48h
mov     [ebp+8], edx
push    16h
push    8
mov     eax, [ebp+8]
push    eax
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    7
push    4
push    1Ah
mov     ecx, [ebp+8]
push    ecx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    0Fh
mov     edx, [ebp+8]
push    edx
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    2
push    7
push    5
mov     eax, [ebp+8]
push    eax
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    3
push    1Bh
mov     ecx, [ebp+8]
push    ecx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 4A7BFE98h
mov     [ebp+8], edx
push    8
push    15h
mov     eax, [ebp+8]
push    eax
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
mov     ecx, [ebp+8]
```

```
                xor     ecx, 71912DB8h
                mov     [ebp+8], ecx
                push    9
                push    1
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    9
                push    8
                push    17h
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc2
                add     esp, 10h
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                pop     ebp
                retn
    }
}

__declspec(naked) void  Block3FuncB(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                mov     eax, [ebp+8]
                xor     eax, 3F4C93CAh
                mov     [ebp+8], eax
                push    12h
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc3
                add     esp, 8
                mov     [ebp+8], eax
                push    0Ah
                push    9
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc0
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                xor     eax, 16F12999h
                mov     [ebp+8], eax
                push    3
                push    4
                mov     ecx, [ebp+8]
                push    ecx
```

```
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    4
push    5
push    0Bh
mov     edx, [ebp+8]
push    edx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    1
push    18h
mov     eax, [ebp+8]
push    eax
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    15h
push    2
mov     ecx, [ebp+8]
push    ecx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    9
push    5
push    18h
mov     edx, [ebp+8]
push    edx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    3
push    12h
mov     eax, [ebp+8]
push    eax
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    0Bh
mov     ecx, [ebp+8]
push    ecx
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    0Eh
push    8
mov     edx, [ebp+8]
push    edx
call    SubFunc1
add     esp, 0Ch
```

```
                        mov     [ebp+8], eax
                        push    1
                        push    1Bh
                        mov     eax, [ebp+8]
                        push    eax
                        call    SubFunc0
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    8
                        push    5
                        push    16h
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    SubFunc2
                        add     esp, 10h
                        mov     [ebp+8], eax
                        mov     edx, [ebp+8]
                        xor     edx, 0A59EEE9Ah
                        mov     [ebp+8], edx
                        push    0Eh
                        mov     eax, [ebp+8]
                        push    eax
                        call    SubFunc3
                        add     esp, 8
                        mov     [ebp+8], eax
                        push    19h
                        push    3
                        push    1Bh
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    SubFunc2
                        add     esp, 10h
                        mov     [ebp+8], eax
                        mov     eax, [ebp+8]
                        pop     ebp
                        retn
     }
}

__declspec(naked) void  Block3FuncC(void)
{
     __asm
     {
                        push    ebp
                        mov     ebp, esp
                        mov     eax, [ebp+8]
                        xor     eax, 10AFC3E4h
                        mov     [ebp+8], eax
                        push    8
                        push    6
                        push    0Bh
                        mov     ecx, [ebp+8]
```

```
push    ecx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    4
mov     edx, [ebp+8]
push    edx
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    2
push    0Ah
push    8
mov     eax, [ebp+8]
push    eax
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 8C6C2052h
mov     [ebp+8], ecx
push    0Bh
push    3
mov     edx, [ebp+8]
push    edx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    2
push    3
push    7
mov     eax, [ebp+8]
push    eax
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    0Eh
push    6
mov     ecx, [ebp+8]
push    ecx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    19h
push    2
mov     edx, [ebp+8]
push    edx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
mov     eax, [ebp+8]
xor     eax, 2ACA701Ah
```

```
                 mov     [ebp+8], eax
                 push    0Ah
                 push    4
                 mov     ecx, [ebp+8]
                 push    ecx
                 call    SubFunc1
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 push    9
                 push    2
                 push    0Eh
                 mov     edx, [ebp+8]
                 push    edx
                 call    SubFunc2
                 add     esp, 10h
                 mov     [ebp+8], eax
                 push    0Ch
                 push    6
                 mov     eax, [ebp+8]
                 push    eax
                 call    SubFunc1
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 mov     eax, [ebp+8]
                 pop     ebp
                 retn
     }
}

__declspec(naked) void  Block3FuncD(void)
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 mov     eax, [ebp+8]
                 xor     eax, 96174FB5h
                 mov     [ebp+8], eax
                 push    0Ch
                 push    7
                 mov     ecx, [ebp+8]
                 push    ecx
                 call    SubFunc1
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 push    0Ch
                 push    0Dh
                 mov     edx, [ebp+8]
                 push    edx
                 call    SubFunc0
                 add     esp, 0Ch
                 mov     [ebp+8], eax
```

```
push    14h
mov     eax, [ebp+8]
push    eax
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    0
push    0Fh
mov     ecx, [ebp+8]
push    ecx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    17h
push    4
mov     edx, [ebp+8]
push    edx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    6
push    1
push    14h
mov     eax, [ebp+8]
push    eax
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 0B4324752h
mov     [ebp+8], ecx
push    0Ch
push    3
mov     edx, [ebp+8]
push    edx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    1Fh
mov     eax, [ebp+8]
push    eax
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    6
push    19h
mov     ecx, [ebp+8]
push    ecx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    5
push    8
push    10h
mov     edx, [ebp+8]
push    edx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    0Dh
push    7
mov     eax, [ebp+8]
push    eax
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 414B8E93h
mov     [ebp+8], ecx
push    16h
push    0
push    1Dh
mov     edx, [ebp+8]
push    edx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    12h
push    0Bh
mov     eax, [ebp+8]
push    eax
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    14h
push    3
push    1Bh
mov     ecx, [ebp+8]
push    ecx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    18h
mov     edx, [ebp+8]
push    edx
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    7
push    0Eh
mov     eax, [ebp+8]
push    eax
call    SubFunc0
add     esp, 0Ch
```

```
                  mov     [ebp+8], eax
                  push    1Bh
                  mov     ecx, [ebp+8]
                  push    ecx
                  call    SubFunc3
                  add     esp, 8
                  mov     [ebp+8], eax
                  mov     eax, [ebp+8]
                  pop     ebp
                  retn
     }
}

__declspec(naked) void  Block3FuncE(void)
{
     __asm
     {
                  push    ebp
                  mov     ebp, esp
                  mov     eax, [ebp+8]
                  xor     eax, 8B8BAF82h
                  mov     [ebp+8], eax
                  push    16h
                  push    7
                  mov     ecx, [ebp+8]
                  push    ecx
                  call    SubFunc1
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    11h
                  push    0Bh
                  mov     edx, [ebp+8]
                  push    edx
                  call    SubFunc0
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    14h
                  push    5
                  mov     eax, [ebp+8]
                  push    eax
                  call    SubFunc1
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    4
                  push    13h
                  push    0Bh
                  mov     ecx, [ebp+8]
                  push    ecx
                  call    SubFunc2
                  add     esp, 10h
                  mov     [ebp+8], eax
                  mov     edx, [ebp+8]
```

```
xor     edx, 0DDF185A2h
mov     [ebp+8], edx
push    5
push    9
push    12h
mov     eax, [ebp+8]
push    eax
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    0Eh
push    10h
mov     ecx, [ebp+8]
push    ecx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    16h
push    8
mov     edx, [ebp+8]
push    edx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    2
push    1
push    7
mov     eax, [ebp+8]
push    eax
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    14h
mov     ecx, [ebp+8]
push    ecx
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    2
push    0Bh
mov     edx, [ebp+8]
push    edx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    0Bh
mov     eax, [ebp+8]
push    eax
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    3
```

```
                push    4
                push    15h
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc2
                add     esp, 10h
                mov     [ebp+8], eax
                push    15h
                push    7
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc1
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                pop     ebp
                retn
    }
}

__declspec(naked) void  Block3FuncF(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                mov     eax, [ebp+8]
                xor     eax, 6760502h
                mov     [ebp+8], eax
                push    13h
                push    7
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc1
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     edx, [ebp+8]
                xor     edx, 17019638h
                mov     [ebp+8], edx
                push    4
                push    2
                push    0Eh
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc2
                add     esp, 10h
                mov     [ebp+8], eax
                push    5
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc3
                add     esp, 8
```

```
mov     [ebp+8], eax
push    1Ah
push    4
mov     edx, [ebp+8]
push    edx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    16h
push    6
mov     eax, [ebp+8]
push    eax
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    7
push    8
push    17h
mov     ecx, [ebp+8]
push    ecx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    1
push    1Dh
mov     edx, [ebp+8]
push    edx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    4
push    5
push    17h
mov     eax, [ebp+8]
push    eax
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    12h
mov     ecx, [ebp+8]
push    ecx
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 87E29F3Ch
mov     [ebp+8], edx
push    1
push    1Ah
mov     eax, [ebp+8]
push    eax
call    SubFunc0
```

```
add     esp, 0Ch
mov     [ebp+8], eax
push    9
push    2
mov     ecx, [ebp+8]
push    ecx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    0Bh
push    0
push    1Fh
mov     edx, [ebp+8]
push    edx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    3
push    15h
mov     eax, [ebp+8]
push    eax
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 1AD7F4D0h
mov     [ebp+8], ecx
push    0Eh
push    6
mov     edx, [ebp+8]
push    edx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    3
push    18h
mov     eax, [ebp+8]
push    eax
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 0552CA9h
mov     [ebp+8], ecx
push    0Ah
push    0
push    12h
mov     edx, [ebp+8]
push    edx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
```

```
                    push    2
                    push    1Bh
                    mov     eax, [ebp+8]
                    push    eax
                    call    SubFunc0
                    add     esp, 0Ch
                    mov     [ebp+8], eax
                    mov     eax, [ebp+8]
                    pop     ebp
                    retn
        }
}


////////////////////////////////////////
// block 6
////////////////////////////////////////

//
// called by the ASM dispatcher, this routine finds
// the address where the execution will be routed
//
DWORD Dispatcher(DWORD Address)
{
        int I = 0;

        while(true)
        {
                if(Address == B6OriginalFuncs[i])
                {
                        return (DWORD)Block6DynamicFuncs[i];
                }
                i++;
        }
        //
        // we should never get here
        //
        return NULL;
}


//
// static routines will dispatch their dynamic calls towards this
// ASM routine. Dynamic calls originally were  CALL [EBP - 8],
that
// is the address to be checked in the lookaside arrays
//
__declspec(naked) void AsmDispatcher(void)
{
        __asm
        {
                mov       eax, dword ptr [ebp - 8]
                push eax
                call Dispatcher
                add     esp, 4
                jmp     eax
                retn    //we should never get here
        }
}



__declspec(naked) void sub_48DC2D(void) {   __asm    {





             push    ebp
             mov     ebp, esp
             sub     esp, 40h
             mov     dword ptr [ebp-30h], 5Eh
             mov     dword ptr [ebp-2Ch], 0Bh
             mov     dword ptr [ebp-28h], 0D5h
             mov     dword ptr [ebp-24h], 60h
             mov     dword ptr [ebp-20h], 0E5h
             mov     dword ptr [ebp-1Ch], 0E8h
             mov     dword ptr [ebp-18h], 0C5h
             mov     dword ptr [ebp-14h], 5Eh
             mov     dword ptr [ebp-10h], 0Fh
             mov     dword ptr [ebp-40h], 7
             mov     eax, [ebp+8]
             shr     eax, 0Fh
             and     eax, 7
             mov     ecx, [ebp+eax*4-30h]
             mov     [ebp-3Ch], ecx
             mov     eax, [ebp-3Ch]
             cdq
             and     edx, 0Fh
             add     eax, edx
             sar     eax, 4
             mov     [ebp-34h], eax
             mov     edx, [ebp-3Ch]
             and     edx, 8000000Fh
             jns     short loc_48DCA8
             dec     edx
             or      edx, 0FFFFFFF0h
             inc     edx

loc_48DCA8:
             mov     [ebp-38h], edx
             mov     eax, [ebp-34h]
             cmp     eax, [ebp-38h]
             jnz     short loc_48DCC9
```

```
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48DCC6
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48DCC6:
                mov     [ebp-38h], ecx

loc_48DCC9:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_48DD35(void) {  __asm  {
```

```
push    ebp
mov     ebp, esp
sub     esp, 40h
mov     dword ptr [ebp-30h], 0E3h
mov     dword ptr [ebp-2Ch], 0F9h
mov     dword ptr [ebp-28h], 54h
mov     dword ptr [ebp-24h], 90h
mov     dword ptr [ebp-20h], 15h
mov     dword ptr [ebp-1Ch], 0BFh
mov     dword ptr [ebp-18h], 35h
mov     dword ptr [ebp-14h], 27h
mov     dword ptr [ebp-10h], 10h
mov     dword ptr [ebp-40h], 7
mov     eax, [ebp+8]
shr     eax, 10h
and     eax, 7
mov     ecx, [ebp+eax*4-30h]
mov     [ebp-3Ch], ecx
mov     eax, [ebp-3Ch]
cdq
and     edx, 0Fh
add     eax, edx
sar     eax, 4
mov     [ebp-34h], eax
mov     edx, [ebp-3Ch]
and     edx, 8000000Fh
jns     short loc_48DDB0
dec     edx
or      edx, 0FFFFFFF0h
inc     edx
```

```
loc_48DDB0:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48DDD1
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48DDCE
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48DDCE:
                mov     [ebp-38h], ecx

loc_48DDD1:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_48DE3D(void) {  __asm  {
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0F4h
                mov     dword ptr [ebp-2Ch], 7Eh
                mov     dword ptr [ebp-28h], 0C6h
                mov     dword ptr [ebp-24h], 0A1h
                mov     dword ptr [ebp-20h], 5Ah
                mov     dword ptr [ebp-1Ch], 0A6h
                mov     dword ptr [ebp-18h], 6Fh
                mov     dword ptr [ebp-14h], 8Bh
                mov     dword ptr [ebp-10h], 6
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 6
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48DEB8
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48DEB8:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48DED9
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48DED6
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48DED6:
                mov     [ebp-38h], ecx

loc_48DED9:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}


__declspec(naked) void sub_48DF45(void) {  __asm  {

                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 23h
                mov     dword ptr [ebp-2Ch], 1Dh
                mov     dword ptr [ebp-28h], 0AAh
                mov     dword ptr [ebp-24h], 0D3h
                mov     dword ptr [ebp-20h], 0B5h
                mov     dword ptr [ebp-1Ch], 35h
                mov     dword ptr [ebp-18h], 2Eh
                mov     dword ptr [ebp-14h], 6Dh
                mov     dword ptr [ebp-10h], 0Ah
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ah
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48DFC0
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48DFC0:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48DFE1
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48DFDE
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48DFDE:
                mov     [ebp-38h], ecx

loc_48DFE1:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}



__declspec(naked) void sub_48E04D(void) {  __asm  {
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 83h
                mov     dword ptr [ebp-2Ch], 0C7h
                mov     dword ptr [ebp-28h], 2Dh
                mov     dword ptr [ebp-24h], 89h
                mov     dword ptr [ebp-20h], 0D5h
                mov     dword ptr [ebp-1Ch], 0A3h
                mov     dword ptr [ebp-18h], 39h
                mov     dword ptr [ebp-14h], 0DAh
                mov     dword ptr [ebp-10h], 0Ah
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ah
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48E0C8
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48E0C8:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48E0E9
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48E0E6
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48E0E6:
                mov     [ebp-38h], ecx

loc_48E0E9:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}


__declspec(naked) void sub_48E155(void) {  __asm  {

                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0F1h
                mov     dword ptr [ebp-2Ch], 0Eh
                mov     dword ptr [ebp-28h], 15h
                mov     dword ptr [ebp-24h], 0F6h
                mov     dword ptr [ebp-20h], 0D5h
                mov     dword ptr [ebp-1Ch], 0A6h
                mov     dword ptr [ebp-18h], 5Dh
                mov     dword ptr [ebp-14h], 2Ch
                mov     dword ptr [ebp-10h], 2
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 2
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48E1D0
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48E1D0:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48E1F1
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48E1EE
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48E1EE:
                mov     [ebp-38h], ecx

loc_48E1F1:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_48E25D(void) {  __asm  {

                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 17h
                mov     dword ptr [ebp-2Ch], 0A0h
                mov     dword ptr [ebp-28h], 36h
                mov     dword ptr [ebp-24h], 7Eh
                mov     dword ptr [ebp-20h], 3Ah
                mov     dword ptr [ebp-1Ch], 0B6h
                mov     dword ptr [ebp-18h], 2Bh
                mov     dword ptr [ebp-14h], 0AEh
                mov     dword ptr [ebp-10h], 1
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 1
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48E2D7
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48E2D7:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48E2F8
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48E2F5
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48E2F5:
                mov     [ebp-38h], ecx

loc_48E2F8:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}

__declspec(naked) void sub_48E364(void) {  __asm  {
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0F0h
                mov     dword ptr [ebp-2Ch], 56h
                mov     dword ptr [ebp-28h], 0D3h
                mov     dword ptr [ebp-24h], 47h
                mov     dword ptr [ebp-20h], 20h
                mov     dword ptr [ebp-1Ch], 5Ah
                mov     dword ptr [ebp-18h], 15h
                mov     dword ptr [ebp-14h], 0DCh
                mov     dword ptr [ebp-10h], 0Fh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Fh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48E3DF
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48E3DF:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48E400
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48E3FD
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48E3FD:
```

```
                mov     [ebp-38h], ecx

loc_48E400:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}


__declspec(naked) void sub_48E46C(void) {  __asm  {
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 6Eh
                mov     dword ptr [ebp-2Ch], 0B1h
                mov     dword ptr [ebp-28h], 0AEh
                mov     dword ptr [ebp-24h], 0E4h
                mov     dword ptr [ebp-20h], 4Dh
                mov     dword ptr [ebp-1Ch], 4
                mov     dword ptr [ebp-18h], 76h
                mov     dword ptr [ebp-14h], 0B8h
                mov     dword ptr [ebp-10h], 3
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 3
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48E4E7
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48E4E7:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48E508
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48E505
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48E505:
                mov     [ebp-38h], ecx

loc_48E508:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}


__declspec(naked) void sub_48E574(void) {  __asm  {
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 58h
                mov     dword ptr [ebp-2Ch], 8Dh
                mov     dword ptr [ebp-28h], 8Bh
                mov     dword ptr [ebp-24h], 44h
                mov     dword ptr [ebp-20h], 64h
                mov     dword ptr [ebp-1Ch], 57h
                mov     dword ptr [ebp-18h], 48h
                mov     dword ptr [ebp-14h], 0DAh
                mov     dword ptr [ebp-10h], 0Ch
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ch
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48E5EF
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48E5EF:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48E610
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48E60D
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48E60D:
                mov     [ebp-38h], ecx

loc_48E610:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}

__declspec(naked) void sub_48E67C(void) {  __asm  {
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 91h
                mov     dword ptr [ebp-2Ch], 24h
                mov     dword ptr [ebp-28h], 3
                mov     dword ptr [ebp-24h], 6
                mov     dword ptr [ebp-20h], 0B7h
                mov     dword ptr [ebp-1Ch], 0B0h
                mov     dword ptr [ebp-18h], 26h
                mov     dword ptr [ebp-14h], 0E3h
                mov     dword ptr [ebp-10h], 1
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 1
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48E6F6
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48E6F6:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48E717
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48E714
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48E714:
                mov     [ebp-38h], ecx

loc_48E717:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}

__declspec(naked) void sub_48E783(void) {  __asm  {

                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 99h
                mov     dword ptr [ebp-2Ch], 71h
                mov     dword ptr [ebp-28h], 8Ah
                mov     dword ptr [ebp-24h], 65h
                mov     dword ptr [ebp-20h], 0A4h
                mov     dword ptr [ebp-1Ch], 4Ah
                mov     dword ptr [ebp-18h], 0D9h
                mov     dword ptr [ebp-14h], 3Fh
                mov     dword ptr [ebp-10h], 13h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 13h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48E7FE
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48E7FE:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48E81F
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48E81C
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48E81C:
                mov     [ebp-38h], ecx

loc_48E81F:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}


__declspec(naked) void sub_48E88B(void) {  __asm  {
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 38h
                mov     dword ptr [ebp-2Ch], 29h
                mov     dword ptr [ebp-28h], 5Ch
                mov     dword ptr [ebp-24h], 7Fh
                mov     dword ptr [ebp-20h], 0E4h
                mov     dword ptr [ebp-1Ch], 60h
                mov     dword ptr [ebp-18h], 3Dh
                mov     dword ptr [ebp-14h], 0B8h
                mov     dword ptr [ebp-10h], 6
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 6
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48E906
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48E906:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48E927
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48E924
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48E924:
                mov     [ebp-38h], ecx
```

```
loc_48E927:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}

__declspec(naked) void sub_48E993(void) {  __asm  {
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0
                mov     dword ptr [ebp-2Ch], 85h
                mov     dword ptr [ebp-28h], 0BAh
                mov     dword ptr [ebp-24h], 39h
                mov     dword ptr [ebp-20h], 0A0h
                mov     dword ptr [ebp-1Ch], 99h
                mov     dword ptr [ebp-18h], 0D9h
                mov     dword ptr [ebp-14h], 47h
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48EA0E
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48EA0E:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48EA2F
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48EA2C
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48EA2C:
                mov     [ebp-38h], ecx

loc_48EA2F:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}


__declspec(naked) void sub_48EA9B(void) {  __asm  {
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0E8h
                mov     dword ptr [ebp-2Ch], 0B7h
                mov     dword ptr [ebp-28h], 64h
                mov     dword ptr [ebp-24h], 0BBh
                mov     dword ptr [ebp-20h], 0ACh
                mov     dword ptr [ebp-1Ch], 0E9h
                mov     dword ptr [ebp-18h], 6Ch
                mov     dword ptr [ebp-14h], 2Eh
                mov     dword ptr [ebp-10h], 0Eh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Eh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48EB16
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48EB16:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48EB37
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48EB34
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48EB34:
                mov     [ebp-38h], ecx

loc_48EB37:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}


__declspec(naked) void sub_48EBA3(void) {  __asm  {

                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0E7h
                mov     dword ptr [ebp-2Ch], 0B4h
                mov     dword ptr [ebp-28h], 71h
                mov     dword ptr [ebp-24h], 21h
                mov     dword ptr [ebp-20h], 9Dh
                mov     dword ptr [ebp-1Ch], 0D3h
                mov     dword ptr [ebp-18h], 4Dh
                mov     dword ptr [ebp-14h], 0Fh
                mov     dword ptr [ebp-10h], 3
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 3
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48EC1E
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48EC1E:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48EC3F
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48EC3C
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48EC3C:
                mov     [ebp-38h], ecx

loc_48EC3F:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}


__declspec(naked) void sub_48ECAB(void) {  __asm  {

                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0E2h
                mov     dword ptr [ebp-2Ch], 71h
                mov     dword ptr [ebp-28h], 48h
                mov     dword ptr [ebp-24h], 22h
                mov     dword ptr [ebp-20h], 42h
                mov     dword ptr [ebp-1Ch], 13h
                mov     dword ptr [ebp-18h], 0B1h
                mov     dword ptr [ebp-14h], 8Bh
                mov     dword ptr [ebp-10h], 0Eh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Eh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48ED26
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48ED26:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48ED47
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48ED44
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48ED44:
                mov     [ebp-38h], ecx

loc_48ED47:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}


__declspec(naked) void sub_48EDB3(void) {  __asm  {
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 8Bh
                mov     dword ptr [ebp-2Ch], 1
                mov     dword ptr [ebp-28h], 43h
                mov     dword ptr [ebp-24h], 4
                mov     dword ptr [ebp-20h], 2Eh
                mov     dword ptr [ebp-1Ch], 0C6h
                mov     dword ptr [ebp-18h], 0A2h
                mov     dword ptr [ebp-14h], 0C9h
                mov     dword ptr [ebp-10h], 2
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 2
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48EE2E
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48EE2E:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48EE4F
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48EE4C
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48EE4C:
                mov     [ebp-38h], ecx
```

```
loc_48EE4F:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}

__declspec(naked) void sub_48EEBB(void) {  __asm  {

                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0C9h
                mov     dword ptr [ebp-2Ch], 4Fh
                mov     dword ptr [ebp-28h], 0E8h
                mov     dword ptr [ebp-24h], 58h
                mov     dword ptr [ebp-20h], 8Eh
                mov     dword ptr [ebp-1Ch], 37h
                mov     dword ptr [ebp-18h], 64h
                mov     dword ptr [ebp-14h], 9Eh
                mov     dword ptr [ebp-10h], 0Eh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Eh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48EF36
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48EF36:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48EF57
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48EF54
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48EF54:
                mov     [ebp-38h], ecx

loc_48EF57:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_48EFC3(void) {  __asm  {













                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0FAh
                mov     dword ptr [ebp-2Ch], 0D3h
                mov     dword ptr [ebp-28h], 8Dh
                mov     dword ptr [ebp-24h], 0DAh
                mov     dword ptr [ebp-20h], 0B9h
                mov     dword ptr [ebp-1Ch], 0E3h
                mov     dword ptr [ebp-18h], 16h
                mov     dword ptr [ebp-14h], 7Fh
                mov     dword ptr [ebp-10h], 10h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 10h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48F03E
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48F03E:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48F05F
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48F05C
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48F05C:
                mov     [ebp-38h], ecx

loc_48F05F:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}



__declspec(naked) void sub_48F0CB(void) {  __asm  {



                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 74h
                mov     dword ptr [ebp-2Ch], 7Fh
                mov     dword ptr [ebp-28h], 21h
                mov     dword ptr [ebp-24h], 77h
                mov     dword ptr [ebp-20h], 0C3h
                mov     dword ptr [ebp-1Ch], 76h
                mov     dword ptr [ebp-18h], 6Bh
                mov     dword ptr [ebp-14h], 0F8h
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48F146
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48F146:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48F167
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48F164
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48F164:
                mov     [ebp-38h], ecx

loc_48F167:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}





__declspec(naked) void sub_48F1D3(void) {  __asm  {









                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 59h
                mov     dword ptr [ebp-2Ch], 90h
                mov     dword ptr [ebp-28h], 0B0h
                mov     dword ptr [ebp-24h], 0C0h
                mov     dword ptr [ebp-20h], 73h
                mov     dword ptr [ebp-1Ch], 86h
                mov     dword ptr [ebp-18h], 0A0h
                mov     dword ptr [ebp-14h], 0D2h
                mov     dword ptr [ebp-10h], 10h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 10h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48F24E
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48F24E:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48F26F
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48F26C
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48F26C:
                mov     [ebp-38h], ecx

loc_48F26F:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_48F2DB(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 77h
                mov     dword ptr [ebp-2Ch], 32h
                mov     dword ptr [ebp-28h], 3Dh
                mov     dword ptr [ebp-24h], 32h
                mov     dword ptr [ebp-20h], 78h
                mov     dword ptr [ebp-1Ch], 89h
                mov     dword ptr [ebp-18h], 0F9h
                mov     dword ptr [ebp-14h], 0C1h
                mov     dword ptr [ebp-10h], 8
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 8
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48F356
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48F356:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48F377
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48F374
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48F374:
                mov     [ebp-38h], ecx

loc_48F377:
```

```
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_48F3E3(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0CBh
                mov     dword ptr [ebp-2Ch], 13h
                mov     dword ptr [ebp-28h], 34h
                mov     dword ptr [ebp-24h], 28h
                mov     dword ptr [ebp-20h], 0F8h
                mov     dword ptr [ebp-1Ch], 0AEh
                mov     dword ptr [ebp-18h], 50h
                mov     dword ptr [ebp-14h], 30h
                mov     dword ptr [ebp-10h], 0Ah
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ah
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48F45E
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48F45E:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48F47F
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48F47C
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx
```

```
loc_48F47C:
                mov     [ebp-38h], ecx

loc_48F47F:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_48F4EB(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0CEh
                mov     dword ptr [ebp-2Ch], 48h
                mov     dword ptr [ebp-28h], 4Dh
                mov     dword ptr [ebp-24h], 3Bh
                mov     dword ptr [ebp-20h], 0B3h
                mov     dword ptr [ebp-1Ch], 0C5h
                mov     dword ptr [ebp-18h], 0BBh
                mov     dword ptr [ebp-14h], 0C4h
                mov     dword ptr [ebp-10h], 0Eh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Eh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48F566
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48F566:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48F587
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48F584
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48F584:
                mov     [ebp-38h], ecx

loc_48F587:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_48F5F3(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 7Fh
                mov     dword ptr [ebp-2Ch], 0EDh
                mov     dword ptr [ebp-28h], 0Eh
                mov     dword ptr [ebp-24h], 53h
                mov     dword ptr [ebp-20h], 45h
                mov     dword ptr [ebp-1Ch], 11h
                mov     dword ptr [ebp-18h], 0D3h
                mov     dword ptr [ebp-14h], 0CDh
                mov     dword ptr [ebp-10h], 0Fh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Fh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48F66E
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48F66E:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48F68F
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48F68C
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48F68C:
                mov     [ebp-38h], ecx

loc_48F68F:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_48F6FB(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 1Bh
                mov     dword ptr [ebp-2Ch], 0Dh
                mov     dword ptr [ebp-28h], 0F6h
                mov     dword ptr [ebp-24h], 97h
                mov     dword ptr [ebp-20h], 42h
                mov     dword ptr [ebp-1Ch], 11h
                mov     dword ptr [ebp-18h], 0C6h
                mov     dword ptr [ebp-14h], 0DEh
                mov     dword ptr [ebp-10h], 4
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 4
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48F776
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48F776:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48F797
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48F794
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48F794:
                mov     [ebp-38h], ecx

loc_48F797:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_48F803(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 74h
                mov     dword ptr [ebp-2Ch], 0C1h
                mov     dword ptr [ebp-28h], 9
                mov     dword ptr [ebp-24h], 0Eh
                mov     dword ptr [ebp-20h], 9
                mov     dword ptr [ebp-1Ch], 0D4h
                mov     dword ptr [ebp-18h], 5Fh
                mov     dword ptr [ebp-14h], 29h
                mov     dword ptr [ebp-10h], 14h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 14h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48F87E
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48F87E:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48F89F
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48F89C
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48F89C:
                mov     [ebp-38h], ecx

loc_48F89F:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_48F90B(void) {  __asm  {














                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0E5h
                mov     dword ptr [ebp-2Ch], 6Fh
                mov     dword ptr [ebp-28h], 85h
                mov     dword ptr [ebp-24h], 16h
                mov     dword ptr [ebp-20h], 0Bh
                mov     dword ptr [ebp-1Ch], 0D1h
                mov     dword ptr [ebp-18h], 45h
                mov     dword ptr [ebp-14h], 0BBh
                mov     dword ptr [ebp-10h], 0Fh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Fh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48F986
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48F986:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48F9A7
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48F9A4
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48F9A4:
                mov     [ebp-38h], ecx

loc_48F9A7:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_48FA13(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0DDh
                mov     dword ptr [ebp-2Ch], 4Ah
                mov     dword ptr [ebp-28h], 0B6h
                mov     dword ptr [ebp-24h], 11h
                mov     dword ptr [ebp-20h], 55h
                mov     dword ptr [ebp-1Ch], 9Fh
                mov     dword ptr [ebp-18h], 0F6h
                mov     dword ptr [ebp-14h], 76h
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48FA8E
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48FA8E:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48FAAF
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48FAAC
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48FAAC:
                mov     [ebp-38h], ecx

loc_48FAAF:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_48FB1B(void) {  __asm  {










                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0D7h
                mov     dword ptr [ebp-2Ch], 47h
                mov     dword ptr [ebp-28h], 0EFh
                mov     dword ptr [ebp-24h], 36h
                mov     dword ptr [ebp-20h], 0F1h
                mov     dword ptr [ebp-1Ch], 0B2h
                mov     dword ptr [ebp-18h], 57h
                mov     dword ptr [ebp-14h], 47h
                mov     dword ptr [ebp-10h], 0
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48FB93
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48FB93:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48FBB4
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48FBB1
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48FBB1:
                mov     [ebp-38h], ecx

loc_48FBB4:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_48FC20(void) {  __asm  {











                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0F1h
                mov     dword ptr [ebp-2Ch], 22h
                mov     dword ptr [ebp-28h], 52h
                mov     dword ptr [ebp-24h], 0E0h
                mov     dword ptr [ebp-20h], 34h
                mov     dword ptr [ebp-1Ch], 79h
                mov     dword ptr [ebp-18h], 0D9h
                mov     dword ptr [ebp-14h], 2
                mov     dword ptr [ebp-10h], 12h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 12h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48FC9B
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48FC9B:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48FCBC
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48FCB9
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48FCB9:
                mov     [ebp-38h], ecx

loc_48FCBC:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}






__declspec(naked) void sub_48FD28(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 45h
                mov     dword ptr [ebp-2Ch], 39h
                mov     dword ptr [ebp-28h], 9Fh
                mov     dword ptr [ebp-24h], 83h
                mov     dword ptr [ebp-20h], 51h
                mov     dword ptr [ebp-1Ch], 48h
                mov     dword ptr [ebp-18h], 2Dh
                mov     dword ptr [ebp-14h], 71h
                mov     dword ptr [ebp-10h], 1
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 1
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48FDA2
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48FDA2:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48FDC3
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48FDC0
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48FDC0:
                mov     [ebp-38h], ecx

loc_48FDC3:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_48FE2F(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 53h
                mov     dword ptr [ebp-2Ch], 57h
                mov     dword ptr [ebp-28h], 9Ah
                mov     dword ptr [ebp-24h], 62h
                mov     dword ptr [ebp-20h], 14h
                mov     dword ptr [ebp-1Ch], 8Bh
                mov     dword ptr [ebp-18h], 2Ah
                mov     dword ptr [ebp-14h], 19h
                mov     dword ptr [ebp-10h], 9
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 9
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48FEAA
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48FEAA:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48FECB
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48FEC8
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48FEC8:
                mov     [ebp-38h], ecx
```

```
loc_48FECB:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_48FF37(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 6
                mov     dword ptr [ebp-2Ch], 0Ah
                mov     dword ptr [ebp-28h], 19h
                mov     dword ptr [ebp-24h], 3Ah
                mov     dword ptr [ebp-20h], 0A0h
                mov     dword ptr [ebp-1Ch], 0B1h
                mov     dword ptr [ebp-18h], 0D5h
                mov     dword ptr [ebp-14h], 9Dh
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_48FFB2
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_48FFB2:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_48FFD3
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_48FFD0
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_48FFD0:
                mov     [ebp-38h], ecx

loc_48FFD3:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}





__declspec(naked) void sub_49003F(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0C4h
                mov     dword ptr [ebp-2Ch], 0EAh
                mov     dword ptr [ebp-28h], 4Eh
                mov     dword ptr [ebp-24h], 0BDh
                mov     dword ptr [ebp-20h], 29h
                mov     dword ptr [ebp-1Ch], 9Bh
                mov     dword ptr [ebp-18h], 0ECh
                mov     dword ptr [ebp-14h], 0F6h
                mov     dword ptr [ebp-10h], 6
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 6
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4900BA
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4900BA:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4900DB
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4900D8
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4900D8:
                mov     [ebp-38h], ecx

loc_4900DB:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_490147(void) {  __asm  {
```

```
push    ebp
mov     ebp, esp
sub     esp, 40h
mov     dword ptr [ebp-30h], 6Fh
mov     dword ptr [ebp-2Ch], 0A9h
mov     dword ptr [ebp-28h], 0B8h
mov     dword ptr [ebp-24h], 9Ch
mov     dword ptr [ebp-20h], 2Eh
mov     dword ptr [ebp-1Ch], 21h
mov     dword ptr [ebp-18h], 0AFh
mov     dword ptr [ebp-14h], 0D5h
mov     dword ptr [ebp-10h], 14h
mov     dword ptr [ebp-40h], 7
mov     eax, [ebp+8]
shr     eax, 14h
and     eax, 7
mov     ecx, [ebp+eax*4-30h]
mov     [ebp-3Ch], ecx
mov     eax, [ebp-3Ch]
cdq
and     edx, 0Fh
add     eax, edx
sar     eax, 4
mov     [ebp-34h], eax
mov     edx, [ebp-3Ch]
and     edx, 8000000Fh
jns     short loc_4901C2
dec     edx
or      edx, 0FFFFFFF0h
inc     edx
```

```
loc_4901C2:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4901E3
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4901E0
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4901E0:
                mov     [ebp-38h], ecx

loc_4901E3:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49024F(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 28h
                mov     dword ptr [ebp-2Ch], 0AAh
                mov     dword ptr [ebp-28h], 60h
                mov     dword ptr [ebp-24h], 8Dh
                mov     dword ptr [ebp-20h], 0A8h
                mov     dword ptr [ebp-1Ch], 77h
                mov     dword ptr [ebp-18h], 2Ah
                mov     dword ptr [ebp-14h], 99h
                mov     dword ptr [ebp-10h], 1
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 1
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4902C9
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4902C9:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4902EA
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4902E7
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4902E7:
                mov     [ebp-38h], ecx

loc_4902EA:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}

__declspec(naked) void sub_490356(void) {  __asm  {

                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0A3h
                mov     dword ptr [ebp-2Ch], 4Ah
                mov     dword ptr [ebp-28h], 46h
                mov     dword ptr [ebp-24h], 46h
                mov     dword ptr [ebp-20h], 0B8h
                mov     dword ptr [ebp-1Ch], 0E5h
                mov     dword ptr [ebp-18h], 3Bh
                mov     dword ptr [ebp-14h], 0A0h
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4903D1
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4903D1:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4903F2
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4903EF
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4903EF:
                mov     [ebp-38h], ecx

loc_4903F2:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49045E(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0D3h
                mov     dword ptr [ebp-2Ch], 0E2h
                mov     dword ptr [ebp-28h], 15h
                mov     dword ptr [ebp-24h], 0D7h
                mov     dword ptr [ebp-20h], 98h
                mov     dword ptr [ebp-1Ch], 8Dh
                mov     dword ptr [ebp-18h], 10h
                mov     dword ptr [ebp-14h], 0F2h
                mov     dword ptr [ebp-10h], 0Bh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Bh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4904D9
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4904D9:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4904FA
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4904F7
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4904F7:
                mov     [ebp-38h], ecx

loc_4904FA:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}



__declspec(naked) void sub_490566(void) {  __asm  {











                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 1Eh
                mov     dword ptr [ebp-2Ch], 0D3h
                mov     dword ptr [ebp-28h], 6Fh
                mov     dword ptr [ebp-24h], 0E4h
                mov     dword ptr [ebp-20h], 34h
                mov     dword ptr [ebp-1Ch], 0F7h
                mov     dword ptr [ebp-18h], 22h
                mov     dword ptr [ebp-14h], 43h
                mov     dword ptr [ebp-10h], 11h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 11h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4905E1
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4905E1:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_490602
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4905FF
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4905FF:
                mov     [ebp-38h], ecx

loc_490602:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49066E(void) {  __asm  {













                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0E2h
                mov     dword ptr [ebp-2Ch], 0A7h
                mov     dword ptr [ebp-28h], 0C5h
                mov     dword ptr [ebp-24h], 0
                mov     dword ptr [ebp-20h], 0DFh
                mov     dword ptr [ebp-1Ch], 37h
                mov     dword ptr [ebp-18h], 85h
                mov     dword ptr [ebp-14h], 93h
                mov     dword ptr [ebp-10h], 13h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 13h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4906E9
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4906E9:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49070A
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_490707
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_490707:
                mov     [ebp-38h], ecx

loc_49070A:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_490776(void) {  __asm  {




















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0EDh
                mov     dword ptr [ebp-2Ch], 71h
                mov     dword ptr [ebp-28h], 6Ch
                mov     dword ptr [ebp-24h], 5Bh
                mov     dword ptr [ebp-20h], 7Fh
                mov     dword ptr [ebp-1Ch], 7Ch
                mov     dword ptr [ebp-18h], 1Eh
                mov     dword ptr [ebp-14h], 20h
                mov     dword ptr [ebp-10h], 0Dh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Dh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4907F1
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4907F1:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_490812
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49080F
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49080F:
                mov     [ebp-38h], ecx

loc_490812:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49087E(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 9Fh
                mov     dword ptr [ebp-2Ch], 60h
                mov     dword ptr [ebp-28h], 0CAh
                mov     dword ptr [ebp-24h], 0C8h
                mov     dword ptr [ebp-20h], 80h
                mov     dword ptr [ebp-1Ch], 62h
                mov     dword ptr [ebp-18h], 0DFh
                mov     dword ptr [ebp-14h], 53h
                mov     dword ptr [ebp-10h], 0
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4908F6
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4908F6:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_490917
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_490914
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_490914:
                mov     [ebp-38h], ecx

loc_490917:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_490983(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 84h
                mov     dword ptr [ebp-2Ch], 0A7h
                mov     dword ptr [ebp-28h], 0CAh
                mov     dword ptr [ebp-24h], 54h
                mov     dword ptr [ebp-20h], 37h
                mov     dword ptr [ebp-1Ch], 0C7h
                mov     dword ptr [ebp-18h], 0C4h
                mov     dword ptr [ebp-14h], 51h
                mov     dword ptr [ebp-10h], 14h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 14h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4909FE
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4909FE:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_490A1F
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_490A1C
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx
```

```
loc_490A1C:
                mov     [ebp-38h], ecx

loc_490A1F:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}





__declspec(naked) void sub_490A8B(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 36h
                mov     dword ptr [ebp-2Ch], 7Fh
                mov     dword ptr [ebp-28h], 0B5h
                mov     dword ptr [ebp-24h], 0AFh
                mov     dword ptr [ebp-20h], 0ACh
                mov     dword ptr [ebp-1Ch], 65h
                mov     dword ptr [ebp-18h], 0CFh
                mov     dword ptr [ebp-14h], 0D2h
                mov     dword ptr [ebp-10h], 4
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 4
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_490B06
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_490B06:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_490B27
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_490B24
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_490B24:
                mov     [ebp-38h], ecx

loc_490B27:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_490B93(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0CFh
                mov     dword ptr [ebp-2Ch], 5Dh
                mov     dword ptr [ebp-28h], 5Ah
                mov     dword ptr [ebp-24h], 0DBh
                mov     dword ptr [ebp-20h], 9
                mov     dword ptr [ebp-1Ch], 0BEh
                mov     dword ptr [ebp-18h], 65h
                mov     dword ptr [ebp-14h], 77h
                mov     dword ptr [ebp-10h], 2
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 2
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_490C0E
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_490C0E:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_490C2F
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_490C2C
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_490C2C:
                mov     [ebp-38h], ecx

loc_490C2F:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_490C9B(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 91h
                mov     dword ptr [ebp-2Ch], 6Ah
                mov     dword ptr [ebp-28h], 42h
                mov     dword ptr [ebp-24h], 7Dh
                mov     dword ptr [ebp-20h], 0DBh
                mov     dword ptr [ebp-1Ch], 0Ch
                mov     dword ptr [ebp-18h], 89h
                mov     dword ptr [ebp-14h], 89h
                mov     dword ptr [ebp-10h], 12h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 12h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_490D16
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_490D16:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_490D37
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_490D34
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_490D34:
                mov     [ebp-38h], ecx

loc_490D37:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_490DA3(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 24h
                mov     dword ptr [ebp-2Ch], 0DBh
                mov     dword ptr [ebp-28h], 0A0h
                mov     dword ptr [ebp-24h], 0AFh
                mov     dword ptr [ebp-20h], 74h
                mov     dword ptr [ebp-1Ch], 0DCh
                mov     dword ptr [ebp-18h], 52h
                mov     dword ptr [ebp-14h], 34h
                mov     dword ptr [ebp-10h], 7
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 7
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_490E1E
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_490E1E:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_490E3F
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_490E3C
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_490E3C:
                mov     [ebp-38h], ecx

loc_490E3F:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_490EAB(void) {  __asm  {














                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 6Eh
                mov     dword ptr [ebp-2Ch], 4Bh
                mov     dword ptr [ebp-28h], 79h
                mov     dword ptr [ebp-24h], 0D0h
                mov     dword ptr [ebp-20h], 24h
                mov     dword ptr [ebp-1Ch], 0CFh
                mov     dword ptr [ebp-18h], 2Ah
                mov     dword ptr [ebp-14h], 58h
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_490F26
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_490F26:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_490F47
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_490F44
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_490F44:
                mov     [ebp-38h], ecx

loc_490F47:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_490FB3(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 96h
                mov     dword ptr [ebp-2Ch], 12h
                mov     dword ptr [ebp-28h], 0CDh
                mov     dword ptr [ebp-24h], 0EDh
                mov     dword ptr [ebp-20h], 47h
                mov     dword ptr [ebp-1Ch], 87h
                mov     dword ptr [ebp-18h], 9Ah
                mov     dword ptr [ebp-14h], 0Ch
                mov     dword ptr [ebp-10h], 0Ah
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ah
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49102E
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49102E:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49104F
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49104C
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49104C:
                mov     [ebp-38h], ecx

loc_49104F:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4910BB(void) {  __asm  {














                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 32h
                mov     dword ptr [ebp-2Ch], 26h
                mov     dword ptr [ebp-28h], 9Eh
                mov     dword ptr [ebp-24h], 16h
                mov     dword ptr [ebp-20h], 3Dh
                mov     dword ptr [ebp-1Ch], 94h
                mov     dword ptr [ebp-18h], 0C9h
                mov     dword ptr [ebp-14h], 0B9h
                mov     dword ptr [ebp-10h], 0Eh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Eh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_491136
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_491136:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_491157
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_491154
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_491154:
                mov     [ebp-38h], ecx

loc_491157:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4911C3(void) {  __asm  {




















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0C1h
                mov     dword ptr [ebp-2Ch], 42h
                mov     dword ptr [ebp-28h], 49h
                mov     dword ptr [ebp-24h], 1Bh
                mov     dword ptr [ebp-20h], 54h
                mov     dword ptr [ebp-1Ch], 0EAh
                mov     dword ptr [ebp-18h], 4Ch
                mov     dword ptr [ebp-14h], 0Bh
                mov     dword ptr [ebp-10h], 11h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 11h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49123E
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49123E:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49125F
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49125C
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49125C:
                mov     [ebp-38h], ecx

loc_49125F:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4912CB(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0EBh
                mov     dword ptr [ebp-2Ch], 11h
                mov     dword ptr [ebp-28h], 82h
                mov     dword ptr [ebp-24h], 7
                mov     dword ptr [ebp-20h], 4Eh
                mov     dword ptr [ebp-1Ch], 0F5h
                mov     dword ptr [ebp-18h], 4
                mov     dword ptr [ebp-14h], 0A1h
                mov     dword ptr [ebp-10h], 2
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 2
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_491346
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_491346:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_491367
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_491364
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_491364:
                mov     [ebp-38h], ecx

loc_491367:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4913D3(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 44h
                mov     dword ptr [ebp-2Ch], 2Fh
                mov     dword ptr [ebp-28h], 0Eh
                mov     dword ptr [ebp-24h], 8
                mov     dword ptr [ebp-20h], 36h
                mov     dword ptr [ebp-1Ch], 81h
                mov     dword ptr [ebp-18h], 60h
                mov     dword ptr [ebp-14h], 63h
                mov     dword ptr [ebp-10h], 4
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 4
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49144E
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49144E:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49146F
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49146C
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49146C:
                mov     [ebp-38h], ecx
```

```
loc_49146F:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4914DB(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0EFh
                mov     dword ptr [ebp-2Ch], 10h
                mov     dword ptr [ebp-28h], 9
                mov     dword ptr [ebp-24h], 9Ch
                mov     dword ptr [ebp-20h], 6Eh
                mov     dword ptr [ebp-1Ch], 0F6h
                mov     dword ptr [ebp-18h], 0B4h
                mov     dword ptr [ebp-14h], 0B1h
                mov     dword ptr [ebp-10h], 15h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 15h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_491556
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_491556:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_491577
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_491574
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_491574:
                mov     [ebp-38h], ecx

loc_491577:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4915E3(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0EAh
                mov     dword ptr [ebp-2Ch], 0F7h
                mov     dword ptr [ebp-28h], 55h
                mov     dword ptr [ebp-24h], 74h
                mov     dword ptr [ebp-20h], 8Ah
                mov     dword ptr [ebp-1Ch], 0ECh
                mov     dword ptr [ebp-18h], 83h
                mov     dword ptr [ebp-14h], 0Fh
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49165E
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49165E:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49167F
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49167C
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49167C:
                mov     [ebp-38h], ecx

loc_49167F:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4916EB(void) {  __asm  {
```

```
push    ebp
mov     ebp, esp
sub     esp, 40h
mov     dword ptr [ebp-30h], 0E6h
mov     dword ptr [ebp-2Ch], 75h
mov     dword ptr [ebp-28h], 8Fh
mov     dword ptr [ebp-24h], 0B6h
mov     dword ptr [ebp-20h], 0C5h
mov     dword ptr [ebp-1Ch], 0A1h
mov     dword ptr [ebp-18h], 0C1h
mov     dword ptr [ebp-14h], 53h
mov     dword ptr [ebp-10h], 9
mov     dword ptr [ebp-40h], 7
mov     eax, [ebp+8]
shr     eax, 9
and     eax, 7
mov     ecx, [ebp+eax*4-30h]
mov     [ebp-3Ch], ecx
mov     eax, [ebp-3Ch]
cdq
and     edx, 0Fh
add     eax, edx
sar     eax, 4
mov     [ebp-34h], eax
mov     edx, [ebp-3Ch]
and     edx, 8000000Fh
jns     short loc_491766
dec     edx
or      edx, 0FFFFFFF0h
inc     edx
```

```
loc_491766:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_491787
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_491784
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_491784:
                mov     [ebp-38h], ecx

loc_491787:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_4917F3(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0Ch
                mov     dword ptr [ebp-2Ch], 0A3h
                mov     dword ptr [ebp-28h], 0D6h
                mov     dword ptr [ebp-24h], 0C0h
                mov     dword ptr [ebp-20h], 0F2h
                mov     dword ptr [ebp-1Ch], 31h
                mov     dword ptr [ebp-18h], 0C5h
                mov     dword ptr [ebp-14h], 17h
                mov     dword ptr [ebp-10h], 0Fh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Fh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49186E
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49186E:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49188F
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49188C
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49188C:
                mov     [ebp-38h], ecx

loc_49188F:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}

__declspec(naked) void sub_4918FB(void) {  __asm  {

                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 2Fh
                mov     dword ptr [ebp-2Ch], 75h
                mov     dword ptr [ebp-28h], 9Bh
                mov     dword ptr [ebp-24h], 55h
                mov     dword ptr [ebp-20h], 8Ah
                mov     dword ptr [ebp-1Ch], 0A9h
                mov     dword ptr [ebp-18h], 3Ah
                mov     dword ptr [ebp-14h], 72h
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_491976
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_491976:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_491997
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_491994
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_491994:
                mov     [ebp-38h], ecx

loc_491997:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_491A03(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 7Fh
                mov     dword ptr [ebp-2Ch], 0F7h
                mov     dword ptr [ebp-28h], 2Dh
                mov     dword ptr [ebp-24h], 70h
                mov     dword ptr [ebp-20h], 7Fh
                mov     dword ptr [ebp-1Ch], 0B7h
                mov     dword ptr [ebp-18h], 73h
                mov     dword ptr [ebp-14h], 0B4h
                mov     dword ptr [ebp-10h], 10h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 10h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_491A7E
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_491A7E:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_491A9F
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_491A9C
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_491A9C:
                mov     [ebp-38h], ecx

loc_491A9F:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_491B0B(void) {  __asm  {















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0F4h
                mov     dword ptr [ebp-2Ch], 0E2h
                mov     dword ptr [ebp-28h], 50h
                mov     dword ptr [ebp-24h], 37h
                mov     dword ptr [ebp-20h], 57h
                mov     dword ptr [ebp-1Ch], 0CCh
                mov     dword ptr [ebp-18h], 61h
                mov     dword ptr [ebp-14h], 0BBh
                mov     dword ptr [ebp-10h], 4
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 4
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_491B86
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_491B86:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_491BA7
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_491BA4
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_491BA4:
                mov     [ebp-38h], ecx

loc_491BA7:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}



__declspec(naked) void sub_491C13(void) {  __asm  {



                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 5Eh
                mov     dword ptr [ebp-2Ch], 0DEh
                mov     dword ptr [ebp-28h], 6
                mov     dword ptr [ebp-24h], 0BCh
                mov     dword ptr [ebp-20h], 0C7h
                mov     dword ptr [ebp-1Ch], 0F0h
                mov     dword ptr [ebp-18h], 42h
                mov     dword ptr [ebp-14h], 66h
                mov     dword ptr [ebp-10h], 0Ah
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ah
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_491C8E
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_491C8E:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_491CAF
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_491CAC
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_491CAC:
                mov     [ebp-38h], ecx

loc_491CAF:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_491D1B(void) {  __asm  {




















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0F8h
                mov     dword ptr [ebp-2Ch], 67h
                mov     dword ptr [ebp-28h], 0C4h
                mov     dword ptr [ebp-24h], 0C9h
                mov     dword ptr [ebp-20h], 66h
                mov     dword ptr [ebp-1Ch], 0E6h
                mov     dword ptr [ebp-18h], 0ACh
                mov     dword ptr [ebp-14h], 4Ah
                mov     dword ptr [ebp-10h], 0
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_491D93
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_491D93:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_491DB4
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_491DB1
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_491DB1:
                mov     [ebp-38h], ecx

loc_491DB4:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_491E20(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 17h
                mov     dword ptr [ebp-2Ch], 0A1h
                mov     dword ptr [ebp-28h], 82h
                mov     dword ptr [ebp-24h], 1Eh
                mov     dword ptr [ebp-20h], 8Dh
                mov     dword ptr [ebp-1Ch], 72h
                mov     dword ptr [ebp-18h], 6Ah
                mov     dword ptr [ebp-14h], 0F1h
                mov     dword ptr [ebp-10h], 7
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 7
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_491E9B
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_491E9B:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_491EBC
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_491EB9
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_491EB9:
                mov     [ebp-38h], ecx

loc_491EBC:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_491F28(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0BBh
                mov     dword ptr [ebp-2Ch], 5Ch
                mov     dword ptr [ebp-28h], 0Ah
                mov     dword ptr [ebp-24h], 51h
                mov     dword ptr [ebp-20h], 0A0h
                mov     dword ptr [ebp-1Ch], 8Eh
                mov     dword ptr [ebp-18h], 17h
                mov     dword ptr [ebp-14h], 0C0h
                mov     dword ptr [ebp-10h], 0Ch
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ch
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_491FA3
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_491FA3:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_491FC4
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_491FC1
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx
```

```
loc_491FC1:
                mov     [ebp-38h], ecx

loc_491FC4:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_492030(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 6Ch
                mov     dword ptr [ebp-2Ch], 79h
                mov     dword ptr [ebp-28h], 87h
                mov     dword ptr [ebp-24h], 0F9h
                mov     dword ptr [ebp-20h], 0A1h
                mov     dword ptr [ebp-1Ch], 0C0h
                mov     dword ptr [ebp-18h], 0Ah
                mov     dword ptr [ebp-14h], 9Bh
                mov     dword ptr [ebp-10h], 0Fh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Fh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4920AB
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4920AB:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4920CC
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4920C9
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4920C9:
                mov     [ebp-38h], ecx

loc_4920CC:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_492138(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 14h
                mov     dword ptr [ebp-2Ch], 0B2h
                mov     dword ptr [ebp-28h], 62h
                mov     dword ptr [ebp-24h], 31h
                mov     dword ptr [ebp-20h], 0D2h
                mov     dword ptr [ebp-1Ch], 28h
                mov     dword ptr [ebp-18h], 0BBh
                mov     dword ptr [ebp-14h], 66h
                mov     dword ptr [ebp-10h], 13h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 13h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4921B3
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4921B3:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4921D4
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4921D1
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4921D1:
                mov     [ebp-38h], ecx

loc_4921D4:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_492240(void) {  __asm  {
```

```
push    ebp
mov     ebp, esp
sub     esp, 40h
mov     dword ptr [ebp-30h], 4
mov     dword ptr [ebp-2Ch], 94h
mov     dword ptr [ebp-28h], 6
mov     dword ptr [ebp-24h], 0E0h
mov     dword ptr [ebp-20h], 47h
mov     dword ptr [ebp-1Ch], 0A7h
mov     dword ptr [ebp-18h], 67h
mov     dword ptr [ebp-14h], 67h
mov     dword ptr [ebp-10h], 0
mov     dword ptr [ebp-40h], 7
mov     eax, [ebp+8]
and     eax, 7
mov     ecx, [ebp+eax*4-30h]
mov     [ebp-3Ch], ecx
mov     eax, [ebp-3Ch]
cdq
and     edx, 0Fh
add     eax, edx
sar     eax, 4
mov     [ebp-34h], eax
mov     edx, [ebp-3Ch]
and     edx, 8000000Fh
jns     short loc_4922B8
dec     edx
or      edx, 0FFFFFFF0h
inc     edx
```

```
loc_4922B8:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4922D9
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4922D6
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4922D6:
                mov     [ebp-38h], ecx

loc_4922D9:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_492345(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0A2h
                mov     dword ptr [ebp-2Ch], 41h
                mov     dword ptr [ebp-28h], 51h
                mov     dword ptr [ebp-24h], 18h
                mov     dword ptr [ebp-20h], 5Ch
                mov     dword ptr [ebp-1Ch], 5Dh
                mov     dword ptr [ebp-18h], 0F6h
                mov     dword ptr [ebp-14h], 0C3h
                mov     dword ptr [ebp-10h], 11h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 11h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4923C0
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4923C0:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4923E1
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4923DE
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4923DE:
                mov     [ebp-38h], ecx

loc_4923E1:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49244D(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 73h
                mov     dword ptr [ebp-2Ch], 43h
                mov     dword ptr [ebp-28h], 7Dh
                mov     dword ptr [ebp-24h], 0F9h
                mov     dword ptr [ebp-20h], 65h
                mov     dword ptr [ebp-1Ch], 3Fh
                mov     dword ptr [ebp-18h], 0EEh
                mov     dword ptr [ebp-14h], 0F2h
                mov     dword ptr [ebp-10h], 9
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 9
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4924C8
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4924C8:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4924E9
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4924E6
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4924E6:
                mov     [ebp-38h], ecx

loc_4924E9:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_492555(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0AEh
                mov     dword ptr [ebp-2Ch], 68h
                mov     dword ptr [ebp-28h], 57h
                mov     dword ptr [ebp-24h], 34h
                mov     dword ptr [ebp-20h], 0D7h
                mov     dword ptr [ebp-1Ch], 0B6h
                mov     dword ptr [ebp-18h], 36h
                mov     dword ptr [ebp-14h], 5Dh
                mov     dword ptr [ebp-10h], 1
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 1
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4925CF
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4925CF:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4925F0
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4925ED
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4925ED:
                mov     [ebp-38h], ecx

loc_4925F0:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49265C(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 2Ah
                mov     dword ptr [ebp-2Ch], 0F6h
                mov     dword ptr [ebp-28h], 0BBh
                mov     dword ptr [ebp-24h], 0BDh
                mov     dword ptr [ebp-20h], 0EEh
                mov     dword ptr [ebp-1Ch], 61h
                mov     dword ptr [ebp-18h], 0ECh
                mov     dword ptr [ebp-14h], 95h
                mov     dword ptr [ebp-10h], 9
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 9
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4926D7
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4926D7:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4926F8
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4926F5
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4926F5:
                mov     [ebp-38h], ecx

loc_4926F8:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_492764(void) {  __asm  {












                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 67h
                mov     dword ptr [ebp-2Ch], 0ACh
                mov     dword ptr [ebp-28h], 20h
                mov     dword ptr [ebp-24h], 8Ch
                mov     dword ptr [ebp-20h], 0C4h
                mov     dword ptr [ebp-1Ch], 47h
                mov     dword ptr [ebp-18h], 0
                mov     dword ptr [ebp-14h], 9
                mov     dword ptr [ebp-10h], 0Ch
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ch
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4927DF
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4927DF:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_492800
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4927FD
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4927FD:
                mov     [ebp-38h], ecx

loc_492800:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49286C(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 83h
                mov     dword ptr [ebp-2Ch], 5Bh
                mov     dword ptr [ebp-28h], 2
                mov     dword ptr [ebp-24h], 0B1h
                mov     dword ptr [ebp-20h], 26h
                mov     dword ptr [ebp-1Ch], 0A6h
                mov     dword ptr [ebp-18h], 50h
                mov     dword ptr [ebp-14h], 46h
                mov     dword ptr [ebp-10h], 0Ah
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ah
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4928E7
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4928E7:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_492908
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_492905
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_492905:
                mov     [ebp-38h], ecx

loc_492908:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_492974(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 77h
                mov     dword ptr [ebp-2Ch], 79h
                mov     dword ptr [ebp-28h], 0FBh
                mov     dword ptr [ebp-24h], 47h
                mov     dword ptr [ebp-20h], 5Bh
                mov     dword ptr [ebp-1Ch], 0E2h
                mov     dword ptr [ebp-18h], 1
                mov     dword ptr [ebp-14h], 0C9h
                mov     dword ptr [ebp-10h], 6
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 6
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4929EF
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4929EF:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_492A10
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_492A0D
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_492A0D:
                mov     [ebp-38h], ecx
```

```
loc_492A10:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}





__declspec(naked) void sub_492A7C(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0E5h
                mov     dword ptr [ebp-2Ch], 47h
                mov     dword ptr [ebp-28h], 7Ch
                mov     dword ptr [ebp-24h], 4
                mov     dword ptr [ebp-20h], 26h
                mov     dword ptr [ebp-1Ch], 66h
                mov     dword ptr [ebp-18h], 4Bh
                mov     dword ptr [ebp-14h], 9Bh
                mov     dword ptr [ebp-10h], 0
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_492AF4
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_492AF4:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_492B15
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_492B12
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx
```

```
loc_492B12:
                mov     [ebp-38h], ecx

loc_492B15:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_492B81(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 90h
                mov     dword ptr [ebp-2Ch], 0Eh
                mov     dword ptr [ebp-28h], 0A5h
                mov     dword ptr [ebp-24h], 1Ch
                mov     dword ptr [ebp-20h], 3Fh
                mov     dword ptr [ebp-1Ch], 38h
                mov     dword ptr [ebp-18h], 92h
                mov     dword ptr [ebp-14h], 0DCh
                mov     dword ptr [ebp-10h], 3
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 3
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_492BFC
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_492BFC:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_492C1D
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_492C1A
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_492C1A:
                mov     [ebp-38h], ecx

loc_492C1D:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_492C89(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 36h
                mov     dword ptr [ebp-2Ch], 0B0h
                mov     dword ptr [ebp-28h], 38h
                mov     dword ptr [ebp-24h], 2Ch
                mov     dword ptr [ebp-20h], 13h
                mov     dword ptr [ebp-1Ch], 37h
                mov     dword ptr [ebp-18h], 99h
                mov     dword ptr [ebp-14h], 97h
                mov     dword ptr [ebp-10h], 3
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 3
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_492D04
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_492D04:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_492D25
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_492D22
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_492D22:
                mov     [ebp-38h], ecx

loc_492D25:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_492D91(void) {  __asm  {
```

```
push    ebp
mov     ebp, esp
sub     esp, 40h
mov     dword ptr [ebp-30h], 78h
mov     dword ptr [ebp-2Ch], 0C0h
mov     dword ptr [ebp-28h], 60h
mov     dword ptr [ebp-24h], 0D3h
mov     dword ptr [ebp-20h], 0EFh
mov     dword ptr [ebp-1Ch], 2Dh
mov     dword ptr [ebp-18h], 0B4h
mov     dword ptr [ebp-14h], 0C9h
mov     dword ptr [ebp-10h], 0Eh
mov     dword ptr [ebp-40h], 7
mov     eax, [ebp+8]
shr     eax, 0Eh
and     eax, 7
mov     ecx, [ebp+eax*4-30h]
mov     [ebp-3Ch], ecx
mov     eax, [ebp-3Ch]
cdq
and     edx, 0Fh
add     eax, edx
sar     eax, 4
mov     [ebp-34h], eax
mov     edx, [ebp-3Ch]
and     edx, 8000000Fh
jns     short loc_492E0C
dec     edx
```

```
                or      edx, 0FFFFFFF0h
                inc     edx

loc_492E0C:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_492E2D
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_492E2A
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_492E2A:
                mov     [ebp-38h], ecx

loc_492E2D:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_492E99(void) {  __asm  {
```

```
push    ebp
mov     ebp, esp
sub     esp, 40h
mov     dword ptr [ebp-30h], 0Fh
mov     dword ptr [ebp-2Ch], 56h
mov     dword ptr [ebp-28h], 4Dh
mov     dword ptr [ebp-24h], 0CDh
mov     dword ptr [ebp-20h], 73h
mov     dword ptr [ebp-1Ch], 0D1h
mov     dword ptr [ebp-18h], 20h
mov     dword ptr [ebp-14h], 87h
mov     dword ptr [ebp-10h], 0Ah
mov     dword ptr [ebp-40h], 7
mov     eax, [ebp+8]
shr     eax, 0Ah
and     eax, 7
mov     ecx, [ebp+eax*4-30h]
mov     [ebp-3Ch], ecx
mov     eax, [ebp-3Ch]
cdq
and     edx, 0Fh
add     eax, edx
sar     eax, 4
```

```
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_492F14
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_492F14:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_492F35
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_492F32
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_492F32:
                mov     [ebp-38h], ecx

loc_492F35:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_492FA1(void) {  __asm  {











                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 35h
                mov     dword ptr [ebp-2Ch], 8Ah
                mov     dword ptr [ebp-28h], 0D8h
                mov     dword ptr [ebp-24h], 0A6h
                mov     dword ptr [ebp-20h], 0EDh
                mov     dword ptr [ebp-1Ch], 54h
                mov     dword ptr [ebp-18h], 0A6h
                mov     dword ptr [ebp-14h], 0BBh
                mov     dword ptr [ebp-10h], 15h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 15h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49301C
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49301C:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49303D
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49303A
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49303A:
                mov     [ebp-38h], ecx

loc_49303D:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_4930A9(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0A8h
                mov     dword ptr [ebp-2Ch], 0D7h
                mov     dword ptr [ebp-28h], 16h
                mov     dword ptr [ebp-24h], 0AFh
                mov     dword ptr [ebp-20h], 77h
                mov     dword ptr [ebp-1Ch], 2Dh
                mov     dword ptr [ebp-18h], 0CCh
                mov     dword ptr [ebp-14h], 9Bh
                mov     dword ptr [ebp-10h], 8
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 8
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_493124
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_493124:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_493145
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_493142
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_493142:
                mov     [ebp-38h], ecx

loc_493145:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_4931B1(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0B3h
                mov     dword ptr [ebp-2Ch], 74h
                mov     dword ptr [ebp-28h], 6Eh
                mov     dword ptr [ebp-24h], 0F6h
                mov     dword ptr [ebp-20h], 91h
                mov     dword ptr [ebp-1Ch], 9Bh
                mov     dword ptr [ebp-18h], 0A7h
                mov     dword ptr [ebp-14h], 64h
                mov     dword ptr [ebp-10h], 3
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 3
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49322C
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49322C:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49324D
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49324A
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49324A:
                mov     [ebp-38h], ecx

loc_49324D:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4932B9(void) {  __asm  {
















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 7Ah
                mov     dword ptr [ebp-2Ch], 5
                mov     dword ptr [ebp-28h], 0Bh
                mov     dword ptr [ebp-24h], 0D6h
                mov     dword ptr [ebp-20h], 5Fh
                mov     dword ptr [ebp-1Ch], 7Bh
                mov     dword ptr [ebp-18h], 72h
                mov     dword ptr [ebp-14h], 0E0h
                mov     dword ptr [ebp-10h], 14h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 14h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_493334
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_493334:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_493355
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_493352
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_493352:
                mov     [ebp-38h], ecx

loc_493355:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4933C1(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0C8h
                mov     dword ptr [ebp-2Ch], 0C7h
                mov     dword ptr [ebp-28h], 0EBh
                mov     dword ptr [ebp-24h], 0E4h
                mov     dword ptr [ebp-20h], 7Fh
                mov     dword ptr [ebp-1Ch], 0F5h
                mov     dword ptr [ebp-18h], 34h
                mov     dword ptr [ebp-14h], 0Ah
                mov     dword ptr [ebp-10h], 8
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 8
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49343C
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49343C:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49345D
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49345A
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49345A:
                mov     [ebp-38h], ecx

loc_49345D:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4934C9(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0E6h
                mov     dword ptr [ebp-2Ch], 14h
                mov     dword ptr [ebp-28h], 72h
                mov     dword ptr [ebp-24h], 47h
                mov     dword ptr [ebp-20h], 0F7h
                mov     dword ptr [ebp-1Ch], 91h
                mov     dword ptr [ebp-18h], 0E9h
                mov     dword ptr [ebp-14h], 0E1h
                mov     dword ptr [ebp-10h], 0Fh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Fh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_493544
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_493544:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_493565
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_493562
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_493562:
```

```
                mov     [ebp-38h], ecx

loc_493565:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4935D1(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 3Ah
                mov     dword ptr [ebp-2Ch], 0Ah
                mov     dword ptr [ebp-28h], 0CAh
                mov     dword ptr [ebp-24h], 38h
                mov     dword ptr [ebp-20h], 90h
                mov     dword ptr [ebp-1Ch], 85h
                mov     dword ptr [ebp-18h], 33h
                mov     dword ptr [ebp-14h], 4Bh
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49364C
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49364C:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49366D
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49366A
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49366A:
                mov     [ebp-38h], ecx

loc_49366D:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4936D9(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0A5h
                mov     dword ptr [ebp-2Ch], 1Fh
                mov     dword ptr [ebp-28h], 1Dh
                mov     dword ptr [ebp-24h], 6Bh
                mov     dword ptr [ebp-20h], 5Bh
                mov     dword ptr [ebp-1Ch], 1Ah
                mov     dword ptr [ebp-18h], 1Ah
                mov     dword ptr [ebp-14h], 0E1h
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_493754
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_493754:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_493775
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_493772
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_493772:
                mov     [ebp-38h], ecx

loc_493775:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4937E1(void) {  __asm  {
```

```
push    ebp
mov     ebp, esp
sub     esp, 40h
mov     dword ptr [ebp-30h], 31h
mov     dword ptr [ebp-2Ch], 77h
mov     dword ptr [ebp-28h], 0D3h
mov     dword ptr [ebp-24h], 35h
mov     dword ptr [ebp-20h], 9Bh
mov     dword ptr [ebp-1Ch], 65h
mov     dword ptr [ebp-18h], 6Fh
mov     dword ptr [ebp-14h], 0F5h
mov     dword ptr [ebp-10h], 8
mov     dword ptr [ebp-40h], 7
mov     eax, [ebp+8]
shr     eax, 8
and     eax, 7
mov     ecx, [ebp+eax*4-30h]
mov     [ebp-3Ch], ecx
mov     eax, [ebp-3Ch]
cdq
and     edx, 0Fh
add     eax, edx
sar     eax, 4
mov     [ebp-34h], eax
mov     edx, [ebp-3Ch]
and     edx, 8000000Fh
jns     short loc_49385C
dec     edx
or      edx, 0FFFFFFF0h
inc     edx
```

```
loc_49385C:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49387D
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49387A
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49387A:
                mov     [ebp-38h], ecx

loc_49387D:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_4938E9(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0Dh
                mov     dword ptr [ebp-2Ch], 22h
                mov     dword ptr [ebp-28h], 52h
                mov     dword ptr [ebp-24h], 0AEh
                mov     dword ptr [ebp-20h], 65h
                mov     dword ptr [ebp-1Ch], 86h
                mov     dword ptr [ebp-18h], 9Ch
                mov     dword ptr [ebp-14h], 28h
                mov     dword ptr [ebp-10h], 0Eh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Eh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_493964
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_493964:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_493985
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_493982
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_493982:
                mov     [ebp-38h], ecx

loc_493985:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4939F1(void) {  __asm  {















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 5
                mov     dword ptr [ebp-2Ch], 32h
                mov     dword ptr [ebp-28h], 0C6h
                mov     dword ptr [ebp-24h], 2Ch
                mov     dword ptr [ebp-20h], 55h
                mov     dword ptr [ebp-1Ch], 0F3h
                mov     dword ptr [ebp-18h], 2Ch
                mov     dword ptr [ebp-14h], 0DAh
                mov     dword ptr [ebp-10h], 15h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 15h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_493A6C
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_493A6C:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_493A8D
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_493A8A
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_493A8A:
                mov     [ebp-38h], ecx

loc_493A8D:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_493AF9(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 6Dh
                mov     dword ptr [ebp-2Ch], 62h
                mov     dword ptr [ebp-28h], 61h
                mov     dword ptr [ebp-24h], 0C6h
                mov     dword ptr [ebp-20h], 39h
                mov     dword ptr [ebp-1Ch], 0A4h
                mov     dword ptr [ebp-18h], 0ACh
                mov     dword ptr [ebp-14h], 56h
                mov     dword ptr [ebp-10h], 0Ah
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ah
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_493B74
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_493B74:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_493B95
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_493B92
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_493B92:
                mov     [ebp-38h], ecx

loc_493B95:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_493C01(void) {  __asm  {




















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 61h
                mov     dword ptr [ebp-2Ch], 7Eh
                mov     dword ptr [ebp-28h], 6Ah
                mov     dword ptr [ebp-24h], 21h
                mov     dword ptr [ebp-20h], 0EFh
                mov     dword ptr [ebp-1Ch], 0D6h
                mov     dword ptr [ebp-18h], 0ABh
                mov     dword ptr [ebp-14h], 7Fh
                mov     dword ptr [ebp-10h], 3
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 3
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_493C7C
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_493C7C:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_493C9D
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_493C9A
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_493C9A:
                mov     [ebp-38h], ecx

loc_493C9D:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_493D09(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 57h
                mov     dword ptr [ebp-2Ch], 0C9h
                mov     dword ptr [ebp-28h], 0BBh
                mov     dword ptr [ebp-24h], 7Ah
                mov     dword ptr [ebp-20h], 0C1h
                mov     dword ptr [ebp-1Ch], 95h
                mov     dword ptr [ebp-18h], 9Eh
                mov     dword ptr [ebp-14h], 1Ah
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_493D84
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_493D84:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_493DA5
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_493DA2
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_493DA2:
                mov     [ebp-38h], ecx

loc_493DA5:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_493E11(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 75h
                mov     dword ptr [ebp-2Ch], 0B2h
                mov     dword ptr [ebp-28h], 85h
                mov     dword ptr [ebp-24h], 0AFh
                mov     dword ptr [ebp-20h], 4Ah
                mov     dword ptr [ebp-1Ch], 8Fh
                mov     dword ptr [ebp-18h], 0D6h
                mov     dword ptr [ebp-14h], 0F1h
                mov     dword ptr [ebp-10h], 4
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 4
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_493E8C
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_493E8C:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_493EAD
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_493EAA
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_493EAA:
                mov     [ebp-38h], ecx

loc_493EAD:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_493F19(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0E6h
                mov     dword ptr [ebp-2Ch], 85h
                mov     dword ptr [ebp-28h], 0C7h
                mov     dword ptr [ebp-24h], 72h
                mov     dword ptr [ebp-20h], 12h
                mov     dword ptr [ebp-1Ch], 5Dh
                mov     dword ptr [ebp-18h], 6Ch
                mov     dword ptr [ebp-14h], 1Dh
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_493F94
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_493F94:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_493FB5
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_493FB2
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_493FB2:
                mov     [ebp-38h], ecx
```

```
loc_493FB5:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_494021(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0CAh
                mov     dword ptr [ebp-2Ch], 0DFh
                mov     dword ptr [ebp-28h], 52h
                mov     dword ptr [ebp-24h], 27h
                mov     dword ptr [ebp-20h], 3Bh
                mov     dword ptr [ebp-1Ch], 28h
                mov     dword ptr [ebp-18h], 76h
                mov     dword ptr [ebp-14h], 87h
                mov     dword ptr [ebp-10h], 9
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 9
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49409C
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49409C:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4940BD
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4940BA
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4940BA:
                mov     [ebp-38h], ecx

loc_4940BD:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_494129(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 9
                mov     dword ptr [ebp-2Ch], 1Dh
                mov     dword ptr [ebp-28h], 5Eh
                mov     dword ptr [ebp-24h], 7Ch
                mov     dword ptr [ebp-20h], 86h
                mov     dword ptr [ebp-1Ch], 36h
                mov     dword ptr [ebp-18h], 5Ch
                mov     dword ptr [ebp-14h], 21h
                mov     dword ptr [ebp-10h], 9
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 9
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4941A4
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4941A4:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4941C5
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4941C2
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4941C2:
                mov     [ebp-38h], ecx

loc_4941C5:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_494231(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0FBh
                mov     dword ptr [ebp-2Ch], 0CBh
                mov     dword ptr [ebp-28h], 24h
                mov     dword ptr [ebp-24h], 97h
                mov     dword ptr [ebp-20h], 73h
                mov     dword ptr [ebp-1Ch], 0C5h
                mov     dword ptr [ebp-18h], 10h
                mov     dword ptr [ebp-14h], 8
                mov     dword ptr [ebp-10h], 12h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 12h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4942AC
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4942AC:
```

```
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4942CD
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4942CA
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4942CA:
                mov     [ebp-38h], ecx

loc_4942CD:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_494339(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0FBh
                mov     dword ptr [ebp-2Ch], 0D2h
                mov     dword ptr [ebp-28h], 9Bh
                mov     dword ptr [ebp-24h], 5Eh
                mov     dword ptr [ebp-20h], 65h
                mov     dword ptr [ebp-1Ch], 52h
                mov     dword ptr [ebp-18h], 0CCh
                mov     dword ptr [ebp-14h], 7
                mov     dword ptr [ebp-10h], 14h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 14h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4943B4
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4943B4:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4943D5
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4943D2
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4943D2:
                mov     [ebp-38h], ecx

loc_4943D5:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
```

```
}}

__declspec(naked) void sub_494441(void) {  __asm  {

                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 98h
                mov     dword ptr [ebp-2Ch], 0CCh
                mov     dword ptr [ebp-28h], 47h
                mov     dword ptr [ebp-24h], 0F6h
                mov     dword ptr [ebp-20h], 86h
                mov     dword ptr [ebp-1Ch], 66h
                mov     dword ptr [ebp-18h], 63h
                mov     dword ptr [ebp-14h], 71h
                mov     dword ptr [ebp-10h], 8
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 8
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4944BC
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4944BC:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4944DD
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4944DA
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4944DA:
                mov     [ebp-38h], ecx

loc_4944DD:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_494549(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 61h
                mov     dword ptr [ebp-2Ch], 91h
                mov     dword ptr [ebp-28h], 6Bh
                mov     dword ptr [ebp-24h], 0D6h
                mov     dword ptr [ebp-20h], 75h
                mov     dword ptr [ebp-1Ch], 0AAh
                mov     dword ptr [ebp-18h], 0F6h
                mov     dword ptr [ebp-14h], 0F5h
                mov     dword ptr [ebp-10h], 0Ah
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ah
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4945C4
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4945C4:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4945E5
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4945E2
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4945E2:
                mov     [ebp-38h], ecx

loc_4945E5:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_494651(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0E9h
                mov     dword ptr [ebp-2Ch], 7
                mov     dword ptr [ebp-28h], 0
                mov     dword ptr [ebp-24h], 0Fh
                mov     dword ptr [ebp-20h], 0B9h
                mov     dword ptr [ebp-1Ch], 8
                mov     dword ptr [ebp-18h], 0B7h
                mov     dword ptr [ebp-14h], 25h
                mov     dword ptr [ebp-10h], 3
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 3
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4946CC
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4946CC:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4946ED
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4946EA
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4946EA:
                mov     [ebp-38h], ecx

loc_4946ED:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_494759(void) {  __asm  {










                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 7Fh
                mov     dword ptr [ebp-2Ch], 63h
                mov     dword ptr [ebp-28h], 0C5h
                mov     dword ptr [ebp-24h], 3Dh
                mov     dword ptr [ebp-20h], 6Eh
                mov     dword ptr [ebp-1Ch], 0B1h
                mov     dword ptr [ebp-18h], 31h
                mov     dword ptr [ebp-14h], 5Ch
                mov     dword ptr [ebp-10h], 10h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 10h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4947D4
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4947D4:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4947F5
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4947F2
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4947F2:
                mov     [ebp-38h], ecx

loc_4947F5:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_494861(void) {  __asm  {















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 1Ch
                mov     dword ptr [ebp-2Ch], 68h
                mov     dword ptr [ebp-28h], 9
                mov     dword ptr [ebp-24h], 10h
                mov     dword ptr [ebp-20h], 0BDh
                mov     dword ptr [ebp-1Ch], 57h
                mov     dword ptr [ebp-18h], 53h
                mov     dword ptr [ebp-14h], 88h
                mov     dword ptr [ebp-10h], 13h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 13h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4948DC
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4948DC:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4948FD
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4948FA
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4948FA:
                mov     [ebp-38h], ecx

loc_4948FD:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_494969(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0CDh
                mov     dword ptr [ebp-2Ch], 94h
                mov     dword ptr [ebp-28h], 0FBh
                mov     dword ptr [ebp-24h], 2
                mov     dword ptr [ebp-20h], 49h
                mov     dword ptr [ebp-1Ch], 0BCh
                mov     dword ptr [ebp-18h], 6
                mov     dword ptr [ebp-14h], 0D5h
                mov     dword ptr [ebp-10h], 2
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 2
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4949E4
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4949E4:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_494A05
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_494A02
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_494A02:
                mov     [ebp-38h], ecx

loc_494A05:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_494A71(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0B2h
                mov     dword ptr [ebp-2Ch], 3Ah
                mov     dword ptr [ebp-28h], 11h
                mov     dword ptr [ebp-24h], 5Ah
                mov     dword ptr [ebp-20h], 6Eh
                mov     dword ptr [ebp-1Ch], 0EBh
                mov     dword ptr [ebp-18h], 93h
                mov     dword ptr [ebp-14h], 4Ah
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_494AEC
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_494AEC:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_494B0D
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_494B0A
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx
```

```
loc_494B0A:
                mov     [ebp-38h], ecx

loc_494B0D:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_494B79(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0F6h
                mov     dword ptr [ebp-2Ch], 32h
                mov     dword ptr [ebp-28h], 0AAh
                mov     dword ptr [ebp-24h], 17h
                mov     dword ptr [ebp-20h], 90h
                mov     dword ptr [ebp-1Ch], 83h
                mov     dword ptr [ebp-18h], 0BDh
                mov     dword ptr [ebp-14h], 1Bh
                mov     dword ptr [ebp-10h], 9
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 9
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_494BF4
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_494BF4:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_494C15
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_494C12
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_494C12:
                mov     [ebp-38h], ecx

loc_494C15:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_494C81(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0A4h
                mov     dword ptr [ebp-2Ch], 3Bh
                mov     dword ptr [ebp-28h], 30h
                mov     dword ptr [ebp-24h], 28h
                mov     dword ptr [ebp-20h], 8
                mov     dword ptr [ebp-1Ch], 0C5h
                mov     dword ptr [ebp-18h], 6Fh
                mov     dword ptr [ebp-14h], 35h
                mov     dword ptr [ebp-10h], 9
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 9
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_494CFC
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_494CFC:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_494D1D
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_494D1A
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_494D1A:
                mov     [ebp-38h], ecx

loc_494D1D:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_494D89(void) {  __asm  {









                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0B9h
                mov     dword ptr [ebp-2Ch], 0A6h
                mov     dword ptr [ebp-28h], 50h
                mov     dword ptr [ebp-24h], 0ADh
                mov     dword ptr [ebp-20h], 82h
                mov     dword ptr [ebp-1Ch], 0E0h
                mov     dword ptr [ebp-18h], 59h
                mov     dword ptr [ebp-14h], 0ABh
                mov     dword ptr [ebp-10h], 0Ah
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ah
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_494E04
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_494E04:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_494E25
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_494E22
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_494E22:
                mov     [ebp-38h], ecx

loc_494E25:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_494E91(void) {  __asm  {
```

```
push    ebp
mov     ebp, esp
sub     esp, 40h
mov     dword ptr [ebp-30h], 0B5h
mov     dword ptr [ebp-2Ch], 0B6h
mov     dword ptr [ebp-28h], 0C7h
mov     dword ptr [ebp-24h], 6Fh
mov     dword ptr [ebp-20h], 0F6h
mov     dword ptr [ebp-1Ch], 99h
mov     dword ptr [ebp-18h], 11h
mov     dword ptr [ebp-14h], 8Dh
mov     dword ptr [ebp-10h], 0Ah
mov     dword ptr [ebp-40h], 7
mov     eax, [ebp+8]
shr     eax, 0Ah
and     eax, 7
mov     ecx, [ebp+eax*4-30h]
mov     [ebp-3Ch], ecx
mov     eax, [ebp-3Ch]
cdq
and     edx, 0Fh
add     eax, edx
sar     eax, 4
mov     [ebp-34h], eax
```

```
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_494F0C
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_494F0C:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_494F2D
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_494F2A
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_494F2A:
                mov     [ebp-38h], ecx

loc_494F2D:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_494F99(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 9Dh
                mov     dword ptr [ebp-2Ch], 32h
                mov     dword ptr [ebp-28h], 37h
                mov     dword ptr [ebp-24h], 0A8h
                mov     dword ptr [ebp-20h], 0B8h
                mov     dword ptr [ebp-1Ch], 0Bh
                mov     dword ptr [ebp-18h], 3Bh
                mov     dword ptr [ebp-14h], 0C9h
                mov     dword ptr [ebp-10h], 0Dh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Dh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_495014
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_495014:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_495035
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_495032
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_495032:
                mov     [ebp-38h], ecx

loc_495035:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_4950A1(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 98h
                mov     dword ptr [ebp-2Ch], 1Bh
                mov     dword ptr [ebp-28h], 0F4h
                mov     dword ptr [ebp-24h], 3Ah
                mov     dword ptr [ebp-20h], 33h
                mov     dword ptr [ebp-1Ch], 0EEh
                mov     dword ptr [ebp-18h], 82h
                mov     dword ptr [ebp-14h], 78h
                mov     dword ptr [ebp-10h], 14h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 14h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49511C
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49511C:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49513D
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49513A
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49513A:
                mov     [ebp-38h], ecx

loc_49513D:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_4951A9(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 17h
                mov     dword ptr [ebp-2Ch], 0DEh
                mov     dword ptr [ebp-28h], 3Fh
                mov     dword ptr [ebp-24h], 0D5h
                mov     dword ptr [ebp-20h], 2Bh
                mov     dword ptr [ebp-1Ch], 44h
                mov     dword ptr [ebp-18h], 72h
                mov     dword ptr [ebp-14h], 63h
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_495224
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_495224:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_495245
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_495242
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_495242:
                mov     [ebp-38h], ecx

loc_495245:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4952B1(void) {  __asm  {
















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0F5h
                mov     dword ptr [ebp-2Ch], 0C9h
                mov     dword ptr [ebp-28h], 31h
                mov     dword ptr [ebp-24h], 2
                mov     dword ptr [ebp-20h], 0ECh
                mov     dword ptr [ebp-1Ch], 0C5h
                mov     dword ptr [ebp-18h], 0
                mov     dword ptr [ebp-14h], 0E1h
                mov     dword ptr [ebp-10h], 0Ah
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ah
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49532C
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49532C:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49534D
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49534A
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49534A:
                mov     [ebp-38h], ecx

loc_49534D:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4953B9(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 5Eh
                mov     dword ptr [ebp-2Ch], 56h
                mov     dword ptr [ebp-28h], 49h
                mov     dword ptr [ebp-24h], 0C3h
                mov     dword ptr [ebp-20h], 0EEh
                mov     dword ptr [ebp-1Ch], 0A3h
                mov     dword ptr [ebp-18h], 71h
                mov     dword ptr [ebp-14h], 43h
                mov     dword ptr [ebp-10h], 0
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_495431
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_495431:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_495452
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49544F
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49544F:
                mov     [ebp-38h], ecx

loc_495452:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4954BE(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 6Ah
                mov     dword ptr [ebp-2Ch], 9Ch
                mov     dword ptr [ebp-28h], 2Fh
                mov     dword ptr [ebp-24h], 0A6h
                mov     dword ptr [ebp-20h], 6Eh
                mov     dword ptr [ebp-1Ch], 8Fh
                mov     dword ptr [ebp-18h], 0Dh
                mov     dword ptr [ebp-14h], 0ADh
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_495539
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_495539:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49555A
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_495557
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_495557:
                mov     [ebp-38h], ecx
```

```
loc_49555A:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}





__declspec(naked) void sub_4955C6(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0E2h
                mov     dword ptr [ebp-2Ch], 65h
                mov     dword ptr [ebp-28h], 0E7h
                mov     dword ptr [ebp-24h], 72h
                mov     dword ptr [ebp-20h], 4Bh
                mov     dword ptr [ebp-1Ch], 0DBh
                mov     dword ptr [ebp-18h], 0A7h
                mov     dword ptr [ebp-14h], 12h
                mov     dword ptr [ebp-10h], 0Ch
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ch
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_495641
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_495641:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_495662
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49565F
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49565F:
                mov     [ebp-38h], ecx

loc_495662:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4956CE(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 70h
                mov     dword ptr [ebp-2Ch], 0B4h
                mov     dword ptr [ebp-28h], 0C2h
                mov     dword ptr [ebp-24h], 98h
                mov     dword ptr [ebp-20h], 1Ah
                mov     dword ptr [ebp-1Ch], 0DFh
                mov     dword ptr [ebp-18h], 22h
                mov     dword ptr [ebp-14h], 0C6h
                mov     dword ptr [ebp-10h], 0Eh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Eh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_495749
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_495749:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49576A
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_495767
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_495767:
                mov     [ebp-38h], ecx

loc_49576A:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4957D6(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 53h
                mov     dword ptr [ebp-2Ch], 0F2h
                mov     dword ptr [ebp-28h], 65h
                mov     dword ptr [ebp-24h], 93h
                mov     dword ptr [ebp-20h], 82h
                mov     dword ptr [ebp-1Ch], 62h
                mov     dword ptr [ebp-18h], 34h
                mov     dword ptr [ebp-14h], 0Bh
                mov     dword ptr [ebp-10h], 6
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 6
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_495851
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_495851:
```

```
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_495872
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49586F
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49586F:
                mov     [ebp-38h], ecx

loc_495872:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_4958DE(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0D7h
                mov     dword ptr [ebp-2Ch], 27h
                mov     dword ptr [ebp-28h], 62h
                mov     dword ptr [ebp-24h], 10h
                mov     dword ptr [ebp-20h], 0D0h
                mov     dword ptr [ebp-1Ch], 18h
                mov     dword ptr [ebp-18h], 79h
                mov     dword ptr [ebp-14h], 6Ah
                mov     dword ptr [ebp-10h], 14h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 14h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_495959
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_495959:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49597A
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_495977
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_495977:
                mov     [ebp-38h], ecx

loc_49597A:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
```

```
}}

__declspec(naked) void sub_4959E6(void) {  __asm  {

                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0E1h
                mov     dword ptr [ebp-2Ch], 6Dh
                mov     dword ptr [ebp-28h], 30h
                mov     dword ptr [ebp-24h], 36h
                mov     dword ptr [ebp-20h], 0F9h
                mov     dword ptr [ebp-1Ch], 0C9h
                mov     dword ptr [ebp-18h], 77h
                mov     dword ptr [ebp-14h], 4Ah
                mov     dword ptr [ebp-10h], 10h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 10h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_495A61
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_495A61:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_495A82
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_495A7F
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_495A7F:
                mov     [ebp-38h], ecx

loc_495A82:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_495AEE(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 6Dh
                mov     dword ptr [ebp-2Ch], 42h
                mov     dword ptr [ebp-28h], 14h
                mov     dword ptr [ebp-24h], 63h
                mov     dword ptr [ebp-20h], 59h
                mov     dword ptr [ebp-1Ch], 0B7h
                mov     dword ptr [ebp-18h], 0EAh
                mov     dword ptr [ebp-14h], 9Eh
                mov     dword ptr [ebp-10h], 10h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 10h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_495B69
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_495B69:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_495B8A
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_495B87
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_495B87:
                mov     [ebp-38h], ecx

loc_495B8A:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_495BF6(void) {  __asm  {










                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 3Ch
                mov     dword ptr [ebp-2Ch], 21h
                mov     dword ptr [ebp-28h], 36h
                mov     dword ptr [ebp-24h], 2Eh
                mov     dword ptr [ebp-20h], 0A3h
                mov     dword ptr [ebp-1Ch], 80h
                mov     dword ptr [ebp-18h], 0F0h
                mov     dword ptr [ebp-14h], 0B0h
                mov     dword ptr [ebp-10h], 0Eh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Eh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_495C71
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_495C71:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_495C92
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_495C8F
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_495C8F:
                mov     [ebp-38h], ecx

loc_495C92:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_495CFE(void) {  __asm  {



















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 5
                mov     dword ptr [ebp-2Ch], 0A9h
                mov     dword ptr [ebp-28h], 0E9h
                mov     dword ptr [ebp-24h], 83h
                mov     dword ptr [ebp-20h], 81h
                mov     dword ptr [ebp-1Ch], 36h
                mov     dword ptr [ebp-18h], 69h
                mov     dword ptr [ebp-14h], 2Dh
                mov     dword ptr [ebp-10h], 15h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 15h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_495D79
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_495D79:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_495D9A
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_495D97
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_495D97:
                mov     [ebp-38h], ecx

loc_495D9A:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_495E06(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 2
                mov     dword ptr [ebp-2Ch], 3Dh
                mov     dword ptr [ebp-28h], 0CFh
                mov     dword ptr [ebp-24h], 0DFh
                mov     dword ptr [ebp-20h], 24h
                mov     dword ptr [ebp-1Ch], 0B0h
                mov     dword ptr [ebp-18h], 0BEh
                mov     dword ptr [ebp-14h], 47h
                mov     dword ptr [ebp-10h], 11h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 11h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_495E81
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_495E81:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_495EA2
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_495E9F
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_495E9F:
                mov     [ebp-38h], ecx

loc_495EA2:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_495F0E(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0BFh
                mov     dword ptr [ebp-2Ch], 24h
                mov     dword ptr [ebp-28h], 32h
                mov     dword ptr [ebp-24h], 0C3h
                mov     dword ptr [ebp-20h], 0FBh
                mov     dword ptr [ebp-1Ch], 64h
                mov     dword ptr [ebp-18h], 87h
                mov     dword ptr [ebp-14h], 0CAh
                mov     dword ptr [ebp-10h], 4
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 4
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_495F89
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_495F89:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_495FAA
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_495FA7
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_495FA7:
                mov     [ebp-38h], ecx

loc_495FAA:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_496016(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 2Dh
                mov     dword ptr [ebp-2Ch], 44h
                mov     dword ptr [ebp-28h], 0C1h
                mov     dword ptr [ebp-24h], 4Bh
                mov     dword ptr [ebp-20h], 46h
                mov     dword ptr [ebp-1Ch], 8Ch
                mov     dword ptr [ebp-18h], 0B6h
                mov     dword ptr [ebp-14h], 0FBh
                mov     dword ptr [ebp-10h], 3
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 3
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_496091
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_496091:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4960B2
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4960AF
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx
```

```
loc_4960AF:
                mov     [ebp-38h], ecx

loc_4960B2:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49611E(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 83h
                mov     dword ptr [ebp-2Ch], 0EEh
                mov     dword ptr [ebp-28h], 36h
                mov     dword ptr [ebp-24h], 7
                mov     dword ptr [ebp-20h], 0E0h
                mov     dword ptr [ebp-1Ch], 0EFh
                mov     dword ptr [ebp-18h], 0A8h
                mov     dword ptr [ebp-14h], 0A9h
                mov     dword ptr [ebp-10h], 0Ch
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ch
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_496199
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_496199:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4961BA
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4961B7
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4961B7:
                mov     [ebp-38h], ecx

loc_4961BA:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_496226(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0CAh
                mov     dword ptr [ebp-2Ch], 0E2h
                mov     dword ptr [ebp-28h], 0B7h
                mov     dword ptr [ebp-24h], 51h
                mov     dword ptr [ebp-20h], 0F6h
                mov     dword ptr [ebp-1Ch], 34h
                mov     dword ptr [ebp-18h], 8Dh
                mov     dword ptr [ebp-14h], 0DCh
                mov     dword ptr [ebp-10h], 15h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 15h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4962A1
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4962A1:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4962C2
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4962BF
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4962BF:
                mov     [ebp-38h], ecx

loc_4962C2:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49632E(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 90h
                mov     dword ptr [ebp-2Ch], 2Ch
                mov     dword ptr [ebp-28h], 0BEh
                mov     dword ptr [ebp-24h], 34h
                mov     dword ptr [ebp-20h], 36h
                mov     dword ptr [ebp-1Ch], 28h
                mov     dword ptr [ebp-18h], 8Bh
                mov     dword ptr [ebp-14h], 0B6h
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4963A9
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4963A9:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4963CA
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4963C7
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4963C7:
                mov     [ebp-38h], ecx

loc_4963CA:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_496436(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 7Bh
                mov     dword ptr [ebp-2Ch], 0D2h
                mov     dword ptr [ebp-28h], 0F7h
                mov     dword ptr [ebp-24h], 0Ch
                mov     dword ptr [ebp-20h], 84h
                mov     dword ptr [ebp-1Ch], 0B3h
                mov     dword ptr [ebp-18h], 0BAh
                mov     dword ptr [ebp-14h], 92h
                mov     dword ptr [ebp-10h], 8
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 8
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4964B1
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4964B1:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4964D2
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4964CF
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4964CF:
                mov     [ebp-38h], ecx

loc_4964D2:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}



__declspec(naked) void sub_49653E(void) {  __asm  {



                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 44h
                mov     dword ptr [ebp-2Ch], 6Eh
                mov     dword ptr [ebp-28h], 0D2h
                mov     dword ptr [ebp-24h], 44h
                mov     dword ptr [ebp-20h], 0CBh
                mov     dword ptr [ebp-1Ch], 88h
                mov     dword ptr [ebp-18h], 9Bh
                mov     dword ptr [ebp-14h], 0C9h
                mov     dword ptr [ebp-10h], 1
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 1
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4965B8
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4965B8:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4965D9
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4965D6
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4965D6:
                mov     [ebp-38h], ecx

loc_4965D9:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_496645(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0A5h
                mov     dword ptr [ebp-2Ch], 73h
                mov     dword ptr [ebp-28h], 8Ah
                mov     dword ptr [ebp-24h], 51h
                mov     dword ptr [ebp-20h], 2Bh
                mov     dword ptr [ebp-1Ch], 0F0h
                mov     dword ptr [ebp-18h], 0C0h
                mov     dword ptr [ebp-14h], 96h
                mov     dword ptr [ebp-10h], 0Fh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Fh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4966C0
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4966C0:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4966E1
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4966DE
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4966DE:
                mov     [ebp-38h], ecx

loc_4966E1:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49674D(void) {  __asm  {




                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0F9h
                mov     dword ptr [ebp-2Ch], 8Ch
                mov     dword ptr [ebp-28h], 1Ch
                mov     dword ptr [ebp-24h], 6Eh
                mov     dword ptr [ebp-20h], 0E3h
                mov     dword ptr [ebp-1Ch], 0F5h
                mov     dword ptr [ebp-18h], 0E4h
                mov     dword ptr [ebp-14h], 81h
                mov     dword ptr [ebp-10h], 2
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 2
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4967C8
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4967C8:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4967E9
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4967E6
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4967E6:
                mov     [ebp-38h], ecx

loc_4967E9:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_496855(void) {  __asm  {






















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 20h
                mov     dword ptr [ebp-2Ch], 8Eh
                mov     dword ptr [ebp-28h], 0D9h
                mov     dword ptr [ebp-24h], 67h
                mov     dword ptr [ebp-20h], 0B9h
                mov     dword ptr [ebp-1Ch], 88h
                mov     dword ptr [ebp-18h], 71h
                mov     dword ptr [ebp-14h], 90h
                mov     dword ptr [ebp-10h], 1
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 1
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4968CF
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4968CF:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4968F0
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4968ED
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4968ED:
                mov     [ebp-38h], ecx

loc_4968F0:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49695C(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 5Fh
                mov     dword ptr [ebp-2Ch], 0F9h
                mov     dword ptr [ebp-28h], 0EFh
                mov     dword ptr [ebp-24h], 5Ah
                mov     dword ptr [ebp-20h], 12h
                mov     dword ptr [ebp-1Ch], 8Bh
                mov     dword ptr [ebp-18h], 5Dh
                mov     dword ptr [ebp-14h], 37h
                mov     dword ptr [ebp-10h], 11h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 11h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4969D7
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4969D7:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4969F8
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4969F5
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4969F5:
                mov     [ebp-38h], ecx

loc_4969F8:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}






__declspec(naked) void sub_496A64(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0DDh
                mov     dword ptr [ebp-2Ch], 0DAh
                mov     dword ptr [ebp-28h], 71h
                mov     dword ptr [ebp-24h], 58h
                mov     dword ptr [ebp-20h], 51h
                mov     dword ptr [ebp-1Ch], 96h
                mov     dword ptr [ebp-18h], 60h
                mov     dword ptr [ebp-14h], 8Dh
                mov     dword ptr [ebp-10h], 7
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 7
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_496ADF
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_496ADF:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_496B00
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_496AFD
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_496AFD:
                mov     [ebp-38h], ecx
```

```
loc_496B00:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_496B6C(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 9Bh
                mov     dword ptr [ebp-2Ch], 0D1h
                mov     dword ptr [ebp-28h], 43h
                mov     dword ptr [ebp-24h], 55h
                mov     dword ptr [ebp-20h], 89h
                mov     dword ptr [ebp-1Ch], 5Fh
                mov     dword ptr [ebp-18h], 33h
                mov     dword ptr [ebp-14h], 0E4h
                mov     dword ptr [ebp-10h], 8
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 8
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_496BE7
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_496BE7:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_496C08
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_496C05
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_496C05:
                mov     [ebp-38h], ecx

loc_496C08:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_496C74(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0CFh
                mov     dword ptr [ebp-2Ch], 76h
                mov     dword ptr [ebp-28h], 0Dh
                mov     dword ptr [ebp-24h], 0E5h
                mov     dword ptr [ebp-20h], 6Fh
                mov     dword ptr [ebp-1Ch], 19h
                mov     dword ptr [ebp-18h], 73h
                mov     dword ptr [ebp-14h], 0AAh
                mov     dword ptr [ebp-10h], 8
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 8
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_496CEF
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_496CEF:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_496D10
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_496D0D
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_496D0D:
                mov     [ebp-38h], ecx

loc_496D10:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_496D7C(void) {  __asm  {
```

```
push    ebp
mov     ebp, esp
sub     esp, 40h
mov     dword ptr [ebp-30h], 19h
mov     dword ptr [ebp-2Ch], 0Ah
mov     dword ptr [ebp-28h], 8
mov     dword ptr [ebp-24h], 0A0h
mov     dword ptr [ebp-20h], 3Dh
mov     dword ptr [ebp-1Ch], 97h
mov     dword ptr [ebp-18h], 0D7h
mov     dword ptr [ebp-14h], 94h
mov     dword ptr [ebp-10h], 0Fh
mov     dword ptr [ebp-40h], 7
mov     eax, [ebp+8]
shr     eax, 0Fh
and     eax, 7
mov     ecx, [ebp+eax*4-30h]
mov     [ebp-3Ch], ecx
mov     eax, [ebp-3Ch]
cdq
and     edx, 0Fh
add     eax, edx
sar     eax, 4
mov     [ebp-34h], eax
mov     edx, [ebp-3Ch]
and     edx, 8000000Fh
jns     short loc_496DF7
dec     edx
or      edx, 0FFFFFFF0h
inc     edx
```

```
loc_496DF7:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_496E18
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_496E15
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_496E15:
                mov     [ebp-38h], ecx

loc_496E18:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_496E84(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 38h
                mov     dword ptr [ebp-2Ch], 0BDh
                mov     dword ptr [ebp-28h], 0BEh
                mov     dword ptr [ebp-24h], 46h
                mov     dword ptr [ebp-20h], 2Fh
                mov     dword ptr [ebp-1Ch], 0C6h
                mov     dword ptr [ebp-18h], 54h
                mov     dword ptr [ebp-14h], 0BFh
                mov     dword ptr [ebp-10h], 9
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 9
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_496EFF
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_496EFF:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_496F20
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_496F1D
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_496F1D:
                mov     [ebp-38h], ecx

loc_496F20:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_496F8C(void) {  __asm  {



















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 26h
                mov     dword ptr [ebp-2Ch], 47h
                mov     dword ptr [ebp-28h], 84h
                mov     dword ptr [ebp-24h], 0F4h
                mov     dword ptr [ebp-20h], 1
                mov     dword ptr [ebp-1Ch], 2Fh
                mov     dword ptr [ebp-18h], 0C0h
                mov     dword ptr [ebp-14h], 0E8h
                mov     dword ptr [ebp-10h], 8
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 8
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_497007
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_497007:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_497028
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_497025
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_497025:
                mov     [ebp-38h], ecx

loc_497028:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_497094(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 65h
                mov     dword ptr [ebp-2Ch], 65h
                mov     dword ptr [ebp-28h], 7Bh
                mov     dword ptr [ebp-24h], 0Bh
                mov     dword ptr [ebp-20h], 0DFh
                mov     dword ptr [ebp-1Ch], 0CBh
                mov     dword ptr [ebp-18h], 0Ah
                mov     dword ptr [ebp-14h], 36h
                mov     dword ptr [ebp-10h], 4
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 4
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49710F
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49710F:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_497130
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49712D
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49712D:
                mov     [ebp-38h], ecx

loc_497130:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49719C(void) {  __asm  {














                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 84h
                mov     dword ptr [ebp-2Ch], 19h
                mov     dword ptr [ebp-28h], 7Eh
                mov     dword ptr [ebp-24h], 95h
                mov     dword ptr [ebp-20h], 5Eh
                mov     dword ptr [ebp-1Ch], 6Fh
                mov     dword ptr [ebp-18h], 94h
                mov     dword ptr [ebp-14h], 0B3h
                mov     dword ptr [ebp-10h], 10h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 10h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_497217
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_497217:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_497238
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_497235
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_497235:
                mov     [ebp-38h], ecx

loc_497238:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4972A4(void) {  __asm  {
















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0C6h
                mov     dword ptr [ebp-2Ch], 87h
                mov     dword ptr [ebp-28h], 24h
                mov     dword ptr [ebp-24h], 78h
                mov     dword ptr [ebp-20h], 92h
                mov     dword ptr [ebp-1Ch], 0ABh
                mov     dword ptr [ebp-18h], 0BBh
                mov     dword ptr [ebp-14h], 0ADh
                mov     dword ptr [ebp-10h], 9
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 9
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49731F
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49731F:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_497340
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49733D
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49733D:
                mov     [ebp-38h], ecx

loc_497340:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}





__declspec(naked) void sub_4973AC(void) {  __asm  {





                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 8
                mov     dword ptr [ebp-2Ch], 37h
                mov     dword ptr [ebp-28h], 68h
                mov     dword ptr [ebp-24h], 13h
                mov     dword ptr [ebp-20h], 7Ch
                mov     dword ptr [ebp-1Ch], 27h
                mov     dword ptr [ebp-18h], 16h
                mov     dword ptr [ebp-14h], 5Ah
                mov     dword ptr [ebp-10h], 15h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 15h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_497427
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_497427:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_497448
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_497445
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_497445:
                mov     [ebp-38h], ecx

loc_497448:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4974B4(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 8Bh
                mov     dword ptr [ebp-2Ch], 0B4h
                mov     dword ptr [ebp-28h], 45h
                mov     dword ptr [ebp-24h], 94h
                mov     dword ptr [ebp-20h], 52h
                mov     dword ptr [ebp-1Ch], 5Fh
                mov     dword ptr [ebp-18h], 0DBh
                mov     dword ptr [ebp-14h], 84h
                mov     dword ptr [ebp-10h], 0Dh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Dh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49752F
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49752F:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_497550
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49754D
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49754D:
                mov     [ebp-38h], ecx

loc_497550:
```

```
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4975BC(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 6Bh
                mov     dword ptr [ebp-2Ch], 0Bh
                mov     dword ptr [ebp-28h], 64h
                mov     dword ptr [ebp-24h], 0F6h
                mov     dword ptr [ebp-20h], 0B0h
                mov     dword ptr [ebp-1Ch], 3Bh
                mov     dword ptr [ebp-18h], 0F4h
                mov     dword ptr [ebp-14h], 0B7h
                mov     dword ptr [ebp-10h], 0
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_497634
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_497634:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_497655
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_497652
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx
```

```
loc_497652:
                mov     [ebp-38h], ecx

loc_497655:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4976C1(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 36h
                mov     dword ptr [ebp-2Ch], 92h
                mov     dword ptr [ebp-28h], 7Eh
                mov     dword ptr [ebp-24h], 7Dh
                mov     dword ptr [ebp-20h], 0FBh
                mov     dword ptr [ebp-1Ch], 0E3h
                mov     dword ptr [ebp-18h], 85h
                mov     dword ptr [ebp-14h], 5Dh
                mov     dword ptr [ebp-10h], 0Bh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Bh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49773C
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49773C:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49775D
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49775A
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49775A:
                mov     [ebp-38h], ecx

loc_49775D:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4977C9(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 30h
                mov     dword ptr [ebp-2Ch], 0DEh
                mov     dword ptr [ebp-28h], 0EDh
                mov     dword ptr [ebp-24h], 8
                mov     dword ptr [ebp-20h], 38h
                mov     dword ptr [ebp-1Ch], 8
                mov     dword ptr [ebp-18h], 0Ch
                mov     dword ptr [ebp-14h], 1
                mov     dword ptr [ebp-10h], 12h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 12h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_497844
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_497844:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_497865
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_497862
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_497862:
                mov     [ebp-38h], ecx

loc_497865:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_4978D1(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0E3h
                mov     dword ptr [ebp-2Ch], 89h
                mov     dword ptr [ebp-28h], 6
                mov     dword ptr [ebp-24h], 1Ch
                mov     dword ptr [ebp-20h], 0B9h
                mov     dword ptr [ebp-1Ch], 3Bh
                mov     dword ptr [ebp-18h], 42h
                mov     dword ptr [ebp-14h], 1Eh
                mov     dword ptr [ebp-10h], 8
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 8
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49794C
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49794C:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49796D
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49796A
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49796A:
                mov     [ebp-38h], ecx

loc_49796D:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_4979D9(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 6
                mov     dword ptr [ebp-2Ch], 0E8h
                mov     dword ptr [ebp-28h], 14h
                mov     dword ptr [ebp-24h], 0B8h
                mov     dword ptr [ebp-20h], 4
                mov     dword ptr [ebp-1Ch], 0Bh
                mov     dword ptr [ebp-18h], 0FAh
                mov     dword ptr [ebp-14h], 28h
                mov     dword ptr [ebp-10h], 0Fh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Fh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_497A54
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_497A54:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_497A75
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_497A72
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_497A72:
                mov     [ebp-38h], ecx

loc_497A75:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_497AE1(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 44h
                mov     dword ptr [ebp-2Ch], 6Ch
                mov     dword ptr [ebp-28h], 0A5h
                mov     dword ptr [ebp-24h], 0F3h
                mov     dword ptr [ebp-20h], 5Bh
                mov     dword ptr [ebp-1Ch], 0C8h
                mov     dword ptr [ebp-18h], 0E9h
                mov     dword ptr [ebp-14h], 0ABh
                mov     dword ptr [ebp-10h], 9
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 9
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_497B5C
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_497B5C:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_497B7D
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_497B7A
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_497B7A:
                mov     [ebp-38h], ecx

loc_497B7D:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_497BE9(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 8Ch
                mov     dword ptr [ebp-2Ch], 51h
                mov     dword ptr [ebp-28h], 7Dh
                mov     dword ptr [ebp-24h], 4Ch
                mov     dword ptr [ebp-20h], 48h
                mov     dword ptr [ebp-1Ch], 0D8h
                mov     dword ptr [ebp-18h], 19h
                mov     dword ptr [ebp-14h], 0A3h
                mov     dword ptr [ebp-10h], 7
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 7
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_497C64
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_497C64:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_497C85
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_497C82
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_497C82:
                mov     [ebp-38h], ecx

loc_497C85:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_497CF1(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0Dh
                mov     dword ptr [ebp-2Ch], 22h
                mov     dword ptr [ebp-28h], 18h
                mov     dword ptr [ebp-24h], 0C3h
                mov     dword ptr [ebp-20h], 0F8h
                mov     dword ptr [ebp-1Ch], 1Dh
                mov     dword ptr [ebp-18h], 0A1h
                mov     dword ptr [ebp-14h], 8
                mov     dword ptr [ebp-10h], 0Fh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Fh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_497D6C
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_497D6C:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_497D8D
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_497D8A
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_497D8A:
                mov     [ebp-38h], ecx

loc_497D8D:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_497DF9(void) {  __asm  {













                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 65h
                mov     dword ptr [ebp-2Ch], 8Ch
                mov     dword ptr [ebp-28h], 75h
                mov     dword ptr [ebp-24h], 58h
                mov     dword ptr [ebp-20h], 4
                mov     dword ptr [ebp-1Ch], 6
                mov     dword ptr [ebp-18h], 0A5h
                mov     dword ptr [ebp-14h], 48h
                mov     dword ptr [ebp-10h], 6
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 6
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_497E74
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_497E74:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_497E95
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_497E92
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_497E92:
                mov     [ebp-38h], ecx

loc_497E95:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_497F01(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0Ch
                mov     dword ptr [ebp-2Ch], 46h
                mov     dword ptr [ebp-28h], 9Ah
                mov     dword ptr [ebp-24h], 0C1h
                mov     dword ptr [ebp-20h], 0EEh
                mov     dword ptr [ebp-1Ch], 0DBh
                mov     dword ptr [ebp-18h], 0E0h
                mov     dword ptr [ebp-14h], 0B1h
                mov     dword ptr [ebp-10h], 7
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 7
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_497F7C
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_497F7C:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_497F9D
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_497F9A
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_497F9A:
                mov     [ebp-38h], ecx

loc_497F9D:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}



__declspec(naked) void sub_498009(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0C4h
                mov     dword ptr [ebp-2Ch], 0CCh
                mov     dword ptr [ebp-28h], 95h
                mov     dword ptr [ebp-24h], 0BBh
                mov     dword ptr [ebp-20h], 2Eh
                mov     dword ptr [ebp-1Ch], 0C4h
                mov     dword ptr [ebp-18h], 2Ch
                mov     dword ptr [ebp-14h], 42h
                mov     dword ptr [ebp-10h], 2
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 2
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_498084
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_498084:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4980A5
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4980A2
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4980A2:
                mov     [ebp-38h], ecx
```

```
loc_4980A5:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_498111(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 7Dh
                mov     dword ptr [ebp-2Ch], 0D4h
                mov     dword ptr [ebp-28h], 0B0h
                mov     dword ptr [ebp-24h], 20h
                mov     dword ptr [ebp-20h], 0B8h
                mov     dword ptr [ebp-1Ch], 0C0h
                mov     dword ptr [ebp-18h], 0F4h
                mov     dword ptr [ebp-14h], 0E1h
                mov     dword ptr [ebp-10h], 13h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 13h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49818C
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49818C:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4981AD
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4981AA
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4981AA:
                mov     [ebp-38h], ecx

loc_4981AD:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_498219(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 3Ch
                mov     dword ptr [ebp-2Ch], 7Fh
                mov     dword ptr [ebp-28h], 56h
                mov     dword ptr [ebp-24h], 58h
                mov     dword ptr [ebp-20h], 42h
                mov     dword ptr [ebp-1Ch], 52h
                mov     dword ptr [ebp-18h], 0D7h
                mov     dword ptr [ebp-14h], 0F9h
                mov     dword ptr [ebp-10h], 15h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 15h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_498294
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_498294:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4982B5
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4982B2
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4982B2:
                mov     [ebp-38h], ecx

loc_4982B5:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_498321(void) {  __asm  {
```

```
push    ebp
mov     ebp, esp
sub     esp, 40h
mov     dword ptr [ebp-30h], 0B3h
mov     dword ptr [ebp-2Ch], 0B9h
mov     dword ptr [ebp-28h], 64h
mov     dword ptr [ebp-24h], 8
mov     dword ptr [ebp-20h], 0C3h
mov     dword ptr [ebp-1Ch], 0B3h
mov     dword ptr [ebp-18h], 95h
mov     dword ptr [ebp-14h], 39h
mov     dword ptr [ebp-10h], 6
mov     dword ptr [ebp-40h], 7
mov     eax, [ebp+8]
shr     eax, 6
and     eax, 7
mov     ecx, [ebp+eax*4-30h]
mov     [ebp-3Ch], ecx
mov     eax, [ebp-3Ch]
cdq
and     edx, 0Fh
add     eax, edx
sar     eax, 4
mov     [ebp-34h], eax
mov     edx, [ebp-3Ch]
and     edx, 8000000Fh
jns     short loc_49839C
dec     edx
or      edx, 0FFFFFFF0h
inc     edx
```

```
loc_49839C:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4983BD
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4983BA
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4983BA:
                mov     [ebp-38h], ecx

loc_4983BD:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_498429(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 4Ah
                mov     dword ptr [ebp-2Ch], 49h
                mov     dword ptr [ebp-28h], 1
                mov     dword ptr [ebp-24h], 93h
                mov     dword ptr [ebp-20h], 9
                mov     dword ptr [ebp-1Ch], 0A2h
                mov     dword ptr [ebp-18h], 24h
                mov     dword ptr [ebp-14h], 66h
                mov     dword ptr [ebp-10h], 9
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 9
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4984A4
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4984A4:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4984C5
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4984C2
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4984C2:
                mov     [ebp-38h], ecx

loc_4984C5:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_498531(void) {  __asm  {




















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0ABh
                mov     dword ptr [ebp-2Ch], 0D4h
                mov     dword ptr [ebp-28h], 23h
                mov     dword ptr [ebp-24h], 44h
                mov     dword ptr [ebp-20h], 3Ch
                mov     dword ptr [ebp-1Ch], 6Dh
                mov     dword ptr [ebp-18h], 9Bh
                mov     dword ptr [ebp-14h], 9Eh
                mov     dword ptr [ebp-10h], 6
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 6
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4985AC
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4985AC:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4985CD
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4985CA
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4985CA:
                mov     [ebp-38h], ecx

loc_4985CD:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_498639(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 8Eh
                mov     dword ptr [ebp-2Ch], 0BFh
                mov     dword ptr [ebp-28h], 6Bh
                mov     dword ptr [ebp-24h], 0B2h
                mov     dword ptr [ebp-20h], 2Ch
                mov     dword ptr [ebp-1Ch], 40h
                mov     dword ptr [ebp-18h], 97h
                mov     dword ptr [ebp-14h], 47h
                mov     dword ptr [ebp-10h], 1
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 1
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4986B3
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4986B3:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4986D4
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4986D1
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4986D1:
                mov     [ebp-38h], ecx

loc_4986D4:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_498740(void) {  __asm  {




















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0CAh
                mov     dword ptr [ebp-2Ch], 0E1h
                mov     dword ptr [ebp-28h], 55h
                mov     dword ptr [ebp-24h], 0AFh
                mov     dword ptr [ebp-20h], 0E5h
                mov     dword ptr [ebp-1Ch], 9Eh
                mov     dword ptr [ebp-18h], 3Fh
                mov     dword ptr [ebp-14h], 53h
                mov     dword ptr [ebp-10h], 9
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 9
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4987BB
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4987BB:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4987DC
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4987D9
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4987D9:
                mov     [ebp-38h], ecx

loc_4987DC:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}



__declspec(naked) void sub_498848(void) {  __asm  {



                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 3Fh
                mov     dword ptr [ebp-2Ch], 2Fh
                mov     dword ptr [ebp-28h], 0Eh
                mov     dword ptr [ebp-24h], 19h
                mov     dword ptr [ebp-20h], 7Eh
                mov     dword ptr [ebp-1Ch], 37h
                mov     dword ptr [ebp-18h], 2Fh
                mov     dword ptr [ebp-14h], 0B4h
                mov     dword ptr [ebp-10h], 0Bh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Bh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4988C3
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4988C3:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4988E4
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4988E1
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4988E1:
                mov     [ebp-38h], ecx

loc_4988E4:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_498950(void) {  __asm  {












                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 16h
                mov     dword ptr [ebp-2Ch], 8Eh
                mov     dword ptr [ebp-28h], 37h
                mov     dword ptr [ebp-24h], 92h
                mov     dword ptr [ebp-20h], 58h
                mov     dword ptr [ebp-1Ch], 29h
                mov     dword ptr [ebp-18h], 0A1h
                mov     dword ptr [ebp-14h], 89h
                mov     dword ptr [ebp-10h], 4
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 4
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_4989CB
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_4989CB:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_4989EC
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_4989E9
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_4989E9:
                mov     [ebp-38h], ecx

loc_4989EC:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}





__declspec(naked) void sub_498A58(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0A4h
                mov     dword ptr [ebp-2Ch], 58h
                mov     dword ptr [ebp-28h], 0EBh
                mov     dword ptr [ebp-24h], 0E5h
                mov     dword ptr [ebp-20h], 50h
                mov     dword ptr [ebp-1Ch], 25h
                mov     dword ptr [ebp-18h], 44h
                mov     dword ptr [ebp-14h], 6Eh
                mov     dword ptr [ebp-10h], 0
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_498AD0
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_498AD0:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_498AF1
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_498AEE
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_498AEE:
                mov     [ebp-38h], ecx

loc_498AF1:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_498B5D(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 6
                mov     dword ptr [ebp-2Ch], 7
                mov     dword ptr [ebp-28h], 0AAh
                mov     dword ptr [ebp-24h], 0A0h
                mov     dword ptr [ebp-20h], 7Dh
                mov     dword ptr [ebp-1Ch], 12h
                mov     dword ptr [ebp-18h], 7Ah
                mov     dword ptr [ebp-14h], 21h
                mov     dword ptr [ebp-10h], 10h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 10h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_498BD8
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_498BD8:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_498BF9
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_498BF6
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx
```

```
loc_498BF6:
                mov     [ebp-38h], ecx

loc_498BF9:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_498C65(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0BDh
                mov     dword ptr [ebp-2Ch], 0A9h
                mov     dword ptr [ebp-28h], 85h
                mov     dword ptr [ebp-24h], 0A9h
                mov     dword ptr [ebp-20h], 0ECh
                mov     dword ptr [ebp-1Ch], 0E1h
                mov     dword ptr [ebp-18h], 0F9h
                mov     dword ptr [ebp-14h], 25h
                mov     dword ptr [ebp-10h], 4
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 4
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_498CE0
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_498CE0:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_498D01
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_498CFE
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_498CFE:
                mov     [ebp-38h], ecx

loc_498D01:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_498D6D(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 1Eh
                mov     dword ptr [ebp-2Ch], 7Eh
                mov     dword ptr [ebp-28h], 0DBh
                mov     dword ptr [ebp-24h], 36h
                mov     dword ptr [ebp-20h], 14h
                mov     dword ptr [ebp-1Ch], 23h
                mov     dword ptr [ebp-18h], 95h
                mov     dword ptr [ebp-14h], 83h
                mov     dword ptr [ebp-10h], 9
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 9
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_498DE8
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_498DE8:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_498E09
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_498E06
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_498E06:
                mov     [ebp-38h], ecx

loc_498E09:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_498E75(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 43h
                mov     dword ptr [ebp-2Ch], 0AEh
                mov     dword ptr [ebp-28h], 11h
                mov     dword ptr [ebp-24h], 5Dh
                mov     dword ptr [ebp-20h], 0A7h
                mov     dword ptr [ebp-1Ch], 9Dh
                mov     dword ptr [ebp-18h], 0DBh
                mov     dword ptr [ebp-14h], 14h
                mov     dword ptr [ebp-10h], 6
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 6
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_498EF0
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_498EF0:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_498F11
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_498F0E
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_498F0E:
                mov     [ebp-38h], ecx

loc_498F11:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_498F7D(void) {  __asm  {
```

```
push    ebp
mov     ebp, esp
sub     esp, 40h
mov     dword ptr [ebp-30h], 90h
mov     dword ptr [ebp-2Ch], 0A1h
mov     dword ptr [ebp-28h], 0C3h
mov     dword ptr [ebp-24h], 37h
mov     dword ptr [ebp-20h], 0E8h
mov     dword ptr [ebp-1Ch], 0B6h
mov     dword ptr [ebp-18h], 3Eh
mov     dword ptr [ebp-14h], 0A1h
mov     dword ptr [ebp-10h], 0Bh
mov     dword ptr [ebp-40h], 7
mov     eax, [ebp+8]
shr     eax, 0Bh
and     eax, 7
mov     ecx, [ebp+eax*4-30h]
mov     [ebp-3Ch], ecx
mov     eax, [ebp-3Ch]
cdq
and     edx, 0Fh
add     eax, edx
sar     eax, 4
mov     [ebp-34h], eax
```

```
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_498FF8
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_498FF8:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_499019
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_499016
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_499016:
                mov     [ebp-38h], ecx

loc_499019:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_499085(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0EDh
                mov     dword ptr [ebp-2Ch], 0B1h
                mov     dword ptr [ebp-28h], 0FAh
                mov     dword ptr [ebp-24h], 6Fh
                mov     dword ptr [ebp-20h], 76h
                mov     dword ptr [ebp-1Ch], 49h
                mov     dword ptr [ebp-18h], 7Ah
                mov     dword ptr [ebp-14h], 6Dh
                mov     dword ptr [ebp-10h], 3
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 3
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_499100
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_499100:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_499121
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49911E
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49911E:
                mov     [ebp-38h], ecx

loc_499121:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49918D(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 79h
                mov     dword ptr [ebp-2Ch], 0A3h
                mov     dword ptr [ebp-28h], 0CCh
                mov     dword ptr [ebp-24h], 38h
                mov     dword ptr [ebp-20h], 3
                mov     dword ptr [ebp-1Ch], 0B6h
                mov     dword ptr [ebp-18h], 50h
                mov     dword ptr [ebp-14h], 0E7h
                mov     dword ptr [ebp-10h], 0Dh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Dh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_499208
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_499208:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_499229
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_499226
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_499226:
                mov     [ebp-38h], ecx

loc_499229:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_499295(void) {  __asm  {













                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 99h
                mov     dword ptr [ebp-2Ch], 75h
                mov     dword ptr [ebp-28h], 8Bh
                mov     dword ptr [ebp-24h], 0F5h
                mov     dword ptr [ebp-20h], 0F2h
                mov     dword ptr [ebp-1Ch], 0E1h
                mov     dword ptr [ebp-18h], 4Eh
                mov     dword ptr [ebp-14h], 1Dh
                mov     dword ptr [ebp-10h], 10h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 10h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_499310
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_499310:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_499331
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49932E
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49932E:
                mov     [ebp-38h], ecx

loc_499331:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49939D(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0D3h
                mov     dword ptr [ebp-2Ch], 8Fh
                mov     dword ptr [ebp-28h], 0F9h
                mov     dword ptr [ebp-24h], 0B3h
                mov     dword ptr [ebp-20h], 0DCh
                mov     dword ptr [ebp-1Ch], 69h
                mov     dword ptr [ebp-18h], 37h
                mov     dword ptr [ebp-14h], 9Eh
                mov     dword ptr [ebp-10h], 2
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 2
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_499418
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_499418:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_499439
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_499436
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_499436:
                mov     [ebp-38h], ecx

loc_499439:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4994A5(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 2Ch
                mov     dword ptr [ebp-2Ch], 0CDh
                mov     dword ptr [ebp-28h], 37h
                mov     dword ptr [ebp-24h], 3Fh
                mov     dword ptr [ebp-20h], 4Fh
                mov     dword ptr [ebp-1Ch], 63h
                mov     dword ptr [ebp-18h], 33h
                mov     dword ptr [ebp-14h], 0AAh
                mov     dword ptr [ebp-10h], 0
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49951D
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49951D:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49953E
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49953B
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49953B:
                mov     [ebp-38h], ecx

loc_49953E:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}





__declspec(naked) void sub_4995AA(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0D7h
                mov     dword ptr [ebp-2Ch], 19h
                mov     dword ptr [ebp-28h], 0E0h
                mov     dword ptr [ebp-24h], 71h
                mov     dword ptr [ebp-20h], 76h
                mov     dword ptr [ebp-1Ch], 0B4h
                mov     dword ptr [ebp-18h], 78h
                mov     dword ptr [ebp-14h], 0CAh
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_499625
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_499625:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_499646
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_499643
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_499643:
                mov     [ebp-38h], ecx
```

```
loc_499646:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}





__declspec(naked) void sub_4996B2(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 8Ch
                mov     dword ptr [ebp-2Ch], 0A4h
                mov     dword ptr [ebp-28h], 0E3h
                mov     dword ptr [ebp-24h], 0C5h
                mov     dword ptr [ebp-20h], 0E6h
                mov     dword ptr [ebp-1Ch], 8Fh
                mov     dword ptr [ebp-18h], 0EBh
                mov     dword ptr [ebp-14h], 0D9h
                mov     dword ptr [ebp-10h], 0Ah
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ah
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49972D
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49972D:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49974E
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49974B
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49974B:
                mov     [ebp-38h], ecx

loc_49974E:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4997BA(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 7
                mov     dword ptr [ebp-2Ch], 0DBh
                mov     dword ptr [ebp-28h], 5Eh
                mov     dword ptr [ebp-24h], 2Eh
                mov     dword ptr [ebp-20h], 0AAh
                mov     dword ptr [ebp-1Ch], 0B4h
                mov     dword ptr [ebp-18h], 0
                mov     dword ptr [ebp-14h], 98h
                mov     dword ptr [ebp-10h], 4
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 4
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_499835
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_499835:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_499856
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_499853
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_499853:
                mov     [ebp-38h], ecx

loc_499856:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_4998C2(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0E0h
                mov     dword ptr [ebp-2Ch], 0C3h
                mov     dword ptr [ebp-28h], 0A1h
                mov     dword ptr [ebp-24h], 1Dh
                mov     dword ptr [ebp-20h], 3Dh
                mov     dword ptr [ebp-1Ch], 77h
                mov     dword ptr [ebp-18h], 0F0h
                mov     dword ptr [ebp-14h], 0CFh
                mov     dword ptr [ebp-10h], 11h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 11h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49993D
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49993D:
```

```
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49995E
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49995B
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49995B:
                mov     [ebp-38h], ecx

loc_49995E:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_4999CA(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 6
                mov     dword ptr [ebp-2Ch], 12h
                mov     dword ptr [ebp-28h], 0F6h
                mov     dword ptr [ebp-24h], 5Bh
                mov     dword ptr [ebp-20h], 13h
                mov     dword ptr [ebp-1Ch], 2Eh
                mov     dword ptr [ebp-18h], 2Dh
                mov     dword ptr [ebp-14h], 56h
                mov     dword ptr [ebp-10h], 13h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 13h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_499A45
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_499A45:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_499A66
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_499A63
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_499A63:
                mov     [ebp-38h], ecx

loc_499A66:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
```

```
}}

__declspec(naked) void sub_499AD2(void) {  __asm  {

                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 17h
                mov     dword ptr [ebp-2Ch], 0CFh
                mov     dword ptr [ebp-28h], 3
                mov     dword ptr [ebp-24h], 0DEh
                mov     dword ptr [ebp-20h], 82h
                mov     dword ptr [ebp-1Ch], 68h
                mov     dword ptr [ebp-18h], 29h
                mov     dword ptr [ebp-14h], 5Fh
                mov     dword ptr [ebp-10h], 9
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 9
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_499B4D
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_499B4D:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_499B6E
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_499B6B
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_499B6B:
                mov     [ebp-38h], ecx

loc_499B6E:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_499BDA(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0B8h
                mov     dword ptr [ebp-2Ch], 40h
                mov     dword ptr [ebp-28h], 69h
                mov     dword ptr [ebp-24h], 21h
                mov     dword ptr [ebp-20h], 0F7h
                mov     dword ptr [ebp-1Ch], 67h
                mov     dword ptr [ebp-18h], 0EDh
                mov     dword ptr [ebp-14h], 0AAh
                mov     dword ptr [ebp-10h], 0
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_499C52
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_499C52:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_499C73
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_499C70
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_499C70:
                mov     [ebp-38h], ecx

loc_499C73:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_499CDF(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 92h
                mov     dword ptr [ebp-2Ch], 0BCh
                mov     dword ptr [ebp-28h], 3Ch
                mov     dword ptr [ebp-24h], 0F6h
                mov     dword ptr [ebp-20h], 0CDh
                mov     dword ptr [ebp-1Ch], 28h
                mov     dword ptr [ebp-18h], 68h
                mov     dword ptr [ebp-14h], 84h
                mov     dword ptr [ebp-10h], 0Eh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Eh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_499D5A
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_499D5A:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_499D7B
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_499D78
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_499D78:
                mov     [ebp-38h], ecx

loc_499D7B:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_499DE7(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0DEh
                mov     dword ptr [ebp-2Ch], 43h
                mov     dword ptr [ebp-28h], 2Fh
                mov     dword ptr [ebp-24h], 31h
                mov     dword ptr [ebp-20h], 54h
                mov     dword ptr [ebp-1Ch], 71h
                mov     dword ptr [ebp-18h], 0Ah
                mov     dword ptr [ebp-14h], 0A8h
                mov     dword ptr [ebp-10h], 10h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 10h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_499E62
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_499E62:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_499E83
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_499E80
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_499E80:
                mov     [ebp-38h], ecx

loc_499E83:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_499EEF(void) {  __asm  {













                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 5Dh
                mov     dword ptr [ebp-2Ch], 5Eh
                mov     dword ptr [ebp-28h], 8Eh
                mov     dword ptr [ebp-24h], 2Ch
                mov     dword ptr [ebp-20h], 40h
                mov     dword ptr [ebp-1Ch], 86h
                mov     dword ptr [ebp-18h], 0D1h
                mov     dword ptr [ebp-14h], 9Bh
                mov     dword ptr [ebp-10h], 8
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 8
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_499F6A
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_499F6A:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_499F8B
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_499F88
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_499F88:
                mov     [ebp-38h], ecx

loc_499F8B:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_499FF7(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 3Ah
                mov     dword ptr [ebp-2Ch], 8Eh
                mov     dword ptr [ebp-28h], 5Bh
                mov     dword ptr [ebp-24h], 87h
                mov     dword ptr [ebp-20h], 0Ah
                mov     dword ptr [ebp-1Ch], 6Ch
                mov     dword ptr [ebp-18h], 6Ah
                mov     dword ptr [ebp-14h], 4Ch
                mov     dword ptr [ebp-10h], 8
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 8
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49A072
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49A072:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49A093
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49A090
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49A090:
                mov     [ebp-38h], ecx

loc_49A093:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49A0FF(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 13h
                mov     dword ptr [ebp-2Ch], 0C4h
                mov     dword ptr [ebp-28h], 0FAh
                mov     dword ptr [ebp-24h], 0E2h
                mov     dword ptr [ebp-20h], 7
                mov     dword ptr [ebp-1Ch], 69h
                mov     dword ptr [ebp-18h], 2Dh
                mov     dword ptr [ebp-14h], 0FAh
                mov     dword ptr [ebp-10h], 4
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 4
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49A17A
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49A17A:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49A19B
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49A198
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49A198:
```

```
                mov     [ebp-38h], ecx

loc_49A19B:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49A207(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0F3h
                mov     dword ptr [ebp-2Ch], 84h
                mov     dword ptr [ebp-28h], 71h
                mov     dword ptr [ebp-24h], 3
                mov     dword ptr [ebp-20h], 0CCh
                mov     dword ptr [ebp-1Ch], 34h
                mov     dword ptr [ebp-18h], 32h
                mov     dword ptr [ebp-14h], 6Bh
                mov     dword ptr [ebp-10h], 13h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 13h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49A282
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49A282:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49A2A3
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49A2A0
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49A2A0:
                mov     [ebp-38h], ecx

loc_49A2A3:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49A30F(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 37h
                mov     dword ptr [ebp-2Ch], 0EBh
                mov     dword ptr [ebp-28h], 46h
                mov     dword ptr [ebp-24h], 0ADh
                mov     dword ptr [ebp-20h], 0B3h
                mov     dword ptr [ebp-1Ch], 39h
                mov     dword ptr [ebp-18h], 4Ah
                mov     dword ptr [ebp-14h], 9Dh
                mov     dword ptr [ebp-10h], 14h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 14h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49A38A
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49A38A:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49A3AB
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49A3A8
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49A3A8:
                mov     [ebp-38h], ecx

loc_49A3AB:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49A417(void) {  __asm  {
```

```
push    ebp
mov     ebp, esp
sub     esp, 40h
mov     dword ptr [ebp-30h], 0E0h
mov     dword ptr [ebp-2Ch], 72h
mov     dword ptr [ebp-28h], 74h
mov     dword ptr [ebp-24h], 4Eh
mov     dword ptr [ebp-20h], 48h
mov     dword ptr [ebp-1Ch], 0D8h
mov     dword ptr [ebp-18h], 4Eh
mov     dword ptr [ebp-14h], 57h
mov     dword ptr [ebp-10h], 7
mov     dword ptr [ebp-40h], 7
mov     eax, [ebp+8]
shr     eax, 7
and     eax, 7
mov     ecx, [ebp+eax*4-30h]
mov     [ebp-3Ch], ecx
mov     eax, [ebp-3Ch]
cdq
and     edx, 0Fh
add     eax, edx
sar     eax, 4
mov     [ebp-34h], eax
mov     edx, [ebp-3Ch]
and     edx, 8000000Fh
jns     short loc_49A492
dec     edx
or      edx, 0FFFFFFF0h
inc     edx
```

```
loc_49A492:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49A4B3
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49A4B0
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49A4B0:
                mov     [ebp-38h], ecx

loc_49A4B3:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49A51F(void) {  __asm  {
```

```
push    ebp
mov     ebp, esp
sub     esp, 40h
mov     dword ptr [ebp-30h], 8Bh
mov     dword ptr [ebp-2Ch], 0D9h
mov     dword ptr [ebp-28h], 0E7h
mov     dword ptr [ebp-24h], 4
mov     dword ptr [ebp-20h], 9Bh
mov     dword ptr [ebp-1Ch], 58h
mov     dword ptr [ebp-18h], 5Bh
mov     dword ptr [ebp-14h], 0E6h
mov     dword ptr [ebp-10h], 13h
mov     dword ptr [ebp-40h], 7
mov     eax, [ebp+8]
shr     eax, 13h
and     eax, 7
mov     ecx, [ebp+eax*4-30h]
mov     [ebp-3Ch], ecx
mov     eax, [ebp-3Ch]
cdq
and     edx, 0Fh
add     eax, edx
sar     eax, 4
mov     [ebp-34h], eax
mov     edx, [ebp-3Ch]
```

```
                and     edx, 8000000Fh
                jns     short loc_49A59A
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49A59A:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49A5BB
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49A5B8
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49A5B8:
                mov     [ebp-38h], ecx

loc_49A5BB:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49A627(void) {  __asm  {















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 8Ah
                mov     dword ptr [ebp-2Ch], 49h
                mov     dword ptr [ebp-28h], 0CDh
                mov     dword ptr [ebp-24h], 0C3h
                mov     dword ptr [ebp-20h], 0ABh
                mov     dword ptr [ebp-1Ch], 0D3h
                mov     dword ptr [ebp-18h], 5Ah
                mov     dword ptr [ebp-14h], 0D2h
                mov     dword ptr [ebp-10h], 0Ah
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ah
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49A6A2
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49A6A2:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49A6C3
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49A6C0
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49A6C0:
                mov     [ebp-38h], ecx

loc_49A6C3:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}





__declspec(naked) void sub_49A72F(void) {  __asm  {
















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 8
                mov     dword ptr [ebp-2Ch], 0C5h
                mov     dword ptr [ebp-28h], 0C5h
                mov     dword ptr [ebp-24h], 0FAh
                mov     dword ptr [ebp-20h], 26h
                mov     dword ptr [ebp-1Ch], 0CCh
                mov     dword ptr [ebp-18h], 13h
                mov     dword ptr [ebp-14h], 9
                mov     dword ptr [ebp-10h], 1
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 1
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49A7A9
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49A7A9:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49A7CA
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49A7C7
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49A7C7:
                mov     [ebp-38h], ecx

loc_49A7CA:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49A836(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 81h
                mov     dword ptr [ebp-2Ch], 0A8h
                mov     dword ptr [ebp-28h], 9Ch
                mov     dword ptr [ebp-24h], 78h
                mov     dword ptr [ebp-20h], 68h
                mov     dword ptr [ebp-1Ch], 17h
                mov     dword ptr [ebp-18h], 2Eh
                mov     dword ptr [ebp-14h], 0CEh
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49A8B1
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49A8B1:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49A8D2
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49A8CF
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49A8CF:
                mov     [ebp-38h], ecx

loc_49A8D2:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49A93E(void) {  __asm  {








                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0F7h
                mov     dword ptr [ebp-2Ch], 8Dh
                mov     dword ptr [ebp-28h], 8Ah
                mov     dword ptr [ebp-24h], 0E8h
                mov     dword ptr [ebp-20h], 0F9h
                mov     dword ptr [ebp-1Ch], 2
                mov     dword ptr [ebp-18h], 0FBh
                mov     dword ptr [ebp-14h], 39h
                mov     dword ptr [ebp-10h], 0Fh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Fh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49A9B9
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49A9B9:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49A9DA
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49A9D7
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49A9D7:
                mov     [ebp-38h], ecx

loc_49A9DA:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49AA46(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 6Ah
                mov     dword ptr [ebp-2Ch], 4Bh
                mov     dword ptr [ebp-28h], 0B4h
                mov     dword ptr [ebp-24h], 90h
                mov     dword ptr [ebp-20h], 0
                mov     dword ptr [ebp-1Ch], 2Ch
                mov     dword ptr [ebp-18h], 0DBh
                mov     dword ptr [ebp-14h], 47h
                mov     dword ptr [ebp-10h], 6
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 6
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49AAC1
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49AAC1:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49AAE2
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49AADF
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49AADF:
                mov     [ebp-38h], ecx

loc_49AAE2:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49AB4E(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 5Fh
                mov     dword ptr [ebp-2Ch], 69h
                mov     dword ptr [ebp-28h], 9Ch
                mov     dword ptr [ebp-24h], 5Bh
                mov     dword ptr [ebp-20h], 2Ch
                mov     dword ptr [ebp-1Ch], 0C6h
                mov     dword ptr [ebp-18h], 0B7h
                mov     dword ptr [ebp-14h], 0D5h
                mov     dword ptr [ebp-10h], 0Ch
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ch
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49ABC9
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49ABC9:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49ABEA
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49ABE7
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49ABE7:
                mov     [ebp-38h], ecx
```

```
loc_49ABEA:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49AC56(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 39h
                mov     dword ptr [ebp-2Ch], 59h
                mov     dword ptr [ebp-28h], 45h
                mov     dword ptr [ebp-24h], 0DEh
                mov     dword ptr [ebp-20h], 4Fh
                mov     dword ptr [ebp-1Ch], 0D4h
                mov     dword ptr [ebp-18h], 94h
                mov     dword ptr [ebp-14h], 2Ch
                mov     dword ptr [ebp-10h], 14h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 14h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49ACD1
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49ACD1:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49ACF2
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49ACEF
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49ACEF:
                mov     [ebp-38h], ecx

loc_49ACF2:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49AD5E(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 5Dh
                mov     dword ptr [ebp-2Ch], 0E0h
                mov     dword ptr [ebp-28h], 6Fh
                mov     dword ptr [ebp-24h], 5Fh
                mov     dword ptr [ebp-20h], 49h
                mov     dword ptr [ebp-1Ch], 62h
                mov     dword ptr [ebp-18h], 2Eh
                mov     dword ptr [ebp-14h], 0B4h
                mov     dword ptr [ebp-10h], 7
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 7
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49ADD9
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49ADD9:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49ADFA
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49ADF7
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49ADF7:
                mov     [ebp-38h], ecx

loc_49ADFA:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49AE66(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 8Dh
                mov     dword ptr [ebp-2Ch], 0F6h
                mov     dword ptr [ebp-28h], 0A1h
                mov     dword ptr [ebp-24h], 0D1h
                mov     dword ptr [ebp-20h], 14h
                mov     dword ptr [ebp-1Ch], 0BCh
                mov     dword ptr [ebp-18h], 7Eh
                mov     dword ptr [ebp-14h], 0A7h
                mov     dword ptr [ebp-10h], 15h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 15h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49AEE1
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49AEE1:
```

```
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49AF02
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49AEFF
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49AEFF:
                mov     [ebp-38h], ecx

loc_49AF02:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49AF6E(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0A3h
                mov     dword ptr [ebp-2Ch], 2Ch
                mov     dword ptr [ebp-28h], 88h
                mov     dword ptr [ebp-24h], 0Fh
                mov     dword ptr [ebp-20h], 0E1h
                mov     dword ptr [ebp-1Ch], 38h
                mov     dword ptr [ebp-18h], 8Fh
                mov     dword ptr [ebp-14h], 7Eh
                mov     dword ptr [ebp-10h], 6
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 6
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49AFE9
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49AFE9:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49B00A
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49B007
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49B007:
                mov     [ebp-38h], ecx

loc_49B00A:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
```

```
}}
```

```
__declspec(naked) void sub_49B076(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 55h
                mov     dword ptr [ebp-2Ch], 0FBh
                mov     dword ptr [ebp-28h], 85h
                mov     dword ptr [ebp-24h], 0B9h
                mov     dword ptr [ebp-20h], 8
                mov     dword ptr [ebp-1Ch], 0A1h
                mov     dword ptr [ebp-18h], 0EEh
                mov     dword ptr [ebp-14h], 56h
                mov     dword ptr [ebp-10h], 14h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 14h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49B0F1
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49B0F1:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49B112
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49B10F
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49B10F:
                mov     [ebp-38h], ecx

loc_49B112:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49B17E(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0D0h
                mov     dword ptr [ebp-2Ch], 6Dh
                mov     dword ptr [ebp-28h], 18h
                mov     dword ptr [ebp-24h], 0E9h
                mov     dword ptr [ebp-20h], 50h
                mov     dword ptr [ebp-1Ch], 7Fh
                mov     dword ptr [ebp-18h], 42h
                mov     dword ptr [ebp-14h], 0C5h
                mov     dword ptr [ebp-10h], 0Eh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Eh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49B1F9
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49B1F9:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49B21A
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49B217
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49B217:
                mov     [ebp-38h], ecx

loc_49B21A:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49B286(void) {  __asm  {
















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 7Fh
                mov     dword ptr [ebp-2Ch], 42h
                mov     dword ptr [ebp-28h], 6Bh
                mov     dword ptr [ebp-24h], 54h
                mov     dword ptr [ebp-20h], 9Bh
                mov     dword ptr [ebp-1Ch], 74h
                mov     dword ptr [ebp-18h], 5Fh
                mov     dword ptr [ebp-14h], 84h
                mov     dword ptr [ebp-10h], 8
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 8
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49B301
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49B301:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49B322
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49B31F
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49B31F:
                mov     [ebp-38h], ecx

loc_49B322:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49B38E(void) {  __asm  {




















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 2Ah
                mov     dword ptr [ebp-2Ch], 0F4h
                mov     dword ptr [ebp-28h], 0D2h
                mov     dword ptr [ebp-24h], 25h
                mov     dword ptr [ebp-20h], 0F0h
                mov     dword ptr [ebp-1Ch], 45h
                mov     dword ptr [ebp-18h], 78h
                mov     dword ptr [ebp-14h], 0C1h
                mov     dword ptr [ebp-10h], 10h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 10h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49B409
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49B409:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49B42A
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49B427
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49B427:
                mov     [ebp-38h], ecx

loc_49B42A:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49B496(void) {  __asm  {




















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 16h
                mov     dword ptr [ebp-2Ch], 28h
                mov     dword ptr [ebp-28h], 21h
                mov     dword ptr [ebp-24h], 0EFh
                mov     dword ptr [ebp-20h], 5Dh
                mov     dword ptr [ebp-1Ch], 60h
                mov     dword ptr [ebp-18h], 8Dh
                mov     dword ptr [ebp-14h], 76h
                mov     dword ptr [ebp-10h], 0Ah
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ah
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49B511
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49B511:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49B532
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49B52F
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49B52F:
                mov     [ebp-38h], ecx

loc_49B532:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49B59E(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0EDh
                mov     dword ptr [ebp-2Ch], 0C9h
                mov     dword ptr [ebp-28h], 5Eh
                mov     dword ptr [ebp-24h], 64h
                mov     dword ptr [ebp-20h], 5Dh
                mov     dword ptr [ebp-1Ch], 3Ch
                mov     dword ptr [ebp-18h], 0AFh
                mov     dword ptr [ebp-14h], 0B0h
                mov     dword ptr [ebp-10h], 0Dh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Dh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49B619
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49B619:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49B63A
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49B637
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49B637:
                mov     [ebp-38h], ecx

loc_49B63A:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49B6A6(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 1Eh
                mov     dword ptr [ebp-2Ch], 0DBh
                mov     dword ptr [ebp-28h], 8Fh
                mov     dword ptr [ebp-24h], 0C6h
                mov     dword ptr [ebp-20h], 7Ah
                mov     dword ptr [ebp-1Ch], 89h
                mov     dword ptr [ebp-18h], 0BFh
                mov     dword ptr [ebp-14h], 2Ch
                mov     dword ptr [ebp-10h], 0Dh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Dh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49B721
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49B721:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49B742
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49B73F
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx
```

```
loc_49B73F:
                mov     [ebp-38h], ecx

loc_49B742:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49B7AE(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 52h
                mov     dword ptr [ebp-2Ch], 0B9h
                mov     dword ptr [ebp-28h], 46h
                mov     dword ptr [ebp-24h], 24h
                mov     dword ptr [ebp-20h], 89h
                mov     dword ptr [ebp-1Ch], 0B4h
                mov     dword ptr [ebp-18h], 23h
                mov     dword ptr [ebp-14h], 0C9h
                mov     dword ptr [ebp-10h], 12h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 12h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49B829
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49B829:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49B84A
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49B847
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49B847:
                mov     [ebp-38h], ecx

loc_49B84A:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49B8B6(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 75h
                mov     dword ptr [ebp-2Ch], 13h
                mov     dword ptr [ebp-28h], 0D3h
                mov     dword ptr [ebp-24h], 0DBh
                mov     dword ptr [ebp-20h], 57h
                mov     dword ptr [ebp-1Ch], 13h
                mov     dword ptr [ebp-18h], 0E8h
                mov     dword ptr [ebp-14h], 8Ah
                mov     dword ptr [ebp-10h], 0Ch
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ch
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49B931
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49B931:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49B952
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49B94F
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49B94F:
                mov     [ebp-38h], ecx

loc_49B952:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49B9BE(void) {  __asm  {
```

```
push    ebp
mov     ebp, esp
sub     esp, 40h
mov     dword ptr [ebp-30h], 38h
mov     dword ptr [ebp-2Ch], 0D9h
mov     dword ptr [ebp-28h], 0C0h
mov     dword ptr [ebp-24h], 1Fh
mov     dword ptr [ebp-20h], 26h
mov     dword ptr [ebp-1Ch], 78h
mov     dword ptr [ebp-18h], 0F7h
mov     dword ptr [ebp-14h], 0D2h
mov     dword ptr [ebp-10h], 14h
mov     dword ptr [ebp-40h], 7
mov     eax, [ebp+8]
shr     eax, 14h
and     eax, 7
mov     ecx, [ebp+eax*4-30h]
mov     [ebp-3Ch], ecx
mov     eax, [ebp-3Ch]
cdq
and     edx, 0Fh
add     eax, edx
sar     eax, 4
mov     [ebp-34h], eax
mov     edx, [ebp-3Ch]
and     edx, 8000000Fh
jns     short loc_49BA39
dec     edx
or      edx, 0FFFFFFF0h
```

```
                inc     edx

loc_49BA39:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49BA5A
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49BA57
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49BA57:
                mov     [ebp-38h], ecx

loc_49BA5A:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49BAC6(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0BFh
                mov     dword ptr [ebp-2Ch], 6Dh
                mov     dword ptr [ebp-28h], 30h
                mov     dword ptr [ebp-24h], 0B2h
                mov     dword ptr [ebp-20h], 0DEh
                mov     dword ptr [ebp-1Ch], 0E0h
                mov     dword ptr [ebp-18h], 0
                mov     dword ptr [ebp-14h], 0EFh
                mov     dword ptr [ebp-10h], 11h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 11h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49BB41
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49BB41:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49BB62
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49BB5F
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49BB5F:
                mov     [ebp-38h], ecx

loc_49BB62:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49BBCE(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0ECh
                mov     dword ptr [ebp-2Ch], 0EEh
                mov     dword ptr [ebp-28h], 0F9h
                mov     dword ptr [ebp-24h], 94h
                mov     dword ptr [ebp-20h], 75h
                mov     dword ptr [ebp-1Ch], 70h
                mov     dword ptr [ebp-18h], 16h
                mov     dword ptr [ebp-14h], 85h
                mov     dword ptr [ebp-10h], 15h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 15h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49BC49
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49BC49:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49BC6A
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49BC67
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49BC67:
                mov     [ebp-38h], ecx

loc_49BC6A:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49BCD6(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 89h
                mov     dword ptr [ebp-2Ch], 0AEh
                mov     dword ptr [ebp-28h], 9
                mov     dword ptr [ebp-24h], 44h
                mov     dword ptr [ebp-20h], 36h
                mov     dword ptr [ebp-1Ch], 0D9h
                mov     dword ptr [ebp-18h], 0A7h
                mov     dword ptr [ebp-14h], 0B1h
                mov     dword ptr [ebp-10h], 14h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 14h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49BD51
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49BD51:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49BD72
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49BD6F
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49BD6F:
                mov     [ebp-38h], ecx

loc_49BD72:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}



__declspec(naked) void sub_49BDDE(void) {  __asm  {








                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 1Eh
                mov     dword ptr [ebp-2Ch], 0C7h
                mov     dword ptr [ebp-28h], 50h
                mov     dword ptr [ebp-24h], 18h
                mov     dword ptr [ebp-20h], 89h
                mov     dword ptr [ebp-1Ch], 0EDh
                mov     dword ptr [ebp-18h], 0DDh
                mov     dword ptr [ebp-14h], 14h
                mov     dword ptr [ebp-10h], 0Dh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Dh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49BE59
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49BE59:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49BE7A
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49BE77
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49BE77:
                mov     [ebp-38h], ecx

loc_49BE7A:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49BEE6(void) {  __asm  {




                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0D9h
                mov     dword ptr [ebp-2Ch], 45h
                mov     dword ptr [ebp-28h], 0C5h
                mov     dword ptr [ebp-24h], 0CFh
                mov     dword ptr [ebp-20h], 0C9h
                mov     dword ptr [ebp-1Ch], 8Dh
                mov     dword ptr [ebp-18h], 0E3h
                mov     dword ptr [ebp-14h], 0DCh
                mov     dword ptr [ebp-10h], 0Fh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Fh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49BF61
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49BF61:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49BF82
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49BF7F
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49BF7F:
                mov     [ebp-38h], ecx

loc_49BF82:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49BFEE(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 75h
                mov     dword ptr [ebp-2Ch], 70h
                mov     dword ptr [ebp-28h], 70h
                mov     dword ptr [ebp-24h], 49h
                mov     dword ptr [ebp-20h], 7
                mov     dword ptr [ebp-1Ch], 33h
                mov     dword ptr [ebp-18h], 6Eh
                mov     dword ptr [ebp-14h], 3Eh
                mov     dword ptr [ebp-10h], 0Dh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Dh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49C069
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49C069:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49C08A
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49C087
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49C087:
                mov     [ebp-38h], ecx

loc_49C08A:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49C0F6(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 2Ch
                mov     dword ptr [ebp-2Ch], 34h
                mov     dword ptr [ebp-28h], 0E6h
                mov     dword ptr [ebp-24h], 96h
                mov     dword ptr [ebp-20h], 0D8h
                mov     dword ptr [ebp-1Ch], 7Bh
                mov     dword ptr [ebp-18h], 0EBh
                mov     dword ptr [ebp-14h], 3Fh
                mov     dword ptr [ebp-10h], 0
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49C16E
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49C16E:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49C18F
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49C18C
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49C18C:
                mov     [ebp-38h], ecx
```

```
loc_49C18F:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49C1FB(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 36h
                mov     dword ptr [ebp-2Ch], 0DEh
                mov     dword ptr [ebp-28h], 76h
                mov     dword ptr [ebp-24h], 6Ah
                mov     dword ptr [ebp-20h], 97h
                mov     dword ptr [ebp-1Ch], 87h
                mov     dword ptr [ebp-18h], 0CDh
                mov     dword ptr [ebp-14h], 4
                mov     dword ptr [ebp-10h], 6
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 6
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49C276
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49C276:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49C297
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49C294
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49C294:
                mov     [ebp-38h], ecx

loc_49C297:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49C303(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0A3h
                mov     dword ptr [ebp-2Ch], 0E0h
                mov     dword ptr [ebp-28h], 7Ah
                mov     dword ptr [ebp-24h], 0EEh
                mov     dword ptr [ebp-20h], 37h
                mov     dword ptr [ebp-1Ch], 5
                mov     dword ptr [ebp-18h], 12h
                mov     dword ptr [ebp-14h], 44h
                mov     dword ptr [ebp-10h], 7
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 7
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49C37E
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49C37E:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49C39F
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49C39C
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49C39C:
                mov     [ebp-38h], ecx

loc_49C39F:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49C40B(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 2Bh
                mov     dword ptr [ebp-2Ch], 0BEh
                mov     dword ptr [ebp-28h], 0E2h
                mov     dword ptr [ebp-24h], 9
                mov     dword ptr [ebp-20h], 51h
                mov     dword ptr [ebp-1Ch], 0E7h
                mov     dword ptr [ebp-18h], 0B4h
                mov     dword ptr [ebp-14h], 0C9h
                mov     dword ptr [ebp-10h], 1
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 1
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49C485
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49C485:
```

```
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49C4A6
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49C4A3
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49C4A3:
                mov     [ebp-38h], ecx

loc_49C4A6:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49C512(void) {  __asm  {
```

```
push    ebp
mov     ebp, esp
sub     esp, 40h
mov     dword ptr [ebp-30h], 89h
mov     dword ptr [ebp-2Ch], 23h
mov     dword ptr [ebp-28h], 46h
mov     dword ptr [ebp-24h], 92h
mov     dword ptr [ebp-20h], 0B6h
mov     dword ptr [ebp-1Ch], 0E8h
mov     dword ptr [ebp-18h], 0BDh
mov     dword ptr [ebp-14h], 2Ch
mov     dword ptr [ebp-10h], 0Dh
mov     dword ptr [ebp-40h], 7
mov     eax, [ebp+8]
shr     eax, 0Dh
and     eax, 7
mov     ecx, [ebp+eax*4-30h]
mov     [ebp-3Ch], ecx
mov     eax, [ebp-3Ch]
cdq
and     edx, 0Fh
add     eax, edx
sar     eax, 4
mov     [ebp-34h], eax
mov     edx, [ebp-3Ch]
and     edx, 8000000Fh
jns     short loc_49C58D
```

```
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49C58D:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49C5AE
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49C5AB
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49C5AB:
                mov     [ebp-38h], ecx

loc_49C5AE:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
```

```
}}

__declspec(naked) void sub_49C61A(void) {  __asm  {

                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0Ch
                mov     dword ptr [ebp-2Ch], 72h
                mov     dword ptr [ebp-28h], 13h
                mov     dword ptr [ebp-24h], 3Dh
                mov     dword ptr [ebp-20h], 67h
                mov     dword ptr [ebp-1Ch], 48h
                mov     dword ptr [ebp-18h], 0AFh
                mov     dword ptr [ebp-14h], 0B3h
                mov     dword ptr [ebp-10h], 12h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 12h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49C695
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49C695:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49C6B6
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49C6B3
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49C6B3:
                mov     [ebp-38h], ecx

loc_49C6B6:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49C722(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 94h
                mov     dword ptr [ebp-2Ch], 20h
                mov     dword ptr [ebp-28h], 5Ch
                mov     dword ptr [ebp-24h], 0D6h
                mov     dword ptr [ebp-20h], 20h
                mov     dword ptr [ebp-1Ch], 89h
                mov     dword ptr [ebp-18h], 0Fh
                mov     dword ptr [ebp-14h], 79h
                mov     dword ptr [ebp-10h], 10h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 10h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49C79D
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49C79D:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49C7BE
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49C7BB
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49C7BB:
                mov     [ebp-38h], ecx

loc_49C7BE:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49C82A(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0D8h
                mov     dword ptr [ebp-2Ch], 0FAh
                mov     dword ptr [ebp-28h], 0A9h
                mov     dword ptr [ebp-24h], 1Fh
                mov     dword ptr [ebp-20h], 0F5h
                mov     dword ptr [ebp-1Ch], 4Dh
                mov     dword ptr [ebp-18h], 6Eh
                mov     dword ptr [ebp-14h], 9Fh
                mov     dword ptr [ebp-10h], 8
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 8
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49C8A5
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49C8A5:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49C8C6
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49C8C3
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49C8C3:
                mov     [ebp-38h], ecx

loc_49C8C6:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49C932(void) {  __asm  {



                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0B7h
                mov     dword ptr [ebp-2Ch], 0Ch
                mov     dword ptr [ebp-28h], 0DDh
                mov     dword ptr [ebp-24h], 0E1h
                mov     dword ptr [ebp-20h], 0C6h
                mov     dword ptr [ebp-1Ch], 0Ch
                mov     dword ptr [ebp-18h], 7Dh
                mov     dword ptr [ebp-14h], 43h
                mov     dword ptr [ebp-10h], 0Eh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Eh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49C9AD
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49C9AD:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49C9CE
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49C9CB
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49C9CB:
                mov     [ebp-38h], ecx

loc_49C9CE:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49CA3A(void) {  __asm  {











                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 4
                mov     dword ptr [ebp-2Ch], 4Ch
                mov     dword ptr [ebp-28h], 35h
                mov     dword ptr [ebp-24h], 6Bh
                mov     dword ptr [ebp-20h], 31h
                mov     dword ptr [ebp-1Ch], 4Dh
                mov     dword ptr [ebp-18h], 8Bh
                mov     dword ptr [ebp-14h], 0B0h
                mov     dword ptr [ebp-10h], 0Ah
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ah
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49CAB5
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49CAB5:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49CAD6
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49CAD3
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49CAD3:
                mov     [ebp-38h], ecx

loc_49CAD6:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49CB42(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0CEh
                mov     dword ptr [ebp-2Ch], 78h
                mov     dword ptr [ebp-28h], 8Dh
                mov     dword ptr [ebp-24h], 0EBh
                mov     dword ptr [ebp-20h], 62h
                mov     dword ptr [ebp-1Ch], 7Ch
                mov     dword ptr [ebp-18h], 4Ah
                mov     dword ptr [ebp-14h], 0CCh
                mov     dword ptr [ebp-10h], 0Fh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Fh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49CBBD
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49CBBD:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49CBDE
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49CBDB
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49CBDB:
                mov     [ebp-38h], ecx

loc_49CBDE:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49CC4A(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 63h
                mov     dword ptr [ebp-2Ch], 9Ch
                mov     dword ptr [ebp-28h], 19h
                mov     dword ptr [ebp-24h], 85h
                mov     dword ptr [ebp-20h], 0D1h
                mov     dword ptr [ebp-1Ch], 24h
                mov     dword ptr [ebp-18h], 0A8h
                mov     dword ptr [ebp-14h], 93h
                mov     dword ptr [ebp-10h], 9
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 9
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49CCC5
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49CCC5:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49CCE6
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49CCE3
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx
```

```
loc_49CCE3:
                mov     [ebp-38h], ecx

loc_49CCE6:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49CD52(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 5Ah
                mov     dword ptr [ebp-2Ch], 62h
                mov     dword ptr [ebp-28h], 48h
                mov     dword ptr [ebp-24h], 0CBh
                mov     dword ptr [ebp-20h], 4Fh
                mov     dword ptr [ebp-1Ch], 6Bh
                mov     dword ptr [ebp-18h], 0E0h
                mov     dword ptr [ebp-14h], 2
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49CDCD
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49CDCD:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49CDEE
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49CDEB
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49CDEB:
                mov     [ebp-38h], ecx

loc_49CDEE:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49CE5A(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 60h
                mov     dword ptr [ebp-2Ch], 0BAh
                mov     dword ptr [ebp-28h], 3
                mov     dword ptr [ebp-24h], 64h
                mov     dword ptr [ebp-20h], 0DDh
                mov     dword ptr [ebp-1Ch], 4Ch
                mov     dword ptr [ebp-18h], 4
                mov     dword ptr [ebp-14h], 7Bh
                mov     dword ptr [ebp-10h], 11h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 11h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49CED5
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49CED5:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49CEF6
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49CEF3
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49CEF3:
                mov     [ebp-38h], ecx

loc_49CEF6:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49CF62(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 67h
                mov     dword ptr [ebp-2Ch], 0E9h
                mov     dword ptr [ebp-28h], 22h
                mov     dword ptr [ebp-24h], 0F6h
                mov     dword ptr [ebp-20h], 5Bh
                mov     dword ptr [ebp-1Ch], 0E7h
                mov     dword ptr [ebp-18h], 0A2h
                mov     dword ptr [ebp-14h], 0F9h
                mov     dword ptr [ebp-10h], 12h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 12h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49CFDD
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49CFDD:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49CFFE
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49CFFB
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49CFFB:
                mov     [ebp-38h], ecx

loc_49CFFE:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49D06A(void) {  __asm  {
```

```
        push    ebp
        mov     ebp, esp
        sub     esp, 40h
        mov     dword ptr [ebp-30h], 7Eh
        mov     dword ptr [ebp-2Ch], 0F7h
        mov     dword ptr [ebp-28h], 3Ch
        mov     dword ptr [ebp-24h], 4
        mov     dword ptr [ebp-20h], 35h
        mov     dword ptr [ebp-1Ch], 4Dh
        mov     dword ptr [ebp-18h], 0ADh
        mov     dword ptr [ebp-14h], 0CDh
        mov     dword ptr [ebp-10h], 11h
        mov     dword ptr [ebp-40h], 7
        mov     eax, [ebp+8]
        shr     eax, 11h
        and     eax, 7
        mov     ecx, [ebp+eax*4-30h]
        mov     [ebp-3Ch], ecx
        mov     eax, [ebp-3Ch]
        cdq
        and     edx, 0Fh
        add     eax, edx
        sar     eax, 4
        mov     [ebp-34h], eax
```

```
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49D0E5
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49D0E5:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49D106
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49D103
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49D103:
                mov     [ebp-38h], ecx

loc_49D106:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}

__declspec(naked) void sub_49D172(void) {  __asm  {

                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 6Eh
                mov     dword ptr [ebp-2Ch], 0D8h
                mov     dword ptr [ebp-28h], 0B5h
                mov     dword ptr [ebp-24h], 0A6h
                mov     dword ptr [ebp-20h], 84h
                mov     dword ptr [ebp-1Ch], 5Bh
                mov     dword ptr [ebp-18h], 0CAh
                mov     dword ptr [ebp-14h], 8Ch
                mov     dword ptr [ebp-10h], 8
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 8
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49D1ED
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49D1ED:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49D20E
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49D20B
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49D20B:
                mov     [ebp-38h], ecx

loc_49D20E:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49D27A(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 92h
                mov     dword ptr [ebp-2Ch], 76h
                mov     dword ptr [ebp-28h], 0D1h
                mov     dword ptr [ebp-24h], 91h
                mov     dword ptr [ebp-20h], 0FBh
                mov     dword ptr [ebp-1Ch], 1Bh
                mov     dword ptr [ebp-18h], 0DCh
                mov     dword ptr [ebp-14h], 0FBh
                mov     dword ptr [ebp-10h], 0Eh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Eh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49D2F5
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49D2F5:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49D316
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49D313
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49D313:
                mov     [ebp-38h], ecx

loc_49D316:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49D382(void) {  __asm  {




                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 4Ah
                mov     dword ptr [ebp-2Ch], 95h
                mov     dword ptr [ebp-28h], 4
                mov     dword ptr [ebp-24h], 0C4h
                mov     dword ptr [ebp-20h], 0D5h
                mov     dword ptr [ebp-1Ch], 6
                mov     dword ptr [ebp-18h], 97h
                mov     dword ptr [ebp-14h], 80h
                mov     dword ptr [ebp-10h], 14h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 14h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49D3FD
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49D3FD:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49D41E
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49D41B
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49D41B:
                mov     [ebp-38h], ecx

loc_49D41E:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49D48A(void) {  __asm  {




                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 8Bh
                mov     dword ptr [ebp-2Ch], 33h
                mov     dword ptr [ebp-28h], 99h
                mov     dword ptr [ebp-24h], 0D3h
                mov     dword ptr [ebp-20h], 0D1h
                mov     dword ptr [ebp-1Ch], 9Bh
                mov     dword ptr [ebp-18h], 1Ch
                mov     dword ptr [ebp-14h], 4Fh
                mov     dword ptr [ebp-10h], 14h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 14h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49D505
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49D505:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49D526
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49D523
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49D523:
                mov     [ebp-38h], ecx

loc_49D526:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49D592(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0ECh
                mov     dword ptr [ebp-2Ch], 0E4h
                mov     dword ptr [ebp-28h], 0D6h
                mov     dword ptr [ebp-24h], 0C3h
                mov     dword ptr [ebp-20h], 0B3h
                mov     dword ptr [ebp-1Ch], 46h
                mov     dword ptr [ebp-18h], 0BFh
                mov     dword ptr [ebp-14h], 0D9h
                mov     dword ptr [ebp-10h], 6
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 6
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49D60D
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49D60D:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49D62E
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49D62B
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49D62B:
                mov     [ebp-38h], ecx

loc_49D62E:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49D69A(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 5Eh
                mov     dword ptr [ebp-2Ch], 23h
                mov     dword ptr [ebp-28h], 5Ch
                mov     dword ptr [ebp-24h], 2Ch
                mov     dword ptr [ebp-20h], 0BBh
                mov     dword ptr [ebp-1Ch], 7Ch
                mov     dword ptr [ebp-18h], 9
                mov     dword ptr [ebp-14h], 58h
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49D715
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49D715:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49D736
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49D733
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49D733:
                mov     [ebp-38h], ecx
```

```
loc_49D736:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}





__declspec(naked) void sub_49D7A2(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 15h
                mov     dword ptr [ebp-2Ch], 0EFh
                mov     dword ptr [ebp-28h], 89h
                mov     dword ptr [ebp-24h], 58h
                mov     dword ptr [ebp-20h], 15h
                mov     dword ptr [ebp-1Ch], 0D3h
                mov     dword ptr [ebp-18h], 0
                mov     dword ptr [ebp-14h], 0F5h
                mov     dword ptr [ebp-10h], 3
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 3
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49D81D
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49D81D:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49D83E
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49D83B
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49D83B:
                mov     [ebp-38h], ecx

loc_49D83E:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49D8AA(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0D0h
                mov     dword ptr [ebp-2Ch], 1Bh
                mov     dword ptr [ebp-28h], 62h
                mov     dword ptr [ebp-24h], 31h
                mov     dword ptr [ebp-20h], 0B9h
                mov     dword ptr [ebp-1Ch], 19h
                mov     dword ptr [ebp-18h], 4Bh
                mov     dword ptr [ebp-14h], 62h
                mov     dword ptr [ebp-10h], 0Fh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Fh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49D925
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49D925:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49D946
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49D943
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49D943:
                mov     [ebp-38h], ecx

loc_49D946:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49D9B2(void) {  __asm  {
```

```
push    ebp
mov     ebp, esp
sub     esp, 40h
mov     dword ptr [ebp-30h], 4Fh
mov     dword ptr [ebp-2Ch], 0A7h
mov     dword ptr [ebp-28h], 0B8h
mov     dword ptr [ebp-24h], 45h
mov     dword ptr [ebp-20h], 0A2h
mov     dword ptr [ebp-1Ch], 76h
mov     dword ptr [ebp-18h], 0F3h
mov     dword ptr [ebp-14h], 0A6h
mov     dword ptr [ebp-10h], 2
mov     dword ptr [ebp-40h], 7
mov     eax, [ebp+8]
shr     eax, 2
and     eax, 7
mov     ecx, [ebp+eax*4-30h]
mov     [ebp-3Ch], ecx
mov     eax, [ebp-3Ch]
cdq
and     edx, 0Fh
add     eax, edx
sar     eax, 4
mov     [ebp-34h], eax
mov     edx, [ebp-3Ch]
and     edx, 8000000Fh
jns     short loc_49DA2D
dec     edx
or      edx, 0FFFFFFF0h
inc     edx
```

```
loc_49DA2D:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49DA4E
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49DA4B
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49DA4B:
                mov     [ebp-38h], ecx

loc_49DA4E:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49DABA(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 61h
                mov     dword ptr [ebp-2Ch], 0D6h
                mov     dword ptr [ebp-28h], 0BDh
                mov     dword ptr [ebp-24h], 16h
                mov     dword ptr [ebp-20h], 15h
                mov     dword ptr [ebp-1Ch], 0E7h
                mov     dword ptr [ebp-18h], 0B1h
                mov     dword ptr [ebp-14h], 75h
                mov     dword ptr [ebp-10h], 5
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 5
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49DB35
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49DB35:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49DB56
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49DB53
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49DB53:
                mov     [ebp-38h], ecx

loc_49DB56:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}



__declspec(naked) void sub_49DBC2(void) {  __asm  {



















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0B1h
                mov     dword ptr [ebp-2Ch], 0DFh
                mov     dword ptr [ebp-28h], 21h
                mov     dword ptr [ebp-24h], 41h
                mov     dword ptr [ebp-20h], 40h
                mov     dword ptr [ebp-1Ch], 7Ch
                mov     dword ptr [ebp-18h], 0FAh
                mov     dword ptr [ebp-14h], 4Dh
                mov     dword ptr [ebp-10h], 0Ah
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Ah
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49DC3D
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49DC3D:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49DC5E
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49DC5B
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49DC5B:
                mov     [ebp-38h], ecx

loc_49DC5E:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49DCCA(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 90h
                mov     dword ptr [ebp-2Ch], 68h
                mov     dword ptr [ebp-28h], 53h
                mov     dword ptr [ebp-24h], 0E7h
                mov     dword ptr [ebp-20h], 0EEh
                mov     dword ptr [ebp-1Ch], 57h
                mov     dword ptr [ebp-18h], 8
                mov     dword ptr [ebp-14h], 2Fh
                mov     dword ptr [ebp-10h], 6
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 6
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49DD45
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49DD45:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49DD66
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49DD63
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49DD63:
                mov     [ebp-38h], ecx

loc_49DD66:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}
```

```
__declspec(naked) void sub_49DDD2(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0C9h
                mov     dword ptr [ebp-2Ch], 6Ah
                mov     dword ptr [ebp-28h], 9Ch
                mov     dword ptr [ebp-24h], 0B7h
                mov     dword ptr [ebp-20h], 0A8h
                mov     dword ptr [ebp-1Ch], 0D6h
                mov     dword ptr [ebp-18h], 79h
                mov     dword ptr [ebp-14h], 20h
                mov     dword ptr [ebp-10h], 0
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49DE4A
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49DE4A:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49DE6B
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49DE68
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49DE68:
                mov     [ebp-38h], ecx

loc_49DE6B:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49DED7(void) {  __asm  {
















                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 1Fh
                mov     dword ptr [ebp-2Ch], 7Dh
                mov     dword ptr [ebp-28h], 0B8h
                mov     dword ptr [ebp-24h], 60h
                mov     dword ptr [ebp-20h], 0B5h
                mov     dword ptr [ebp-1Ch], 11h
                mov     dword ptr [ebp-18h], 0DAh
                mov     dword ptr [ebp-14h], 9
                mov     dword ptr [ebp-10h], 6
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 6
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49DF52
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49DF52:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49DF73
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49DF70
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49DF70:
                mov     [ebp-38h], ecx

loc_49DF73:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49DFDF(void) {  __asm  {












                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 74h
                mov     dword ptr [ebp-2Ch], 2
                mov     dword ptr [ebp-28h], 75h
                mov     dword ptr [ebp-24h], 90h
                mov     dword ptr [ebp-20h], 79h
                mov     dword ptr [ebp-1Ch], 0DCh
                mov     dword ptr [ebp-18h], 50h
                mov     dword ptr [ebp-14h], 6
                mov     dword ptr [ebp-10h], 0Fh
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 0Fh
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49E05A
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49E05A:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49E07B
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49E078
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49E078:
                mov     [ebp-38h], ecx

loc_49E07B:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49E0E7(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0Ah
                mov     dword ptr [ebp-2Ch], 0FBh
                mov     dword ptr [ebp-28h], 6Fh
                mov     dword ptr [ebp-24h], 3
                mov     dword ptr [ebp-20h], 67h
                mov     dword ptr [ebp-1Ch], 6Ah
                mov     dword ptr [ebp-18h], 36h
                mov     dword ptr [ebp-14h], 8Eh
                mov     dword ptr [ebp-10h], 7
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 7
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49E162
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49E162:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49E183
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49E180
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49E180:
                mov     [ebp-38h], ecx

loc_49E183:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49E1EF(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 0DDh
                mov     dword ptr [ebp-2Ch], 33h
                mov     dword ptr [ebp-28h], 5
                mov     dword ptr [ebp-24h], 79h
                mov     dword ptr [ebp-20h], 66h
                mov     dword ptr [ebp-1Ch], 4Bh
                mov     dword ptr [ebp-18h], 0B1h
                mov     dword ptr [ebp-14h], 3Eh
                mov     dword ptr [ebp-10h], 11h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 11h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49E26A
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49E26A:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49E28B
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49E288
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx
```

```
loc_49E288:
                mov     [ebp-38h], ecx

loc_49E28B:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                /*call    [ebp-8]*/ call AsmDispatcher
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}




__declspec(naked) void sub_49E2F7(void) {  __asm  {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 40h
                mov     dword ptr [ebp-30h], 5Eh
                mov     dword ptr [ebp-2Ch], 5Fh
                mov     dword ptr [ebp-28h], 8Fh
                mov     dword ptr [ebp-24h], 0E7h
                mov     dword ptr [ebp-20h], 0E0h
                mov     dword ptr [ebp-1Ch], 97h
                mov     dword ptr [ebp-18h], 0E5h
                mov     dword ptr [ebp-14h], 0CFh
                mov     dword ptr [ebp-10h], 12h
                mov     dword ptr [ebp-40h], 7
                mov     eax, [ebp+8]
                shr     eax, 12h
                and     eax, 7
                mov     ecx, [ebp+eax*4-30h]
                mov     [ebp-3Ch], ecx
                mov     eax, [ebp-3Ch]
                cdq
                and     edx, 0Fh
                add     eax, edx
                sar     eax, 4
                mov     [ebp-34h], eax
                mov     edx, [ebp-3Ch]
                and     edx, 8000000Fh
                jns     short loc_49E372
                dec     edx
                or      edx, 0FFFFFFF0h
                inc     edx

loc_49E372:
                mov     [ebp-38h], edx
                mov     eax, [ebp-34h]
                cmp     eax, [ebp-38h]
                jnz     short loc_49E393
                mov     ecx, [ebp-38h]
                add     ecx, 1
                and     ecx, 8000000Fh
                jns     short loc_49E390
                dec     ecx
                or      ecx, 0FFFFFFF0h
                inc     ecx

loc_49E390:
                mov     [ebp-38h], ecx

loc_49E393:
                mov     edx, [ebp-3Ch]
                mov     eax, [ebp-34h]
                mov     ecx, dword ptr dword_4DF3C0[edx*4]
                xor     ecx, dword ptr dword_4D92CC[eax*4]
                mov     edx, [ebp-38h]
                xor     ecx, dword ptr dword_4D92CC[edx*4]
                mov     [ebp-8], ecx
                mov     eax, [ebp+0Ch]
                push    eax
                mov     ecx, [ebp-3Ch]
                movsx   edx, dword ptr byte_4DDBA0[ecx]
                call    dword ptr Block3Func1Data1[edx*4]
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp+10h]
                push    eax
                mov     ecx, [ebp-4]
                push    ecx
                add     esp, 8
                push    eax
                mov     edx, [ebp-3Ch]
                movsx   eax, dword ptr byte_4DDBA0[edx]
                call    dword ptr off_4DDCDC[eax*4]
                add     esp, 4
                mov     [ebp-0Ch], eax
                mov     eax, [ebp-0Ch]
                and     eax, 1
                mov     esp, ebp
                pop     ebp
                retn
}}

// block 6 sixteen functions array

__declspec(naked) void SixBlock0(void)
{
     __asm {
                 push    ebp
                 mov     ebp, esp
                 push    7
                 push    16h
                 mov     eax, [ebp+8]
                 push    eax
                 call    SubFunc0
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 mov     ecx, [ebp+8]
                 xor     ecx, 0D972B853h
                 mov     [ebp+8], ecx
                 push    13h
                 push    8
                 mov     edx, [ebp+8]
                 push    edx
                 call    SubFunc1
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 push    0FFFFFFFBh
                 push    0Eh
                 push    7
                 mov     eax, [ebp+8]
                 push    eax
                 call    SubFunc2
                 add     esp, 10h
                 mov     [ebp+8], eax
                 mov     ecx, [ebp+8]
                 xor     ecx, 4B487412h
                 mov     [ebp+8], ecx
                 push    0Ch
                 push    9
                 mov     edx, [ebp+8]
                 push    edx
                 call    SubFunc0
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 push    0Dh
                 push    2
                 mov     eax, [ebp+8]
                 push    eax
                 call    SubFunc1
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 push    5
                 push    15h
                 mov     ecx, [ebp+8]
                 push    ecx
                 call    SubFunc0
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 mov     edx, [ebp+8]
                 xor     edx, 6E4957A8h
                 mov     [ebp+8], edx
                 mov     eax, [ebp+8]
                 pop     ebp
                 retn
     }
```

```
}

__declspec(naked) void SixBlock1(void)
{
     __asm {
                push    ebp
                mov     ebp, esp
                mov     eax, [ebp+8]
                xor     eax, 40174D5Bh
                mov     [ebp+8], eax
                push    0FFFFFFF4h
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc3
                add     esp, 8
                mov     [ebp+8], eax
                push    0
                push    1Eh
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc0
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    14h
                push    1
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc1
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     ecx, [ebp+8]
                xor     ecx, 0A52B3D68h
                mov     [ebp+8], ecx
                push    0FFFFFFF9h
                push    14h
                push    0Bh
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc2
                add     esp, 10h
                mov     [ebp+8], eax
                push    0FFFFFFE6h
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc3
                add     esp, 8
                mov     [ebp+8], eax
                mov     ecx, [ebp+8]
                xor     ecx, 8B9D36E9h
                mov     [ebp+8], ecx
                push    0FFFFFFF3h
                push    2
```

```
                    push    14h
                    mov     edx, [ebp+8]
                    push    edx
                    call    SubFunc2
                    add     esp, 10h
                    mov     [ebp+8], eax
                    push    0Ah
                    push    0Ah
                    mov     eax, [ebp+8]
                    push    eax
                    call    SubFunc0
                    add     esp, 0Ch
                    mov     [ebp+8], eax
                    push    0FFFFFFEAh
                    push    0
                    push    1Ch
                    mov     ecx, [ebp+8]
                    push    ecx
                    call    SubFunc2
                    add     esp, 10h
                    mov     [ebp+8], eax
                    push    1Ah
                    push    5
                    mov     edx, [ebp+8]
                    push    edx
                    call    SubFunc0
                    add     esp, 0Ch
                    mov     [ebp+8], eax
                    mov     eax, [ebp+8]
                    xor     eax, 0FB1E52AFh
                    mov     [ebp+8], eax
                    mov     eax, [ebp+8]
                    pop     ebp
                    retn
    }
}

__declspec(naked) void SixBlock2(void)
{
    __asm {
                    push    ebp
                    mov     ebp, esp
                    push    0FFFFFFF1h
                    mov     eax, [ebp+8]
                    push    eax
                    call    SubFunc3
                    add     esp, 8
                    mov     [ebp+8], eax
                    mov     ecx, [ebp+8]
                    xor     ecx, 0F185A47Ch
                    mov     [ebp+8], ecx
                    push    0FFFFFFFEh
                    push    0Fh
                    push    9
                    mov     edx, [ebp+8]
                    push    edx
                    call    SubFunc2
                    add     esp, 10h
                    mov     [ebp+8], eax
                    push    0FFFFFFF1h
                    mov     eax, [ebp+8]
                    push    eax
                    call    SubFunc3
                    add     esp, 8
                    mov     [ebp+8], eax
                    push    4
                    push    17h
                    mov     ecx, [ebp+8]
                    push    ecx
                    call    SubFunc0
                    add     esp, 0Ch
                    mov     [ebp+8], eax
                    push    9
                    push    2
                    mov     edx, [ebp+8]
                    push    edx
                    call    SubFunc1
                    add     esp, 0Ch
                    mov     [ebp+8], eax
                    push    0FFFFFFF7h
                    push    1
                    push    1Eh
                    mov     eax, [ebp+8]
                    push    eax
                    call    SubFunc2
                    add     esp, 10h
                    mov     [ebp+8], eax
                    push    0Eh
                    push    8
                    mov     ecx, [ebp+8]
                    push    ecx
                    call    SubFunc1
                    add     esp, 0Ch
                    mov     [ebp+8], eax
                    push    0
                    push    11h
                    mov     edx, [ebp+8]
                    push    edx
                    call    SubFunc0
                    add     esp, 0Ch
                    mov     [ebp+8], eax
                    push    0FFFFFFF5h
                    push    0
                    push    18h
```

```
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc2
                add     esp, 10h
                mov     [ebp+8], eax
                mov     ecx, [ebp+8]
                xor     ecx, 81A5E699h
                mov     [ebp+8], ecx
                push    0FFFFFFE6h
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc3
                add     esp, 8
                mov     [ebp+8], eax
                push    0FFFFFFFAh
                push    9
                push    0Eh
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc2
                add     esp, 10h
                mov     [ebp+8], eax
                push    0FFFFFFEFh
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc3
                add     esp, 8
                mov     [ebp+8], eax
                push    17h
                push    4
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc1
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                xor     eax, 0FBBD38E7h
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                pop     ebp
                retn
    }
}

__declspec(naked) void SixBlock3(void)
{
    __asm {
                push    ebp
                mov     ebp, esp
                push    0FFFFFFFAh
                push    3
                push    0Dh
```

```
mov     eax, [ebp+8]
push    eax
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    0FFFFFFEEh
mov     ecx, [ebp+8]
push    ecx
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    0Fh
push    5
mov     edx, [ebp+8]
push    edx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFFBh
push    7
push    8
mov     eax, [ebp+8]
push    eax
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    16h
push    2
mov     ecx, [ebp+8]
push    ecx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFF9h
push    0Ah
push    0Bh
mov     edx, [ebp+8]
push    edx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    7
push    8
mov     eax, [ebp+8]
push    eax
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    0Bh
push    7
mov     ecx, [ebp+8]
push    ecx
```

```
                        call    SubFunc1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    0FFFFFFFBh
                        push    7
                        push    11h
                        mov     edx, [ebp+8]
                        push    edx
                        call    SubFunc2
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    6
                        push    0Ah
                        mov     eax, [ebp+8]
                        push    eax
                        call    SubFunc0
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    0Ah
                        push    3
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    SubFunc1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        mov     edx, [ebp+8]
                        xor     edx, 5C2ACD4Dh
                        mov     [ebp+8], edx
                        mov     eax, [ebp+8]
                        pop     ebp
                        retn
        }
}

__declspec(naked) void SixBlock4(void)
{
        __asm {
                        push    ebp
                        mov     ebp, esp
                        push    0FFFFFFFBh
                        push    7
                        push    17h
                        mov     eax, [ebp+8]
                        push    eax
                        call    SubFunc2
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    0FFFFFFEDh
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    SubFunc3
                        add     esp, 8
```

```
mov     [ebp+8], eax
push    0Ch
push    0Fh
mov     edx, [ebp+8]
push    edx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    0Eh
push    1
mov     eax, [ebp+8]
push    eax
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 89C7C9A6h
mov     [ebp+8], ecx
push    9
push    8
mov     edx, [ebp+8]
push    edx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    3
push    0Fh
mov     eax, [ebp+8]
push    eax
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    15h
push    4
mov     ecx, [ebp+8]
push    ecx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFEBh
mov     edx, [ebp+8]
push    edx
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    12h
push    5
mov     eax, [ebp+8]
push    eax
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
```

```
push    6
push    12h
mov     ecx, [ebp+8]
push    ecx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    14h
push    1
mov     edx, [ebp+8]
push    edx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    0
push    1Fh
mov     eax, [ebp+8]
push    eax
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    0Ah
push    7
mov     ecx, [ebp+8]
push    ecx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFFDh
push    0Dh
push    8
mov     edx, [ebp+8]
push    edx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    0FFFFFFE5h
mov     eax, [ebp+8]
push    eax
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    13h
push    5
mov     ecx, [ebp+8]
push    ecx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    7
push    0Fh
mov     edx, [ebp+8]
```

```
                 push    edx
                 call    SubFunc0
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 push    0FFFFFFE9h
                 push    0
                 push    1Ch
                 mov     eax, [ebp+8]
                 push    eax
                 call    SubFunc2
                 add     esp, 10h
                 mov     [ebp+8], eax
                 push    0FFFFFFFCh
                 mov     ecx, [ebp+8]
                 push    ecx
                 call    SubFunc3
                 add     esp, 8
                 mov     [ebp+8], eax
                 mov     edx, [ebp+8]
                 xor     edx, 47D8FFBDh
                 mov     [ebp+8], edx
                 mov     eax, [ebp+8]
                 pop     ebp
                 retn
     }
}

__declspec(naked) void SixBlock5(void)
{
     __asm {
                 push    ebp
                 mov     ebp, esp
                 push    0
                 push    1Fh
                 mov     eax, [ebp+8]
                 push    eax
                 call    SubFunc0
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 push    0FFFFFFFAh
                 push    3
                 push    16h
                 mov     ecx, [ebp+8]
                 push    ecx
                 call    SubFunc2
                 add     esp, 10h
                 mov     [ebp+8], eax
                 push    0Ch
                 push    1
                 mov     edx, [ebp+8]
                 push    edx
                 call    SubFunc1
```

```
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFFCh
push    0Bh
push    6
mov     eax, [ebp+8]
push    eax
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    0Bh
push    4
mov     ecx, [ebp+8]
push    ecx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 7C3547F7h
mov     [ebp+8], edx
push    0FFFFFFEBh
mov     eax, [ebp+8]
push    eax
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    14h
push    4
mov     ecx, [ebp+8]
push    ecx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFF7h
push    0
push    0Dh
mov     edx, [ebp+8]
push    edx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    0Bh
push    1
mov     eax, [ebp+8]
push    eax
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFE5h
push    0
push    1Ch
mov     ecx, [ebp+8]
```

```
push    ecx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    0FFFFFFFBh
mov     edx, [ebp+8]
push    edx
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    0FFFFFFFBh
push    1
push    1Eh
mov     eax, [ebp+8]
push    eax
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    0Eh
push    1
mov     ecx, [ebp+8]
push    ecx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 5BFB6B28h
mov     [ebp+8], edx
push    0FFFFFFF2h
mov     eax, [ebp+8]
push    eax
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 8B465658h
mov     [ebp+8], ecx
push    0FFFFFFEDh
push    0
push    1Fh
mov     edx, [ebp+8]
push    edx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    10h
push    2
mov     eax, [ebp+8]
push    eax
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
```

```
                        push    17h
                        push    8
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    SubFunc0
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        mov     edx, [ebp+8]
                        xor     edx, 24A4A3B5h
                        mov     [ebp+8], edx
                        mov     eax, [ebp+8]
                        pop     ebp
                        retn
        }
}

__declspec(naked) void SixBlock6(void)
{
        __asm {
                        push    ebp
                        mov     ebp, esp
                        push    0Dh
                        push    0Bh
                        mov     eax, [ebp+8]
                        push    eax
                        call    SubFunc0
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    0FFFFFFF1h
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    SubFunc3
                        add     esp, 8
                        mov     [ebp+8], eax
                        mov     edx, [ebp+8]
                        xor     edx, 251AFCFEh
                        mov     [ebp+8], edx
                        push    0FFFFFFFDh
                        mov     eax, [ebp+8]
                        push    eax
                        call    SubFunc3
                        add     esp, 8
                        mov     [ebp+8], eax
                        push    0FFFFFFF5h
                        push    4
                        push    19h
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    SubFunc2
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    0FFFFFFF2h
```

```
                  mov     edx, [ebp+8]
                  push    edx
                  call    SubFunc3
                  add     esp, 8
                  mov     [ebp+8], eax
                  push    0FFFFFFFCh
                  push    8
                  push    13h
                  mov     eax, [ebp+8]
                  push    eax
                  call    SubFunc2
                  add     esp, 10h
                  mov     [ebp+8], eax
                  mov     ecx, [ebp+8]
                  xor     ecx, 6BA330BFh
                  mov     [ebp+8], ecx
                  push    0Ah
                  push    12h
                  mov     edx, [ebp+8]
                  push    edx
                  call    SubFunc0
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    0Ch
                  push    8
                  mov     eax, [ebp+8]
                  push    eax
                  call    SubFunc1
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  mov     ecx, [ebp+8]
                  xor     ecx, 5ED8936Ch
                  mov     [ebp+8], ecx
                  push    0FFFFFFEFh
                  mov     edx, [ebp+8]
                  push    edx
                  call    SubFunc3
                  add     esp, 8
                  mov     [ebp+8], eax
                  mov     eax, [ebp+8]
                  xor     eax, 34473F96h
                  mov     [ebp+8], eax
                  mov     eax, [ebp+8]
                  pop     ebp
                  retn
     }
}

__declspec(naked) void SixBlock7(void)
{
     __asm {
                  push    ebp
```

```
mov     ebp, esp
push    0
push    7
mov     eax, [ebp+8]
push    eax
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFE1h
mov     ecx, [ebp+8]
push    ecx
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    0FFFFFFFBh
push    0Ah
push    0Fh
mov     edx, [ebp+8]
push    edx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    0FFFFFFFCh
mov     eax, [ebp+8]
push    eax
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 3A22CD09h
mov     [ebp+8], ecx
push    13h
push    4
mov     edx, [ebp+8]
push    edx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
mov     eax, [ebp+8]
xor     eax, 0F6547803h
mov     [ebp+8], eax
push    0FFFFFFF3h
mov     ecx, [ebp+8]
push    ecx
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 0ABA4CE9Ah
mov     [ebp+8], edx
push    11h
push    9
```

```
mov     eax, [ebp+8]
push    eax
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    11h
push    5
mov     ecx, [ebp+8]
push    ecx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFF0h
push    0
push    1Fh
mov     edx, [ebp+8]
push    edx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    0Fh
push    2
mov     eax, [ebp+8]
push    eax
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 49253F31h
mov     [ebp+8], ecx
push    0
push    1Eh
mov     edx, [ebp+8]
push    edx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
mov     eax, [ebp+8]
xor     eax, 3773D297h
mov     [ebp+8], eax
push    3
push    11h
mov     ecx, [ebp+8]
push    ecx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 70648B0Dh
mov     [ebp+8], edx
push    0Dh
push    6
```

```
                 mov     eax, [ebp+8]
                 push    eax
                 call    SubFunc1
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 push    0FFFFFFF5h
                 mov     ecx, [ebp+8]
                 push    ecx
                 call    SubFunc3
                 add     esp, 8
                 mov     [ebp+8], eax
                 push    10h
                 push    0Bh
                 mov     edx, [ebp+8]
                 push    edx
                 call    SubFunc0
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 mov     eax, [ebp+8]
                 xor     eax, 5A07B2A1h
                 mov     [ebp+8], eax
                 mov     eax, [ebp+8]
                 pop     ebp
                 retn
     }
}

__declspec(naked) void SixBlock8(void)
{
     __asm {
                 push    ebp
                 mov     ebp, esp
                 push    0FFFFFFF5h
                 mov     eax, [ebp+8]
                 push    eax
                 call    SubFunc3
                 add     esp, 8
                 mov     [ebp+8], eax
                 push    5
                 push    19h
                 mov     ecx, [ebp+8]
                 push    ecx
                 call    SubFunc0
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 push    10h
                 push    5
                 mov     edx, [ebp+8]
                 push    edx
                 call    SubFunc1
                 add     esp, 0Ch
                 mov     [ebp+8], eax
```

```
                        mov     eax, [ebp+8]
                        xor     eax, 0A8DB534Eh
                        mov     [ebp+8], eax
                        push    0FFFFFFEBh
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    SubFunc3
                        add     esp, 8
                        mov     [ebp+8], eax
                        push    0FFFFFFF4h
                        push    6
                        push    0Fh
                        mov     edx, [ebp+8]
                        push    edx
                        call    SubFunc2
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    0FFFFFFE3h
                        mov     eax, [ebp+8]
                        push    eax
                        call    SubFunc3
                        add     esp, 8
                        mov     [ebp+8], eax
                        push    5
                        push    5
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    SubFunc0
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        mov     edx, [ebp+8]
                        xor     edx, 58887477h
                        mov     [ebp+8], edx
                        push    0FFFFFFF9h
                        push    9
                        push    10h
                        mov     eax, [ebp+8]
                        push    eax
                        call    SubFunc2
                        add     esp, 10h
                        mov     [ebp+8], eax
                        mov     ecx, [ebp+8]
                        xor     ecx, 949D26F0h
                        mov     [ebp+8], ecx
                        mov     eax, [ebp+8]
                        pop     ebp
                        retn
        }
}

__declspec(naked) void SixBlock9(void)
{
```

```
__asm {
            push    ebp
            mov     ebp, esp
            push    0Eh
            push    2
            mov     eax, [ebp+8]
            push    eax
            call    SubFunc1
            add     esp, 0Ch
            mov     [ebp+8], eax
            mov     ecx, [ebp+8]
            xor     ecx, 71B455A8h
            mov     [ebp+8], ecx
            push    2
            push    1Dh
            mov     edx, [ebp+8]
            push    edx
            call    SubFunc0
            add     esp, 0Ch
            mov     [ebp+8], eax
            push    0Bh
            push    8
            mov     eax, [ebp+8]
            push    eax
            call    SubFunc1
            add     esp, 0Ch
            mov     [ebp+8], eax
            push    0FFFFFFE7h
            push    1
            push    1Eh
            mov     ecx, [ebp+8]
            push    ecx
            call    SubFunc2
            add     esp, 10h
            mov     [ebp+8], eax
            push    0FFFFFFF6h
            mov     edx, [ebp+8]
            push    edx
            call    SubFunc3
            add     esp, 8
            mov     [ebp+8], eax
            push    14h
            push    1
            mov     eax, [ebp+8]
            push    eax
            call    SubFunc1
            add     esp, 0Ch
            mov     [ebp+8], eax
            mov     ecx, [ebp+8]
            xor     ecx, 247C11F6h
            mov     [ebp+8], ecx
            push    0FFFFFFEBh
```

```
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc3
                add     esp, 8
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                xor     eax, 0F097965Dh
                mov     [ebp+8], eax
                push    0Eh
                push    2
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0Eh
                push    0Fh
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc0
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0FFFFFFFEh
                push    17h
                push    3
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc2
                add     esp, 10h
                mov     [ebp+8], eax
                push    0FFFFFFFAh
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc3
                add     esp, 8
                mov     [ebp+8], eax
                mov     edx, [ebp+8]
                xor     edx, 0EECED199h
                mov     [ebp+8], edx
                mov     eax, [ebp+8]
                pop     ebp
                retn
    }
}

__declspec(naked) void SixBlockA(void)
{
    __asm {
                push    ebp
                mov     ebp, esp
                push    0FFFFFFF7h
                push    8
                push    17h
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc2
                add     esp, 10h
                mov     [ebp+8], eax
                push    9
                push    1
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc1
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     edx, [ebp+8]
                xor     edx, 71912DB8h
                mov     [ebp+8], edx
                push    8
                push    15h
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc0
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     ecx, [ebp+8]
                xor     ecx, 4A7BFE98h
                mov     [ebp+8], ecx
                push    3
                push    1Bh
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc0
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0FFFFFFFEh
                push    7
                push    5
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc2
                add     esp, 10h
                mov     [ebp+8], eax
                push    0FFFFFFF1h
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc3
                add     esp, 8
                mov     [ebp+8], eax
                push    0FFFFFFF9h
                push    4
                push    1Ah
                mov     edx, [ebp+8]
                push    edx
```

```
                         call    SubFunc2
                         add     esp, 10h
                         mov     [ebp+8], eax
                         push    16h
                         push    8
                         mov     eax, [ebp+8]
                         push    eax
                         call    SubFunc1
                         add     esp, 0Ch
                         mov     [ebp+8], eax
                         mov     ecx, [ebp+8]
                         xor     ecx, 0DCA20F48h
                         mov     [ebp+8], ecx
                         push    10h
                         push    7
                         mov     edx, [ebp+8]
                         push    edx
                         call    SubFunc0
                         add     esp, 0Ch
                         mov     [ebp+8], eax
                         push    0Ch
                         push    7
                         mov     eax, [ebp+8]
                         push    eax
                         call    SubFunc1
                         add     esp, 0Ch
                         mov     [ebp+8], eax
                         push    0FFFFFFFEh
                         push    8
                         push    4
                         mov     ecx, [ebp+8]
                         push    ecx
                         call    SubFunc2
                         add     esp, 10h
                         mov     [ebp+8], eax
                         push    0FFFFFFFFh
                         mov     edx, [ebp+8]
                         push    edx
                         call    SubFunc3
                         add     esp, 8
                         mov     [ebp+8], eax
                         mov     eax, [ebp+8]
                         xor     eax, 0FDEB38C7h
                         mov     [ebp+8], eax
                         mov     eax, [ebp+8]
                         pop     ebp
                         retn
        }
}

__declspec(naked) void SixBlockB(void)
{
```

```
__asm {
            push    ebp
            mov     ebp, esp
            push    0FFFFFFE7h
            push    3
            push    1Bh
            mov     eax, [ebp+8]
            push    eax
            call    SubFunc2
            add     esp, 10h
            mov     [ebp+8], eax
            push    0FFFFFFF2h
            mov     ecx, [ebp+8]
            push    ecx
            call    SubFunc3
            add     esp, 8
            mov     [ebp+8], eax
            mov     edx, [ebp+8]
            xor     edx, 0A59EEE9Ah
            mov     [ebp+8], edx
            push    0FFFFFFF8h
            push    5
            push    16h
            mov     eax, [ebp+8]
            push    eax
            call    SubFunc2
            add     esp, 10h
            mov     [ebp+8], eax
            push    1
            push    1Bh
            mov     ecx, [ebp+8]
            push    ecx
            call    SubFunc0
            add     esp, 0Ch
            mov     [ebp+8], eax
            push    0Eh
            push    8
            mov     edx, [ebp+8]
            push    edx
            call    SubFunc1
            add     esp, 0Ch
            mov     [ebp+8], eax
            push    0FFFFFFF5h
            mov     eax, [ebp+8]
            push    eax
            call    SubFunc3
            add     esp, 8
            mov     [ebp+8], eax
            push    3
            push    12h
            mov     ecx, [ebp+8]
            push    ecx
```

```
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFF7h
push    5
push    18h
mov     edx, [ebp+8]
push    edx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    15h
push    2
mov     eax, [ebp+8]
push    eax
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    1
push    18h
mov     ecx, [ebp+8]
push    ecx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFFCh
push    5
push    0Bh
mov     edx, [ebp+8]
push    edx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    3
push    4
mov     eax, [ebp+8]
push    eax
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 16F12999h
mov     [ebp+8], ecx
push    0Ah
push    9
mov     edx, [ebp+8]
push    edx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFEEh
mov     eax, [ebp+8]
```

```
                        push    eax
                        call    SubFunc3
                        add     esp, 8
                        mov     [ebp+8], eax
                        mov     ecx, [ebp+8]
                        xor     ecx, 3F4C93CAh
                        mov     [ebp+8], ecx
                        mov     eax, [ebp+8]
                        pop     ebp
                        retn
    }
}

__declspec(naked) void SixBlockC(void)
{
    __asm {
                        push    ebp
                        mov     ebp, esp
                        push    0Ch
                        push    6
                        mov     eax, [ebp+8]
                        push    eax
                        call    SubFunc1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    0FFFFFFF7h
                        push    2
                        push    0Eh
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    SubFunc2
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    0Ah
                        push    4
                        mov     edx, [ebp+8]
                        push    edx
                        call    SubFunc1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        mov     eax, [ebp+8]
                        xor     eax, 2ACA701Ah
                        mov     [ebp+8], eax
                        push    19h
                        push    2
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    SubFunc0
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    0Eh
                        push    6
```

```
            mov     edx, [ebp+8]
            push    edx
            call    SubFunc1
            add     esp, 0Ch
            mov     [ebp+8], eax
            push    0FFFFFFFEh
            push    3
            push    7
            mov     eax, [ebp+8]
            push    eax
            call    SubFunc2
            add     esp, 10h
            mov     [ebp+8], eax
            push    0Bh
            push    3
            mov     ecx, [ebp+8]
            push    ecx
            call    SubFunc1
            add     esp, 0Ch
            mov     [ebp+8], eax
            mov     edx, [ebp+8]
            xor     edx, 8C6C2052h
            mov     [ebp+8], edx
            push    0FFFFFFFEh
            push    0Ah
            push    8
            mov     eax, [ebp+8]
            push    eax
            call    SubFunc2
            add     esp, 10h
            mov     [ebp+8], eax
            push    0FFFFFFFCh
            mov     ecx, [ebp+8]
            push    ecx
            call    SubFunc3
            add     esp, 8
            mov     [ebp+8], eax
            push    0FFFFFFF8h
            push    6
            push    0Bh
            mov     edx, [ebp+8]
            push    edx
            call    SubFunc2
            add     esp, 10h
            mov     [ebp+8], eax
            mov     eax, [ebp+8]
            xor     eax, 10AFC3E4h
            mov     [ebp+8], eax
            mov     eax, [ebp+8]
            pop     ebp
            retn
}
```

```
}

__declspec(naked) void SixBlockD(void)
{
     __asm {
                  push    ebp
                  mov     ebp, esp
                  push    0FFFFFFE5h
                  mov     eax, [ebp+8]
                  push    eax
                  call    SubFunc3
                  add     esp, 8
                  mov     [ebp+8], eax
                  push    7
                  push    0Eh
                  mov     ecx, [ebp+8]
                  push    ecx
                  call    SubFunc0
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    0FFFFFFE8h
                  mov     edx, [ebp+8]
                  push    edx
                  call    SubFunc3
                  add     esp, 8
                  mov     [ebp+8], eax
                  push    0FFFFFFECh
                  push    3
                  push    1Bh
                  mov     eax, [ebp+8]
                  push    eax
                  call    SubFunc2
                  add     esp, 10h
                  mov     [ebp+8], eax
                  push    12h
                  push    0Bh
                  mov     ecx, [ebp+8]
                  push    ecx
                  call    SubFunc0
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    0FFFFFFEAh
                  push    0
                  push    1Dh
                  mov     edx, [ebp+8]
                  push    edx
                  call    SubFunc2
                  add     esp, 10h
                  mov     [ebp+8], eax
                  mov     eax, [ebp+8]
                  xor     eax, 414B8E93h
                  mov     [ebp+8], eax
```

```
push    0Dh
push    7
mov     ecx, [ebp+8]
push    ecx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFFBh
push    8
push    10h
mov     edx, [ebp+8]
push    edx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    6
push    19h
mov     eax, [ebp+8]
push    eax
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFE1h
mov     ecx, [ebp+8]
push    ecx
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    0Ch
push    3
mov     edx, [ebp+8]
push    edx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
mov     eax, [ebp+8]
xor     eax, 0B4324752h
mov     [ebp+8], eax
push    0FFFFFFFAh
push    1
push    14h
mov     ecx, [ebp+8]
push    ecx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    17h
push    4
mov     edx, [ebp+8]
push    edx
call    SubFunc1
add     esp, 0Ch
```

```
                mov     [ebp+8], eax
                push    0
                push    0Fh
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc0
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0FFFFFFECh
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc3
                add     esp, 8
                mov     [ebp+8], eax
                push    0Ch
                push    0Dh
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc0
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0Ch
                push    7
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc1
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     ecx, [ebp+8]
                xor     ecx, 96174FB5h
                mov     [ebp+8], ecx
                mov     eax, [ebp+8]
                pop     ebp
                retn
     }
}

__declspec(naked) void SixBlockE(void)
{
     __asm {
                push    ebp
                mov     ebp, esp
                push    15h
                push    7
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0FFFFFFFDh
                push    4
                push    15h
```

```
mov     ecx, [ebp+8]
push    ecx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    0FFFFFFF5h
mov     edx, [ebp+8]
push    edx
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    2
push    0Bh
mov     eax, [ebp+8]
push    eax
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFECh
mov     ecx, [ebp+8]
push    ecx
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    0FFFFFFFEh
push    1
push    7
mov     edx, [ebp+8]
push    edx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    16h
push    8
mov     eax, [ebp+8]
push    eax
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    0Eh
push    10h
mov     ecx, [ebp+8]
push    ecx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFFBh
push    9
push    12h
mov     edx, [ebp+8]
push    edx
call    SubFunc2
```

```
                add     esp, 10h
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                xor     eax, 0DDF185A2h
                mov     [ebp+8], eax
                push    0FFFFFFFCh
                push    13h
                push    0Bh
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc2
                add     esp, 10h
                mov     [ebp+8], eax
                push    14h
                push    5
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    11h
                push    0Bh
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc0
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    16h
                push    7
                mov     ecx, [ebp+8]
                push    ecx
                call    SubFunc1
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     edx, [ebp+8]
                xor     edx, 8B8BAF82h
                mov     [ebp+8], edx
                mov     eax, [ebp+8]
                pop     ebp
                retn
    }
}

__declspec(naked) void SixBlockF(void)
{
    __asm {
                push    ebp
                mov     ebp, esp
                push    2
                push    1Bh
                mov     eax, [ebp+8]
                push    eax
```

```
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFF6h
push    0
push    12h
mov     ecx, [ebp+8]
push    ecx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, offset dword_552CA9
mov     [ebp+8], edx
push    3
push    18h
mov     eax, [ebp+8]
push    eax
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    0Eh
push    6
mov     ecx, [ebp+8]
push    ecx
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 1AD7F4D0h
mov     [ebp+8], edx
push    3
push    15h
mov     eax, [ebp+8]
push    eax
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFF5h
push    0
push    1Fh
mov     ecx, [ebp+8]
push    ecx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    9
push    2
mov     edx, [ebp+8]
push    edx
call    SubFunc1
add     esp, 0Ch
```

```
mov     [ebp+8], eax
push    1
push    1Ah
mov     eax, [ebp+8]
push    eax
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 87E29F3Ch
mov     [ebp+8], ecx
push    0FFFFFFEEh
mov     edx, [ebp+8]
push    edx
call    SubFunc3
add     esp, 8
mov     [ebp+8], eax
push    0FFFFFFFCh
push    5
push    17h
mov     eax, [ebp+8]
push    eax
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    1
push    1Dh
mov     ecx, [ebp+8]
push    ecx
call    SubFunc0
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFF9h
push    8
push    17h
mov     edx, [ebp+8]
push    edx
call    SubFunc2
add     esp, 10h
mov     [ebp+8], eax
push    16h
push    6
mov     eax, [ebp+8]
push    eax
call    SubFunc1
add     esp, 0Ch
mov     [ebp+8], eax
push    1Ah
push    4
mov     ecx, [ebp+8]
push    ecx
call    SubFunc0
```

```
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0FFFFFFFBh
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc3
                add     esp, 8
                mov     [ebp+8], eax
                push    0FFFFFFFCh
                push    2
                push    0Eh
                mov     eax, [ebp+8]
                push    eax
                call    SubFunc2
                add     esp, 10h
                mov     [ebp+8], eax
                mov     ecx, [ebp+8]
                xor     ecx, 17019638h
                mov     [ebp+8], ecx
                push    13h
                push    7
                mov     edx, [ebp+8]
                push    edx
                call    SubFunc1
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                xor     eax, 6760502h
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                pop     ebp
                retn
     }
}


// block 6 dynamic functions

__declspec(naked) void sub_48D4EE(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCDC_2
                add     esp, 4
```

```
mov     [ebp-4], eax
mov     eax, [ebp-4]
push    edx
mov     edx, [ebp+0Ch]
mov     edx, 0FFFFh
and     eax, edx
push    ebx
push    eax
mov     bh, 7
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
and     eax, 800h
bswap   ecx
pop     eax
bswap   ecx
and     ah, bh
mov     bl, 86h
sub     bl, 7
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
sub     bl, 1Ah
sub     bl, 20h
not     bx
bswap   eax
not     bx
bswap   eax
and     al, bl
mov     eax, eax
test    eax, eax
jnz     loc_48D60C
pop     ebx
pop     edx
mov     eax, [ebp-4]
push    edx
mov     edx, 0FFFFh
and     eax, edx
push    ebx
push    eax
mov     bh, 7
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
```

```
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 98h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 0Ch
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    0Dh
                pop     ebx
                jo      short loc_48D5D1
                jl      short loc_48D5CF

loc_48D5CA:
                jmp     short loc_48D5D3

loc_48D5CF:
                jz      short loc_48D5CA

loc_48D5D1:
                jmp     short loc_48D5CA


loc_48D5D3:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                and     eax, 41h
                dec     bl
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 2500h
                dec     dh
                sub     dh, 3
                dec     dh
                sub     dh, 18h
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_48D60C
                and     eax, 0
                jmp     short loc_48D610

loc_48D60C:
                and     eax, 0
                inc     eax

loc_48D610:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9370
                xor     ecx, ds:dword_4D9374
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48D633
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48D633:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDC9C
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
        }
 }


__declspec(naked) void sub_48D2A4(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE8
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Dh
                dec     ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 7
                dec     ch
                dec     ch
                and     ah, ch
                mov     cl, 77h
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     cl
                bswap   edx
                not     cl
                bswap   edx
                dec     cl
                dec     cl
                push    eax
                dec     cl
                dec     cl
                sub     cl, 12h
                dec     cl
                jo      short loc_48D30C
                jl      short loc_48D30A

loc_48D307:

                jmp     short loc_48D30E

loc_48D30A:
                jz      short loc_48D307

loc_48D30C:
                jmp     short loc_48D307

loc_48D30E:
                dec     cl
                and     eax, 40h
                dec     cl
                dec     cl
                dec     cl
                add     cl, 0Eh
                dec     cl
                dec     cl
                and     eax, 800h
                sub     cl, 1Fh
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                pop     eax
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D937C
                xor     ecx, ds:dword_4D9380
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48D366
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48D366:
```

```
                    mov     eax, [ebp-8]
                    push    eax
                    call    ds:off_4DDCA8
                    add     esp, 4
                    pop     edi
                    pop     esi
                    pop     ebx
                    mov     esp, ebp
                    pop     ebp
                    retn
           }
}


__declspec(naked) void sub_482654(void)
{
     __asm
     {
                    push    ebp
                    mov     ebp, esp
                    sub     esp, 8
                    push    ebx
                    push    esi
                    push    edi
                    mov     eax, [ebp+8]
                    dec     bh
                    and     eax, 800h
                    jo      short loc_482670
                    jl      short loc_48266E

 loc_48266B:

                    jmp     short loc_482672
 loc_48266E:
                    jz      short loc_48266B

 loc_482670:
                    jmp     short loc_48266B

 loc_482672:
                    mov     ebx, 4
                    and     eax, ebx
                    mov     ch, 52h
                    dec     ch
                    mov     ebx, [ebp+0Ch]
                    test    ebx, ebx
                    jo      short loc_48268B
                    jl      short loc_482689

 loc_482686:

                    jmp     short loc_48268D
```

```
loc_482689:
                jz      short loc_482686

loc_48268B:
                jmp     short loc_482686

loc_48268D:
                jz      short loc_482698
                dec     edi
                sub     ch, 3
                and     eax, 0
                jmp     short loc_4826B9

loc_482698:
                dec     edi
                dec     ecx
                sub     ch, 2
                dec     ch
                dec     ch
                sub     ch, 8
                jo      short loc_4826AD
                jl      short loc_4826AB

loc_4826A8:

                jmp     short loc_4826AF

loc_4826AB:
                jz      short loc_4826A8

loc_4826AD:
                jmp     short loc_4826A8

loc_4826AF:
                and     eax, 0
                dec     ecx
                sub     ch, 2
                inc     eax
                dec     ch

loc_4826B9:
                mov     [ebp-8], eax
                mov     eax, ds:dword_4D93AC
                xor     eax, ds:dword_4D93B0
                shl     eax, 1
                mov     [ebp-4], eax
                cmp     dword ptr [ebp-8], 0
                jz      short loc_4826DB
                mov     ecx, [ebp-4]
                or      ecx, 1
                mov     [ebp-4], ecx
```

```
 loc_4826DB:
                mov     edx, [ebp-4]
                push    edx
                call    ds:off_4DDCD8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
 }


__declspec(naked) void sub_48A696(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD14
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                mov     ecx, 800h
                mov     ecx, 0Ah
                not     ecx
                bswap   eax
                not     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     edx
                inc     ecx
                inc     ecx
                dec     edx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     edx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     cl
                dec     edx
                inc     ecx
                inc     ecx
                dec     edx
                inc     ecx
                inc     ecx
                inc     cl
                dec     edx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     ecx
                dec     edx
                dec     edx
                inc     ecx
                inc     cl
                inc     cl
                dec     dl
                inc     cl
                add     ecx, 0Bh
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Ah
                dec     ecx
                push    edx
                mov     edx, 4
                add     ecx, edx
                inc     ecx
                pop     edx
                bswap   eax
                and     eax, ecx
                pop     ecx
                neg     eax
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A8
                xor     ecx, ds:dword_4D93AC
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48A735
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48A735:
```

```
                        mov     eax, [ebp-8]
                        push    eax
                        call    ds:off_4DDCD4
                        add     esp, 4
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
        }
}



__declspec(naked) void sub_48B8EB(void)
{
        __asm
        {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDCE4
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    ebx
                        mov     ebx, 0FFFFh
                        and     eax, ebx
                        push    ecx
                        mov     ch, 2Dh
                        dec     ch
                        sub     ch, 1
                        sub     ch, 20h
                        dec     ch
                        dec     ch
                        sub     ch, 7
                        dec     ch
                        dec     ch
                        and     ah, ch
                        mov     cl, 0BDh
                        sub     cl, 2
                        dec     cl
                        dec     cl
                        dec     cl
                        dec     cl
                        dec     cl
                        dec     cl
                        dec     cl
                        not     cl
                        bswap   edx
                        not     cl
                        bswap   edx
                        dec     cl
                        dec     cl
                        dec     cl
                        dec     cl
                        push    eax
                        dec     cl
                        dec     cl
                        sub     cl, 12h
                        dec     cl
                        dec     cl
                        sub     cl, 3
                        dec     cl
                        and     eax, 40h
                        dec     cl
                        dec     cl
                        dec     cl
                        add     cl, 0Eh
                        dec     cl
                        dec     cl
                        and     eax, 80h
                        sub     cl, 1Fh
                        dec     cl
                        dec     cl
                        dec     cl
                        not     ecx
                        bswap   eax
                        not     ecx
                        bswap   eax
                        pop     eax
                        inc     cl
                        inc     cl
                        inc     cl
                        and     al, cl
                        mov     eax, eax
                        pop     ecx
                        neg     eax
                        sbb     eax, eax
                        inc     eax
                        pop     ebx
                        push    eax
                        mov     eax, [ebp-4]
                        mov     edx, 0C00h
                        sub     dh, 1
                        dec     dh
                        dec     dh
                        dec     dh
```

```
                and     eax, edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                cmp     eax, edx
                setz    cl
                mov     al, cl
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9378
                xor     ecx, ds:dword_4D937C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48B9DB
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48B9DB:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_48B4C8(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD10
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                mov     ecx, 41h
                not     ecx
                bswap   eax
                not     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                add     ecx, 0Dh
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Fh
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Ah
                dec     ecx
                push    edx
                mov     edx, 4
                add     ecx, edx
                inc     ecx
                pop     edx
                bswap   eax
                add     ecx, 3
                and     eax, ecx
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                push    eax
                mov     eax, [ebp-4]
                mov     edx, 0F00h
                sub     dh, 1
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                dec     dh
```

```
                dec     dh
                and     eax, edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                jo      short loc_48B559
                jl      short loc_48B557

loc_48B554:

                jmp     short loc_48B55B



loc_48B557:
                jz      short loc_48B554

loc_48B559:
                jmp     short loc_48B554


loc_48B55B:
                cmp     eax, edx
                jo      short loc_48B566
                jl      short loc_48B564

loc_48B561:

                jmp     short loc_48B568



loc_48B564:
                jz      short loc_48B561

loc_48B566:
                jmp     short loc_48B561


loc_48B568:
                jz      short loc_48B57B
                and     eax, 0
                jo      short loc_48B576
                jl      short loc_48B574

loc_48B571:

                jmp     short loc_48B578
```

```
 loc_48B574:
                jz      short loc_48B571

 loc_48B576:
                jmp     short loc_48B571


 loc_48B578:
                inc     eax
                jmp     short loc_48B57E


 loc_48B57B:
                and     eax, 0

 loc_48B57E:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A4
                xor     ecx, ds:dword_4D93A8
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48B5A1
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48B5A1:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_47F71C(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF4
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1E00h
                pop     ebx
                jo      short loc_47F752
                jl      short loc_47F750

loc_47F74B:

                jmp     short loc_47F754

loc_47F750:
                jz      short loc_47F74B

loc_47F752:
                jmp     short loc_47F74B

loc_47F754:
                sub     bh, 4
                dec     bh
                push    eax
                dec     bh
                dec     bh
                jo      short loc_47F769
                jl      short loc_47F767

loc_47F762:

                jmp     short loc_47F76B

loc_47F767:
                jz      short loc_47F762

loc_47F769:
                jmp     short loc_47F762

loc_47F76B:
                and     eax, 40h
                sub     bh, 13h
                sub     bh, 3
                pop     eax
                dec     bh
                and     ah, bh
                mov     edx, 12h
                dec     dl
                jo      short loc_47F78B
                jl      short loc_47F789

loc_47F784:

                jmp     short loc_47F78D

loc_47F789:
                jz      short loc_47F784

loc_47F78B:
                jmp     short loc_47F784

loc_47F78D:
                sub     dl, 1
                dec     dl
                sub     dl, 7
                dec     dl
                dec     dl
                sub     dl, 2
                jo      short loc_47F7A7
                jl      short loc_47F7A5

loc_47F7A0:

                jmp     short loc_47F7A9

loc_47F7A5:
                jz      short loc_47F7A0

loc_47F7A7:
                jmp     short loc_47F7A0

loc_47F7A9:
                and     al, dl
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9388
                xor     ecx, ds:dword_4D938C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_47F7D5
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_47F7D5:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48DA6B(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD10
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 80h
                jmp     short loc_48DA94
                mov     ebx, 4
 loc_48DA94:
                mov     ebx, 32h
                not     ebx
                bswap   eax
                not     ebx
                inc     ebx
                inc     ebx
                add     ebx, 8
                dec     ebx
                push    ecx
                mov     ecx, 4
```

```
                 add     ebx, ecx
                 inc     ebx
                 pop     ecx
                 bswap   eax
                 and     eax, ebx
                 pop     ebx
                 neg     eax
                 sbb     eax, eax
                 inc     eax
                 pop     edx
                 mov     [ebp-0Ch], eax
                 mov     ecx, ds:dword_4D93A4
                 xor     ecx, ds:dword_4D93A8
                 shl     ecx, 1
                 mov     [ebp-8], ecx
                 cmp     dword ptr [ebp-0Ch], 0
                 jz      short loc_48DADD
                 mov     edx, [ebp-8]
                 or      edx, 1
                 mov     [ebp-8], edx
loc_48DADD:
                 mov     eax, [ebp-8]
                 push    eax
                 call    ds:off_4DDCD0
                 add     esp, 4
                 pop     edi
                 pop     esi
                 pop     ebx
                 mov     esp, ebp
                 pop     ebp
                 retn
     }
}



__declspec(naked) void sub_48A34A(void)
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 sub     esp, 0Ch
                 push    ebx
                 push    esi
                 push    edi
                 mov     eax, [ebp+8]
                 push    eax
                 call    ds:off_4DDCE8
                 add     esp, 4
                 mov     [ebp-4], eax
                 mov     eax, [ebp-4]
```

```
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    5
                pop     ebx
                dec     bl
                dec     bl
                sub     bl, 2
                dec     bl
                and     al, bl
                mov     dh, 0Dh
                and     dl, 0
                sub     dh, 3
                dec     dh
                sub     dh, 1
                and     ah, dh
                pop     ebx
                pop     edx
                test    eax, eax
                jz      short loc_48A39A
                not     eax
                add     eax, 1
                stc
                jmp     short loc_48A3A0

loc_48A39A:
                not     eax
                add     eax, 1
                clc

loc_48A3A0:
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D937C
                xor     ecx, ds:dword_4D9380
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48A3C7
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48A3C7:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_48778D(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD00
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Dh
                dec     ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 7
                dec     ch
                dec     ch
                and     ah, ch
                mov     cl, 75h
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     cl
                bswap   edx
                not     cl
                bswap   edx
                dec     cl
                dec     cl
                push    eax
                dec     cl
                dec     cl
                sub     cl, 12h
                dec     cl
                jo      short loc_4877F2
                jl      short loc_4877F0

loc_4877ED:
                jmp     short loc_4877F4

loc_4877F0:
                jz      short loc_4877ED

loc_4877F2:
                jmp     short loc_4877ED

loc_4877F4:
                dec     cl
                and     eax, 40h
                dec     cl
                dec     cl
                dec     cl
                add     cl, 0Eh
                dec     cl
                dec     cl
                and     eax, 800h
                sub     cl, 1Fh
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                pop     eax
                and     al, cl
                mov     eax, eax
                pop     ecx
                pop     ebx
                test    eax, eax
                jnz     loc_4878DB
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                inc     bh
                inc     bh
                inc     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 80h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 98h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 0Ch
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                push    4
                pop     ecx
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                and     al, cl
                mov     bh, 0Fh
                and     bl, 0
                sub     bh, 4
                dec     bh
```

```
                sub     bh, 1
                dec     bh
                and     ah, bh
                pop     ecx
                pop     ebx
                test    eax, eax
                jz      short loc_4878C6
                not     eax
                add     eax, 1
                stc
                jmp     short loc_4878CC

loc_4878C6:
                not     eax
                add     eax, 1
                clc

loc_4878CC:
                sbb     eax, eax
                add     eax, 1
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_4878DB
                and     eax, 0
                jmp     short loc_4878DF

loc_4878DB:
                and     eax, 0
                inc     eax

loc_4878DF:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9394
                xor     ecx, ds:dword_4D9398
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_487902
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_487902:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
```

```
                        retn
        }
}





__declspec(naked) void sub_48BCA8(void)
{
        __asm
        {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDCE8
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    edx
                        mov     edx, [ebp+0Ch]
                        mov     edx, 0FFFFh
                        and     eax, edx
                        push    ebx
                        push    eax
                        mov     bh, 7
                        dec     bh
                        dec     bh
                        dec     bh
                        dec     bh
                        dec     bh
                        dec     bh
                        dec     bh
                        and     eax, 800h
                        bswap   ecx
                        pop     eax
                        bswap   ecx
                        and     ah, bh
                        mov     bl, 86h
                        sub     bl, 5
                        dec     bl
                        dec     bl
                        dec     bl
                        dec     bl
                        dec     bl
                        dec     bl
                        dec     bl
                        sub     bl, 3Ah
                        not     bx
                        bswap   eax
                        not     bx
                        bswap   eax
                        and     al, bl
                        pop     ebx
                        pop     edx
                        test    eax, eax
                        jnz     loc_48BDFA
                        mov     eax, [ebp-4]
                        push    edx
                        mov     edx, 0FFFFh
                        and     eax, edx
                        push    ebx
                        push    eax
                        mov     bh, 4
                        dec     bh
                        dec     bh
                        dec     bh
                        dec     bh
                        and     eax, 800h
                        bswap   ecx
                        pop     eax
                        bswap   ecx
                        and     ah, bh
                        mov     bl, 98h
                        sub     bl, 5
                        dec     bl
                        dec     bl
                        dec     bl
                        dec     bl
                        dec     bl
                        dec     bl
                        dec     bl
                        sub     bl, 0Ch
                        not     bx
                        bswap   eax
                        not     bx
                        bswap   eax
                        and     al, bl
                        mov     eax, eax
                        pop     ebx
                        neg     eax
                        sbb     eax, eax
                        inc     eax
                        pop     edx
                        mov     ecx, eax
                        push    ecx
                        mov     eax, [ebp-4]
                        push    edx
```

```
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_48BD84
                jl      short loc_48BD82

loc_48BD7D:

                jmp     short loc_48BD86

loc_48BD82:
                jz      short loc_48BD7D

loc_48BD84:
                jmp     short loc_48BD7D

loc_48BD86:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                jo      short loc_48BD99
                jl      short loc_48BD97

loc_48BD94:

                jmp     short loc_48BD9B

loc_48BD97:
                jz      short loc_48BD94

loc_48BD99:
                jmp     short loc_48BD94

loc_48BD9B:
                and     eax, 40h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1200h
                dec     dh
                sub     dh, 1
                dec     dh
                sub     dh, 7
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_48BDCF
                jl      short loc_48BDCD

loc_48BDC8:

                jmp     short loc_48BDD1

loc_48BDCD:
                jz      short loc_48BDC8

loc_48BDCF:
                jmp     short loc_48BDC8

loc_48BDD1:
                inc     eax
                dec     eax
                jo      short loc_48BDDE
                jl      short loc_48BDDC

loc_48BDD7:

                jmp     short loc_48BDE0

loc_48BDDC:
                jz      short loc_48BDD7

loc_48BDDE:
                jmp     short loc_48BDD7

loc_48BDE0:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_48BDED
                jl      short loc_48BDEB

loc_48BDE8:

                jmp     short loc_48BDEF

loc_48BDEB:
                jz      short loc_48BDE8

loc_48BDED:
                jmp     short loc_48BDE8
```

```
 loc_48BDEF:
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_48BDFA
                and     eax, 0
                jmp     short loc_48BDFE

 loc_48BDFA:

                and     eax, 0
                inc     eax

 loc_48BDFE:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D937C
                xor     ecx, ds:dword_4D9380
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48BE21
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48BE21:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_489D38(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD08
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Dh
                dec     ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 7
                dec     ch
                dec     ch
                and     ah, ch
                mov     cl, 77h
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     cl
                bswap   edx
                not     cl
                bswap   edx
                dec     cl
                dec     cl
                push    eax
                dec     cl
                dec     cl
                sub     cl, 12h
                dec     cl
                jo      short loc_489DA0
                jl      short loc_489D9E

loc_489D9B:
                jmp     short loc_489DA2

loc_489D9E:
                jz      short loc_489D9B

loc_489DA0:
                jmp     short loc_489D9B
loc_489DA2:
                dec     cl
                and     eax, 40h
                dec     cl
                dec     cl
                dec     cl
                add     cl, 0Eh
                dec     cl
                dec     cl
                and     eax, 800h
                sub     cl, 1Fh
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                pop     eax
                and     al, cl
                mov     eax, eax
                pop     ecx
                pop     ebx
                test    eax, eax
                jnz     loc_489E7F
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 98h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 0Ch
                not     bx
                bswap   eax
```

```
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                push    4
                pop     ecx
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                and     al, cl
                mov     bh, 0Fh
                and     bl, 0
                dec     bh
                sub     bh, 3
                dec     bh
                sub     bh, 1
                dec     bh
                and     ah, bh
                pop     ecx
                pop     ebx
                test    eax, eax
                jz      short loc_489E6A
                not     eax
                add     eax, 1
                stc
                jmp     short loc_489E70
loc_489E6A:
                not     eax
                add     eax, 1
                clc

loc_489E70:
                sbb     eax, eax
                add     eax, 1
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_489E7F
                and     eax, 0
                jmp     short loc_489E83
```

```
 loc_489E7F:
                and     eax, 0
                inc     eax

 loc_489E83:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D939C
                xor     ecx, ds:dword_4D93A0
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_489EA6
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_489EA6:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_486E92(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD0C
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                mov     ecx, 800h
                mov     ecx, 0Ch
                not     ecx
                bswap   eax
                not     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                dec     ecx
                dec     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                inc     ecx
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Dh
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Ah
                dec     ecx
```

```
push    edx
mov     edx, 4
add     ecx, edx
inc     ecx
pop     edx
bswap   eax
and     eax, ecx
pop     ecx
pop     edx
test    eax, eax
jnz     loc_487009
mov     eax, [ebp-4]
push    ebx
mov     ebx, 0FFFFh
and     eax, ebx
push    ecx
mov     ch, 2Ch
sub     ch, 1
sub     ch, 20h
dec     ch
dec     ch
sub     ch, 4
dec     ch
sub     ch, 3
dec     ch
and     ah, ch
mov     cl, 0AEh
sub     cl, 2
dec     cl
dec     cl
sub     cl, 6
not     al
bswap   ecx
not     al
bswap   ecx
dec     cl
dec     cl
sub     cl, 10h
dec     cl
dec     cl
add     cl, 0Ch
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
sub     cl, 10h
sub     cl, 1
dec     cl
dec     cl
dec     cl
```

```
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                jo      short loc_486F93
                jl      short loc_486F91

loc_486F8C:
                jmp     short loc_486F95

loc_486F91:
                jz      short loc_486F8C

loc_486F93:
                jmp     short loc_486F8C

loc_486F95:
                and     al, cl
                pop     ecx
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_486FBB
                jl      short loc_486FB9

loc_486FB4:
                jmp     short loc_486FBD

loc_486FB9:
                jz      short loc_486FB4

loc_486FBB:
                jmp     short loc_486FB4

loc_486FBD:
```

```
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                and     eax, 41h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1500h
                dec     dh
                sub     dh, 7
                dec     dh
                sub     dh, 3
                dec     dh
                jo      short loc_486FF3
                jl      short loc_486FF1

loc_486FEC:
                jmp     short loc_486FF5

loc_486FF1:
                jz      short loc_486FEC

loc_486FF3:
                jmp     short loc_486FEC

loc_486FF5:
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_487009
                and     eax, 0
                inc     eax
                jmp     short loc_48700C

loc_487009:
                and     eax, 0

loc_48700C:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A0
                xor     ecx, ds:dword_4D93A4
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48702F
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48702F:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCCC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}





__declspec(naked) void sub_485AD2(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD0C
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Dh
                dec     ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
```

```
sub     ch, 7
dec     ch
dec     ch
and     ah, ch
mov     cl, 0BDh
sub     cl, 2
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
not     cl
bswap   edx
not     cl
bswap   edx
dec     cl
dec     cl
dec     cl
dec     cl
push    eax
dec     cl
dec     cl
sub     cl, 12h
dec     cl
dec     cl
sub     cl, 3
dec     cl
and     eax, 40h
dec     cl
dec     cl
dec     cl
add     cl, 0Eh
dec     cl
dec     cl
and     eax, 80h
sub     cl, 1Fh
dec     cl
dec     cl
dec     cl
not     ecx
bswap   eax
not     ecx
bswap   eax
pop     eax
inc     cl
inc     cl
inc     cl
and     al, cl
mov     eax, eax
pop     ecx
```

```
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A0
                xor     ecx, ds:dword_4D93A4
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_485B9E
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_485B9E:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCCC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



 __declspec(naked) void sub_48A4A6(void)
 {
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF4
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 800h
                jmp     short loc_48A4CF
                mov     ebx, 80h
```

```
loc_48A4CF:
                mov     ebx, 72h
                not     ebx
                bswap   eax
                not     ebx
                inc     ebx
                inc     ebx
                add     ebx, 8
                dec     ebx
                push    ecx
                mov     ecx, 4
                add     ebx, ecx
                inc     ebx
                pop     ecx
                bswap   eax
                and     eax, ebx
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                push    eax
                mov     eax, [ebp-4]
                mov     edx, 0F00h
                sub     dh, 1
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                and     eax, edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                jo      short loc_48A522
                jl      short loc_48A520

loc_48A51D:
                jmp     short loc_48A524

loc_48A520:
                jz      short loc_48A51D

loc_48A522:
                jmp     short loc_48A51D

loc_48A524:
```

```
                cmp     eax, edx
                jo      short loc_48A52F
                jl      short loc_48A52D

loc_48A52A:
                jmp     short loc_48A531

loc_48A52D:
                jz      short loc_48A52A

loc_48A52F:
                jmp     short loc_48A52A

loc_48A531:
                jnz     short loc_48A543
                jo      short loc_48A53C
                jl      short loc_48A53A

loc_48A537:
                jmp     short loc_48A53E

loc_48A53A:
                jz      short loc_48A537

loc_48A53C:
                jmp     short loc_48A537

loc_48A53E:
                and     eax, 0
                jmp     short loc_48A552

loc_48A543:
                and     eax, 0
                jo      short loc_48A54F
                jl      short loc_48A54D

loc_48A54A:
                jmp     short loc_48A551

loc_48A54D:
                jz      short loc_48A54A

loc_48A54F:
                jmp     short loc_48A54A

loc_48A551:
                inc     eax

loc_48A552:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9388
                xor     ecx, ds:dword_4D938C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48A575
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48A575:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_484B20(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE0
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 87h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9374
                xor     ecx, ds:dword_4D9378
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_484BB6
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_484BB6:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
```

```
     }
}




__declspec(naked) void sub_48873D(void)
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 sub     esp, 0Ch
                 push    ebx
                 push    esi
                 push    edi
                 mov     eax, [ebp+8]
                 push    eax
                 call    ds:off_4DDD10
                 add     esp, 4
                 mov     [ebp-4], eax
                 mov     eax, [ebp-4]
                 jo      short loc_488762
                 jl      short loc_488760

 loc_48875D:
                 jmp     short loc_488764

 loc_488760:
                 jz      short loc_48875D

 loc_488762:
                 jmp     short loc_48875D

 loc_488764:
                 push    edx
                 mov     edx, 0FFFFh
                 and     eax, edx
                 push    ebx
                 push    eax
                 mov     bh, 7
                 xor     bh, 7
                 and     eax, 800h
                 bswap   ecx
                 pop     eax
                 bswap   ecx
                 and     ah, bh
                 jo      short loc_488788
                 jl      short loc_488786

 loc_488783:
```

```
                jmp     short loc_48878A
loc_488786:
                jz      short loc_488783

loc_488788:
                jmp     short loc_488783

loc_48878A:
                mov     bl, 0C6h
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                jo      short loc_4887BF
                jl      short loc_4887BD

loc_4887BA:
                jmp     short loc_4887C1

loc_4887BD:
                jz      short loc_4887BA

loc_4887BF:
                jmp     short loc_4887BA

loc_4887C1:
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A4
                xor     ecx, ds:dword_4D93A8
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4887EF
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_4887EF:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48D647(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF8
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 800h
                jmp     short loc_48D670
                mov     ebx, 80h

loc_48D670:
                mov     ebx, 72h
                not     ebx
                bswap   eax
                not     ebx
                inc     ebx
                inc     ebx
                add     ebx, 8
                dec     ebx
                push    ecx
                mov     ecx, 4
                add     ebx, ecx
                inc     ebx
                pop     ecx
                bswap   eax
                and     eax, ebx
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                push    eax
                mov     eax, [ebp-4]
                mov     edx, 0F00h
                sub     dh, 1
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                and     eax, edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                cmp     eax, edx
                setz    cl
                mov     al, cl
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D938C
                xor     ecx, ds:dword_4D9390
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48D6E4
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48D6E4:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48649F(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCEC
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 5
                dec     bh
                sub     bh, 1
                dec     bh
                sub     bh, 2
                and     eax, 80h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 0A1h
                dec     bl
                dec     esi
                dec     bl
                sub     bl, 0Eh
                dec     bl
                dec     bl
                dec     bl
                dec     edi
                dec     bl
                dec     bl
                sub     bl, 0Fh
                dec     bl
                dec     bl
                dec     edi
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9380
                xor     ecx, ds:dword_4D9384
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48653B
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48653B:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCAC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_4857D3(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD10
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    100h
                pop     ebx
                dec     bh
                jo      short loc_485809
                jl      short loc_485807

 loc_485804:
                jmp     short loc_48580B

 loc_485807:
                jz      short loc_485804

 loc_485809:
                jmp     short loc_485804

 loc_48580B:
                add     bh, 0FFh
                add     bh, 0FFh
                add     bh, 0FFh
                add     bh, 0FFh
                inc     bh
                inc     bh
                inc     bh
                inc     bh
                and     ah, bh
                jo      short loc_48582A
                jl      short loc_485828

 loc_485825:
                jmp     short loc_48582C

 loc_485828:
                jz      short loc_485825
```

```
loc_48582A:
                jmp     short loc_485825

loc_48582C:
                mov     bl, 14h
                dec     bl
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 3
                and     al, bl
                pop     ebx
                pop     edx
                test    eax, eax
                jz      short loc_485855
                not     eax
                add     eax, 1
                stc
                jmp     short loc_48585B

loc_485855:
                not     eax
                add     eax, 1
                clc

loc_48585B:
                sbb     eax, eax
                neg     eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A4
                xor     ecx, ds:dword_4D93A8
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_485884
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_485884:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD0
                add     esp, 4
                pop     edi
                pop     esi
```

```
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
        }
}




__declspec(naked) void sub_48A2BE(void)
{
        __asm
        {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDCFC
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    edx
                        mov     edx, 0FFFFh
                        and     eax, edx
                        push    ebx
                        push    2
                        pop     ebx
                        dec     bl
                        dec     bl
                        and     al, bl
                        mov     dh, 0Eh
                        and     dl, 0
                        sub     dh, 4
                        dec     dh
                        sub     dh, 1
                        and     ah, dh
                        pop     ebx
                        pop     edx
                        test    eax, eax
                        jz      short loc_48A309
                        not     eax
                        add     eax, 1
                        stc
                        jmp     short loc_48A30F
 loc_48A309:
                        not     eax
```

```
                add     eax, 1
                clc

 loc_48A30F:
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9390
                xor     ecx, ds:dword_4D9394
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48A336
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48A336:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCBC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48ABB6(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD04
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                jo      short loc_48ABDB
                jl      short loc_48ABD9

loc_48ABD6:
                jmp     short loc_48ABDD

loc_48ABD9:
                jz      short loc_48ABD6

loc_48ABDB:
                jmp     short loc_48ABD6

loc_48ABDD:
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 4
                dec     ch
                sub     ch, 3
                dec     ch
                and     ah, ch
                mov     cl, 70h
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 6
                not     al
                bswap   ecx
                not     al
                bswap   ecx
                dec     cl
                dec     cl
                sub     cl, 10h
                dec     cl
                dec     cl
                add     cl, 0Ch
                dec     cl
                dec     cl
                dec     cl
                jo      short loc_48AC31
                jl      short loc_48AC2F

loc_48AC2C:
                jmp     short loc_48AC33
```

```
loc_48AC2F:
                jz      short loc_48AC2C

loc_48AC31:
                jmp     short loc_48AC2C

loc_48AC33:
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 10h
                sub     cl, 1
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                and     al, cl
                mov     eax, eax
                pop     ecx
                pop     ebx
                test    eax, eax
                jnz     loc_48AD70
                mov     eax, [ebp-4]
                jo      short loc_48AC78
                jl      short loc_48AC76

loc_48AC73:

                jmp     short loc_48AC7A

loc_48AC76:
                jz      short loc_48AC73

loc_48AC78:
                jmp     short loc_48AC73

loc_48AC7A:
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                jo      short loc_48ACA9
                jl      short loc_48ACA7

loc_48ACA4:

                jmp     short loc_48ACAB

loc_48ACA7:
                jz      short loc_48ACA4

loc_48ACA9:
                jmp     short loc_48ACA4

loc_48ACAB:
                mov     bl, 0C6h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_48ACF8
                jl      short loc_48ACF6

loc_48ACF1:

                jmp     short loc_48ACFA

loc_48ACF6:
                jz      short loc_48ACF1

loc_48ACF8:
                jmp     short loc_48ACF1

loc_48ACFA:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                jo      short loc_48AD0D
                jl      short loc_48AD0B

loc_48AD08:
                jmp     short loc_48AD0F

loc_48AD0B:
                jz      short loc_48AD08

loc_48AD0D:
                jmp     short loc_48AD08

loc_48AD0F:
                and     eax, 40h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1200h
                dec     dh
                sub     dh, 1
                dec     dh
                sub     dh, 7
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_48AD43
                jl      short loc_48AD41

loc_48AD3C:
                jmp     short loc_48AD45

loc_48AD41:
                jz      short loc_48AD3C

loc_48AD43:
                jmp     short loc_48AD3C

loc_48AD45:
                inc     eax
                dec     eax
                jo      short loc_48AD52
                jl      short loc_48AD50

loc_48AD4B:

                jmp     short loc_48AD54

loc_48AD50:
                jz      short loc_48AD4B

loc_48AD52:
                jmp     short loc_48AD4B

loc_48AD54:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_48AD63
                jl      short loc_48AD61

loc_48AD5C:
                jmp     short loc_48AD65

loc_48AD61:
                jz      short loc_48AD5C

loc_48AD63:
                jmp     short loc_48AD5C
```

```
 loc_48AD65:
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_48AD70
                and     eax, 0
                jmp     short loc_48AD74
 loc_48AD70:

                and     eax, 0
                inc     eax

 loc_48AD74:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48AD97
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48AD97:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_489EBA(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCDC_2
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1F00h
                pop     ebx
                jo      short loc_489EF0
                jl      short loc_489EEE

loc_489EE9:
                jmp     short loc_489EF2

loc_489EEE:
                jz      short loc_489EE9

loc_489EF0:
                jmp     short loc_489EE9

loc_489EF2:
                sub     bh, 5
                dec     bh
                push    eax
                dec     bh
                dec     bh
                and     eax, 41h
                dec     bh
                sub     bh, 12h
                sub     bh, 3
                pop     eax
                dec     bh
                and     ah, bh
                mov     edx, 15h
                dec     dl
                sub     dl, 3
                dec     dl
                sub     dl, 7
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                and     al, dl
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_489F3A
                jl      short loc_489F38

loc_489F33:
                jmp     short loc_489F3C

loc_489F38:
                jz      short loc_489F33

loc_489F3A:
                jmp     short loc_489F33

loc_489F3C:
                inc     eax
                dec     eax
                jo      short loc_489F49
                jl      short loc_489F47

loc_489F42:
                jmp     short loc_489F4B

loc_489F47:
                jz      short loc_489F42

loc_489F49:
                jmp     short loc_489F42

loc_489F4B:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_489F58
                jl      short loc_489F56

loc_489F53:
                jmp     short loc_489F5A

loc_489F56:
                jz      short loc_489F53

loc_489F58:
                jmp     short loc_489F53

loc_489F5A:
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9370
                xor     ecx, ds:dword_4D9374
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_489F7E
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_489F7E:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDC9C
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_486D34(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD08
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
```

```
and     eax, 800h
bswap   ecx
pop     eax
bswap   ecx
and     ah, bh
mov     bl, 86h
sub     bl, 5
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
sub     bl, 1Ah
dec     bl
sub     bl, 1Fh
not     bx
bswap   eax
not     bx
bswap   eax
and     al, bl
mov     eax, eax
test    eax, eax
jnz     loc_486E57
pop     ebx
pop     edx
mov     eax, [ebp-4]
push    edx
mov     edx, 0FFFFh
and     eax, edx
push    ebx
push    eax
mov     bh, 7
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
and     eax, 800h
bswap   ecx
pop     eax
bswap   ecx
and     ah, bh
mov     bl, 98h
sub     bl, 5
dec     bl
dec     bl
dec     bl
dec     bl
```

```
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 0Ch
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    0Dh
                pop     ebx
                jo      short loc_486E1A
                jl      short loc_486E18

loc_486E13:
                jmp     short loc_486E1C

loc_486E18:
                jz      short loc_486E13

loc_486E1A:
                jmp     short loc_486E13

loc_486E1C:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                and     eax, 41h
                dec     bl
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 2500h
                dec     dh
                sub     dh, 3
                dec     dh
                sub     dh, 17h
                dec     dh
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_486E57
                and     eax, 0
                jmp     short loc_486E5B
 loc_486E57:
                and     eax, 0
                inc     eax

 loc_486E5B:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D939C
                xor     ecx, ds:dword_4D93A0
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_486E7E
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_486E7E:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_48AEC3(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF0
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     dh, 2
                dec     dh
                dec     dh
                and     ah, dh
                mov     dl, 0Eh
                sub     dl, 0FFh
                jo      short loc_48AEF6
                jl      short loc_48AEF4

loc_48AEF1:
                jmp     short loc_48AEF8

loc_48AEF4:
                jz      short loc_48AEF1

loc_48AEF6:
                jmp     short loc_48AEF1

loc_48AEF8:
                sub     dl, 0FFh
                sub     dl, 0FFh
                sub     dl, 0Ah
                sub     dl, 0FFh
                sub     dl, 0FFh
                sub     dl, 5
                dec     dl
                dec     dl
                dec     dl
                sub     dl, 3
                sub     dl, 0FFh
                dec     dl
                inc     dl
                inc     dl
                inc     dl
                and     al, dl
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9384
                xor     ecx, ds:dword_4D9388
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48AF49
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48AF49:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_4850F1(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF0
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     dh, 2
                dec     dh
                dec     dh
                and     ah, dh
                mov     dl, 0Eh
                sub     dl, 0FFh
                jo      short loc_485124
                jl      short loc_485122
```

```
loc_48511F:
                jmp     short loc_485126

loc_485122:
                jz      short loc_48511F

loc_485124:
                jmp     short loc_48511F

loc_485126:
                sub     dl, 0FEh
                dec     dl
                sub     dl, 0FFh
                sub     dl, 0Ah
                sub     dl, 0FFh
                sub     dl, 0FFh
                sub     dl, 5
                dec     dl
                jo      short loc_485145
                jl      short loc_485143

loc_485140:
                jmp     short loc_485147

loc_485143:
                jz      short loc_485140

loc_485145:
                jmp     short loc_485140

loc_485147:
                dec     dl
                dec     dl
                sub     dl, 3
                sub     dl, 0FFh
                dec     dl
                inc     dl
                inc     dl
                inc     dl
                jo      short loc_485162
                jl      short loc_485160

loc_48515D:
                jmp     short loc_485164

loc_485160:
                jz      short loc_48515D

loc_485162:
                jmp     short loc_48515D
```

```
 loc_485164:
                inc     dl
                dec     dl
                inc     dl
                dec     dl
                dec     dl
                inc     dl
                dec     dl
                inc     dl
                inc     dl
                inc     dl
                dec     dl
                inc     dl
                dec     dl
                inc     dl
                inc     dl
                dec     dl
                dec     dl
                dec     dl
                and     al, dl
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9384
                xor     ecx, ds:dword_4D9388
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4851AE
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_4851AE:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_4821A5(void)
{
     __asm
```

```
{
            push    ebp
            mov     ebp, esp
            sub     esp, 0Ch
            push    ebx
            push    esi
            push    edi
            mov     eax, [ebp+8]
            push    eax
            call    ds:off_4DDCFC
            add     esp, 4
            mov     [ebp-4], eax
            mov     eax, [ebp-4]
            push    ebx
            mov     ebx, 0FFFFh
            and     eax, ebx
            push    ecx
            mov     ch, 2Ch
            sub     ch, 1
            sub     ch, 20h
            dec     ch
            dec     ch
            sub     ch, 4
            dec     ch
            sub     ch, 3
            dec     ch
            and     ah, ch
            mov     cl, 0AEh
            sub     cl, 2
            dec     cl
            dec     cl
            sub     cl, 6
            not     al
            bswap   ecx
            not     al
            bswap   ecx
            dec     cl
            dec     cl
            sub     cl, 10h
            dec     cl
            dec     cl
            add     cl, 0Ch
            dec     cl
            dec     cl
            dec     cl
            dec     cl
            dec     cl
            dec     cl
            sub     cl, 10h
            sub     cl, 1
            dec     cl
            dec     cl
```

```
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ebx
                push    eax
                mov     eax, [ebp-4]
                mov     edx, 200h
                inc     dh
                inc     dh
                dec     dh
                inc     dh
                inc     dh
                inc     dh
                inc     dh
                inc     dh
                and     eax, edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                cmp     eax, edx
                jo      short loc_48226E
                jl      short loc_48226C

loc_482269:
                jmp     short loc_482270

loc_48226C:
                jz      short loc_482269

loc_48226E:
                jmp     short loc_482269

loc_482270:
                jnz     short loc_482277
                and     eax, 0
                jmp     short loc_48227B

 loc_482277:
                and     eax, 0
                inc     eax

 loc_48227B:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9390
                xor     ecx, ds:dword_4D9394
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48229E
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48229E:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCBC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_487EE3(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD04
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1E00h
                pop     ebx
                jo      short loc_487F19
                jl      short loc_487F17

loc_487F12:
                jmp     short loc_487F1B

loc_487F17:
                jz      short loc_487F12

loc_487F19:
                jmp     short loc_487F12

loc_487F1B:
                sub     bh, 2
                sub     bh, 3
                push    eax
                dec     bh
                dec     bh
                jo      short loc_487F2F
                jl      short loc_487F2D

loc_487F2A:
                jmp     short loc_487F31

loc_487F2D:
                jz      short loc_487F2A

loc_487F2F:
                jmp     short loc_487F2A

loc_487F31:
                and     eax, 800h
                dec     bh
                sub     bh, 14h
                sub     bh, 2
                pop     eax
                dec     bh
                inc     bh
                and     ah, bh
                mov     edx, 13h
                dec     dl
                dec     dl
                sub     dl, 1
                dec     dl
                sub     dl, 9
                dec     dl
                dec     dl
                and     al, dl
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_487F87
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_487F87:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_4896E4(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD00
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                push    800h
                pop     ecx
                dec     ch
                dec     ch
                sub     ch, 3
                sub     ch, 1
                dec     ch
                dec     ch
                and     ah, ch
                mov     cl, 14h
                dec     cl
                dec     cl
                sub     cl, 2
                dec     cl
                dec     dl
                sub     cl, 1
                dec     cl
                dec     cl
                dec     dl
                dec     cl
                dec     dl
                dec     cl
                sub     cl, 3
                dec     cl
                dec     dl
                sub     cl, 1
                dec     cl
                and     al, cl
                pop     ecx
                pop     ebx
                test    eax, eax
                jz      short loc_489757
                not     eax
                add     eax, 1
                stc
                jmp     short loc_48975D
loc_489757:
                not     eax
                add     eax, 1
                clc

loc_48975D:
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9394
                xor     ecx, ds:dword_4D9398
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_489784
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_489784:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_48BAC0(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF0
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 8
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 41h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 87h
                sub     bl, 4
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9384
                xor     ecx, ds:dword_4D9388
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48BB57
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48BB57:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
```

```
    }
}




__declspec(naked) void sub_484F0B(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD10
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 80h
                jmp     short loc_484F34
                mov     ebx, 4

 loc_484F34:
                mov     ebx, 32h
                not     ebx
                bswap   eax
                not     ebx
                inc     ebx
                inc     ebx
                inc     ebx
                add     ebx, 8
                dec     ebx
                push    ecx
                mov     ecx, 4
                add     ebx, ecx
                inc     ebx
                pop     ecx
                bswap   eax
                and     eax, ebx
                pop     ebx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A4
                xor     ecx, ds:dword_4D93A8
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_484F7F
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_484F7F:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48D8AF(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCFC
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_48D8E2
                jl      short loc_48D8E0

 loc_48D8DB:
                jmp     short loc_48D8E4
```

```
loc_48D8E0:
                jz      short loc_48D8DB

loc_48D8E2:
                jmp     short loc_48D8DB

loc_48D8E4:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                jo      short loc_48D8F9
                jl      short loc_48D8F7

loc_48D8F2:
                jmp     short loc_48D8FB

loc_48D8F7:
                jz      short loc_48D8F2

loc_48D8F9:
                jmp     short loc_48D8F2

loc_48D8FB:
                and     eax, 40h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1200h
                dec     dh
                sub     dh, 1
                dec     dh
                sub     dh, 7
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_48D92F
                jl      short loc_48D92D

loc_48D928:
                jmp     short loc_48D931

loc_48D92D:
```

```
                jz      short loc_48D928

loc_48D92F:
                jmp     short loc_48D928

loc_48D931:
                inc     eax
                dec     eax
                jo      short loc_48D93E
                jl      short loc_48D93C

loc_48D937:
                jmp     short loc_48D940

loc_48D93C:
                jz      short loc_48D937

loc_48D93E:
                jmp     short loc_48D937

loc_48D940:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_48D94D
                jl      short loc_48D94B

loc_48D948:
                jmp     short loc_48D94F

loc_48D94B:
                jz      short loc_48D948

loc_48D94D:
                jmp     short loc_48D948

loc_48D94F:
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9390
                xor     ecx, ds:dword_4D9394
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48D973
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48D973:
                mov     eax, [ebp-8]
```

```
                        push    eax
                        call    ds:off_4DDCBC
                        add     esp, 4
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
        }
}





__declspec(naked) void sub_487F9B(void)
{
        __asm
        {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDCF0
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    edx
                        mov     edx, 0FFFFh
                        and     eax, edx
                        push    ebx
                        push    0C00h
                        pop     ebx
                        jo      short loc_487FD1
                        jl      short loc_487FCF

 loc_487FCA:
                        jmp     short loc_487FD3

 loc_487FCF:
                        jz      short loc_487FCA

 loc_487FD1:
                        jmp     short loc_487FCA

 loc_487FD3:
                        dec     bh
```

```
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                push    eax
                dec     bh
                dec     bh
                and     eax, 41h
                dec     bh
                sub     bh, 3
                pop     eax
                dec     bh
                and     ah, bh
                mov     edx, 25h
                dec     dl
                sub     dl, 3
                dec     dl
                sub     dl, 17h
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                jo      short loc_488013
                jl      short loc_488011

loc_48800C:
                jmp     short loc_488015

loc_488011:
                jz      short loc_48800C

loc_488013:
                jmp     short loc_48800C

loc_488015:
                and     al, dl
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9384
                xor     ecx, ds:dword_4D9388
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_488041
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx
```

```
 loc_488041:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_48B85F(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 8
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                bswap   eax
                dec     bh
                bswap   eax
                and     eax, 800h
                jo      short loc_48B87F
                jl      short loc_48B87D

 loc_48B87A:
                jmp     short loc_48B881

 loc_48B87D:
                jz      short loc_48B87A

 loc_48B87F:
                jmp     short loc_48B87A

 loc_48B881:
                mov     ebx, 4
                and     eax, 10h
                dec     ch
                mov     ebx, [ebp+0Ch]
                xor     ecx, ecx
                or      ebx, ecx
                jz      short loc_48B89F
                dec     edi
                sub     ch, 2
                dec     ch
                and     eax, 0
                jmp     short loc_48B8B5
loc_48B89F:
                dec     edi
                jo      short loc_48B8A9
                jl      short loc_48B8A7

loc_48B8A4:
                jmp     short loc_48B8AB

loc_48B8A7:
                jz      short loc_48B8A4

loc_48B8A9:
                jmp     short loc_48B8A4

loc_48B8AB:
                and     eax, 0
                dec     ecx
                sub     ch, 2
                inc     eax
                dec     ch

loc_48B8B5:
                mov     [ebp-8], eax
                mov     eax, ds:dword_4D93AC
                xor     eax, ds:dword_4D93B0
                shl     eax, 1
                mov     [ebp-4], eax
                cmp     dword ptr [ebp-8], 0
                jz      short loc_48B8D7
                mov     ecx, [ebp-4]
                or      ecx, 1
                mov     [ebp-4], ecx

loc_48B8D7:
                mov     edx, [ebp-4]
                push    edx
                call    ds:off_4DDCD8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}
```

```
__declspec(naked) void sub_485051(void)
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 sub     esp, 0Ch
                 push    ebx
                 push    esi
                 push    edi
                 mov     eax, [ebp+8]
                 push    eax
                 call    ds:off_4DDCE0
                 add     esp, 4
                 mov     [ebp-4], eax
                 mov     eax, [ebp-4]
                 push    ebx
                 mov     ebx, [ebp+0Ch]
                 mov     ebx, 0FFFFh
                 and     eax, ebx
                 push    ecx
                 push    40h
                 pop     ecx
                 xor     ecx, 40h
                 and     al, cl
                 mov     bh, 0Fh
                 and     bl, 0
                 dec     bh
                 sub     bh, 3
                 dec     bh
                 sub     bh, 1
                 dec     bh
                 jo      short loc_48509D
                 jl      short loc_48509B

 loc_485096:
                 jmp     short loc_48509F

 loc_48509B:
                 jz      short loc_485096

 loc_48509D:
                 jmp     short loc_485096

 loc_48509F:
                 and     ah, bh
                 pop     ecx
                 pop     ebx
```

```
                test    eax, eax
                jz      short loc_4850AF
                not     eax
                add     eax, 1
                stc
                jmp     short loc_4850B5

loc_4850AF:
                not     eax
                add     eax, 1
                clc

loc_4850B5:
                sbb     eax, eax
                add     eax, 1
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9374
                xor     ecx, ds:dword_4D9378
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4850DD
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_4850DD:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_47F7E9(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
```

```
mov     eax, [ebp+8]
push    eax
call    ds:off_4DDD0C
add     esp, 4
mov     [ebp-4], eax
mov     eax, [ebp-4]
push    ebx
mov     ebx, 0FFFFh
and     eax, ebx
push    ecx
mov     ch, 2Dh
dec     ch
sub     ch, 1
sub     ch, 20h
dec     ch
dec     ch
sub     ch, 7
dec     ch
dec     ch
and     ah, ch
mov     cl, 0BCh
sub     cl, 2
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
inc     cl
dec     cl
dec     cl
inc     cl
not     cl
bswap   edx
not     cl
bswap   edx
dec     cl
dec     cl
dec     cl
dec     cl
push    eax
dec     cl
dec     cl
sub     cl, 12h
sub     cl, 5
dec     cl
and     eax, 40h
dec     cl
dec     cl
dec     cl
add     cl, 0Eh
dec     cl
```

```
                dec     cl
                and     eax, 80h
                sub     cl, 1Fh
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                pop     eax
                inc     cl
                inc     cl
                inc     cl
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ebx
                push    eax
                mov     eax, [ebp-4]
                mov     edx, 0C00h
                sub     dh, 1
                dec     dh
                dec     dh
                dec     dh
                and     eax, edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                cmp     eax, edx
                setz    cl
                mov     al, cl
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A0
                xor     ecx, ds:dword_4D93A4
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_47F8DB
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_47F8DB:
                mov     eax, [ebp-8]
                push    eax
```

```
                    call    ds:off_4DDCCC
                    add     esp, 4
                    pop     edi
                    pop     esi
                    pop     ebx
                    mov     esp, ebp
                    pop     ebp
                    retn
        }
}




__declspec(naked) void sub_485593(void)
{
        __asm
        {
                    push    ebp
                    mov     ebp, esp
                    sub     esp, 0Ch
                    push    ebx
                    push    esi
                    push    edi
                    mov     eax, [ebp+8]
                    push    eax
                    call    ds:off_4DDD04
                    add     esp, 4
                    mov     [ebp-4], eax
                    mov     eax, [ebp-4]
                    push    edx
                    mov     edx, 0FFFFh
                    and     eax, edx
                    push    ebx
                    push    3
                    pop     ebx
                    dec     bl
                    dec     bl
                    dec     bl
                    and     al, bl
                    mov     dh, 0Fh
                    and     dl, 0
                    sub     dh, 5
                    dec     dh
                    sub     dh, 1
                    and     ah, dh
                    pop     ebx
                    pop     edx
                    test    eax, eax
                    jz      short loc_4855ED
                    not     eax
```

```
                add     eax, 1
                stc
                jmp     short loc_4855F3
                jo      short loc_4855EB
                jl      short loc_4855E9

loc_4855E4:
                jmp     short loc_4855ED

loc_4855E9:
                jz      short loc_4855E4

loc_4855EB:
                jmp     short loc_4855E4

loc_4855ED:
                not     eax
                add     eax, 1
                clc

loc_4855F3:
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_485602
                jl      short loc_485600

loc_4855FB:
                jmp     short loc_485604

loc_485600:
                jz      short loc_4855FB

loc_485602:
                jmp     short loc_4855FB

loc_485604:
                inc     eax
                dec     eax
                jo      short loc_485611
                jl      short loc_48560F

loc_48560A:
                jmp     short loc_485613

loc_48560F:
                jz      short loc_48560A

loc_485611:
                jmp     short loc_48560A

loc_485613:
```

```
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_485622
                jl      short loc_485620

 loc_48561B:
                jmp     short loc_485624

 loc_485620:
                jz      short loc_48561B

 loc_485622:
                jmp     short loc_48561B


 loc_485624:
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_485648
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_485648:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48468F(void)
{
    __asm
    {
                push    ebp
```

```
mov     ebp, esp
sub     esp, 0Ch
push    ebx
push    esi
push    edi
mov     eax, [ebp+8]
push    eax
call    ds:off_4DDD00
add     esp, 4
mov     [ebp-4], eax
mov     eax, [ebp-4]
push    ebx
mov     ebx, 0FFFFh
and     eax, ebx
push    ecx
mov     ch, 2Dh
dec     ch
sub     ch, 1
sub     ch, 20h
dec     ch
dec     ch
sub     ch, 7
dec     ch
dec     ch
and     ah, ch
mov     cl, 77h
sub     cl, 2
dec     cl
dec     cl
dec     cl
not     cl
bswap   edx
not     cl
bswap   edx
dec     cl
dec     cl
push    eax
dec     cl
dec     cl
sub     cl, 14h
and     eax, 80h
dec     cl
dec     cl
dec     cl
add     cl, 0Eh
dec     cl
dec     cl
and     eax, 800h
sub     cl, 21h
dec     cl
not     ecx
bswap   eax
```

```
                not     ecx
                bswap   eax
                pop     eax
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9394
                xor     ecx, ds:dword_4D9398
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48473E
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48473E:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_4847E1(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCEC
                add     esp, 4
```

```
mov     [ebp-4], eax
mov     eax, [ebp-4]
push    edx
mov     edx, 0FFFFh
and     eax, edx
push    ebx
push    eax
mov     bh, 1Ch
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
and     eax, 800h
bswap   ecx
pop     eax
bswap   ecx
and     ah, bh
mov     bl, 8Bh
sub     bl, 5
dec     bl
dec     bl
dec     bl
dec     bl
sub     bl, 4
dec     bl
dec     bl
dec     bl
sub     bl, 1Ah
```

```
                        dec     bl
                        dec     bl
                        sub     bl, 1Fh
                        not     bx
                        bswap   eax
                        not     bx
                        bswap   eax
                        and     al, bl
                        mov     eax, eax
                        pop     ebx
                        neg     eax
                        sbb     eax, eax
                        neg     eax
                        pop     edx
                        mov     [ebp-0Ch], eax
                        mov     ecx, ds:dword_4D9380
                        xor     ecx, ds:dword_4D9384
                        shl     ecx, 1
                        mov     [ebp-8], ecx
                        cmp     dword ptr [ebp-0Ch], 0
                        jz      short loc_4848A6
                        mov     edx, [ebp-8]
                        or      edx, 1
                        mov     [ebp-8], edx

 loc_4848A6:
                        mov     eax, [ebp-8]
                        push    eax
                        call    ds:off_4DDCAC
                        add     esp, 4
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
    }
}




__declspec(naked) void sub_487692(void)
{
    __asm
    {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
```

```
mov     eax, [ebp+8]
push    eax
call    ds:off_4DDCE4
add     esp, 4
mov     [ebp-4], eax
mov     eax, [ebp-4]
push    ebx
mov     ebx, [ebp+0Ch]
mov     ebx, 0FFFFh
and     eax, ebx
push    ecx
mov     ch, 2Dh
dec     ch
sub     ch, 1
sub     ch, 20h
dec     ch
dec     ch
sub     ch, 7
dec     ch
dec     ch
and     ah, ch
mov     cl, 0BDh
sub     cl, 2
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
not     cl
bswap   edx
not     cl
bswap   edx
sub     cl, 4
push    eax
dec     cl
dec     cl
sub     cl, 17h
dec     cl
and     eax, 40h
dec     cl
dec     cl
dec     cl
add     cl, 0Eh
dec     cl
dec     cl
and     eax, 80h
sub     cl, 1Fh
dec     cl
dec     cl
dec     cl
```

```
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                pop     eax
                inc     cl
                inc     cl
                inc     cl
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ebx
                push    eax
                mov     eax, [ebp-4]
                mov     edx, 0C00h
                sub     dh, 1
                dec     dh
                dec     dh
                dec     dh
                and     eax, edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                cmp     eax, edx
                setz    cl
                mov     al, cl
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9378
                xor     ecx, ds:dword_4D937C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_487779
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_487779:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
```

```
                    pop     ebp
                    retn
          }
     }




     __declspec(naked) void sub_482781(void)
     {
          __asm
          {
                    push    ebp
                    mov     ebp, esp
                    sub     esp, 0Ch
                    push    ebx
                    push    esi
                    push    edi
                    mov     eax, [ebp+8]
                    push    eax
                    call    ds:off_4DDD14
                    add     esp, 4
                    mov     [ebp-4], eax
                    mov     eax, [ebp-4]
                    push    edx
                    mov     edx, 0FFFFh
                    and     eax, edx
                    push    ebx
                    push    eax
                    mov     bh, 7
                    xor     bh, 7
                    inc     bh
                    dec     bh
                    inc     bh
                    dec     bh
                    inc     bh
                    dec     bh
                    and     eax, 800h
                    bswap   ecx
                    pop     eax
                    bswap   ecx
                    and     ah, bh
                    mov     bl, 86h
                    dec     bl
                    dec     bl
                    dec     bl
                    dec     bl
                    dec     bl
                    dec     bl
                    dec     bl
                    dec     bl
                    dec     bl
```

```
dec     bl
inc     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
sub     bl, 1Fh
not     bx
bswap   eax
not     bx
bswap   eax
and     al, bl
mov     eax, eax
pop     ebx
neg     eax
sbb     eax, eax
inc     eax
pop     edx
mov     [ebp-0Ch], eax
mov     ecx, ds:dword_4D93A8
xor     ecx, ds:dword_4D93AC
shl     ecx, 1
mov     [ebp-8], ecx
cmp     dword ptr [ebp-0Ch], 0
jz      short loc_482853
mov     edx, [ebp-8]
```

```
                or      edx, 1
                mov     [ebp-8], edx

 loc_482853:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_485315(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF4
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                mov     ecx, 800h
                mov     ecx, 4Bh
                not     ecx
                bswap   eax
                not     ecx
                xor     ecx, 19h
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 12h
                add     ecx, 0Ah
                dec     ecx
                push    edx
                mov     edx, 4
                add     ecx, edx
                inc     ecx
                pop     edx
                bswap   eax
                add     ecx, 3
                and     eax, ecx
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                push    eax
                mov     eax, [ebp-4]
                mov     edx, 0E00h
                sub     dh, 1
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                and     eax, edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                cmp     eax, edx
                setz    cl
                mov     al, cl
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9388
                xor     ecx, ds:dword_4D938C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4853C0
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_4853C0:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
        }
}



__declspec(naked) void sub_486128(void)
{
        __asm
        {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCFC
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    410h
                pop     ebx
                dec     bh
                dec     bh
                sub     bh, 0FFh
                sub     bh, 2
                dec     bh
                and     ah, bh
                mov     bl, 0Eh
                sub     bl, 4
                dec     bl
                sub     bl, 1
                sub     bl, 1
                sub     bl, 1
                sub     bl, 1
                sub     bl, 1
                and     al, bl
                pop     ebx
                pop     edx
                test    eax, eax
                jz      short loc_486187
                not     eax
                add     eax, 1
                stc
                jmp     short loc_48618D

 loc_486187:
                not     eax
                add     eax, 1
                clc

 loc_48618D:
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9390
                xor     ecx, ds:dword_4D9394
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4861B4
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_4861B4:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCBC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_48B379(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF0
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                push    800h
                pop     ecx
                dec     ch
                dec     ch
                sub     ch, 4
                dec     ch
                inc     esi
                dec     ch
                and     ah, ch
                mov     cl, 0Fh
                dec     dl
                sub     cl, 3
                dec     cl
                sub     cl, 1
                dec     cl
                dec     edi
                dec     cl
                dec     cl
                inc     esi
                dec     cl
                sub     cl, 1
                dec     cl
                and     al, cl
                pop     ecx
                pop     ebx
                test    eax, eax
                jz      short loc_48B3DD
                not     eax
                add     eax, 1
                stc
                jmp     short loc_48B3E3

loc_48B3DD:
                not     eax
                add     eax, 1
                clc

loc_48B3E3:
                sbb     eax, eax
                neg     eax
```

```
                    mov     [ebp-0Ch], eax
                    mov     ecx, ds:dword_4D9384
                    xor     ecx, ds:dword_4D9388
                    shl     ecx, 1
                    mov     [ebp-8], ecx
                    cmp     dword ptr [ebp-0Ch], 0
                    jz      short loc_48B40A
                    mov     edx, [ebp-8]
                    or      edx, 1
                    mov     [ebp-8], edx

 loc_48B40A:
                    mov     eax, [ebp-8]
                    push    eax
                    call    ds:off_4DDCB0
                    add     esp, 4
                    pop     edi
                    pop     esi
                    pop     ebx
                    mov     esp, ebp
                    pop     ebp
                    retn
    }
}




__declspec(naked) void sub_4838D4(void)
{
    __asm
    {
                    push    ebp
                    mov     ebp, esp
                    sub     esp, 0Ch
                    push    ebx
                    push    esi
                    push    edi
                    mov     eax, [ebp+8]
                    push    eax
                    call    ds:off_4DDCFC
                    add     esp, 4
                    mov     [ebp-4], eax
                    mov     eax, [ebp-4]
                    push    ecx
                    mov     ecx, 800h
                    mov     ecx, 0Dh
                    not     ecx
                    bswap   eax
                    not     ecx
                    inc     ecx
                    inc     ecx
```

```
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Dh
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Ah
                dec     ecx
                push    edx
                mov     edx, 4
                add     ecx, edx
                inc     ecx
                pop     edx
                bswap   eax
                and     eax, ecx
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9390
                xor     ecx, ds:dword_4D9394
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_483962
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_483962:
                mov     eax, [ebp-8]
```

```
                    push    eax
                    call    ds:off_4DDCBC
                    add     esp, 4
                    pop     edi
                    pop     esi
                    pop     ebx
                    mov     esp, ebp
                    pop     ebp
                    retn
     }
}




__declspec(naked) void sub_488E61(void)
{
     __asm
     {
                        push    ebp
                    mov     ebp, esp
                    sub     esp, 0Ch
                    push    ebx
                    push    esi
                    push    edi
                    mov     eax, [ebp+8]
                    push    eax
                    call    ds:off_4DDCF8
                    add     esp, 4
                    mov     [ebp-4], eax
                    mov     eax, [ebp-4]
                    push    edx
                    mov     edx, 0FFFFh
                    and     eax, edx
                    push    ebx
                    push    eax
                    mov     bh, 7
                    dec     bh
                    dec     bh
                    dec     bh
                    dec     bh
                    dec     bh
                    dec     bh
                    dec     bh
                    and     eax, 800h
                    bswap   ecx
                    pop     eax
                    bswap   ecx
                    and     ah, bh
                    mov     bl, 87h
                    sub     bl, 5
                    dec     bl
```

```
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D938C
                xor     ecx, ds:dword_4D9390
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_488EF6
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_488EF6:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_489A35(void)
{
    __asm
```

```
{
            push    ebp
            mov     ebp, esp
            sub     esp, 0Ch
            push    ebx
            push    esi
            push    edi
            mov     eax, [ebp+8]
            push    eax
            call    ds:off_4DDD08
            add     esp, 4
            mov     [ebp-4], eax
            mov     eax, [ebp-4]
            push    edx
            mov     edx, 0FFFFh
            and     eax, edx
            push    ebx
            push    eax
            mov     bh, 7
            dec     bh
            dec     bh
            dec     bh
            dec     bh
            dec     bh
            dec     bh
            dec     bh
            and     eax, 800h
            bswap   ecx
            pop     eax
            bswap   ecx
            and     ah, bh
            mov     bl, 87h
            sub     bl, 5
            dec     bl
            dec     bl
            dec     bl
            dec     bl
            dec     bl
            dec     bl
            dec     bl
            sub     bl, 1Ah
            dec     bl
            dec     bl
            sub     bl, 1Fh
            not     bx
            bswap   eax
            not     bx
            bswap   eax
            and     al, bl
            mov     eax, eax
            pop     ebx
            neg     eax
```

```
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D939C
                xor     ecx, ds:dword_4D93A0
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_489ACC
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_489ACC:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_48A589(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD00
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 80h
                jmp     short loc_48A5B2
                mov     ebx, 4
 loc_48A5B2:
                mov     ebx, 32h
                not     ebx
                bswap   eax
                not     ebx
                inc     ebx
                inc     ebx
                inc     ebx
                add     ebx, 8
                dec     ebx
                push    ecx
                mov     ecx, 4
                add     ebx, ecx
                inc     ebx
                pop     ecx
                bswap   eax
                and     eax, ebx
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9394
                xor     ecx, ds:dword_4D9398
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48A5FB
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48A5FB:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_483784(void)
{
     __asm
```

```
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 8
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 10h
                dec     ch
                dec     ch
                sub     ch, 4
                dec     ch
                sub     ch, 13h
                dec     ch
                mov     ebx, [ebp+0Ch]
                dec     ah
                and     cl, 0
                dec     ah
                xor     edx, edx
                or      ebx, edx
                jz      short loc_4837C5
                dec     edi
                and     eax, 0
                jmp     short loc_4837C9

loc_4837C5:
                and     eax, 0
                inc     eax

loc_4837C9:
                mov     [ebp-8], eax
                mov     eax, ds:dword_4D9384
                xor     eax, ds:dword_4D9388
                shl     eax, 1
                mov     [ebp-4], eax
                cmp     dword ptr [ebp-8], 0
                jz      short loc_4837EB
                mov     ecx, [ebp-4]
                or      ecx, 1
                mov     [ebp-4], ecx

loc_4837EB:
                mov     edx, [ebp-4]
                push    edx
                call    ds:off_4DDCB0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48B048(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD04
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 4
                dec     ch
                sub     ch, 3
                dec     ch
                and     ah, ch
                mov     cl, 0AFh
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 6
                not     al
                bswap   ecx
```

```
not     al
bswap   ecx
dec     cl
dec     cl
sub     cl, 10h
dec     cl
dec     cl
add     cl, 0Ch
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
sub     cl, 10h
sub     cl, 1
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
not     ecx
bswap   eax
not     ecx
bswap   eax
inc     cl
add     cl, 2
and     al, cl
mov     eax, eax
pop     ecx
neg     eax
sbb     eax, eax
inc     eax
pop     ebx
push    eax
mov     eax, [ebp-4]
mov     edx, 300h
inc     dh
inc     dh
dec     dh
inc     dh
inc     dh
inc     dh
inc     dh
and     eax, edx
neg     eax
sbb     eax, eax
inc     eax
mov     edx, eax
```

```
                pop     eax
                xor     ecx, ecx
                cmp     eax, edx
                setz    cl
                mov     al, cl
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48B130
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48B130:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48830C(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE8
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                jo      short loc_488331
                jl      short loc_48832F
```

```
loc_48832C:
                jmp     short loc_488333

loc_48832F:
                jz      short loc_48832C

loc_488331:
                jmp     short loc_48832C

loc_488333:
                push    edx
                mov     dh, 2
                jo      short loc_48833F
                jl      short loc_48833D

loc_48833A:
                jmp     short loc_488341

loc_48833D:
                jz      short loc_48833A

loc_48833F:
                jmp     short loc_48833A

loc_488341:
                dec     dh
                dec     dh
                and     ah, dh
                mov     dl, 3
                sub     dl, 2
                inc     dl
                dec     dl
                inc     dl
                dec     dl
                inc     dl
                dec     dl
                inc     dl
                dec     dl
                and     al, dl
                not     ah
                not     ah
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D937C
                xor     ecx, ds:dword_4D9380
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48838B
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48838B:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48D6F8(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCDC_2
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 4
                dec     ch
                sub     ch, 3
                dec     ch
                and     ah, ch
                mov     cl, 0AEh
```

```
sub     cl, 0Ah
not     al
bswap   ecx
not     al
bswap   ecx
dec     cl
dec     cl
sub     cl, 10h
dec     cl
dec     cl
add     cl, 0Ch
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
sub     cl, 10h
sub     cl, 1
dec     cl
dec     cl
dec     cl
inc     cl
dec     cl
dec     cl
dec     cl
dec     cl
inc     cl
dec     cl
dec     cl
dec     cl
not     ecx
bswap   eax
not     ecx
bswap   eax
inc     cl
add     cl, 2
and     al, cl
mov     eax, eax
pop     ecx
neg     eax
sbb     eax, eax
inc     eax
pop     ebx
push    eax
mov     eax, [ebp-4]
mov     edx, 200h
inc     dh
inc     dh
dec     dh
inc     dh
inc     dh
```

```
                inc     dh
                inc     dh
                inc     dh
                and     eax, edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                cmp     eax, edx
                setz    cl
                mov     al, cl
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9370
                xor     ecx, ds:dword_4D9374
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48D7E1
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48D7E1:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDC9C
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48363D(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
```

```
push    eax
call    ds:off_4DDD14
add     esp, 4
mov     [ebp-4], eax
mov     eax, [ebp-4]
push    ecx
mov     ecx, 800h
mov     ecx, 0Ch
not     ecx
bswap   eax
not     ecx
inc     ecx
inc     ecx
inc     ecx
and     eax, 0
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
dec     ecx
inc     ecx
inc     cl
inc     cl
inc     cl
and     ecx, 40h
inc     cl
inc     cl
inc     cl
inc     cl
inc     cl
inc     cl
inc     cl
inc     eax
inc     cl
inc     cl
add     ecx, 0Ah
dec     ecx
push    edx
mov     edx, 4
add     ecx, edx
inc     ecx
pop     edx
pop     ecx
mov     [ebp-0Ch], eax
```

```
                mov     ecx, ds:dword_4D93A8
                xor     ecx, ds:dword_4D93AC
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4836CC
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_4836CC:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_4808ED(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD0C
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                sub     bl, 5
                dec     bl
                push    eax
                and     eax, ebx
                dec     bl
                dec     bl
                and     eax, 10h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1100h
                sub     dh, 1
                dec     dh
                sub     dh, 7
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A0
                xor     ecx, ds:dword_4D93A4
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48096B
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48096B:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCCC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_4837FF(void)
{
     __asm
```

```
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD0C
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Dh
                dec     ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 7
                dec     ch
                dec     ch
                and     ah, ch
                mov     cl, 77h
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     cl
                bswap   edx
                not     cl
                bswap   edx
                dec     cl
                dec     cl
                push    eax
                dec     cl
                dec     cl
                sub     cl, 12h
                dec     cl
                jo      short loc_483867
                jl      short loc_483865

loc_483862:
                jmp     short loc_483869
loc_483865:
                jz      short loc_483862
loc_483867:
                jmp     short loc_483862

loc_483869:
                dec     cl
                and     eax, 40h
                dec     cl
                dec     cl
                dec     cl
                add     cl, 0Eh
                dec     cl
                dec     cl
                and     eax, 800h
                sub     cl, 1Fh
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                pop     eax
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A0
                xor     ecx, ds:dword_4D93A4
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4838C0
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_4838C0:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCCC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
```

```
}




__declspec(naked) void sub_489F92(void)
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 sub     esp, 0Ch
                 push    ebx
                 push    esi
                 push    edi
                 mov     eax, [ebp+8]
                 push    eax
                 call    ds:off_4DDD10
                 add     esp, 4
                 mov     [ebp-4], eax
                 mov     eax, [ebp-4]
                 push    edx
                 mov     dh, 7
                 dec     dh
                 sub     dh, 2
                 and     dh, 0
                 and     ah, dh
                 mov     dl, 4
                 dec     dl
                 sub     dl, 2
                 inc     dl
                 dec     dl
                 inc     dl
                 dec     dl
                 inc     dl
                 dec     dl
                 inc     dl
                 inc     dl
                 dec     dl
                 dec     dl
                 sub     dl, 0FFh
                 dec     dl
                 and     al, dl
                 pop     edx
                 mov     [ebp-0Ch], eax
                 mov     ecx, ds:dword_4D93A4
                 xor     ecx, ds:dword_4D93A8
                 shl     ecx, 1
                 mov     [ebp-8], ecx
                 cmp     dword ptr [ebp-0Ch], 0
                 jz      short loc_48A001
                 mov     edx, [ebp-8]
```

```
                or      edx, 1
                mov     [ebp-8], edx

 loc_48A001:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48572F(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD04
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, [ebp+0Ch]
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                xor     bh, 3
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 98h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 0Ch
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4857BF
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_4857BF:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}
```

```
__declspec(naked) void sub_486088(void)
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 sub     esp, 0Ch
                 push    ebx
                 push    esi
                 push    edi
                 mov     eax, [ebp+8]
                 push    eax
                 call    ds:off_4DDD00
                 add     esp, 4
                 mov     [ebp-4], eax
                 mov     eax, [ebp-4]
                 push    edx
                 mov     edx, 0FFFFh
                 and     eax, edx
                 push    ebx
                 push    1Fh
                 pop     ebx
                 jo      short loc_4860BB
                 jl      short loc_4860B9

 loc_4860B4:
                 jmp     short loc_4860BD

 loc_4860B9:
                 jz      short loc_4860B4

 loc_4860BB:
                 jmp     short loc_4860B4

 loc_4860BD:
                 sub     bl, 5
                 dec     bl
                 push    eax
                 dec     bl
                 dec     bl
                 and     eax, 41h
                 dec     bl
                 sub     bl, 12h
                 sub     bl, 3
                 pop     eax
                 dec     bl
                 and     al, bl
                 mov     edx, 1500h
                 dec     dh
                 sub     dh, 3
                 dec     dh
                 sub     dh, 7
```

```
                dec     dh
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9394
                xor     ecx, ds:dword_4D9398
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_486114
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_486114:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




_declspec(naked) void sub_48C479(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCFC
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
```

```
push    ebx
mov     ebx, 0FFFFh
and     eax, ebx
push    ecx
mov     ch, 2Dh
dec     ch
sub     ch, 1
sub     ch, 20h
dec     ch
dec     ch
sub     ch, 7
dec     ch
dec     ch
and     ah, ch
mov     cl, 0BDh
sub     cl, 2
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
not     cl
bswap   edx
not     cl
bswap   edx
dec     cl
dec     cl
dec     cl
dec     cl
push    eax
dec     cl
dec     cl
sub     cl, 12h
dec     cl
dec     cl
sub     cl, 3
dec     cl
and     eax, 40h
dec     cl
dec     cl
dec     cl
add     cl, 0Eh
dec     cl
dec     cl
and     eax, 80h
sub     cl, 1Fh
dec     cl
dec     cl
dec     cl
not     ecx
```

```
                bswap   eax
                not     ecx
                bswap   eax
                pop     eax
                inc     cl
                inc     cl
                inc     cl
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9390
                xor     ecx, ds:dword_4D9394
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48C544
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48C544:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCBC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_484526(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD00
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                push    4
                pop     ecx
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                and     al, cl
                mov     bh, 0Fh
                and     bl, 0
                dec     bh
                sub     bh, 3
                dec     bh
                sub     bh, 1
                dec     bh
                and     ah, bh
                pop     ecx
                pop     ebx
                test    eax, eax
                jz      short loc_484579
                not     eax
                add     eax, 1
                stc
                jmp     short loc_48457F

loc_484579:
                not     eax
                add     eax, 1
                clc

loc_48457F:
                sbb     eax, eax
                add     eax, 1
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9394
                xor     ecx, ds:dword_4D9398
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4845A7
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx
```

```
 loc_4845A7:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}






__declspec(naked) void sub_47FD40(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF0
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
```

```
pop     eax
bswap   ecx
and     ah, bh
mov     bl, 98h
sub     bl, 5
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
sub     bl, 0Ch
not     bx
bswap   eax
not     bx
bswap   eax
and     al, bl
mov     eax, eax
pop     ebx
neg     eax
sbb     eax, eax
inc     eax
pop     edx
push    eax
mov     eax, [ebp-4]
mov     edx, 0F00h
sub     dh, 1
dec     dh
dec     dh
dec     dh
dec     dh
dec     dh
dec     dh
and     eax, edx
neg     eax
sbb     eax, eax
inc     eax
mov     edx, eax
pop     eax
xor     ecx, ecx
cmp     eax, edx
setz    cl
mov     al, cl
mov     [ebp-0Ch], eax
mov     ecx, ds:dword_4D9384
xor     ecx, ds:dword_4D9388
shl     ecx, 1
mov     [ebp-8], ecx
cmp     dword ptr [ebp-0Ch], 0
jz      short loc_47FDFB
mov     edx, [ebp-8]
```

```
                or      edx, 1
                mov     [ebp-8], edx

 loc_47FDFB:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_4851C2(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD10
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                bswap   ecx
                not     ecx
                push    eax
                not     eax
                mov     eax, 80h
                xchg    eax, ecx
                mov     ecx, 1
                xchg    eax, ecx
                not     eax
                pop     eax
                not     ecx
                pop     ecx
                push    edx
                mov     dh, 12h
                dec     dh
                dec     dh
                not     ecx
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                bswap   eax
                dec     dh
                dec     dh
                sub     dh, 5
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                bswap   eax
                and     ah, dh
                mov     dl, 9
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                not     ecx
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                add     dl, 1
                and     al, dl
                not     ah
                bswap   eax
                bswap   eax
                not     ah
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A4
                xor     ecx, ds:dword_4D93A8
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48526C
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48526C:
                mov     eax, [ebp-8]
                push    eax
```

```
                        call    ds:off_4DDCD0
                        add     esp, 4
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
        }
}




__declspec(naked) void sub_482511(void)
{
        __asm
        {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDD18
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    ebx
                        mov     ebx, 80h
                        jmp     short loc_48253A
                        mov     ebx, 4

 loc_48253A:
                        mov     ebx, 28h
                        not     ebx
                        bswap   eax
                        not     ebx
                        inc     ebx
                        inc     ebx
                        add     ebx, 0Ah
                        inc     ebx
                        add     ebx, 7
                        push    ecx
                        mov     ecx, 4
                        add     ebx, ecx
                        inc     ebx
                        pop     ecx
                        bswap   eax
```

```
                and     eax, ebx
                pop     ebx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93AC
                xor     ecx, ds:dword_4D93B0
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_482587
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_482587:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_4872DC(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD0C
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     dh, 2
                sub     dh, 0FFh
                dec     dh
                sub     dh, 0FFh
                dec     dh
                sub     dh, 0FFh
                sub     dh, 1
                sub     dh, 1
                dec     dh
                and     ah, dh
                mov     edx, 800h
                mov     dl, 0Fh
                sub     dl, 0FFh
                sub     dl, 0FFh
                sub     dl, 0FFh
                sub     dl, 0Ah
                sub     dl, 0FFh
                sub     dl, 0FFh
                sub     dl, 5
                dec     dl
                dec     dl
                dec     dl
                sub     dl, 3
                sub     dl, 0FFh
                dec     dl
                inc     dl
                inc     dl
                and     al, dl
                not     ah
                not     ah
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A0
                xor     ecx, ds:dword_4D93A4
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48736F
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48736F:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCCC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_486C52(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCFC
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                jo      short loc_486C77
                jl      short loc_486C75

 loc_486C72:
                jmp     short loc_486C79

 loc_486C75:
                jz      short loc_486C72

 loc_486C77:
                jmp     short loc_486C72

 loc_486C79:
                push    ebx
                mov     ebx, [ebp+0Ch]
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 4
                dec     ch
                sub     ch, 3
                dec     ch
                and     ah, ch
                mov     cl, 70h
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 6
                not     al
                bswap   ecx
                not     al
                and     eax, 0
                bswap   ecx
                dec     cl
                dec     cl
                sub     cl, 12h
                add     cl, 0Bh
                dec     cl
                dec     cl
                jo      short loc_486CCD
                jl      short loc_486CCB

loc_486CC8:
                jmp     short loc_486CCF

loc_486CCB:
                jz      short loc_486CC8

loc_486CCD:
                jmp     short loc_486CC8

loc_486CCF:
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 40h
                sub     cl, 1
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                inc     eax
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                pop     ecx
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9390
                xor     ecx, ds:dword_4D9394
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_486D20
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_486D20:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCBC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_489299(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCFC
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 4
                dec     bh
                dec     bh
                dec     bh
                xor     bh, 1
                and     eax, 80h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 86h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 10h
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                pop     ebx
                pop     edx
                test    eax, eax
                jnz     loc_4893DF
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 1
                dec     bh
                and     eax, 41h
                bswap   ecx
```

```
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 97h
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 0Ch
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_489373
                jl      short loc_489371

loc_48936C:

                jmp     short loc_489375

loc_489371:
                jz      short loc_48936C

loc_489373:
                jmp     short loc_48936C

loc_489375:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                and     eax, 40h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                jo      short loc_489398
                jl      short loc_489396

loc_489391:
                jmp     short loc_48939A

loc_489396:
                jz      short loc_489391

loc_489398:
                jmp     short loc_489391


loc_48939A:
                and     al, bl
                mov     edx, 1200h
                dec     dh
                sub     dh, 1
                dec     dh
                sub     dh, 7
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_4893C0
                jl      short loc_4893BE

loc_4893B9:
                jmp     short loc_4893C2

loc_4893BE:
                jz      short loc_4893B9

loc_4893C0:
                jmp     short loc_4893B9

loc_4893C2:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_4893D1
                jl      short loc_4893CF

loc_4893CC:
                jmp     short loc_4893D3
```

```
 loc_4893CF:
                jz      short loc_4893CC

 loc_4893D1:
                jmp     short loc_4893CC

 loc_4893D3:
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_4893DF
                and     eax, 0
                inc     eax
                jmp     short loc_4893E2

 loc_4893DF:

                and     eax, 0

 loc_4893E2:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9390
                xor     ecx, ds:dword_4D9394
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_489405
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_489405:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCBC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_485EFB(void)
```

```
{
     __asm
     {
                    push    ebp
                    mov     ebp, esp
                    sub     esp, 0Ch
                    push    ebx
                    push    esi
                    push    edi
                    mov     eax, [ebp+8]
                    push    eax
                    call    ds:off_4DDCEC
                    add     esp, 4
                    mov     [ebp-4], eax
                    mov     eax, [ebp-4]
                    push    ebx
                    mov     ebx, 0FFFFh
                    and     eax, ebx
                    push    ecx
                    mov     ch, 2Dh
                    dec     ch
                    sub     ch, 1
                    jo      short loc_485F30
                    jl      short loc_485F2E

 loc_485F2B:
                    jmp     short loc_485F32

 loc_485F2E:
                    jz      short loc_485F2B

 loc_485F30:
                    jmp     short loc_485F2B

 loc_485F32:
                    sub     ch, 20h
                    dec     ch
                    dec     ch
                    sub     ch, 7
                    dec     ch
                    dec     ch
                    and     ah, ch
                    mov     cl, 77h
                    sub     cl, 2
                    dec     cl
                    dec     cl
                    dec     cl
                    dec     cl
                    not     cl
                    bswap   edx
                    not     cl
                    bswap   edx
```

```
                dec     cl
                dec     cl
                push    eax
                dec     cl
                dec     cl
                sub     cl, 12h
                dec     cl
                jo      short loc_485F6E
                jl      short loc_485F6C

loc_485F69:
                jmp     short loc_485F70

loc_485F6C:
                jz      short loc_485F69

loc_485F6E:
                jmp     short loc_485F69

loc_485F70:
                dec     cl
                and     eax, 40h
                dec     cl
                dec     cl
                dec     cl
                add     cl, 0Eh
                dec     cl
                dec     cl
                and     eax, 800h
                sub     cl, 1Fh
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                pop     eax
                and     al, cl
                mov     eax, eax
                pop     ecx
                pop     ebx
                test    eax, eax
                jnz     loc_48604D
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 98h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 0Ch
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                push    4
                pop     ecx
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                and     al, cl
                mov     bh, 0Fh
                and     bl, 0
                dec     bh
                sub     bh, 3
                dec     bh
                sub     bh, 1
```

```
                dec     bh
                and     ah, bh
                pop     ecx
                pop     ebx
                test    eax, eax
                jz      short loc_486038
                not     eax
                add     eax, 1
                stc
                jmp     short loc_48603E

loc_486038:
                not     eax
                add     eax, 1
                clc

loc_48603E:
                sbb     eax, eax
                add     eax, 1
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_48604D
                and     eax, 0
                jmp     short loc_486051

loc_48604D:
                and     eax, 0
                inc     eax

loc_486051:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9380
                xor     ecx, ds:dword_4D9384
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_486074
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_486074:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCAC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
```

```
    }
}




__declspec(naked) void sub_4867A8(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD04
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 80h
                jmp     short loc_4867D1
                mov     ebx, 4

 loc_4867D1:
                mov     ebx, 32h
                not     ebx
                bswap   eax
                not     ebx
                inc     ebx
                inc     ebx
                add     ebx, 8
                dec     ebx
                push    ecx
                mov     ecx, 4
                add     ebx, ecx
                inc     ebx
                pop     ecx
                bswap   eax
                and     eax, ebx
                pop     ebx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48681B
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48681B:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_483D1B(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCFC
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 86h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                pop     ebx
                pop     edx
                test    eax, eax
                jnz     loc_483E76
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 98h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
```

```
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 0Ch
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_483DFF
                jl      short loc_483DFD

loc_483DF8:
                jmp     short loc_483E01

loc_483DFD:
                jz      short loc_483DF8

loc_483DFF:
                jmp     short loc_483DF8

loc_483E01:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                jo      short loc_483E14
                jl      short loc_483E12

loc_483E0F:
                jmp     short loc_483E16

loc_483E12:
                jz      short loc_483E0F

loc_483E14:
```

```
                jmp     short loc_483E0F

loc_483E16:
                and     eax, 40h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1200h
                dec     dh
                sub     dh, 1
                dec     dh
                sub     dh, 7
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_483E4A
                jl      short loc_483E48

loc_483E43:
                jmp     short loc_483E4C

loc_483E48:
                jz      short loc_483E43

loc_483E4A:
                jmp     short loc_483E43

loc_483E4C:
                inc     eax
                dec     eax
                jo      short loc_483E59
                jl      short loc_483E57

loc_483E52:
                jmp     short loc_483E5B

loc_483E57:
                jz      short loc_483E52

loc_483E59:
                jmp     short loc_483E52

loc_483E5B:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_483E68
                jl      short loc_483E66

loc_483E63:
                jmp     short loc_483E6A

loc_483E66:
                jz      short loc_483E63

loc_483E68:
                jmp     short loc_483E63

loc_483E6A:
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_483E76
                and     eax, 0
                inc     eax
                jmp     short loc_483E79

loc_483E76:
                and     eax, 0

loc_483E79:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9390
                xor     ecx, ds:dword_4D9394
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_483E9C
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_483E9C:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCBC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}
```

```
__declspec(naked) void sub_48D987(void)
{
     __asm
     {
                  push    ebp
                  mov     ebp, esp
                  sub     esp, 0Ch
                  push    ebx
                  push    esi
                  push    edi
                  mov     eax, [ebp+8]
                  push    eax
                  call    ds:off_4DDD18
                  add     esp, 4
                  mov     [ebp-4], eax
                  mov     eax, [ebp-4]
                  push    edx
                  mov     edx, 0FFFFh
                  and     eax, edx
                  push    ebx
                  push    1F00h
                  pop     ebx
                  jo      short loc_48D9BD
                  jl      short loc_48D9BB

 loc_48D9B6:
                  jmp     short loc_48D9BF

 loc_48D9BB:
                  jz      short loc_48D9B6

 loc_48D9BD:
                  jmp     short loc_48D9B6

 loc_48D9BF:
                  sub     bh, 6
                  push    eax
                  dec     bh
                  dec     bh
                  and     eax, 800h
                  dec     bh
                  sub     bh, 0FFh
                  dec     bh
                  sub     bh, 15h
                  pop     eax
                  dec     bh
                  and     ah, bh
                  mov     edx, 15h
```

```
                dec     dl
                dec     dl
                sub     dl, 0Ah
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                jo      short loc_48D9FC
                jl      short loc_48D9FA

loc_48D9F5:
                jmp     short loc_48D9FE

loc_48D9FA:
                jz      short loc_48D9F5

loc_48D9FC:
                jmp     short loc_48D9F5

loc_48D9FE:
                and     al, dl
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_48DA13
                jl      short loc_48DA11

loc_48DA0C:
                jmp     short loc_48DA15

loc_48DA11:
                jz      short loc_48DA0C

loc_48DA13:
                jmp     short loc_48DA0C

loc_48DA15:
                inc     eax
                dec     eax
                jo      short loc_48DA22
                jl      short loc_48DA20

loc_48DA1B:
                jmp     short loc_48DA24

loc_48DA20:
                jz      short loc_48DA1B
```

```
 loc_48DA22:
                jmp     short loc_48DA1B

 loc_48DA24:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_48DA31
                jl      short loc_48DA2F

 loc_48DA2C:
                jmp     short loc_48DA33

 loc_48DA2F:
                jz      short loc_48DA2C

 loc_48DA31:
                jmp     short loc_48DA2C

 loc_48DA33:
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93AC
                xor     ecx, ds:dword_4D93B0
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48DA57
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48DA57:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48D37A(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD08
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                jo      short loc_48D39F
                jl      short loc_48D39D

loc_48D39A:
                jmp     short loc_48D3A1

loc_48D39D:
                jz      short loc_48D39A

loc_48D39F:
                jmp     short loc_48D39A

loc_48D3A1:
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 4
                dec     ch
                sub     ch, 3
                dec     ch
                and     ah, ch
                mov     cl, 74h
                sub     cl, 8
                dec     cl
                sub     cl, 6
                not     al
                bswap   ecx
                not     al
                bswap   ecx
                dec     cl
                dec     cl
                sub     cl, 10h
                dec     cl
                dec     cl
                add     cl, 0Ch
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 10h
                sub     cl, 1
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                jo      short loc_48D41C
                jl      short loc_48D41A

loc_48D417:
                jmp     short loc_48D41E

loc_48D41A:
                jz      short loc_48D417

loc_48D41C:
                jmp     short loc_48D417

loc_48D41E:
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D939C
                xor     ecx, ds:dword_4D93A0
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48D44D
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48D44D:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_482347(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 8
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Dh
                sub     ch, 2
                sub     ch, 20h
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                mov     ebx, [ebp+0Ch]
                dec     esi
                dec     edi
                xor     edx, edx
                or      ebx, edx
                jz      short loc_48238B
                dec     edi
                and     eax, 0
                jmp     short loc_482392

 loc_48238B:
                dec     edi
                and     eax, 0
                dec     edi
                dec     edi
                inc     eax

 loc_482392:
                mov     [ebp-8], eax
                mov     eax, ds:dword_4D9390
                xor     eax, ds:dword_4D9394
                shl     eax, 1
                mov     [ebp-4], eax
                cmp     dword ptr [ebp-8], 0
                jz      short loc_4823B4
                mov     ecx, [ebp-4]
                or      ecx, 1
                mov     [ebp-4], ecx

 loc_4823B4:
                mov     edx, [ebp-4]
                push    edx
                call    ds:off_4DDCBC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_47FE0F(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
```

```
push    ebx
push    esi
push    edi
mov     eax, [ebp+8]
push    eax
call    ds:off_4DDCF0
add     esp, 4
mov     [ebp-4], eax
mov     eax, [ebp-4]
push    edx
mov     edx, 0FFFFh
and     eax, edx
push    ebx
push    eax
mov     bh, 4
dec     bh
dec     bh
dec     bh
dec     bh
and     eax, 80h
bswap   ecx
pop     eax
bswap   ecx
and     ah, bh
mov     bl, 86h
sub     bl, 5
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
sub     bl, 1Ah
dec     bl
sub     bl, 1Fh
not     bx
bswap   eax
not     bx
bswap   eax
and     al, bl
pop     ebx
pop     edx
test    eax, eax
jnz     loc_47FF33
mov     eax, [ebp-4]
push    edx
mov     edx, 0FFFFh
and     eax, edx
push    ebx
push    eax
mov     bh, 1
```

```
                dec     bh
                and     eax, 41h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 97h
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 0Ch
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_47FED4
                jl      short loc_47FED2

loc_47FECD:
                jmp     short loc_47FED6

loc_47FED2:
                jz      short loc_47FECD

loc_47FED4:
                jmp     short loc_47FECD

loc_47FED6:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                and     eax, 40h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1200h
                dec     dh
                sub     dh, 1
                dec     dh
                sub     dh, 7
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_47FF14
                jl      short loc_47FF12

loc_47FF0D:
                jmp     short loc_47FF16

loc_47FF12:
                jz      short loc_47FF0D

loc_47FF14:
                jmp     short loc_47FF0D

loc_47FF16:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_47FF25
                jl      short loc_47FF23

loc_47FF20:
                jmp     short loc_47FF27

loc_47FF23:
                jz      short loc_47FF20

loc_47FF25:
                jmp     short loc_47FF20

loc_47FF27:
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_47FF33
                and     eax, 0
                inc     eax
                jmp     short loc_47FF36

 loc_47FF33:
                and     eax, 0

 loc_47FF36:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9384
                xor     ecx, ds:dword_4D9388
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_47FF59
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_47FF59:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_48B144(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCDC_2
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 800h
                jmp     short loc_48B16D
                mov     ebx, 80h

loc_48B16D:
                mov     ebx, 72h
                not     ebx
                bswap   eax
                not     ebx
                inc     ebx
                inc     ebx
                add     ebx, 8
                dec     ebx
                push    ecx
                mov     ecx, 4
                add     ebx, ecx
                inc     ebx
                pop     ecx
                bswap   eax
                and     eax, ebx
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                push    eax
                mov     eax, [ebp-4]
                mov     edx, 0F00h
                sub     dh, 5
                xor     dh, 2
                and     eax, edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                cmp     eax, edx
                setz    cl
                mov     al, cl
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9370
                xor     ecx, ds:dword_4D9374
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48B1D8
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx
```

```
 loc_48B1D8:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDC9C
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48C809(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD08
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                mov     ecx, 40h
                mov     ecx, 0Ch
                not     ecx
                bswap   eax
                not     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                jo      short loc_48C842
                jl      short loc_48C840

 loc_48C83D:
                jmp     short loc_48C844

 loc_48C840:
                jz      short loc_48C83D
```

```
loc_48C842:
                jmp     short loc_48C83D

loc_48C844:
                xor     eax, eax
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                inc     cl
                inc     cl
                inc     cl
                and     ecx, 40h
                jo      short loc_48C85E
                jl      short loc_48C85C

loc_48C859:
                jmp     short loc_48C860

loc_48C85C:
                jz      short loc_48C859

loc_48C85E:
                jmp     short loc_48C859

loc_48C860:
                inc     eax
                inc     cl
                inc     cl
                add     ecx, 0Ah
                dec     ecx
                push    edx
                mov     edx, 4
                add     ecx, edx
                inc     ecx
                pop     edx
                pop     ecx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D939C
                xor     ecx, ds:dword_4D93A0
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48C897
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48C897:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_48ADAB(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD04
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 4
                dec     ch
                sub     ch, 3
                dec     ch
                and     ah, ch
                mov     cl, 0AFh
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 6
                not     al
```

```
bswap   ecx
not     al
bswap   ecx
dec     cl
dec     cl
sub     cl, 10h
dec     cl
dec     cl
add     cl, 0Ch
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
sub     cl, 10h
sub     cl, 1
dec     cl
dec     cl
dec     cl
sub     cl, 3
dec     cl
dec     cl
not     ecx
bswap   eax
not     ecx
bswap   eax
inc     cl
add     cl, 2
and     al, cl
mov     eax, eax
pop     ecx
neg     eax
sbb     eax, eax
inc     eax
pop     ebx
push    eax
mov     eax, [ebp-4]
mov     edx, 1400h
inc     dh
dec     dh
inc     dh
sub     dh, 10h
inc     dh
inc     dh
inc     dh
and     eax, edx
neg     eax
sbb     eax, eax
inc     eax
mov     edx, eax
pop     eax
```

```
                xor     ecx, ecx
                jo      short loc_48AE70
                jl      short loc_48AE6E

loc_48AE6B:
                jmp     short loc_48AE72

loc_48AE6E:
                jz      short loc_48AE6B

loc_48AE70:
                jmp     short loc_48AE6B

loc_48AE72:
                cmp     eax, edx
                jz      short loc_48AE89
                jo      short loc_48AE81
                jl      short loc_48AE7F

loc_48AE7A:
                jmp     short loc_48AE83

loc_48AE7F:
                jz      short loc_48AE7A

loc_48AE81:
                jmp     short loc_48AE7A

loc_48AE83:
                and     eax, 0
                inc     eax
                jmp     short loc_48AE8C

loc_48AE89:
                and     eax, 0

loc_48AE8C:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48AEAF
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48AEAF:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
```

```
                        add     esp, 4
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
    }
}




__declspec(naked) void sub_484DC4(void)
{
    __asm
    {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDCFC
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    ebx
                        mov     ebx, 0FFFFh
                        and     eax, ebx
                        push    ecx
                        push    7Fh
                        pop     ecx
                        dec     ecx
                        xor     ecx, 7Eh
                        and     al, cl
                        mov     bh, 0Fh
                        and     bl, 0
                        dec     bh
                        jo      short loc_484E04
                        jl      short loc_484E02

 loc_484DFD:
                        jmp     short loc_484E06

 loc_484E02:
                        jz      short loc_484DFD

 loc_484E04:
                        jmp     short loc_484DFD
```

```
loc_484E06:
                sub     bh, 6
                jo      short loc_484E14
                jl      short loc_484E12

loc_484E0D:
                jmp     short loc_484E16

loc_484E12:
                jz      short loc_484E0D

loc_484E14:
                jmp     short loc_484E0D

loc_484E16:
                and     ah, bh
                pop     ecx
                pop     ebx
                test    eax, eax
                jz      short loc_484E26
                not     eax
                add     eax, 1
                stc
                jmp     short loc_484E2C

loc_484E26:
                not     eax
                add     eax, 1
                clc

loc_484E2C:
                sbb     eax, eax
                add     eax, 1
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9390
                xor     ecx, ds:dword_4D9394
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_484E54
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_484E54:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCBC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48A8A9(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCDC_2
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Dh
                dec     ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 7
                dec     ch
                dec     ch
                and     ah, ch
                mov     cl, 0BDh
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     cl
                bswap   edx
                not     cl
```

```
bswap   edx
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
push    eax
dec     cl
dec     cl
sub     cl, 12h
dec     cl
dec     cl
sub     cl, 3
dec     cl
and     eax, 10h
dec     cl
dec     cl
dec     cl
add     cl, 0Fh
dec     cl
and     eax, 80h
sub     cl, 1Fh
dec     cl
inc     cl
dec     cl
dec     cl
inc     cl
not     ecx
bswap   eax
not     ecx
bswap   eax
pop     eax
inc     cl
inc     cl
inc     cl
and     al, cl
mov     eax, eax
pop     ecx
neg     eax
sbb     eax, eax
neg     eax
pop     ebx
mov     [ebp-0Ch], eax
mov     ecx, ds:dword_4D9370
xor     ecx, ds:dword_4D9374
shl     ecx, 1
mov     [ebp-8], ecx
cmp     dword ptr [ebp-0Ch], 0
jz      short loc_48A97F
```

```
                    mov     edx, [ebp-8]
                    or      edx, 1
                    mov     [ebp-8], edx

 loc_48A97F:
                    mov     eax, [ebp-8]
                    push    eax
                    call    ds:off_4DDC9C
                    add     esp, 4
                    pop     edi
                    pop     esi
                    pop     ebx
                    mov     esp, ebp
                    pop     ebp
                    retn
    }
}



__declspec(naked) void sub_488671(void)
{
    __asm
    {
                    push    ebp
                    mov     ebp, esp
                    sub     esp, 0Ch
                    push    ebx
                    push    esi
                    push    edi
                    mov     eax, [ebp+8]
                    push    eax
                    call    ds:off_4DDCE0
                    add     esp, 4
                    mov     [ebp-4], eax
                    mov     eax, [ebp-4]
                    push    edx
                    mov     edx, 0FFFFh
                    and     eax, edx
                    push    ebx
                    push    410h
                    pop     ebx
                    dec     bh
                    dec     bh
                    sub     bh, 0FFh
                    sub     bh, 2
                    dec     bh
                    and     ah, bh
                    mov     bl, 0Eh
                    sub     bl, 4
                    dec     bl
                    sub     bl, 1
```

```
                sub     bl, 1
                sub     bl, 1
                sub     bl, 1
                sub     bl, 1
                and     al, bl
                pop     ebx
                pop     edx
                test    eax, eax
                jz      short loc_4886D0
                not     eax
                add     eax, 1
                stc
                jmp     short loc_4886D6

loc_4886D0:
                not     eax
                add     eax, 1
                clc

loc_4886D6:
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_4886E5
                jl      short loc_4886E3

loc_4886DE:
                jmp     short loc_4886E7

loc_4886E3:
                jz      short loc_4886DE

loc_4886E5:
                jmp     short loc_4886DE

loc_4886E7:
                inc     eax
                dec     eax
                jo      short loc_4886F4
                jl      short loc_4886F2

loc_4886ED:
                jmp     short loc_4886F6

loc_4886F2:
                jz      short loc_4886ED

loc_4886F4:
                jmp     short loc_4886ED

loc_4886F6:
                inc     eax
```

```
                 dec     eax
                 inc     eax
                 dec     eax
                 jo      short loc_488703
                 jl      short loc_488701

 loc_4886FE:
                 jmp     short loc_488705

 loc_488701:
                 jz      short loc_4886FE

 loc_488703:
                 jmp     short loc_4886FE

 loc_488705:
                 inc     eax
                 mov     [ebp-0Ch], eax
                 mov     ecx, ds:dword_4D9374
                 xor     ecx, ds:dword_4D9378
                 shl     ecx, 1
                 mov     [ebp-8], ecx
                 cmp     dword ptr [ebp-0Ch], 0
                 jz      short loc_488729
                 mov     edx, [ebp-8]
                 or      edx, 1
                 mov     [ebp-8], edx

 loc_488729:
                 mov     eax, [ebp-8]
                 push    eax
                 call    ds:off_4DDCA0
                 add     esp, 4
                 pop     edi
                 pop     esi
                 pop     ebx
                 mov     esp, ebp
                 pop     ebp
                 retn
     }
}




__declspec(naked) void sub_489AE0(void)
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 sub     esp, 0Ch
```

```
push    ebx
push    esi
push    edi
mov     eax, [ebp+8]
push    eax
call    ds:off_4DDD10
add     esp, 4
mov     [ebp-4], eax
mov     eax, [ebp-4]
push    eax
mov     eax, 4
bswap   eax
not     eax
pop     eax
push    edx
mov     dh, 80h
mov     dh, 0
inc     dh
mov     ecx, ecx
inc     dh
inc     dh
inc     esi
inc     dh
dec     edi
inc     dh
dec     dh
inc     dh
push    ebx
inc     dh
push    ecx
bswap   ecx
not     ecx
push    eax
not     eax
mov     eax, 800h
xchg    eax, ecx
mov     ecx, 40h
xchg    eax, ecx
not     eax
pop     eax
not     ecx
pop     ecx
inc     dh
inc     dh
and     ebx, 800h
add     dh, 4
and     ebx, 10h
inc     dh
inc     dh
pop     ebx
sub     dh, 0Dh
dec     dh
```

```
                and     ah, dh
                mov     dl, 5
                sub     dl, 0FFh
                dec     dl
                dec     dl
                inc     dl
                dec     dl
                sub     dl, 0FFh
                dec     dl
                dec     dl
                inc     dl
                dec     dl
                dec     dl
                dec     dl
                and     al, dl
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A4
                xor     ecx, ds:dword_4D93A8
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_489B9E
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_489B9E:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48A3DB(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCDC_2
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Dh
                dec     ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 7
                dec     ch
                dec     ch
                and     ah, ch
                mov     cl, 77h
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                not     cl
                bswap   edx
                not     cl
                bswap   edx
                dec     cl
                dec     cl
                push    eax
                dec     cl
                dec     cl
                sub     cl, 12h
                dec     cl
                dec     cl
                jo      short loc_48A443
                jl      short loc_48A441

loc_48A43E:
                jmp     short loc_48A445

loc_48A441:
                jz      short loc_48A43E

loc_48A443:
```

```
                jmp     short loc_48A43E

loc_48A445:
                and     eax, 40h
                dec     cl
                dec     cl
                dec     cl
                add     cl, 0Eh
                dec     cl
                dec     cl
                and     eax, 41h
                sub     cl, 22h
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                pop     eax
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9370
                xor     ecx, ds:dword_4D9374
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48A492
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48A492:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDC9C
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}
```

```
__declspec(naked) void sub_488260(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCEC
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1F00h
                pop     ebx
                jo      short loc_488296
                jl      short loc_488294

 loc_48828F:
                jmp     short loc_488298

 loc_488294:
                jz      short loc_48828F

 loc_488296:
                jmp     short loc_48828F

 loc_488298:
                sub     bh, 5
                dec     bh
                push    eax
                dec     bh
                dec     bh
                and     eax, 41h
                dec     bh
                sub     bh, 12h
                sub     bh, 3
                pop     eax
                dec     bh
                and     ah, bh
                mov     edx, 15h
                dec     dl
                sub     dl, 3
                dec     dl
                sub     dl, 7
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                and     al, dl
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9380
                xor     ecx, ds:dword_4D9384
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4882F8
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_4882F8:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCAC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_4859B7(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD14
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 800h
                jmp     short loc_4859E0
                mov     ebx, 80h

loc_4859E0:
                mov     ebx, 72h
                not     ebx
                bswap   eax
                not     ebx
                inc     ebx
                inc     ebx
                add     ebx, 8
                dec     ebx
                push    ecx
                mov     ecx, 4
                add     ebx, ecx
                inc     ebx
                pop     ecx
                bswap   eax
                and     eax, ebx
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A8
                xor     ecx, ds:dword_4D93AC
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_485A29
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_485A29:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
```

```
                    retn
        }
}




__declspec(naked) void sub_485898(void)
{
        __asm
        {
                    push    ebp
                    mov     ebp, esp
                    sub     esp, 0Ch
                    push    ebx
                    push    esi
                    push    edi
                    mov     eax, [ebp+8]
                    push    eax
                    call    ds:off_4DDCF4
                    add     esp, 4
                    mov     [ebp-4], eax
                    mov     eax, [ebp-4]
                    push    ebx
                    mov     ebx, 0FFFFh
                    and     eax, ebx
                    push    ecx
                    mov     ch, 2Ch
                    add     ch, 0FFh
                    sub     ch, 20h
                    dec     ch
                    dec     ch
                    sub     ch, 4
                    dec     ch
                    sub     ch, 3
                    dec     ch
                    and     ah, ch
                    mov     cl, 0ADh
                    dec     cl
                    dec     cl
                    dec     cl
                    sub     cl, 6
                    not     al
                    bswap   ecx
                    not     al
                    bswap   ecx
                    dec     cl
                    sub     cl, 11h
                    dec     cl
                    add     cl, 0Ch
                    dec     cl
                    dec     cl
                    dec     cl
                    sub     cl, 3
                    sub     cl, 10h
                    sub     cl, 1
                    dec     cl
                    dec     cl
                    dec     cl
                    dec     cl
                    dec     cl
                    dec     cl
                    dec     cl
                    dec     cl
                    not     ecx
                    bswap   eax
                    not     ecx
                    bswap   eax
                    inc     cl
                    dec     cl
                    add     cl, 2
                    and     al, cl
                    mov     eax, eax
                    pop     ecx
                    neg     eax
                    sbb     eax, eax
                    inc     eax
                    pop     ebx
                    push    eax
                    mov     eax, [ebp-4]
                    mov     edx, 200h
                    inc     dh
                    inc     dh
                    dec     dh
                    inc     dh
                    dec     dh
                    inc     dh
                    inc     dh
                    inc     dh
                    inc     dh
                    dec     dh
                    inc     dh
                    inc     dh
                    dec     dh
                    dec     dh
                    inc     dh
                    inc     dh
                    dec     dh
                    inc     dh
                    inc     dh
                    dec     dh
                    and     eax, edx
                    neg     eax
                    sbb     eax, eax
```

```
                inc     eax
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                cmp     eax, edx
                jo      short loc_485973
                jl      short loc_485971

loc_48596E:
                jmp     short loc_485975


loc_485971:
                jz      short loc_48596E

loc_485973:
                jmp     short loc_48596E

loc_485975:
                jnz     short loc_48597C
                and     eax, 0
                jmp     short loc_485980

loc_48597C:
                and     eax, 0
                inc     eax

loc_485980:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9388
                xor     ecx, ds:dword_4D938C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4859A3
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_4859A3:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}
```

```
__declspec(naked) void sub_48BF8D(void)
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 sub     esp, 0Ch
                 push    ebx
                 push    esi
                 push    edi
                 mov     eax, [ebp+8]
                 push    eax
                 call    ds:off_4DDD04
                 add     esp, 4
                 mov     [ebp-4], eax
                 mov     eax, [ebp-4]
                 jo      short loc_48BFB2
                 jl      short loc_48BFB0

 loc_48BFAD:
                 jmp     short loc_48BFB4

 loc_48BFB0:
                 jz      short loc_48BFAD

 loc_48BFB2:
                 jmp     short loc_48BFAD

 loc_48BFB4:
                 push    ebx
                 mov     ebx, 0FFFFh
                 and     eax, ebx
                 push    ecx
                 mov     ch, 2Ch
                 sub     ch, 1
                 sub     ch, 20h
                 dec     ch
                 dec     ch
                 sub     ch, 4
                 dec     ch
                 sub     ch, 3
                 dec     ch
                 and     ah, ch
                 mov     cl, 70h
                 sub     cl, 2
                 dec     cl
                 dec     cl
```

```
                dec     cl
                sub     cl, 6
                not     al
                bswap   ecx
                not     al
                bswap   ecx
                dec     cl
                dec     cl
                sub     cl, 10h
                dec     cl
                dec     cl
                add     cl, 0Ch
                dec     cl
                dec     cl
                dec     cl
                jo      short loc_48C008
                jl      short loc_48C006

loc_48C003:
                jmp     short loc_48C00A

loc_48C006:
                jz      short loc_48C003

loc_48C008:
                jmp     short loc_48C003

loc_48C00A:
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 10h
                sub     cl, 1
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                and     al, cl
                mov     eax, eax
                pop     ecx
                pop     ebx
                test    eax, eax
                jnz     loc_48C146
                mov     eax, [ebp-4]
                jo      short loc_48C04F
                jl      short loc_48C04D

loc_48C04A:
                jmp     short loc_48C051

loc_48C04D:
                jz      short loc_48C04A

loc_48C04F:
                jmp     short loc_48C04A

loc_48C051:
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                jo      short loc_48C080
                jl      short loc_48C07E

loc_48C07B:
                jmp     short loc_48C082

loc_48C07E:
                jz      short loc_48C07B

loc_48C080:
                jmp     short loc_48C07B

loc_48C082:
                mov     bl, 0C6h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_48C0CF
                jl      short loc_48C0CD

loc_48C0C8:
                jmp     short loc_48C0D1

loc_48C0CD:
                jz      short loc_48C0C8

loc_48C0CF:
                jmp     short loc_48C0C8

loc_48C0D1:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                jo      short loc_48C0E4
                jl      short loc_48C0E2

loc_48C0DF:
                jmp     short loc_48C0E6

loc_48C0E2:
```

```
                jz      short loc_48C0DF

loc_48C0E4:
                jmp     short loc_48C0DF

loc_48C0E6:
                and     eax, 40h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1200h
                dec     dh
                sub     dh, 1
                dec     dh
                sub     dh, 7
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_48C11A
                jl      short loc_48C118

loc_48C113:
                jmp     short loc_48C11C

loc_48C118:
                jz      short loc_48C113

loc_48C11A:
                jmp     short loc_48C113

loc_48C11C:
                inc     eax
                dec     eax
                jo      short loc_48C129
                jl      short loc_48C127

loc_48C122:
                jmp     short loc_48C12B

loc_48C127:
                jz      short loc_48C122

loc_48C129:
                jmp     short loc_48C122
```

```
loc_48C12B:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_48C138
                jl      short loc_48C136

loc_48C133:
                jmp     short loc_48C13A

loc_48C136:
                jz      short loc_48C133

loc_48C138:
                jmp     short loc_48C133

loc_48C13A:
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_48C146
                and     eax, 0
                inc     eax
                jmp     short loc_48C149

loc_48C146:
                and     eax, 0

loc_48C149:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48C16C
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48C16C:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
```

```
    }
}




__declspec(naked) void sub_487BED(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE0
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    0D00h
                pop     ebx
                sub     bh, 3
                dec     bh
                dec     bh
                dec     bh
                push    eax
                dec     bh
                dec     bh
                and     eax, 40h
                dec     bh
                sub     bh, 3
                pop     eax
                dec     bh
                jo      short loc_487C3C
                jl      short loc_487C3A

loc_487C35:
                jmp     short loc_487C3E

loc_487C3A:
                jz      short loc_487C35

loc_487C3C:
                jmp     short loc_487C35
```

```
loc_487C3E:
                and     ah, bh
                mov     edx, 26h
                dec     dl
                dec     dl
                sub     dl, 3
                dec     dl
                sub     dl, 17h
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                jo      short loc_487C66
                jl      short loc_487C64

loc_487C5F:
                jmp     short loc_487C68

loc_487C64:
                jz      short loc_487C5F

loc_487C66:
                jmp     short loc_487C5F

loc_487C68:
                and     al, dl
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9374
                xor     ecx, ds:dword_4D9378
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_487C95
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_487C95:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
        }
}




__declspec(naked) void sub_483136(void)
{
        __asm
        {
                 push    ebp
                 mov     ebp, esp
                 sub     esp, 0Ch
                 push    ebx
                 push    esi
                 push    edi
                 mov     eax, [ebp+8]
                 push    eax
                 call    ds:off_4DDCE4
                 add     esp, 4
                 mov     [ebp-4], eax
                 mov     eax, [ebp-4]
                 push    ebx
                 mov     ebx, 0FFFFh
                 and     eax, ebx
                 push    ecx
                 mov     ch, 2Ch
                 sub     ch, 1
                 sub     ch, 20h
                 dec     ch
                 dec     ch
                 sub     ch, 4
                 dec     ch
                 sub     ch, 3
                 dec     ch
                 and     ah, ch
                 mov     cl, 70h
                 sub     cl, 2
                 dec     cl
                 dec     cl
                 dec     cl
                 sub     cl, 6
                 not     al
                 bswap   ecx
                 not     al
                 bswap   ecx
                 dec     cl
                 dec     cl
                 sub     cl, 10h
                 dec     cl
                 dec     cl
                 add     cl, 0Ch
                 dec     cl
                 dec     cl
                 dec     cl
                 dec     cl
                 dec     cl
                 dec     cl
                 sub     cl, 10h
                 sub     cl, 1
                 dec     cl
                 dec     cl
                 dec     cl
                 dec     cl
                 dec     cl
                 dec     cl
                 dec     cl
                 dec     cl
                 not     ecx
                 bswap   eax
                 not     ecx
                 bswap   eax
                 inc     cl
                 add     cl, 2
                 and     al, cl
                 mov     eax, eax
                 pop     ecx
                 neg     eax
                 sbb     eax, eax
                 inc     eax
                 pop     ebx
                 mov     [ebp-0Ch], eax
                 mov     ecx, ds:dword_4D9378
                 xor     ecx, ds:dword_4D937C
                 shl     ecx, 1
                 mov     [ebp-8], ecx
                 cmp     dword ptr [ebp-0Ch], 0
                 jz      short loc_4831F4
                 mov     edx, [ebp-8]
                 or      edx, 1
                 mov     [ebp-8], edx

 loc_4831F4:
                 mov     eax, [ebp-8]
                 push    eax
                 call    ds:off_4DDCA4
                 add     esp, 4
                 pop     edi
                 pop     esi
                 pop     ebx
                 mov     esp, ebp
                 pop     ebp
```

```
                        retn
        }
}




__declspec(naked) void sub_485479(void)
{
        __asm
        {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDCEC
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    ebx
                        mov     ebx, 0FFFFh
                        and     eax, ebx
                        push    ecx
                        mov     ch, 2Ch
                        sub     ch, 1
                        sub     ch, 20h
                        dec     ch
                        dec     ch
                        sub     ch, 4
                        dec     ch
                        sub     ch, 3
                        dec     ch
                        and     ah, ch
                        mov     cl, 0AFh
                        sub     cl, 2
                        dec     cl
                        dec     cl
                        dec     cl
                        sub     cl, 6
                        not     al
                        bswap   ecx
                        not     al
                        bswap   ecx
                        dec     cl
                        dec     cl
                        sub     cl, 10h
                        dec     cl
                        dec     cl
```

```
                add     cl, 0Ch
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 10h
                sub     cl, 1
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ebx
                push    eax
                mov     eax, [ebp-4]
                mov     edx, 300h
                inc     dh
                inc     dh
                dec     dh
                inc     dh
                inc     dh
                inc     dh
                inc     dh
                and     eax, edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                jo      short loc_485540
                jl      short loc_48553E

loc_48553B:
                jmp     short loc_485542
```

```
loc_48553E:
                jz      short loc_48553B

loc_485540:
                jmp     short loc_48553B

loc_485542:
                cmp     eax, edx
                jz      short loc_485559
                jo      short loc_485551
                jl      short loc_48554F

loc_48554A:
                jmp     short loc_485553

loc_48554F:
                jz      short loc_48554A

loc_485551:
                jmp     short loc_48554A

loc_485553:
                and     eax, 0
                inc     eax
                jmp     short loc_48555C

loc_485559:
                and     eax, 0

loc_48555C:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9380
                xor     ecx, ds:dword_4D9384
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48557F
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48557F:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCAC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
        }
}




__declspec(naked) void sub_48B41E(void)
{
        __asm
        {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD00
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                mov     ecx, 800h
                mov     ecx, 6
                not     ecx
                bswap   eax
                not     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                add     ecx, 4
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                add     ecx, 3
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Dh
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Ah
                dec     ecx
                push    edx
                mov     edx, 4
                add     ecx, edx
                inc     ecx
                pop     edx
                bswap   eax
                and     eax, ecx
                pop     ecx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9394
                xor     ecx, ds:dword_4D9398
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48B4B4
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48B4B4:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_489500(void)
```

```
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 sub     esp, 0Ch
                 push    ebx
                 push    esi
                 push    edi
                 mov     eax, [ebp+8]
                 push    eax
                 call    ds:off_4DDD10
                 add     esp, 4
                 mov     [ebp-4], eax
                 mov     eax, [ebp-4]
                 jo      short loc_489525
                 jl      short loc_489523

 loc_489520:
                 jmp     short loc_489527


 loc_489523:
                 jz      short loc_489520

 loc_489525:
                 jmp     short loc_489520


 loc_489527:
                 push    ebx
                 mov     ebx, [ebp+0Ch]
                 mov     ebx, 0FFFFh
                 and     eax, ebx
                 push    ecx
                 mov     ch, 2Ch
                 sub     ch, 1
                 sub     ch, 20h
                 dec     ch
                 dec     ch
                 sub     ch, 4
                 dec     ch
                 sub     ch, 3
                 dec     ch
                 and     ah, ch
                 mov     cl, 70h
                 sub     cl, 2
                 dec     cl
                 dec     cl
                 dec     cl
                 sub     cl, 6
                 not     al
```

```
                bswap   ecx
                not     al
                bswap   ecx
                dec     cl
                dec     cl
                sub     cl, 10h
                dec     cl
                dec     cl
                add     cl, 0Ch
                dec     cl
                dec     cl
                dec     cl
                jo      short loc_48957E
                jl      short loc_48957C

loc_489579:
                jmp     short loc_489580


loc_48957C:
                jz      short loc_489579

loc_48957E:
                jmp     short loc_489579


loc_489580:
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 13h
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                and     al, cl
                mov     eax, eax
                pop     ecx
                pop     ebx
                test    eax, eax
                jnz     loc_4896A9
                mov     eax, [ebp-4]
                jo      short loc_4895BE
                jl      short loc_4895BC

loc_4895B9:
                jmp     short loc_4895C0


loc_4895BC:
                jz      short loc_4895B9

loc_4895BE:
                jmp     short loc_4895B9


loc_4895C0:
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                and     bh, 0
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                jo      short loc_4895E4
                jl      short loc_4895E2

loc_4895DF:
                jmp     short loc_4895E6


loc_4895E2:
                jz      short loc_4895DF

loc_4895E4:
                jmp     short loc_4895DF


loc_4895E6:
                mov     bl, 0C6h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_489633
                jl      short loc_489631

loc_48962C:
                jmp     short loc_489635


loc_489631:
                jz      short loc_48962C

loc_489633:
                jmp     short loc_48962C


loc_489635:
                sub     bl, 6
                push    eax
                dec     bl
                dec     bl
                jo      short loc_489646
                jl      short loc_489644

loc_489641:
                jmp     short loc_489648


loc_489644:
                jz      short loc_489641

loc_489646:
                jmp     short loc_489641
```

```
loc_489648:
                and     eax, 40h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1200h
                dec     dh
                sub     dh, 1
                dec     dh
                sub     dh, 7
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_48967C
                jl      short loc_48967A

loc_489675:
                jmp     short loc_48967E

loc_48967A:
                jz      short loc_489675

loc_48967C:
                jmp     short loc_489675


loc_48967E:
                inc     eax
                dec     eax
                jo      short loc_48968B
                jl      short loc_489689

loc_489684:
                jmp     short loc_48968D

loc_489689:
                jz      short loc_489684

loc_48968B:
                jmp     short loc_489684


loc_48968D:
```

```
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_48969C
                jl      short loc_48969A

loc_489695:
                jmp     short loc_48969E

loc_48969A:
                jz      short loc_489695

loc_48969C:
                jmp     short loc_489695


loc_48969E:
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_4896A9
                and     eax, 0
                jmp     short loc_4896AD


loc_4896A9:
                and     eax, 0
                inc     eax

loc_4896AD:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A4
                xor     ecx, ds:dword_4D93A8
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4896D0
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_4896D0:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
```

```
                        retn
            }
    }



    __declspec(naked) void sub_4853D4(void)
    {
            __asm
            {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDCE8
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    edx
                        mov     edx, 0FFFFh
                        and     eax, edx
                        push    ebx
                        push    eax
                        mov     bh, 3
                        dec     bh
                        sub     bh, 2
                        and     eax, 800h
                        bswap   ecx
                        pop     eax
                        bswap   ecx
                        and     ah, bh
                        mov     bl, 0B5h
                        dec     bl
                        dec     esi
                        dec     bl
                        dec     bl
                        dec     edi
                        dec     bl
                        sub     bl, 14h
                        dec     bl
                        dec     bl
                        sub     bl, 20h
                        dec     edi
                        sub     bl, 1Ah
                        dec     bl
                        sub     bl, 1Fh
                        not     bx
                        bswap   eax
```

```
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D937C
                xor     ecx, ds:dword_4D9380
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_485465
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_485465:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_480513(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD10
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 86h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                pop     ebx
                pop     edx
                test    eax, eax
                jnz     loc_48066D
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
```

```
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 98h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 0Ch
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_4805F7
                jl      short loc_4805F5

loc_4805F0:
                jmp     short loc_4805F9

loc_4805F5:
                jz      short loc_4805F0

loc_4805F7:
                jmp     short loc_4805F0

loc_4805F9:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                jo      short loc_48060C
                jl      short loc_48060A

loc_480607:
                jmp     short loc_48060E

loc_48060A:
                jz      short loc_480607

loc_48060C:
                jmp     short loc_480607

loc_48060E:
                and     eax, 40h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1200h
                dec     dh
                sub     dh, 1
                dec     dh
                sub     dh, 7
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_480642
                jl      short loc_480640

loc_48063B:
                jmp     short loc_480644

loc_480640:
                jz      short loc_48063B

loc_480642:
                jmp     short loc_48063B

loc_480644:
                inc     eax
                dec     eax
                jo      short loc_480651
                jl      short loc_48064F
```

```
loc_48064A:
                jmp     short loc_480653

loc_48064F:
                jz      short loc_48064A

loc_480651:
                jmp     short loc_48064A

loc_480653:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_480660
                jl      short loc_48065E

loc_48065B:
                jmp     short loc_480662

loc_48065E:
                jz      short loc_48065B

loc_480660:
                jmp     short loc_48065B

loc_480662:
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_48066D
                and     eax, 0
                jmp     short loc_480671

loc_48066D:
                and     eax, 0
                inc     eax

loc_480671:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A4
                xor     ecx, ds:dword_4D93A8
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_480694
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_480694:
```

```
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_480753(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD0C
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_480786
                jl      short loc_480784

 loc_48077F:
                jmp     short loc_480788

 loc_480784:
                jz      short loc_48077F

 loc_480786:
                jmp     short loc_48077F

 loc_480788:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                jo      short loc_48079B
                jl      short loc_480799

loc_480796:
                jmp     short loc_48079D

loc_480799:
                jz      short loc_480796

loc_48079B:
                jmp     short loc_480796

loc_48079D:
                and     eax, 40h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1200h
                dec     dh
                sub     dh, 1
                dec     dh
                sub     dh, 7
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A0
                xor     ecx, ds:dword_4D93A4
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4807E9
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_4807E9:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCCC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_48027A(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCFC
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, [ebp+0Ch]
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 4
                dec     ch
                sub     ch, 3
                dec     ch
                and     ah, ch
                mov     cl, 70h
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 6
                not     al
                bswap   ecx
                not     al
                bswap   ecx
                dec     cl
                dec     cl
                jmp     short loc_4802D9
                and     eax, 1

loc_4802D9:
                sub     cl, 10h
                dec     cl
                dec     cl
                add     cl, 0Ch
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 10h
                sub     cl, 3
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9390
                xor     ecx, ds:dword_4D9394
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48033C
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48033C:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCBC
                add     esp, 4
```

```
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
        }
}



__declspec(naked) void sub_48C558(void)
{
        __asm
        {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDD04
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    edx
                        mov     edx, 0FFFFh
                        and     eax, edx
                        push    ebx
                        push    eax
                        mov     bh, 7
                        dec     bh
                        dec     bh
                        dec     bh
                        and     bh, 0
                        and     eax, 800h
                        bswap   ecx
                        pop     eax
                        bswap   ecx
                        and     ah, bh
                        mov     bl, 98h
                        sub     bl, 5
                        dec     bl
                        dec     bl
                        dec     bl
                        dec     bl
                        dec     bl
                        dec     bl
                        dec     bl
```

```
                sub     bl, 0Ch
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48C5E3
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48C5E3:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48818D(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD0C
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Dh
                dec     ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 7
                dec     ch
                dec     ch
                and     ah, ch
                mov     cl, 77h
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                not     cl
                bswap   edx
                not     cl
                bswap   edx
                dec     cl
                dec     cl
                push    eax
                dec     cl
                dec     cl
                sub     cl, 12h
                dec     cl
                dec     cl
                jo      short loc_4881F5
                jl      short loc_4881F3

loc_4881F0:
                jmp     short loc_4881F7

loc_4881F3:
                jz      short loc_4881F0

loc_4881F5:
                jmp     short loc_4881F0

loc_4881F7:
                and     eax, 40h
                dec     cl
                dec     cl
                dec     cl
                add     cl, 0Eh
                dec     cl
                dec     cl
                and     eax, 80h
                sub     cl, 1Fh
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                pop     eax
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A0
                xor     ecx, ds:dword_4D93A4
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48824C
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48824C:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCCC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_4826EF(void)
{
    __asm
    {
                    push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD10
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     dh, 6
                dec     dh
                jo      short loc_482719
                jl      short loc_482717

loc_482714:
                jmp     short loc_48271B

loc_482717:
                jz      short loc_482714

loc_482719:
                jmp     short loc_482714

loc_48271B:
                sub     dh, 2
                push    eax
                mov     eax, 800h
                bswap   eax
                not     eax
                pop     eax
                sub     dh, 3
                and     ah, dh
                mov     dl, 4
                dec     dl
                sub     dl, 2
                dec     dl
                sub     dl, 0FFh
                and     al, dl
                not     ah
                bswap   eax
                bswap   eax
                not     ah
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A4
                xor     ecx, ds:dword_4D93A8
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48276D
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48276D:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_489C62(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD08
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, [ebp+0Ch]
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 4
                dec     ch
                sub     ch, 3
                dec     ch
                and     ah, ch
                mov     cl, 70h
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 6
                not     al
                bswap   ecx
                not     al
                bswap   ecx
                dec     cl
                dec     cl
                jmp     short loc_489CC1
                and     eax, 1

loc_489CC1:
                sub     cl, 10h
                dec     cl
                dec     cl
                add     cl, 0Ch
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 10h
                sub     cl, 3
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D939C
                xor     ecx, ds:dword_4D93A0
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_489D24
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_489D24:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_486362(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE8
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    0D00h
                pop     ebx
                jo      short loc_486398
                jl      short loc_486396

 loc_486391:
```

```
                jmp     short loc_48639A

loc_486396:
                jz      short loc_486391

loc_486398:
                jmp     short loc_486391

loc_48639A:
                sub     bh, 5
                dec     bh
                push    eax
                dec     bh
                dec     bh
                and     eax, 41h
                dec     bh
                sub     bh, 3
                pop     eax
                dec     bh
                and     ah, bh
                mov     edx, 25h
                dec     dl
                sub     dl, 3
                dec     dl
                sub     dl, 17h
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                and     al, dl
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D937C
                xor     ecx, ds:dword_4D9380
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4863F6
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_4863F6:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_4868CA(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD00
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 4
                dec     ch
                sub     ch, 3
                dec     ch
                and     ah, ch
                mov     cl, 70h
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 6
                not     al
                bswap   ecx
                not     al
                bswap   ecx
                dec     cl
                dec     cl
                sub     cl, 10h
                dec     cl
                dec     cl
                add     cl, 0Ch
                dec     cl
                dec     cl
                dec     cl
                jo      short loc_48693A
                jl      short loc_486938

loc_486935:
                jmp     short loc_48693C

loc_486938:
                jz      short loc_486935

loc_48693A:
                jmp     short loc_486935

loc_48693C:
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 10h
                dec     esi
                inc     edi
                sub     cl, 1
                dec     cl
                dec     cl
                dec     cl
                dec     edi
                dec     cl
                dec     cl
                dec     esi
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                and     al, cl
                pop     ecx
                pop     ebx
                test    eax, eax
                jnz     loc_486A24
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 800h
                jmp     short loc_486987
                mov     ebx, 80h

loc_486987:
                mov     ebx, 72h
                not     ebx
                bswap   eax
                not     ebx
                inc     ebx
                inc     ebx
                add     ebx, 8
                dec     ebx
                push    ecx
                mov     ecx, 4
                add     ebx, ecx
                inc     ebx
                pop     ecx
                bswap   eax
                and     eax, ebx
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_4869CA
                jl      short loc_4869C8

loc_4869C3:
                jmp     short loc_4869CC

loc_4869C8:
                jz      short loc_4869C3

loc_4869CA:
                jmp     short loc_4869C3

loc_4869CC:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                and     eax, 41h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1500h
                dec     dh
                sub     dh, 3
                dec     dh
                sub     dh, 7
                dec     dh
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jo      short loc_486A0E
                jl      short loc_486A0C

loc_486A07:
                jmp     short loc_486A10

loc_486A0C:
                jz      short loc_486A07

loc_486A0E:
                jmp     short loc_486A07

loc_486A10:
                jnz     short loc_486A24
                jo      short loc_486A1D
                jl      short loc_486A1B

loc_486A16:
                jmp     short loc_486A1F

loc_486A1B:
                jz      short loc_486A16

loc_486A1D:
                jmp     short loc_486A16

loc_486A1F:
                and     eax, 0
                jmp     short loc_486A28

loc_486A24:
```

```
                and     eax, 0
                inc     eax

 loc_486A28:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9394
                xor     ecx, ds:dword_4D9398
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_486A4B
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_486A4B:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_48C8AB(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD00
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
```

```
mov     ch, 2Dh
dec     ch
sub     ch, 1
sub     ch, 20h
dec     ch
dec     ch
sub     ch, 7
dec     ch
dec     ch
and     ah, ch
mov     cl, 77h
sub     cl, 2
dec     cl
dec     cl
dec     cl
not     cl
bswap   edx
not     cl
bswap   edx
dec     cl
dec     cl
push    eax
dec     cl
dec     cl
sub     cl, 12h
dec     cl
dec     cl
and     eax, 40h
dec     cl
dec     cl
dec     cl
add     cl, 0Eh
dec     cl
dec     cl
and     eax, 80h
sub     cl, 1Fh
dec     cl
dec     cl
dec     cl
not     ecx
bswap   eax
not     ecx
bswap   eax
pop     eax
and     al, cl
mov     eax, eax
pop     ecx
neg     eax
sbb     eax, eax
inc     eax
pop     ebx
mov     [ebp-0Ch], eax
```

```
                        mov     ecx, ds:dword_4D9394
                        xor     ecx, ds:dword_4D9398
                        shl     ecx, 1
                        mov     [ebp-8], ecx
                        cmp     dword ptr [ebp-0Ch], 0
                        jz      short loc_48C95F
                        mov     edx, [ebp-8]
                        or      edx, 1
                        mov     [ebp-8], edx

 loc_48C95F:
                        mov     eax, [ebp-8]
                        push    eax
                        call    ds:off_4DDCC0
                        add     esp, 4
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
     }
}



__declspec(naked) void sub_483C44(void)
{
     __asm
     {
                        push    ebp
                        mov     ebp, esp

// loc_483C47:
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDD0C
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    ebx
                        mov     ebx, 0FFFFh
                        and     eax, ebx
                        push    ecx
                        mov     ch, 2Ch
                        sub     ch, 1
                        sub     ch, 20h
```

```
dec     ch
dec     ch
sub     ch, 4
dec     ch
sub     ch, 3
dec     ch
and     ah, ch
mov     cl, 72h
sub     cl, 2
dec     cl
dec     cl
dec     cl
sub     cl, 6
not     al
bswap   ecx
not     al
bswap   ecx
dec     cl
dec     cl
sub     cl, 10h
dec     cl
dec     cl
add     cl, 0Ch
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
sub     cl, 10h
sub     cl, 1
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
not     ecx
bswap   eax
not     ecx
bswap   eax
inc     cl
add     cl, 2
and     al, cl
mov     eax, eax
pop     ecx
neg     eax
sbb     eax, eax
```

```
                 neg     eax
                 pop     ebx
                 mov     [ebp-0Ch], eax
                 mov     ecx, ds:dword_4D93A0
                 xor     ecx, ds:dword_4D93A4
                 shl     ecx, 1
                 mov     [ebp-8], ecx
                 cmp     dword ptr [ebp-0Ch], 0
                 jz      short loc_483D07
                 mov     edx, [ebp-8]
                 or      edx, 1
                 mov     [ebp-8], edx

 loc_483D07:
                 mov     eax, [ebp-8]
                 push    eax
                 call    ds:off_4DDCCC
                 add     esp, 4
                 pop     edi
                 pop     esi
                 pop     ebx
                 mov     esp, ebp
                 pop     ebp
                 retn
    }
}



__declspec(naked) void sub_48CE93(void)
{
    __asm
    {
                 push    ebp
                 mov     ebp, esp
                 sub     esp, 0Ch
                 push    ebx
                 push    esi
                 push    edi
                 mov     eax, [ebp+8]
                 push    eax
                 call    ds:off_4DDCEC
                 add     esp, 4
                 mov     [ebp-4], eax
                 mov     eax, [ebp-4]
                 push    ebx
                 mov     ebx, 0FFFFh
                 and     eax, ebx
                 push    ecx
                 push    4
                 pop     ecx
                 dec     cl
```

```
                dec     cl
                dec     cl
                dec     cl
                and     al, cl
                mov     bh, 0Fh
                and     bl, 0
                dec     bh
                sub     bh, 3
                dec     bh
                sub     bh, 1
                dec     bh
                and     ah, bh
                pop     ecx
                pop     ebx
                test    eax, eax
                jz      short loc_48CEE6
                not     eax
                add     eax, 1
                stc
                jmp     short loc_48CEEC

loc_48CEE6:
                not     eax
                add     eax, 1
                clc

loc_48CEEC:
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9380
                xor     ecx, ds:dword_4D9384
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48CF13
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48CF13:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCAC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
```

```
}



__declspec(naked) void sub_4898DB(void)
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 sub     esp, 0Ch
                 push    ebx
                 push    esi
                 push    edi
                 mov     eax, [ebp+8]
                 push    eax
                 call    ds:off_4DDD14
                 add     esp, 4
                 mov     [ebp-4], eax
                 mov     eax, [ebp-4]
                 push    edx
                 mov     edx, 0FFFFh
                 and     eax, edx
                 push    ebx
                 push    0Fh
                 pop     ebx
                 dec     bl
                 dec     bl
                 dec     bl
                 dec     bl
                 sub     bl, 2
                 add     bl, 0FFh
                 dec     bl
                 dec     bl
                 add     bl, 0FFh
                 add     bl, 0FFh
                 dec     bl
                 sub     bl, 1
                 add     bl, 0FFh
                 add     bl, 0FFh
                 and     al, bl
                 mov     dh, 14h
                 and     dl, 0
                 dec     dh
                 sub     dh, 2
                 dec     dh
                 dec     dh
                 sub     dh, 1
                 dec     dh
                 dec     dh
                 inc     dh
```

```
                dec     dh
                dec     dh
                inc     dh
                dec     dh
                inc     dh
                dec     dh
                dec     dh
                inc     dh
                dec     dh
                dec     dh
                dec     dh
                and     ah, dh
                pop     ebx
                pop     edx
                test    eax, eax
                jz      short loc_489965
                not     eax
                add     eax, 1
                stc
                jmp     short loc_48996B

loc_489965:
                not     eax
                add     eax, 1
                clc

loc_48996B:
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A8
                xor     ecx, ds:dword_4D93AC
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_489992
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_489992:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
```

```
}



__declspec(naked) void sub_4806A8(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF0
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     dh, 2
                dec     dh
                dec     dh
                and     ah, dh
                mov     dl, 0Eh
                sub     dl, 0FFh
                jo      short loc_4806DB
                jl      short loc_4806D9

 loc_4806D6:
                jmp     short loc_4806DD

 loc_4806D9:
                jz      short loc_4806D6

 loc_4806DB:
                jmp     short loc_4806D6

 loc_4806DD:
                sub     dl, 0FFh
                sub     dl, 0FFh
                sub     dl, 0Ah
                sub     dl, 0FFh
                sub     dl, 0FFh
                sub     dl, 5
                dec     dl
                jo      short loc_4806FA
                jl      short loc_4806F8

 loc_4806F5:
```

```
                jmp     short loc_4806FC
loc_4806F8:
                jz      short loc_4806F5

loc_4806FA:
                jmp     short loc_4806F5

loc_4806FC:
                dec     dl
                dec     dl
                sub     dl, 3
                sub     dl, 0FFh
                dec     dl
                inc     dl
                inc     dl
                inc     dl
                jo      short loc_480717
                jl      short loc_480715

loc_480712:
                jmp     short loc_480719

loc_480715:
                jz      short loc_480712

loc_480717:
                jmp     short loc_480712

loc_480719:
                and     al, dl
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9384
                xor     ecx, ds:dword_4D9388
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48073F
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48073F:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
```

```
                    retn
     }
}




__declspec(naked) void sub_48839F(void)
{
     __asm
     {
                    push    ebp
                    mov     ebp, esp
                    sub     esp, 0Ch
                    push    ebx
                    push    esi
                    push    edi
                    mov     eax, [ebp+8]
                    push    eax
                    call    ds:off_4DDD0C
                    add     esp, 4
                    mov     [ebp-4], eax
                    mov     eax, [ebp-4]
                    push    ebx
                    mov     ebx, 0FFFFh
                    and     eax, ebx
                    push    ecx
                    mov     ch, 2Dh
                    dec     ch
                    sub     ch, 1
                    sub     ch, 20h
                    dec     ch
                    dec     ch
                    sub     ch, 7
                    dec     ch
                    dec     ch
                    and     ah, ch
                    mov     cl, 77h
                    sub     cl, 2
                    dec     cl
                    dec     cl
                    dec     cl
                    dec     cl
                    not     cl
                    bswap   edx
                    not     cl
                    bswap   edx
                    dec     cl
                    dec     cl
                    push    eax
                    dec     cl
                    dec     cl
```

```
                sub     cl, 12h
                dec     cl
                jo      short loc_488407
                jl      short loc_488405

loc_488402:
                jmp     short loc_488409

loc_488405:
                jz      short loc_488402

loc_488407:
                jmp     short loc_488402

loc_488409:
                dec     cl
                and     eax, 40h
                dec     cl
                dec     cl
                dec     cl
                add     cl, 0Eh
                dec     cl
                dec     cl
                and     eax, 80h
                sub     cl, 1Fh
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                pop     eax
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A0
                xor     ecx, ds:dword_4D93A4
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_488461
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_488461:
```

```
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCCC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_484D34(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD14
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                jo      short loc_484D59
                jl      short loc_484D57

 loc_484D54:
                jmp     short loc_484D5B

 loc_484D57:
                jz      short loc_484D54

 loc_484D59:
                jmp     short loc_484D54

 loc_484D5B:
                push    edx
                jo      short loc_484D65
                jl      short loc_484D63

 loc_484D60:
                jmp     short loc_484D67
```

```
loc_484D63:
                jz      short loc_484D60

loc_484D65:
                jmp     short loc_484D60

loc_484D67:
                mov     dh, 6
                jo      short loc_484D72
                jl      short loc_484D70

loc_484D6D:
                jmp     short loc_484D74

loc_484D70:
                jz      short loc_484D6D

loc_484D72:
                jmp     short loc_484D6D

loc_484D74:
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                and     ah, dh
                mov     dl, 2
                dec     dl
                and     al, dl
                not     ah
                not     ah
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A8
                xor     ecx, ds:dword_4D93AC
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_484DB0
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_484DB0:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_48C5F7(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD08
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 87h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D939C
                xor     ecx, ds:dword_4D93A0
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48C68D
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48C68D:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_48903D(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
```

```
push    eax
call    ds:off_4DDD00
add     esp, 4
mov     [ebp-4], eax
mov     eax, [ebp-4]
push    ebx
mov     ebx, 0FFFFh
and     eax, ebx
push    ecx
mov     ch, 2Dh
dec     ch
sub     ch, 1
sub     ch, 20h
dec     ch
dec     ch
sub     ch, 7
dec     ch
dec     ch
and     ah, ch
mov     cl, 0BDh
sub     cl, 2
sub     cl, 6
dec     cl
not     cl
bswap   edx
not     cl
bswap   edx
dec     cl
dec     cl
sub     cl, 3
dec     cl
dec     cl
dec     cl
push    eax
dec     cl
dec     cl
sub     cl, 4
inc     cl
inc     cl
dec     cl
dec     cl
sub     cl, 11h
dec     cl
and     eax, 10h
dec     cl
dec     cl
dec     cl
add     cl, 0Fh
dec     cl
dec     cl
and     eax, 80h
sub     cl, 1Fh
```

```
                dec     cl
                dec     cl
                inc     cl
                dec     cl
                dec     cl
                inc     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                pop     eax
                inc     cl
                inc     cl
                inc     cl
                add     cl, 2
                dec     cl
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9394
                xor     ecx, ds:dword_4D9398
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_489116
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_489116:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}
```

```
__declspec(naked) void sub_482AC5(void)
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 sub     esp, 0Ch
                 push    ebx
                 push    esi
                 push    edi
                 mov     eax, [ebp+8]
                 push    eax
                 call    ds:off_4DDD04
                 add     esp, 4
                 mov     [ebp-4], eax
                 mov     eax, [ebp-4]
                 jo      short loc_482AEA
                 jl      short loc_482AE8

 loc_482AE5:
                 jmp     short loc_482AEC

 loc_482AE8:
                 jz      short loc_482AE5

 loc_482AEA:
                 jmp     short loc_482AE5

 loc_482AEC:
                 push    edx
                 mov     dh, 2
                 jo      short loc_482AF8
                 jl      short loc_482AF6

 loc_482AF3:
                 jmp     short loc_482AFA

 loc_482AF6:
                 jz      short loc_482AF3

 loc_482AF8:
                 jmp     short loc_482AF3

 loc_482AFA:
                 dec     dh
                 dec     dh
                 and     ah, dh
                 mov     dl, 1
                 and     al, dl
                 not     ah
                 not     ah
```

```
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_482B2C
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_482B2C:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_4879C2(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF0
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Dh
                dec     ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 7
                dec     ch
                dec     ch
                and     ah, ch
                mov     cl, 77h
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     cl
                bswap   edx
                not     cl
                bswap   edx
                dec     cl
                dec     cl
                push    eax
                dec     cl
                dec     cl
                sub     cl, 12h
                dec     cl
                jo      short loc_487A2A
                jl      short loc_487A28

loc_487A25:
                jmp     short loc_487A2C

loc_487A28:
                jz      short loc_487A25

loc_487A2A:
                jmp     short loc_487A25

loc_487A2C:
                dec     cl
                and     eax, 40h
                dec     cl
                dec     cl
                dec     cl
                add     cl, 0Eh
                dec     cl
                dec     cl
                and     eax, 800h
                sub     cl, 1Fh
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
```

```
not     ecx
bswap   eax
pop     eax
and     al, cl
mov     eax, eax
pop     ecx
pop     ebx
test    eax, eax
jnz     loc_487B0A
mov     eax, [ebp-4]
push    edx
mov     edx, 0FFFFh
and     eax, edx
push    ebx
push    eax
mov     bh, 7
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
and     eax, 800h
bswap   ecx
pop     eax
bswap   ecx
and     ah, bh
mov     bl, 98h
sub     bl, 5
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
sub     bl, 0Ch
not     bx
bswap   eax
not     bx
bswap   eax
and     al, bl
mov     eax, eax
pop     ebx
neg     eax
sbb     eax, eax
inc     eax
pop     edx
mov     ecx, eax
push    ecx
mov     eax, [ebp-4]
```

```
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                push    4
                pop     ecx
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                and     al, cl
                mov     bh, 0Fh
                and     bl, 0
                dec     bh
                sub     bh, 3
                dec     bh
                sub     bh, 1
                dec     bh
                and     ah, bh
                pop     ecx
                pop     ebx
                test    eax, eax
                jz      short loc_487AF4
                not     eax
                add     eax, 1
                stc
                jmp     short loc_487AFA

loc_487AF4:
                not     eax
                add     eax, 1
                clc

loc_487AFA:
                sbb     eax, eax
                add     eax, 1
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_487B0A
                and     eax, 0
                inc     eax
                jmp     short loc_487B0D

loc_487B0A:
                and     eax, 0

loc_487B0D:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9384
                xor     ecx, ds:dword_4D9388
                shl     ecx, 1
                mov     [ebp-8], ecx
```

```
                 cmp     dword ptr [ebp-0Ch], 0
                 jz      short loc_487B30
                 mov     edx, [ebp-8]
                 or      edx, 1
                 mov     [ebp-8], edx

 loc_487B30:
                 mov     eax, [ebp-8]
                 push    eax
                 call    ds:off_4DDCB0
                 add     esp, 4
                 pop     edi
                 pop     esi
                 pop     ebx
                 mov     esp, ebp
                 pop     ebp
                 retn
     }
}



__declspec(naked) void sub_4848BA(void)
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 sub     esp, 0Ch
                 push    ebx
                 push    esi
                 push    edi
                 mov     eax, [ebp+8]
                 push    eax
                 call    ds:off_4DDCE4
                 add     esp, 4
                 mov     [ebp-4], eax
                 mov     eax, [ebp-4]
                 jo      short loc_4848DF
                 jl      short loc_4848DD

 loc_4848DA:
                 jmp     short loc_4848E1

 loc_4848DD:
                 jz      short loc_4848DA

 loc_4848DF:
                 jmp     short loc_4848DA

 loc_4848E1:
```

```
                        push    edx
                        mov     dh, 2
                        jo      short loc_4848ED
                        jl      short loc_4848EB

loc_4848E8:
                        jmp     short loc_4848EF

loc_4848EB:
                        jz      short loc_4848E8

loc_4848ED:
                        jmp     short loc_4848E8

loc_4848EF:
                        dec     dh
                        dec     dh
                        and     ah, dh
                        mov     dl, 1
                        and     al, dl
                        not     ah
                        not     ah
                        pop     edx
                        neg     eax
                        sbb     eax, eax
                        inc     eax
                        mov     [ebp-0Ch], eax
                        mov     ecx, ds:dword_4D9378
                        xor     ecx, ds:dword_4D937C
                        shl     ecx, 1
                        mov     [ebp-8], ecx
                        cmp     dword ptr [ebp-0Ch], 0
                        jz      short loc_484926
                        mov     edx, [ebp-8]
                        or      edx, 1
                        mov     [ebp-8], edx

loc_484926:
                        mov     eax, [ebp-8]
                        push    eax
                        call    ds:off_4DDCA4
                        add     esp, 4
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
    }
}
```

```
__declspec(naked) void sub_487CA9(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCEC
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 98h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 0Ch
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9380
                xor     ecx, ds:dword_4D9384
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_487D39
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_487D39:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCAC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48565C(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD04
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
```

```
push    ecx
mov     ch, 2Ch
sub     ch, 1
sub     ch, 20h
dec     ch
dec     ch
sub     ch, 4
dec     ch
sub     ch, 3
dec     ch
and     ah, ch
mov     cl, 70h
sub     cl, 2
dec     cl
dec     cl
dec     cl
sub     cl, 6
not     al
bswap   ecx
not     al
bswap   ecx
dec     cl
dec     cl
sub     cl, 10h
dec     cl
dec     cl
add     cl, 0Ch
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
sub     cl, 10h
sub     cl, 1
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
not     ecx
bswap   eax
not     ecx
bswap   eax
inc     cl
add     cl, 2
and     al, cl
mov     eax, eax
pop     ecx
```

```
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48571B
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48571B:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_485C4D(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 8
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, 800h
                push    ecx
                mov     ch, 41h
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
```

```
                                sub     ch, 4
                                dec     ch
                                sub     ch, 3
                                dec     ch
                                mov     ebx, [ebp+0Ch]
                                dec     esi
                                dec     edi
                                dec     edi
                                xor     edx, edx
                                or      ebx, edx
                                jz      short loc_485C8D
                                dec     edi
                                and     eax, 0
                                jmp     short loc_485C95

                loc_485C8D:
                                dec     edi
                                dec     ecx
                                and     eax, 0
                                dec     ecx
                                dec     edx
                                inc     eax

                loc_485C95:
                                mov     [ebp-8], eax
                                mov     eax, ds:dword_4D9398
                                xor     eax, ds:dword_4D939C
                                shl     eax, 1
                                mov     [ebp-4], eax
                                cmp     dword ptr [ebp-8], 0
                                jz      short loc_485CB7
                                mov     ecx, [ebp-4]
                                or      ecx, 1
                                mov     [ebp-4], ecx

                loc_485CB7:
                                mov     edx, [ebp-4]
                                push    edx
                                call    ds:off_4DDCC4
                                add     esp, 4
                                pop     edi
                                pop     esi
                                pop     ebx
                                mov     esp, ebp
                                pop     ebp
                                retn
                    }
               }
```

```
__declspec(naked) void sub_48AAED(void)
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 sub     esp, 0Ch
                 push    ebx
                 push    esi
                 push    edi
                 mov     eax, [ebp+8]
                 push    eax
                 call    ds:off_4DDCFC
                 add     esp, 4
                 mov     [ebp-4], eax
                 mov     eax, [ebp-4]
                 push    ebx
                 mov     ebx, 0FFFFh
                 and     eax, ebx
                 push    ecx
                 mov     ch, 2Dh
                 dec     ch
                 sub     ch, 1
                 sub     ch, 20h
                 dec     ch
                 dec     ch
                 sub     ch, 7
                 dec     ch
                 dec     ch
                 and     ah, ch
                 mov     cl, 77h
                 sub     cl, 2
                 dec     cl
                 dec     cl
                 dec     cl
                 not     cl
                 bswap   edx
                 not     cl
                 bswap   edx
                 dec     cl
                 dec     cl
                 push    eax
                 dec     cl
                 dec     cl
                 sub     cl, 12h
                 dec     cl
                 dec     cl
                 and     eax, 40h
                 dec     cl
                 dec     cl
                 dec     cl
```

```
                add     cl, 0Eh
                dec     cl
                dec     cl
                and     eax, 80h
                sub     cl, 1Fh
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                pop     eax
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9390
                xor     ecx, ds:dword_4D9394
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48ABA2
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48ABA2:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCBC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_482867(void)
{
    __asm
    {
```

```
push    ebp
mov     ebp, esp
sub     esp, 0Ch
push    ebx
push    esi
push    edi
mov     eax, [ebp+8]
push    eax
call    ds:off_4DDD00
add     esp, 4
mov     [ebp-4], eax
mov     eax, [ebp-4]
push    eax
mov     eax, 4
bswap   eax
not     eax
pop     eax
push    edx
mov     dh, 80h
mov     dh, 0
inc     dh
mov     ecx, ecx
inc     dh
inc     dh
inc     dh
inc     dh
push    ebx
inc     dh
push    ecx
bswap   ecx
not     ecx
push    eax
not     eax
mov     eax, 800h
xchg    eax, ecx
mov     ecx, 40h
xchg    eax, ecx
not     eax
pop     eax
not     ecx
pop     ecx
inc     dh
inc     dh
and     ebx, 800h
inc     dh
inc     dh
inc     dh
inc     dh
and     ebx, 10h
inc     dh
inc     dh
pop     ebx
```

```
                sub     dh, 0Dh
                dec     dh
                and     ah, dh
                mov     dl, 5
                sub     dl, 0FFh
                dec     dl
                dec     dl
                dec     dl
                sub     dl, 0FFh
                dec     dl
                dec     dl
                dec     dl
                and     al, dl
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9394
                xor     ecx, ds:dword_4D9398
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48291C
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48291C:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_484BCA(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
```

```
push    ebx
push    esi
push    edi
mov     eax, [ebp+8]
push    eax
call    ds:off_4DDD08
add     esp, 4
mov     [ebp-4], eax
mov     eax, [ebp-4]
push    ecx
mov     ecx, 800h
mov     ecx, 4Bh
not     ecx
bswap   eax
not     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
dec     ecx
inc     ecx
inc     cl
inc     cl
inc     cl
add     ecx, 0Dh
inc     cl
inc     cl
inc     cl
inc     cl
inc     cl
add     ecx, 0Ah
dec     ecx
push    edx
mov     edx, 4
add     ecx, edx
inc     ecx
pop     edx
bswap   eax
add     ecx, 3
and     eax, ecx
pop     ecx
```

```
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                push    eax
                mov     eax, [ebp-4]
                mov     edx, 0E00h
                sub     dh, 1
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                and     eax, edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                cmp     eax, edx
                setz    cl
                mov     al, cl
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D939C
                xor     ecx, ds:dword_4D93A0
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_484C83
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_484C83:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_483549(void)
```

```
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 sub     esp, 0Ch
                 push    ebx
                 push    esi
                 push    edi
                 mov     eax, [ebp+8]
                 push    eax
                 call    ds:off_4DDD00
                 add     esp, 4
                 mov     [ebp-4], eax
                 mov     eax, [ebp-4]
                 push    edx
                 mov     edx, 0FFFFh
                 and     eax, edx
                 push    ebx
                 push    100h
                 pop     ebx
                 dec     bh
                 jo      short loc_48357F
                 jl      short loc_48357D

 loc_48357A:
                 jmp     short loc_483581

 loc_48357D:
                 jz      short loc_48357A

 loc_48357F:
                 jmp     short loc_48357A

 loc_483581:
                 add     bh, 0FFh
                 add     bh, 0FFh
                 add     bh, 0FFh
                 add     bh, 0FFh
                 inc     bh
                 inc     bh
                 inc     bh
                 inc     bh
                 and     ah, bh
                 jo      short loc_4835A0
                 jl      short loc_48359E

 loc_48359B:
                 jmp     short loc_4835A2

 loc_48359E:
                 jz      short loc_48359B
```

```
loc_4835A0:
                jmp     short loc_48359B

loc_4835A2:
                mov     bl, 15h
                dec     bl
                sub     bl, 6
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                and     al, bl
                pop     ebx
                pop     edx
                test    eax, eax
                jz      short loc_4835CE
                not     eax
                add     eax, 1
                stc
                jmp     short loc_4835D4

loc_4835CE:
                not     eax
                add     eax, 1
                clc

loc_4835D4:
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_4835E3
                jl      short loc_4835E1

loc_4835DC:
                jmp     short loc_4835E5

loc_4835E1:
                jz      short loc_4835DC

loc_4835E3:
                jmp     short loc_4835DC

loc_4835E5:
                inc     eax
                dec     eax
                jo      short loc_4835F2
                jl      short loc_4835F0

loc_4835EB:
                jmp     short loc_4835F4

loc_4835F0:
                jz      short loc_4835EB

loc_4835F2:
                jmp     short loc_4835EB

loc_4835F4:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_483603
                jl      short loc_483601

loc_4835FC:
                jmp     short loc_483605

loc_483601:
                jz      short loc_4835FC

loc_483603:
                jmp     short loc_4835FC

loc_483605:
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9394
                xor     ecx, ds:dword_4D9398
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_483629
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_483629:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
```

```
                        retn
            }
        }




        __declspec(naked) void sub_480ABC(void)
        {
            __asm
            {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 8
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    ebx
                        mov     ebx, 4
                        and     eax, ebx
                        push    ecx
                        mov     ch, 10h
                        sub     ch, 1
                        dec     ch
                        sub     ch, 3
                        dec     ch
                        mov     ebx, [ebp+0Ch]
                        dec     esi
                        dec     edi
                        dec     edi
                        xor     edx, edx
                        or      ebx, edx
                        jz      short loc_480AF9
                        dec     edi
                        sub     ch, 2
                        dec     ch
                        dec     ch
                        sub     ch, 8
                        and     eax, 0
                        jmp     short loc_480B10

         loc_480AF9:
                        dec     edi
                        dec     ecx
                        sub     ch, 2
                        dec     ch
                        dec     ch
                        sub     ch, 8
                        and     eax, 0
                        dec     ecx
```

```
                sub     ch, 2
                dec     ch
                dec     edx
                inc     eax

loc_480B10:
                mov     [ebp-8], eax
                mov     eax, ds:dword_4D9394
                xor     eax, ds:dword_4D9398
                shl     eax, 1
                mov     [ebp-4], eax
                cmp     dword ptr [ebp-8], 0
                jz      short loc_480B32
                mov     ecx, [ebp-4]
                or      ecx, 1
                mov     [ebp-4], ecx

loc_480B32:
                mov     edx, [ebp-4]
                push    edx
                call    ds:off_4DDCC0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_48654F(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF8
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 87h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D938C
                xor     ecx, ds:dword_4D9390
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4865E4
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_4865E4:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_489214(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 8
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                jo      short loc_489229
                jl      short loc_489227

 loc_489224:
                jmp     short loc_48922B

 loc_489227:
                jz      short loc_489224

 loc_489229:
                jmp     short loc_489224

 loc_48922B:
                mov     ebx, 4
                and     eax, ebx
                mov     ch, 52h
                dec     ch
                mov     ebx, [ebp+0Ch]
                dec     esi
                dec     edi
                dec     edi
                xor     ecx, ecx
                or      ebx, ecx
                jz      short loc_48924D
                dec     edi
                sub     ch, 2
                dec     ch
                and     eax, 0
                jmp     short loc_489263

 loc_48924D:
                dec     edi
                dec     ecx
                sub     ch, 2
                dec     ch
                dec     ch
                sub     ch, 8
                and     eax, 0
                dec     ecx
                sub     ch, 2
                inc     eax
                dec     ch

 loc_489263:
                mov     [ebp-8], eax
                mov     eax, ds:dword_4D93A8
                xor     eax, ds:dword_4D93AC
                shl     eax, 1
                mov     [ebp-4], eax
                cmp     dword ptr [ebp-8], 0
                jz      short loc_489285
                mov     ecx, [ebp-4]
                or      ecx, 1
                mov     [ebp-4], ecx

 loc_489285:
                mov     edx, [ebp-4]
                push    edx
                call    ds:off_4DDCD4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48A749(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 8
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                dec     bh
                and     eax, 800h
                jo      short loc_48A765
                jl      short loc_48A763

loc_48A760:
                jmp     short loc_48A767

loc_48A763:
                jz      short loc_48A760

loc_48A765:
                jmp     short loc_48A760

loc_48A767:
                mov     ebx, 4
                and     eax, ebx
                mov     ch, 52h
                dec     ch
                mov     ebx, [ebp+0Ch]
                xor     ecx, ecx
                or      ebx, ecx
                jz      short loc_48A786
                dec     edi
                sub     ch, 2
                dec     ch
                and     eax, 0
                jmp     short loc_48A7A7

loc_48A786:
                dec     edi
                dec     ecx
                sub     ch, 2
                dec     ch
                dec     ch
                sub     ch, 8
                jo      short loc_48A79B
                jl      short loc_48A799

loc_48A796:
                jmp     short loc_48A79D

loc_48A799:
                jz      short loc_48A796

loc_48A79B:
```

```
                jmp     short loc_48A796

loc_48A79D:
                and     eax, 0
                dec     ecx
                sub     ch, 2
                inc     eax
                dec     ch

loc_48A7A7:
                mov     [ebp-8], eax
                mov     eax, ds:dword_4D9370
                xor     eax, ds:dword_4D9374
                shl     eax, 1
                mov     [ebp-4], eax
                cmp     dword ptr [ebp-8], 0
                jz      short loc_48A7C9
                mov     ecx, [ebp-4]
                or      ecx, 1
                mov     [ebp-4], ecx

loc_48A7C9:
                mov     edx, [ebp-4]
                push    edx
                call    ds:off_4DDC9C
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_485280(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD10
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                mov     ecx, 800h
                mov     ecx, 0Ch
                not     ecx
                bswap   eax
                not     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                and     eax, 0
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Dh
                inc     cl
                inc     cl
                inc     cl
                inc     eax
                inc     cl
                inc     cl
                add     ecx, 0Ah
                dec     ecx
                push    edx
                mov     edx, 4
                add     ecx, edx
                inc     ecx
                pop     edx
                pop     ecx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A4
                xor     ecx, ds:dword_4D93A8
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_485301
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_485301:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD0
```

```
                        add     esp, 4
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
        }
}




__declspec(naked) void sub_482FB4(void)
{
        __asm
        {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDD00
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        jo      short loc_482FD9
                        jl      short loc_482FD7

 loc_482FD4:
                        jmp     short loc_482FDB

 loc_482FD7:
                        jz      short loc_482FD4

 loc_482FD9:
                        jmp     short loc_482FD4

 loc_482FDB:
                        push    ebx
                        mov     ebx, 0FFFFh
                        and     eax, ebx
                        push    ecx
                        mov     ch, 2Ch
                        sub     ch, 1
                        sub     ch, 20h
                        dec     ch
                        dec     ch
                        sub     ch, 4
```

```
                dec     ch
                sub     ch, 3
                dec     ch
                and     ah, ch
                mov     cl, 70h
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 6
                not     al
                bswap   ecx
                not     al
                bswap   ecx
                dec     cl
                dec     cl
                sub     cl, 10h
                dec     cl
                dec     cl
                add     cl, 0Ch
                dec     cl
                dec     cl
                dec     cl
                jo      short loc_48302F
                jl      short loc_48302D

loc_48302A:
                jmp     short loc_483031

loc_48302D:
                jz      short loc_48302A

loc_48302F:
                jmp     short loc_48302A

loc_483031:
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 10h
                sub     cl, 1
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                and     al, cl
                pop     ecx
                pop     ebx
                test    eax, eax
                jnz     loc_4830FC
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 800h
                jmp     short loc_483078
                mov     ebx, 80h

loc_483078:
                mov     ebx, 72h
                not     ebx
                bswap   eax
                not     ebx
                inc     ebx
                inc     ebx
                add     ebx, 8
                dec     ebx
                push    ecx
                mov     ecx, 4
                add     ebx, ecx
                inc     ebx
                pop     ecx
                bswap   eax
                and     eax, ebx
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_4830BB
                jl      short loc_4830B9

loc_4830B4:
                jmp     short loc_4830BD

loc_4830B9:
```

```
                jz      short loc_4830B4

loc_4830BB:
                jmp     short loc_4830B4

loc_4830BD:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                and     eax, 41h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1500h
                dec     dh
                sub     dh, 3
                dec     dh
                sub     dh, 7
                dec     dh
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_4830FC
                and     eax, 0
                inc     eax
                jmp     short loc_4830FF

loc_4830FC:
                and     eax, 0

loc_4830FF:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9394
                xor     ecx, ds:dword_4D9398
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_483122
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx
```

```
 loc_483122:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_482EFE(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF0
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1E00h
                pop     ebx
                jo      short loc_482F34
                jl      short loc_482F32

 loc_482F2D:
                jmp     short loc_482F36

 loc_482F32:
                jz      short loc_482F2D

 loc_482F34:
                jmp     short loc_482F2D

 loc_482F36:
```

```
                sub     bh, 4
                dec     bh
                push    eax
                dec     bh
                dec     bh
                jo      short loc_482F4B
                jl      short loc_482F49

loc_482F44:
                jmp     short loc_482F4D

loc_482F49:
                jz      short loc_482F44

loc_482F4B:
                jmp     short loc_482F44

loc_482F4D:
                and     eax, 40h
                dec     bh
                sub     bh, 12h
                sub     bh, 3
                pop     eax
                dec     bh
                and     ah, bh
                mov     edx, 12h
                dec     dl
                sub     dl, 1
                dec     dl
                sub     dl, 7
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                and     al, dl
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9384
                xor     ecx, ds:dword_4D9388
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_482FA0
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_482FA0:
```

```
                        mov     eax, [ebp-8]
                        push    eax
                        call    ds:off_4DDCB0
                        add     esp, 4
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
        }
}



__declspec(naked) void sub_47FA7F(void)
{
        __asm
        {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDCFC
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    ebx
                        mov     ebx, 0FFFFh
                        and     eax, ebx
                        push    ecx
                        push    800h
                        pop     ecx
                        dec     ch
                        dec     ch
                        dec     ch
                        dec     ch
                        dec     ch
                        dec     ch
                        dec     ch
                        dec     ch
                        and     ah, ch
                        mov     cl, 10h
                        dec     dl
                        sub     cl, 2
                        dec     dl
                        sub     cl, 3
                        dec     cl
```

```
                dec     dl
                dec     cl
                dec     cl
                dec     dl
                dec     cl
                dec     dl
                dec     cl
                sub     cl, 1
                dec     cl
                and     al, cl
                pop     ecx
                pop     ebx
                test    eax, eax
                jz      short loc_47FAED
                not     eax
                add     eax, 1
                stc
                jmp     short loc_47FAF3

loc_47FAED:
                not     eax
                add     eax, 1
                clc

loc_47FAF3:
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9390
                xor     ecx, ds:dword_4D9394
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_47FB1A
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_47FB1A:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCBC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}
```

```
__declspec(naked) void sub_4803F9(void)
{
     __asm
     {

                    push    ebp
                    mov     ebp, esp
                    sub     esp, 0Ch
                    push    ebx
                    push    esi
                    push    edi
                    mov     eax, [ebp+8]
                    push    eax
                    call    ds:off_4DDCE4
                    add     esp, 4
                    mov     [ebp-4], eax
                    mov     eax, [ebp-4]
                    jo      short loc_48041E
                    jl      short loc_48041C

 loc_480419:
                    jmp     short loc_480420

 loc_48041C:
                    jz      short loc_480419

 loc_48041E:
                    jmp     short loc_480419

 loc_480420:
                    push    edx
                    mov     dh, 0Eh
                    dec     dh
                    dec     dh
                    dec     dh
                    dec     dh
                    dec     dh
                    dec     dh
                    dec     dh
                    dec     dh
                    dec     dh
                    dec     dh
                    jo      short loc_480440
                    jl      short loc_48043E

 loc_48043B:
                    jmp     short loc_480442
 loc_48043E:
                    jz      short loc_48043B
 loc_480440:
                jmp     short loc_48043B

 loc_480442:
                dec     dh
                sub     dh, 1
                add     dh, 0FEh
                and     ah, dh
                mov     dl, 1
                and     al, dl
                not     ah
                not     ah
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9378
                xor     ecx, ds:dword_4D937C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48047D
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48047D:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_485CCB(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF8
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, [ebp+0Ch]
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Dh
                dec     ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 7
                dec     ch
                dec     ch
                and     ah, ch
                mov     cl, 70h
                sub     cl, 2
                not     cl
                bswap   edx
                not     cl
                bswap   edx
                dec     cl
                dec     cl
                push    eax
                dec     cl
                dec     cl
                sub     cl, 12h
                dec     cl
                jo      short loc_485D2E
                jl      short loc_485D2C

loc_485D29:
                jmp     short loc_485D30

loc_485D2C:
                jz      short loc_485D29

loc_485D2E:
                jmp     short loc_485D29

loc_485D30:
                dec     cl
                and     eax, 40h
```

```
add     cl, 0Eh
dec     cl
dec     cl
and     eax, 800h
sub     cl, 1Fh
dec     cl
dec     cl
dec     cl
not     ecx
bswap   eax
not     ecx
bswap   eax
pop     eax
and     al, cl
mov     eax, eax
pop     ecx
pop     ebx
test    eax, eax
jnz     loc_485E0B
mov     eax, [ebp-4]
push    edx
mov     edx, 0FFFFh
and     eax, edx
push    ebx
push    eax
mov     bh, 8
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
and     eax, 800h
bswap   ecx
pop     eax
bswap   ecx
and     ah, bh
mov     bl, 98h
sub     bl, 5
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     edi
inc     esi
dec     bl
dec     bl
sub     bl, 0Ch
not     bx
```

```
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                push    4
                pop     ecx
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                and     al, cl
                mov     bh, 0Fh
                and     bl, 0
                dec     bh
                sub     bh, 3
                dec     bh
                sub     bh, 1
                dec     bh
                and     ah, bh
                pop     ecx
                pop     ebx
                test    eax, eax
                jz      short loc_485DF6
                not     eax
                add     eax, 1
                stc
                jmp     short loc_485DFC

loc_485DF6:
                not     eax
                add     eax, 1
                clc

loc_485DFC:
                sbb     eax, eax
                add     eax, 1
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_485E0B
                and     eax, 0
                jmp     short loc_485E0F

loc_485E0B:
                and     eax, 0
                inc     eax

loc_485E0F:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D938C
                xor     ecx, ds:dword_4D9390
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_485E32
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_485E32:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_48DAF1(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 8
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    ebx
                mov     ebx, [ebp+0Ch]
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 38h
                sub     ch, 0Bh
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                sub     ch, 4
                dec     ch
                sub     ch, 3
                dec     ch
                mov     ebx, [ebp+0Ch]
                test    ebx, ebx
                jz      short loc_48DB6D
                dec     edi
                and     eax, 0
                jmp     short loc_48DB74

loc_48DB6D:
                dec     edi
                and     eax, 0
```

```
                    dec     edi
                    dec     edi
                    inc     eax

     loc_48DB74:
                    mov     [ebp-8], eax
                    mov     eax, ds:dword_4D9384
                    xor     eax, ds:dword_4D9388
                    shl     eax, 1
                    mov     [ebp-4], eax
                    cmp     dword ptr [ebp-8], 0
                    jz      short loc_48DB96
                    mov     ecx, [ebp-4]
                    or      ecx, 1
                    mov     [ebp-4], ecx

     loc_48DB96:
                    mov     edx, [ebp-4]
                    push    edx
                    call    ds:off_4DDCB0
                    add     esp, 4
                    pop     edi
                    pop     esi
                    pop     ebx
                    mov     esp, ebp
                    pop     ebp
                    retn
     }
}



__declspec(naked) void sub_485A3D(void)
{
     __asm
     {
                    push    ebp
                    mov     ebp, esp
                    sub     esp, 0Ch
                    push    ebx
                    push    esi
                    push    edi
                    mov     eax, [ebp+8]
                    push    eax
                    call    ds:off_4DDCE8
                    add     esp, 4
                    mov     [ebp-4], eax
                    mov     eax, [ebp-4]
                    push    edx
                    mov     edx, 0FFFFh
                    and     eax, edx
```

```
                push    ebx
                push    0Eh
                pop     ebx
                sub     bl, 6
                dec     bl
                push    eax
                dec     bl
                dec     bl
                and     eax, 80h
                dec     bl
                sub     bl, 2
                dec     bl
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 2400h
                dec     dh
                sub     dh, 3
                dec     dh
                sub     dh, 16h
                dec     dh
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D937C
                xor     ecx, ds:dword_4D9380
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_485ABE
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_485ABE:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}
```

```
__declspec(naked) void sub_4840CF(void)
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 sub     esp, 0Ch
                 push    ebx
                 push    esi
                 push    edi
                 mov     eax, [ebp+8]
                 push    eax
                 call    ds:off_4DDCE0
                 add     esp, 4
                 mov     [ebp-4], eax
                 mov     eax, [ebp-4]
                 push    ebx
                 mov     ebx, [ebp+0Ch]
                 mov     ebx, 0FFFFh
                 and     eax, ebx
                 push    ecx
                 mov     ch, 2Dh
                 dec     ch
                 sub     ch, 1
                 sub     ch, 20h
                 dec     ch
                 dec     ch
                 sub     ch, 6
                 dec     ch
                 dec     ch
                 dec     ch
                 jo      short loc_484117
                 jl      short loc_484115

 loc_484112:
                 jmp     short loc_484119

 loc_484115:
                 jz      short loc_484112

 loc_484117:
                 jmp     short loc_484112

 loc_484119:
                 and     ah, ch
                 mov     cl, 87h
                 sub     cl, 12h
                 dec     cl
                 dec     cl
                 sub     cl, 2
                 not     cl
                 bswap   edx
                 not     cl
                 bswap   edx
                 dec     cl
                 dec     cl
                 push    eax
                 dec     cl
                 dec     cl
                 sub     cl, 12h
                 dec     cl
                 dec     cl
                 and     eax, 40h
                 dec     cl
                 dec     cl
                 dec     cl
                 add     cl, 0Eh
                 dec     cl
                 dec     cl
                 and     eax, 800h
                 sub     cl, 1Fh
                 dec     cl
                 dec     cl
                 dec     cl
                 not     ecx
                 bswap   eax
                 not     ecx
                 bswap   eax
                 pop     eax
                 and     al, cl
                 mov     eax, eax
                 pop     ecx
                 pop     ebx
                 test    eax, eax
                 jnz     loc_48421A
                 mov     eax, [ebp-4]
                 push    edx
                 mov     edx, 0FFFFh
                 and     eax, edx
                 push    ebx
                 push    eax
                 mov     bh, 7
                 dec     bh
                 dec     bh
                 dec     bh
                 dec     bh
                 dec     bh
                 dec     bh
                 dec     bh
                 and     eax, 800h
```

```
bswap   ecx
pop     eax
bswap   ecx
and     ah, bh
mov     bl, 98h
sub     bl, 5
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
sub     bl, 0Ch
not     bx
bswap   eax
not     bx
bswap   eax
and     al, bl
mov     eax, eax
pop     ebx
neg     eax
sbb     eax, eax
inc     eax
pop     edx
mov     ecx, eax
push    ecx
mov     eax, [ebp-4]
push    ebx
mov     ebx, 0FFFFh
and     eax, ebx
push    ecx
push    4
pop     ecx
dec     cl
dec     cl
dec     cl
dec     cl
and     al, cl
mov     bh, 0Fh
and     bl, 0
dec     bh
sub     bh, 3
dec     bh
sub     bh, 1
dec     bh
and     ah, bh
pop     ecx
pop     ebx
test    eax, eax
jz      short loc_484205
not     eax
```

```
                add     eax, 1
                stc
                jmp     short loc_48420B

loc_484205:
                not     eax
                add     eax, 1
                clc

loc_48420B:
                sbb     eax, eax
                add     eax, 1
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_48421A
                and     eax, 0
                jmp     short loc_48421E

loc_48421A:
                and     eax, 0
                inc     eax

loc_48421E:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9374
                xor     ecx, ds:dword_4D9378
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_484241
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_484241:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_487137(void)
{
```

```
__asm
{
            push    ebp
            mov     ebp, esp
            sub     esp, 0Ch
            push    ebx
            push    esi
            push    edi
            mov     eax, [ebp+8]
            push    eax
            call    ds:off_4DDCF8
            add     esp, 4
            mov     [ebp-4], eax
            mov     eax, [ebp-4]
            push    ecx
            mov     ecx, 800h
            mov     ecx, 0Ch
            not     ecx
            bswap   eax
            not     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            not     ecx
            not     ecx
            inc     ecx
            inc     ecx
            dec     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            dec     ecx
            inc     ecx
            inc     cl
            inc     cl
            inc     cl
            add     ecx, 0Dh
            inc     cl
            inc     cl
            inc     cl
            inc     cl
            inc     cl
```

```
add     ecx, 0Ah
dec     ecx
push    edx
mov     edx, 4
add     ecx, edx
inc     ecx
pop     edx
bswap   eax
and     eax, ecx
pop     ecx
pop     edx
test    eax, eax
jnz     loc_4872A2
mov     eax, [ebp-4]
push    ebx
mov     ebx, 0FFFFh
and     eax, ebx
push    ecx
mov     ch, 2Ch
sub     ch, 1
sub     ch, 20h
dec     ch
dec     ch
sub     ch, 4
dec     ch
sub     ch, 3
dec     ch
and     ah, ch
mov     cl, 0AEh
sub     cl, 2
dec     cl
dec     cl
sub     cl, 6
not     al
bswap   ecx
not     al
bswap   ecx
dec     cl
dec     cl
sub     cl, 10h
dec     cl
dec     cl
add     cl, 0Ch
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
sub     cl, 10h
sub     cl, 1
dec     cl
```

```
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                jo      short loc_48722C
                jl      short loc_48722A

loc_487225:
                jmp     short loc_48722E

loc_48722A:
                jz      short loc_487225

loc_48722C:
                jmp     short loc_487225

loc_48722E:
                and     al, cl
                pop     ecx
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_487254
                jl      short loc_487252

loc_48724D:
                jmp     short loc_487256

loc_487252:
                jz      short loc_48724D

loc_487254:
                jmp     short loc_48724D
```

```
loc_487256:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                and     eax, 41h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1500h
                dec     dh
                sub     dh, 7
                dec     dh
                sub     dh, 3
                dec     dh
                jo      short loc_48728C
                jl      short loc_48728A

loc_487285:
                jmp     short loc_48728E

loc_48728A:
                jz      short loc_487285

loc_48728C:
                jmp     short loc_487285

loc_48728E:
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_4872A2
                and     eax, 0
                inc     eax
                jmp     short loc_4872A5

loc_4872A2:
                and     eax, 0

loc_4872A5:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D938C
                xor     ecx, ds:dword_4D9390
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4872C8
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_4872C8:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_482930(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCDC_2
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                and     eax, 80h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1400h
                dec     dh
                sub     dh, 3
                dec     dh
                sub     dh, 6
                dec     dh
                jo      short loc_482990
                jl      short loc_48298E

loc_482989:
                jmp     short loc_482992

loc_48298E:
                jz      short loc_482989

loc_482990:
                jmp     short loc_482989

loc_482992:
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9370
                xor     ecx, ds:dword_4D9374
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4829BF
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_4829BF:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDC9C
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_48CAF8(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF4
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 8
                dec     bh
                dec     bh
                sub     bh, 2
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 80h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 86h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                pop     ebx
                pop     edx
                test    eax, eax
                jnz     loc_48CC65
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 1
                dec     bh
                and     eax, 41h
                bswap   ecx
                pop     eax
                bswap   ecx
                jo      short loc_48CB90
                jl      short loc_48CB8E

loc_48CB89:

                jmp     short loc_48CB92

loc_48CB8E:
                jz      short loc_48CB89

loc_48CB90:
                jmp     short loc_48CB89

loc_48CB92:
                and     ah, bh
                jo      short loc_48CB9F
                jl      short loc_48CB9D

loc_48CB98:

                jmp     short loc_48CBA1

loc_48CB9D:
                jz      short loc_48CB98

loc_48CB9F:
                jmp     short loc_48CB98
```

```
loc_48CBA1:
                mov     bl, 97h
                sub     bl, 3
                jo      short loc_48CBB1
                jl      short loc_48CBAF

loc_48CBAA:

                jmp     short loc_48CBB3

loc_48CBAF:
                jz      short loc_48CBAA

loc_48CBB1:
                jmp     short loc_48CBAA

loc_48CBB3:
                sub     bl, 0Ah
                dec     bl
                dec     bl
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_48CBEC
                jl      short loc_48CBEA

loc_48CBE5:

                jmp     short loc_48CBEE

loc_48CBEA:
                jz      short loc_48CBE5

loc_48CBEC:
                jmp     short loc_48CBE5
```

```
loc_48CBEE:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                and     eax, 40h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1200h
                dec     dh
                sub     dh, 1
                dec     dh
                jo      short loc_48CC1F
                jl      short loc_48CC1D

loc_48CC18:

                jmp     short loc_48CC21

loc_48CC1D:
                jz      short loc_48CC18

loc_48CC1F:
                jmp     short loc_48CC18

loc_48CC21:
                sub     dh, 7
                jo      short loc_48CC2F
                jl      short loc_48CC2D

loc_48CC28:

                jmp     short loc_48CC31

loc_48CC2D:
                jz      short loc_48CC28

loc_48CC2F:
                jmp     short loc_48CC28

loc_48CC31:
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_48CC46
                jl      short loc_48CC44

loc_48CC3F:

                jmp     short loc_48CC48

loc_48CC44:
                jz      short loc_48CC3F

loc_48CC46:
                jmp     short loc_48CC3F

loc_48CC48:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_48CC57
                jl      short loc_48CC55

loc_48CC52:

                jmp     short loc_48CC59

loc_48CC55:
                jz      short loc_48CC52

loc_48CC57:
                jmp     short loc_48CC52

loc_48CC59:
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_48CC65
                and     eax, 0
                inc     eax
                jmp     short loc_48CC68

loc_48CC65:

                and     eax, 0

loc_48CC68:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9388
                xor     ecx, ds:dword_4D938C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48CC8B
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48CC8B:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48D461(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD0C
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 80h
                jmp     short loc_48D48A
                mov     ebx, 4

 loc_48D48A:
                mov     ebx, 30h
                not     ebx
                bswap   eax
                not     ebx
                inc     ebx
                inc     ebx
                inc     ebx
                inc     ebx
                inc     ebx
                inc     ebx
                inc     ebx
                inc     ebx
                add     ebx, 4
                dec     ebx
                push    ecx
                mov     ecx, 4
                add     ebx, ecx
                inc     ebx
                pop     ecx
                bswap   eax
                and     eax, ebx
                pop     ebx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A0
                xor     ecx, ds:dword_4D93A4
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48D4DA
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48D4DA:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCCC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_484315(void)
{
    __asm
    {
```

```
push    ebp
mov     ebp, esp
sub     esp, 0Ch
push    ebx
push    esi
push    edi
mov     eax, [ebp+8]
push    eax
call    ds:off_4DDD04
add     esp, 4
mov     [ebp-4], eax
mov     eax, [ebp-4]
push    edx
mov     dh, 2
sub     dh, 0FFh
dec     dh
sub     dh, 0FFh
dec     dh
sub     dh, 0FFh
sub     dh, 1
sub     dh, 1
dec     dh
and     ah, dh
mov     edx, 800h
mov     dl, 0Fh
sub     dl, 0FFh
sub     dl, 0FFh
sub     dl, 0FFh
inc     dl
sub     dl, 0Ah
sub     dl, 0FFh
dec     dl
sub     dl, 0FFh
sub     dl, 5
inc     dl
dec     dl
dec     dl
dec     dl
dec     dl
sub     dl, 3
sub     dl, 0FFh
dec     dl
inc     dl
inc     dl
and     al, dl
not     ah
not     ah
pop     edx
neg     eax
sbb     eax, eax
inc     eax
mov     [ebp-0Ch], eax
```

```
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4843B0
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_4843B0:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_4833EE(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCEC
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 2
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 87h
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     edi
                dec     edi
                dec     bl
                dec     bl
                dec     bl
                sub     cl, 2
                dec     bl
                dec     cl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9380
                xor     ecx, ds:dword_4D9384
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_483489
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_483489:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCAC
                add     esp, 4
```

```
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
        }
}




__declspec(naked) void sub_48A015(void)
{
        __asm
        {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDD00
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    ebx
                        mov     ebx, 800h
                        jmp     short loc_48A03E
                        mov     ebx, 80h

 loc_48A03E:
                        mov     ebx, 6Eh
                        not     ebx
                        bswap   eax
                        not     ebx
                        inc     ebx
                        inc     ebx
                        dec     ebx
                        inc     ebx
                        inc     ebx
                        inc     ebx
                        inc     ebx
                        inc     ebx
                        add     ebx, 8
                        dec     ebx
                        push    ecx
                        mov     ecx, 5
                        add     ebx, ecx
```

```
                pop     ecx
                bswap   eax
                jo      short loc_48A069
                jl      short loc_48A067

loc_48A064:
                jmp     short loc_48A06B

loc_48A067:
                jz      short loc_48A064

loc_48A069:
                jmp     short loc_48A064

loc_48A06B:
                and     eax, ebx
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                push    eax
                mov     eax, [ebp-4]
                mov     edx, 0C00h
                dec     dh
                dec     dh
                dec     dh
                sub     dh, 0FFh
                dec     dh
                dec     dh
                and     eax, edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                cmp     eax, edx
                jo      short loc_48A0A1
                jl      short loc_48A09F

loc_48A09C:
                jmp     short loc_48A0A3

loc_48A09F:
                jz      short loc_48A09C

loc_48A0A1:
                jmp     short loc_48A09C

loc_48A0A3:
                jnz     short loc_48A0B5
                jo      short loc_48A0AE
                jl      short loc_48A0AC

loc_48A0A9:
                jmp     short loc_48A0B0

loc_48A0AC:
                jz      short loc_48A0A9

loc_48A0AE:
                jmp     short loc_48A0A9

loc_48A0B0:
                and     eax, 0
                jmp     short loc_48A0C4

loc_48A0B5:
                and     eax, 0
                jo      short loc_48A0C1
                jl      short loc_48A0BF

loc_48A0BC:
                jmp     short loc_48A0C3

loc_48A0BF:
                jz      short loc_48A0BC

loc_48A0C1:
                jmp     short loc_48A0BC

loc_48A0C3:
                inc     eax

loc_48A0C4:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9394
                xor     ecx, ds:dword_4D9398
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48A0E7
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48A0E7:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_488F97(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD14
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                mov     ecx, 800h
                mov     ecx, 0Ah
                not     ecx
                bswap   eax
                not     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Dh
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Ah
                dec     ecx
                push    edx
                mov     edx, 4
                add     ecx, edx
                inc     ecx
                pop     edx
                bswap   eax
                and     eax, ecx
                pop     ecx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A8
                xor     ecx, ds:dword_4D93AC
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_489029
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_489029:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}
```

```
__declspec(naked) void sub_483208(void)
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 sub     esp, 0Ch
                 push    ebx
                 push    esi
                 push    edi
                 mov     eax, [ebp+8]
                 push    eax
                 call    ds:off_4DDCF0
                 add     esp, 4
                 mov     [ebp-4], eax
                 mov     eax, [ebp-4]
                 push    edx
                 mov     edx, [ebp+0Ch]
                 mov     edx, 0FFFFh
                 and     eax, edx
                 push    ebx
                 push    eax
                 dec     bh
//                 ja      short $+2
                 dec     bh
                 dec     bh
                 dec     bh
                 and     eax, 41h
                 bswap   ecx
                 jo      short loc_483249
                 jl      short loc_483247

 loc_483244:
                 jmp     short loc_48324B

 loc_483247:
                 jz      short loc_483244

 loc_483249:
                 jmp     short loc_483244

 loc_48324B:
                 and     eax, 0
//                 jno     short $+2
                 mov     bl, 85h
                 sub     bl, 20h
                 dec     bl
                 dec     bl
                 sub     bl, 1Ah
                 dec     bl
                 sub     bl, 1Fh
```

```
                not     bx
                jo      short loc_48326D
                jl      short loc_48326B

 loc_483268:
                jmp     short loc_48326F

 loc_48326B:
                jz      short loc_483268

 loc_48326D:
                jmp     short loc_483268

 loc_48326F:
                inc     eax
                dec     bl
                dec     bl
                dec     bl
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9384
                xor     ecx, ds:dword_4D9388
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48329A
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48329A:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_488CA2(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE8
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, [ebp+0Ch]
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 4
                dec     ch
                sub     ch, 3
                dec     ch
                and     ah, ch
                mov     cl, 70h
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 6
                not     al
                bswap   ecx
                not     al
                bswap   ecx
                dec     cl
                dec     cl
                sub     cl, 10h
                dec     cl
                dec     cl
                add     cl, 0Ch
                dec     cl
                dec     cl
                dec     cl
                jo      short loc_488D15
                jl      short loc_488D13

loc_488D10:

                jmp     short loc_488D17
```

```
loc_488D13:
                jz      short loc_488D10

loc_488D15:
                jmp     short loc_488D10

loc_488D17:
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 10h
                sub     cl, 1
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                and     al, cl
                mov     eax, eax
                pop     ecx
                pop     ebx
                test    eax, eax
                jnz     loc_488E26
                mov     eax, [ebp-4]
                jo      short loc_488D5C
                jl      short loc_488D5A

loc_488D57:

                jmp     short loc_488D5E

loc_488D5A:
                jz      short loc_488D57

loc_488D5C:
                jmp     short loc_488D57

loc_488D5E:
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 0C6h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 4
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                and     eax, 40h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
```

```
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1200h
                dec     dh
                sub     dh, 1
                dec     dh
                sub     dh, 7
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_488DFD
                jl      short loc_488DFB

loc_488DF8:

                jmp     short loc_488DFF

loc_488DFB:
                jz      short loc_488DF8

loc_488DFD:
                jmp     short loc_488DF8

loc_488DFF:
                inc     eax
                dec     eax
                jo      short loc_488E0A
                jl      short loc_488E08

loc_488E05:

                jmp     short loc_488E0C

loc_488E08:
                jz      short loc_488E05

loc_488E0A:
                jmp     short loc_488E05

loc_488E0C:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_488E19
                jl      short loc_488E17
```

```
   loc_488E14:

                   jmp     short loc_488E1B

   loc_488E17:
                   jz      short loc_488E14

   loc_488E19:
                   jmp     short loc_488E14

   loc_488E1B:
                   inc     eax
                   pop     ecx
                   cmp     ecx, eax
                   jnz     short loc_488E26
                   and     eax, 0
                   jmp     short loc_488E2A

   loc_488E26:

                   and     eax, 0
                   inc     eax

   loc_488E2A:
                   mov     [ebp-0Ch], eax
                   mov     ecx, ds:dword_4D937C
                   xor     ecx, ds:dword_4D9380
                   shl     ecx, 1
                   mov     [ebp-8], ecx
                   cmp     dword ptr [ebp-0Ch], 0
                   jz      short loc_488E4D
                   mov     edx, [ebp-8]
                   or      edx, 1
                   mov     [ebp-8], edx

   loc_488E4D:
                   mov     eax, [ebp-8]
                   push    eax
                   call    ds:off_4DDCA8
                   add     esp, 4
                   pop     edi
                   pop     esi
                   pop     ebx
                   mov     esp, ebp
                   pop     ebp
                   retn
       }
  }
```

```
__declspec(naked) void sub_487B44(void)
{
    __asm
    {
                    push      ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE0
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 86h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9374
                xor     ecx, ds:dword_4D9378
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_487BD9
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_487BD9:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_4823C8(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD0C
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                mov     ecx, 800h
                mov     ecx, 0Dh
```

```
not     ecx
bswap   eax
not     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
dec     ecx
inc     ecx
inc     cl
inc     cl
inc     cl
add     ecx, 0Dh
inc     cl
inc     cl
inc     cl
inc     cl
inc     cl
add     ecx, 0Ah
dec     ecx
push    edx
mov     edx, 4
add     ecx, edx
inc     ecx
pop     edx
bswap   eax
and     eax, ecx
pop     ecx
neg     eax
sbb     eax, eax
neg     eax
pop     edx
mov     [ebp-0Ch], eax
mov     ecx, ds:dword_4D93A0
xor     ecx, ds:dword_4D93A4
shl     ecx, 1
mov     [ebp-8], ecx
cmp     dword ptr [ebp-0Ch], 0
jz      short loc_482458
```

```
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_482458:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCCC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48097F(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD14
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                mov     ecx, 800h
                mov     ecx, 6
                not     ecx
                bswap   eax
                not     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                add     ecx, 8
                add     ecx, 3
```

```
}
                        inc     ecx
                        inc     ecx
                        inc     ecx
                        inc     ecx
                        inc     ecx
                        inc     ecx
                        dec     ecx
                        inc     ecx
                        inc     cl
                        inc     cl
                        inc     cl
                        add     ecx, 12h
                        add     ecx, 0Ah
                        dec     ecx
                        push    edx
                        mov     edx, 4
                        add     ecx, edx
                        inc     ecx
                        pop     edx
                        bswap   eax
                        and     eax, ecx
                        pop     ecx
                        neg     eax
                        sbb     eax, eax
                        neg     eax
                        pop     edx
                        mov     [ebp-0Ch], eax
                        mov     ecx, ds:dword_4D93A8
                        xor     ecx, ds:dword_4D93AC
                        shl     ecx, 1
                        mov     [ebp-8], ecx
                        cmp     dword ptr [ebp-0Ch], 0
                        jz      short loc_480A07
                        mov     edx, [ebp-8]
                        or      edx, 1
                        mov     [ebp-8], edx

 loc_480A07:
                        mov     eax, [ebp-8]
                        push    eax
                        call    ds:off_4DDCD4
                        add     esp, 4
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
    }
}
```

```
__declspec(naked) void sub_4829D3(void)
{
     __asm
     {
                    push    ebp
                    mov     ebp, esp
                    sub     esp, 0Ch
                    push    ebx
                    push    esi
                    push    edi
                    mov     eax, [ebp+8]
                    push    eax
                    call    ds:off_4DDD0C
                    add     esp, 4
                    mov     [ebp-4], eax
                    mov     eax, [ebp-4]
                    push    edx
                    mov     edx, 0FFFFh
                    and     eax, edx
                    push    ebx
                    push    0Ah
                    pop     ebx
                    dec     bl
                    dec     bl
                    dec     bl
                    add     bl, 0FFh
                    add     bl, 0FFh
                    dec     bl
                    jo      short loc_482A14
                    jl      short loc_482A12

 loc_482A0D:
                    jmp     short loc_482A16

 loc_482A12:
                    jz      short loc_482A0D

 loc_482A14:
                    jmp     short loc_482A0D

 loc_482A16:
                    add     bl, 0FFh
                    add     bl, 0FFh
                    add     bl, 0FFh
                    add     bl, 0FFh
                    and     al, bl
                    jo      short loc_482A2D
                    jl      short loc_482A2B

 loc_482A28:
```

```
                jmp     short loc_482A2F

loc_482A2B:
                jz      short loc_482A28

loc_482A2D:
                jmp     short loc_482A28

loc_482A2F:
                mov     dh, 15h
                and     dl, 0
                dec     dh
                sub     dh, 6
                dec     dh
                dec     dh
                dec     dh
                sub     dh, 1
                dec     dh
                dec     dh
                and     ah, dh
                pop     ebx
                pop     edx
                test    eax, eax
                jz      short loc_482A56
                not     eax
                add     eax, 1
                stc
                jmp     short loc_482A5C

loc_482A56:
                not     eax
                add     eax, 1
                clc

loc_482A5C:
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_482A6B
                jl      short loc_482A69

loc_482A64:
                jmp     short loc_482A6D

loc_482A69:
                jz      short loc_482A64

loc_482A6B:
                jmp     short loc_482A64

loc_482A6D:
                inc     eax
                dec     eax
                jo      short loc_482A7A
                jl      short loc_482A78

loc_482A73:
                jmp     short loc_482A7C

loc_482A78:
                jz      short loc_482A73

loc_482A7A:
                jmp     short loc_482A73

loc_482A7C:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_482A8B
                jl      short loc_482A89

loc_482A84:
                jmp     short loc_482A8D

loc_482A89:
                jz      short loc_482A84

loc_482A8B:
                jmp     short loc_482A84

loc_482A8D:
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A0
                xor     ecx, ds:dword_4D93A4
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_482AB1
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_482AB1:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCCC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
```

```
                    pop     ebp
                    retn
        }
}




__declspec(naked) void sub_480350(void)
{
        __asm
        {
                    push    ebp
                    mov     ebp, esp
                    sub     esp, 0Ch
                    push    ebx
                    push    esi
                    push    edi
                    mov     eax, [ebp+8]
                    push    eax
                    call    ds:off_4DDD0C
                    add     esp, 4
                    mov     [ebp-4], eax
                    mov     eax, [ebp-4]
                    push    edx
                    mov     edx, 0FFFFh
                    and     eax, edx
                    push    ebx
                    push    0D00h
                    pop     ebx
                    jo      short loc_480386
                    jl      short loc_480384

 loc_48037F:
                    jmp     short loc_480388

 loc_480384:
                    jz      short loc_48037F

 loc_480386:
                    jmp     short loc_48037F

 loc_480388:
                    sub     bh, 5
                    dec     bh
                    push    eax
                    dec     bh
                    dec     bh
                    and     eax, 41h
                    dec     bh
                    sub     bh, 3
```

```
                 pop     eax
                 dec     bh
                 and     ah, bh
                 mov     edx, 25h
                 dec     dl
                 sub     dl, 3
                 dec     dl
                 sub     dl, 17h
                 dec     dl
                 dec     dl
                 dec     dl
                 dec     dl
                 dec     dl
                 and     al, dl
                 pop     ebx
                 pop     edx
                 neg     eax
                 sbb     eax, eax
                 neg     eax
                 mov     [ebp-0Ch], eax
                 mov     ecx, ds:dword_4D93A0
                 xor     ecx, ds:dword_4D93A4
                 shl     ecx, 1
                 mov     [ebp-8], ecx
                 cmp     dword ptr [ebp-0Ch], 0
                 jz      short loc_4803E5
                 mov     edx, [ebp-8]
                 or      edx, 1
                 mov     [ebp-8], edx

 loc_4803E5:
                 mov     eax, [ebp-8]
                 push    eax
                 call    ds:off_4DDCCC
                 add     esp, 4
                 pop     edi
                 pop     esi
                 pop     ebx
                 mov     esp, ebp
                 pop     ebp
                 retn
     }
}




__declspec(naked) void sub_4845BB(void)
{
     __asm
     {
                 push    ebp
```

```
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF8
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                bswap   ecx
                not     ecx
                push    eax
                not     eax
                mov     eax, 80h
                xchg    eax, ecx
                mov     ecx, 1
                xchg    eax, ecx
                not     eax
                pop     eax
                not     ecx
                pop     ecx
                push    edx
                mov     dh, 18h
                dec     dh
                dec     dh
                not     ecx
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                bswap   eax
                dec     dh
                dec     dh
                dec     edi
                sub     dh, 3
                dec     dh
                dec     edi
                dec     dh
                sub     dh, 0Bh
                dec     edi
                bswap   eax
                jo      short loc_484620
                jl      short loc_48461E

loc_48461B:
                jmp     short loc_484622

loc_48461E:
                jz      short loc_48461B
```

```
loc_484620:
                jmp     short loc_48461B

loc_484622:
                and     ah, dh
                mov     dl, 9
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                not     ecx
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                add     dl, 5
                sub     dl, 3
                dec     dl
                jo      short loc_48464B
                jl      short loc_484649

loc_484646:
                jmp     short loc_48464D

loc_484649:
                jz      short loc_484646

loc_48464B:
                jmp     short loc_484646

loc_48464D:
                and     al, dl
                not     ah
                bswap   eax
                bswap   eax
                not     ah
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D938C
                xor     ecx, ds:dword_4D9390
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48467B
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48467B:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_484488(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF8
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    0Dh
                pop     ebx
                jo      short loc_4844BB
                jl      short loc_4844B9

 loc_4844B4:
                jmp     short loc_4844BD

 loc_4844B9:
                jz      short loc_4844B4

 loc_4844BB:
                jmp     short loc_4844B4

 loc_4844BD:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                and     eax, 41h
                dec     bl
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 2500h
                dec     dh
                sub     dh, 3
                dec     dh
                sub     dh, 17h
                dec     dh
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D938C
                xor     ecx, ds:dword_4D9390
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_484512
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_484512:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48B65C(void)
{
```

```
__asm
{
            push    ebp
            mov     ebp, esp
            sub     esp, 0Ch
            push    ebx
            push    esi
            push    edi
            mov     eax, [ebp+8]
            push    eax
            call    ds:off_4DDCE4
            add     esp, 4
            mov     [ebp-4], eax
            mov     eax, [ebp-4]
            push    ebx
            mov     ebx, 0FFFFh
            and     eax, ebx
            push    ecx
            mov     ch, 2Dh
            dec     ch
            sub     ch, 2Ah
            dec     ch
            dec     ch
            and     ah, ch
            mov     cl, 0BDh
            sub     cl, 2
            dec     cl
            dec     cl
            dec     cl
            dec     cl
            dec     cl
            dec     cl
            dec     cl
            not     cl
            bswap   edx
            not     cl
            bswap   edx
            dec     cl
            dec     cl
            dec     cl
            dec     cl
            push    eax
            dec     cl
            dec     cl
            sub     cl, 12h
            dec     cl
            dec     cl
            sub     cl, 3
            dec     cl
            and     eax, 40h
            dec     cl
            dec     cl
            dec     cl
            add     cl, 0Eh
            dec     cl
            dec     cl
            and     eax, 80h
            sub     cl, 1Fh
            dec     cl
            dec     cl
            dec     cl
            not     ecx
            bswap   eax
            not     ecx
            bswap   eax
            pop     eax
            inc     cl
            inc     cl
            inc     cl
            and     al, cl
            mov     eax, eax
            pop     ecx
            neg     eax
            sbb     eax, eax
            inc     eax
            pop     ebx
            push    eax
            mov     eax, [ebp-4]
            mov     edx, 0C00h
            sub     dh, 1
            dec     dh
            dec     dh
            dec     dh
            inc     dh
            dec     dh
            inc     dh
            inc     dh
            sub     dh, 2
            and     eax, edx
            neg     eax
            sbb     eax, eax
            inc     eax
            mov     edx, eax
            pop     eax
            xor     ecx, ecx
            cmp     eax, edx
            setz    cl
            mov     al, cl
            mov     [ebp-0Ch], eax
            mov     ecx, ds:dword_4D9378
            xor     ecx, ds:dword_4D937C
            shl     ecx, 1
            mov     [ebp-8], ecx
            cmp     dword ptr [ebp-0Ch], 0
```

```
                jz      short loc_48B74D
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48B74D:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48A993(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD0C
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                bswap   ecx
                not     ecx
                push    eax
                not     eax
                mov     eax, 80h
                xchg    eax, ecx
                mov     ecx, 1
                xchg    eax, ecx
                not     eax
                pop     eax
                not     ecx
                pop     ecx
                push    edx
                mov     dh, 12h
                dec     dh
                dec     dh
                not     ecx
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                bswap   eax
                dec     dh
                dec     dh
                sub     dh, 5
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                bswap   eax
                and     ah, dh
                mov     dl, 9
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                not     ecx
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                add     dl, 1
                and     al, dl
                not     ah
                bswap   eax
                bswap   eax
                not     ah
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A0
                xor     ecx, ds:dword_4D93A4
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48AA38
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48AA38:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCCC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_487DEA(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD04
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                mov     ecx, 800h
                mov     ecx, 40h
                not     ecx
                bswap   eax
                not     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                add     ecx, 0Bh
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Dh
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Ah
                dec     ecx
                push    edx
                mov     edx, 4
                add     ecx, edx
                inc     ecx
                pop     edx
                bswap   eax
                add     ecx, 3
                and     eax, ecx
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                push    eax
                mov     eax, [ebp-4]
                mov     edx, 0F00h
                sub     dh, 1
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                and     eax, edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                jo      short loc_487E87
                jl      short loc_487E85

loc_487E82:
                jmp     short loc_487E89

loc_487E85:
                jz      short loc_487E82

loc_487E87:
                jmp     short loc_487E82
```

```
loc_487E89:
                cmp     eax, edx
                jo      short loc_487E94
                jl      short loc_487E92

loc_487E8F:
                jmp     short loc_487E96

loc_487E92:
                jz      short loc_487E8F

loc_487E94:
                jmp     short loc_487E8F

loc_487E96:
                jz      short loc_487EA9
                and     eax, 0
                jo      short loc_487EA4
                jl      short loc_487EA2

loc_487E9F:
                jmp     short loc_487EA6

loc_487EA2:
                jz      short loc_487E9F

loc_487EA4:
                jmp     short loc_487E9F

loc_487EA6:
                inc     eax
                jmp     short loc_487EAC

loc_487EA9:
                and     eax, 0

loc_487EAC:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_487ECF
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_487ECF:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_482E3D(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCFC
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1E00h
                pop     ebx
                jo      short loc_482E71
                jl      short loc_482E6F

 loc_482E6C:
                jmp     short loc_482E73

 loc_482E6F:
                jz      short loc_482E6C

 loc_482E71:
                jmp     short loc_482E6C

 loc_482E73:
                sub     bh, 4
                inc     bh
                sub     bh, 2
                inc     bh
                inc     bh
                dec     bh
                push    eax
                dec     bh
                dec     bh
                inc     bh
                dec     bh
                dec     bh
                jo      short loc_482E95
                jl      short loc_482E93

loc_482E90:
                jmp     short loc_482E97

loc_482E93:
                jz      short loc_482E90

loc_482E95:
                jmp     short loc_482E90

loc_482E97:
                and     eax, 40h
                dec     bh
                sub     bh, 12h
                sub     bh, 3
                pop     eax
                dec     bh
                and     ah, bh
                mov     edx, 12h
                dec     dl
                sub     dl, 1
                dec     dl
                sub     dl, 9
                inc     dl
                dec     dl
                dec     dl
                dec     dl
                and     al, dl
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9390
                xor     ecx, ds:dword_4D9394
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_482EEA
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_482EEA:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCBC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_488055(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD00
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     dh, 2
                dec     dh
                dec     dh
                and     ah, dh
                mov     dl, 0Eh
                sub     dl, 0FFh
                jo      short loc_488088
                jl      short loc_488086

 loc_488083:
                jmp     short loc_48808A

 loc_488086:
                jz      short loc_488083
```

```
loc_488088:
                jmp     short loc_488083

loc_48808A:
                sub     dl, 0FEh
                dec     dl
                sub     dl, 0FFh
                sub     dl, 0Ah
                sub     dl, 0FFh
                sub     dl, 0FFh
                jo      short loc_4880A4
                jl      short loc_4880A2

loc_48809F:
                jmp     short loc_4880A6

loc_4880A2:
                jz      short loc_48809F

loc_4880A4:
                jmp     short loc_48809F

loc_4880A6:
                sub     dl, 1
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                sub     dl, 3
                sub     dl, 0FFh
                dec     dl
                inc     dl
                inc     dl
                inc     dl
                inc     dl
                and     al, dl
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9394
                xor     ecx, ds:dword_4D9398
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4880F4
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_4880F4:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_48349D(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE4
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    0Ah
                pop     ebx
                dec     bl
                dec     bl
                dec     bl
                add     bl, 0FFh
                add     bl, 0FFh
                dec     bl
                sub     bl, 1
                add     bl, 0FFh
                add     bl, 0FFh
                add     bl, 0FFh
                and     al, bl
                mov     dh, 15h
                and     dl, 0
                dec     dh
                sub     dh, 6
                dec     dh
                dec     dh
                dec     dh
                sub     dh, 1
                dec     dh
                dec     dh
                and     ah, dh
                pop     ebx
                pop     edx
                test    eax, eax
                jz      short loc_483508
                not     eax
                add     eax, 1
                stc
                jmp     short loc_48350E

loc_483508:
                not     eax
                add     eax, 1
                clc

loc_48350E:
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9378
                xor     ecx, ds:dword_4D937C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_483535
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_483535:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
```

```
}



__declspec(naked) void sub_484E68(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCEC
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 87h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                and     eax, 0
                inc     eax
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                mov     eax, eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9380
                xor     ecx, ds:dword_4D9384
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_484EF7
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_484EF7:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCAC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_488108(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 8
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    ebx
                mov     ebx, [ebp+0Ch]
                inc     ebx
                inc     ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 10h
                dec     ch
                dec     ch
                sub     ch, 14h
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                mov     ebx, [ebp+0Ch]
                dec     esi
                dec     edi
                xor     edx, edx
                or      ebx, edx
                jz      short loc_488150
                dec     edi
                and     eax, 0
                jmp     short loc_488157

loc_488150:
                dec     edi
                and     eax, 0
                dec     edi
                dec     edi
                inc     eax

loc_488157:
                mov     [ebp-8], eax
                mov     eax, ds:dword_4D93A0
                xor     eax, ds:dword_4D93A4
                shl     eax, 1
                mov     [ebp-4], eax
                cmp     dword ptr [ebp-8], 0
                jz      short loc_488179
                mov     ecx, [ebp-4]
                or      ecx, 1
                mov     [ebp-4], ecx

loc_488179:
                mov     edx, [ebp-4]
                push    edx
                call    ds:off_4DDCCC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
```

```
    }
}



__declspec(naked) void sub_48246C(void)
{
    __asm
    {
            push    ebp
            mov     ebp, esp
            sub     esp, 0Ch
            push    ebx
            push    esi
            push    edi
            mov     eax, [ebp+8]
            push    eax
            call    ds:off_4DDCF8
            add     esp, 4
            mov     [ebp-4], eax
            mov     eax, [ebp-4]
            push    ebx
            mov     ebx, [ebp+0Ch]
            mov     ebx, 0FFFFh
            and     eax, ebx
            push    ecx
            push    4
            pop     ecx
            dec     cl
            dec     cl
            dec     cl
            dec     cl
            and     al, cl
            mov     bh, 0Dh
            xor     bl, bl
            dec     bh
            dec     bh
            dec     bh
            dec     bh
            dec     bh
            dec     bh
            sub     bh, 1
            dec     bh
            add     bh, 4
            inc     bh
            sub     bh, 1
            dec     bh
            and     ah, bh
            pop     ecx
            pop     ebx
            test    eax, eax
            jz      short loc_4824D0
```

```
                not     eax
                add     eax, 1
                stc
                jmp     short loc_4824D6

loc_4824D0:
                not     eax
                add     eax, 1
                clc

loc_4824D6:
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D938C
                xor     ecx, ds:dword_4D9390
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4824FD
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_4824FD:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_482C9E(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
```

```
call    ds:off_4DDD04
add     esp, 4
mov     [ebp-4], eax
mov     eax, [ebp-4]
push    edx
mov     edx, 0FFFFh
and     eax, edx
push    ebx
push    eax
mov     bh, 3
dec     bh
dec     bh
dec     bh
and     eax, 800h
bswap   ecx
pop     eax
bswap   ecx
and     ah, bh
mov     bl, 9Ah
sub     bl, 5
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
sub     bl, 2
sub     bl, 0Ch
not     bx
bswap   eax
not     bx
bswap   eax
and     al, bl
mov     eax, eax
pop     ebx
neg     eax
sbb     eax, eax
inc     eax
pop     edx
push    eax
mov     eax, [ebp-4]
mov     edx, 500h
inc     dh
inc     dh
inc     dh
and     eax, edx
neg     eax
sbb     eax, eax
inc     eax
mov     edx, eax
pop     eax
```

```
                xor     ecx, ecx
                jo      short loc_482D2A
                jl      short loc_482D28

loc_482D25:
                jmp     short loc_482D2C

loc_482D28:
                jz      short loc_482D25

loc_482D2A:
                jmp     short loc_482D25

loc_482D2C:
                cmp     eax, edx
                jo      short loc_482D37
                jl      short loc_482D35

loc_482D32:
                jmp     short loc_482D39

loc_482D35:
                jz      short loc_482D32

loc_482D37:
                jmp     short loc_482D32

loc_482D39:
                setnz   cl
                mov     al, cl
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_482D61
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_482D61:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
```

```
    }
}



__declspec(naked) void sub_488475(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCFC
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    21h
                pop     ebx
                jo      short loc_4884A8
                jl      short loc_4884A6

 loc_4884A1:
                jmp     short loc_4884AA

 loc_4884A6:
                jz      short loc_4884A1

 loc_4884A8:
                jmp     short loc_4884A1

 loc_4884AA:
                sub     bl, 5
                dec     bl
                sub     bl, 2
                push    eax
                dec     bl
                dec     bl
                jo      short loc_4884C2
                jl      short loc_4884C0

 loc_4884BB:
                jmp     short loc_4884C4
```

```
loc_4884C0:
                jz      short loc_4884BB

loc_4884C2:
                jmp     short loc_4884BB

loc_4884C4:
                and     eax, 40h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1100h
                sub     dh, 1
                dec     dh
                sub     dh, 7
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_4884F6
                jl      short loc_4884F4

loc_4884EF:
                jmp     short loc_4884F8

loc_4884F4:
                jz      short loc_4884EF

loc_4884F6:
                jmp     short loc_4884EF

loc_4884F8:
                inc     eax
                dec     eax
                jo      short loc_488505
                jl      short loc_488503

loc_4884FE:
                jmp     short loc_488507

loc_488503:
                jz      short loc_4884FE

loc_488505:
                jmp     short loc_4884FE
```

```
 loc_488507:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_488514
                jl      short loc_488512

 loc_48850F:
                jmp     short loc_488516

 loc_488512:
                jz      short loc_48850F

 loc_488514:
                jmp     short loc_48850F

 loc_488516:
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9390
                xor     ecx, ds:dword_4D9394
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48853A
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48853A:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCBC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_486299(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF4
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1F00h
                pop     ebx
                jo      short loc_4862CF
                jl      short loc_4862CD

loc_4862C8:
                jmp     short loc_4862D1

loc_4862CD:
                jz      short loc_4862C8

loc_4862CF:
                jmp     short loc_4862C8

loc_4862D1:
                sub     bh, 3
                sub     bh, 3
                push    eax
                dec     bh
                dec     bh
                and     eax, 80h
                dec     bh
                sub     bh, 10h
                sub     bh, 5
                pop     eax
                dec     bh
                jo      short loc_4862F7
                jl      short loc_4862F5

loc_4862F0:
                jmp     short loc_4862F9

loc_4862F5:
                jz      short loc_4862F0

loc_4862F7:
                jmp     short loc_4862F0
```

```
loc_4862F9:
                and     ah, bh
                mov     edx, 16h
                dec     dl
                sub     dl, 3
                dec     dl
                sub     dl, 8
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                jo      short loc_48631F
                jl      short loc_48631D

loc_486318:
                jmp     short loc_486321

loc_48631D:
                jz      short loc_486318

loc_48631F:
                jmp     short loc_486318

loc_486321:
                and     al, dl
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9388
                xor     ecx, ds:dword_4D938C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48634E
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48634E:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_489BB2(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCDC_2
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 0Eh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 85h
                sub     bl, 7
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9370
                xor     ecx, ds:dword_4D9374
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_489C4E
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_489C4E:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDC9C
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48BED9(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE8
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                mov     ecx, 800h
                mov     ecx, 4Bh
                not     ecx
                bswap   eax
                not     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Dh
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                inc     cl
                add     ecx, 0Ah
                dec     ecx
                push    edx
                mov     edx, 4
```

```
                add     ecx, edx
                inc     ecx
                pop     edx
                bswap   eax
                add     ecx, 3
                and     eax, ecx
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D937C
                xor     ecx, ds:dword_4D9380
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48BF79
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48BF79:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_48DBAA(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 8
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    ebx
                mov     ebx, [ebp+0Ch]
                inc     ebx
                inc     ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 10h
                dec     ch
                dec     ch
                sub     ch, 11h
                dec     ch
                sub     ch, 4
                dec     ch
                sub     ch, 3
                dec     ch
                mov     ebx, [ebp+0Ch]
                dec     esi
                dec     edi
                xor     edx, edx
                or      ebx, edx
                jz      short loc_48DBF0
                dec     edi
                and     eax, 0
                jmp     short loc_48DBF7

loc_48DBF0:
                dec     edi
                and     eax, 0
                dec     edi
                dec     edi
                inc     eax

loc_48DBF7:
                mov     [ebp-8], eax
                mov     eax, ds:dword_4D9380
                xor     eax, ds:dword_4D9384
                shl     eax, 1
                mov     [ebp-4], eax
                cmp     dword ptr [ebp-8], 0
                jz      short loc_48DC19
                mov     ecx, [ebp-4]
                or      ecx, 1
                mov     [ebp-4], ecx

loc_48DC19:
                mov     edx, [ebp-4]
                push    edx
                call    ds:off_4DDCAC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_483EB0(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 8
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, 800h
                push    ecx
                mov     ch, 41h
                sub     ch, 1
                sub     ch, 10h
                dec     ch
                dec     ch
                push    ebx
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                mov     ebx, [ebp+0Ch]
                dec     ch
                dec     ch
                dec     ch
                inc     bl
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                inc     bl
                dec     ch
                dec     ch
                dec     ch
                sub     bl, 0Ah
                dec     ch
                sub     ch, 4
```

```
                        dec     ch
                        pop     ebx
                        sub     ch, 3
                        dec     ch
                        mov     ebx, [ebp+0Ch]
                        dec     esi
                        dec     edi
                        dec     edi
                        mov     edx, 4
                        dec     edx
                        dec     edx
                        sub     edx, 2
                        or      ebx, edx
                        jz      short loc_483F24
                        dec     edi
                        and     eax, 0
                        jmp     short loc_483F2C

        loc_483F24:
                        dec     edi
                        dec     ecx
                        and     eax, 0
                        dec     ecx
                        dec     edx
                        inc     eax

        loc_483F2C:
                        mov     [ebp-8], eax
                        mov     eax, ds:dword_4D938C
                        xor     eax, ds:dword_4D9390
                        shl     eax, 1
                        mov     [ebp-4], eax
                        cmp     dword ptr [ebp-8], 0
                        jz      short loc_483F4E
                        mov     ecx, [ebp-4]
                        or      ecx, 1
                        mov     [ebp-4], ecx

        loc_483F4E:
                        mov     edx, [ebp-4]
                        push    edx
                        call    ds:off_4DDCB8
                        add     esp, 4
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
            }
       }
```

```
__declspec(naked) void sub_484003(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCDC_2
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                jo      short loc_484028
                jl      short loc_484026

 loc_484023:
                jmp     short loc_48402A

 loc_484026:
                jz      short loc_484023

 loc_484028:
                jmp     short loc_484023

 loc_48402A:
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 9
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                jo      short loc_48405D
                jl      short loc_48405B

loc_484058:
                jmp     short loc_48405F

loc_48405B:
                jz      short loc_484058

loc_48405D:
                jmp     short loc_484058

loc_48405F:
                mov     bl, 0C5h
                sub     bl, 4
                dec     bl
                dec     bl
                sub     bl, 3
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                jo      short loc_48408A
                jl      short loc_484088

loc_484085:
                jmp     short loc_48408C

loc_484088:
                jz      short loc_484085

loc_48408A:
                jmp     short loc_484085

loc_48408C:
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9370
                xor     ecx, ds:dword_4D9374
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4840BB
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_4840BB:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDC9C
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}





__declspec(naked) void sub_48682F(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD04
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 87h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                and     eax, 0
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                dec     bl
                inc     eax
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4868B6
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_4868B6:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_480022(void)
{
    __asm
    {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE4
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 4
                dec     ch
                sub     ch, 3
                dec     ch
                and     ah, ch
                mov     cl, 70h
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 6
                not     al
                bswap   ecx
                not     al
                bswap   ecx
                dec     cl
                dec     cl
                sub     cl, 10h
                dec     cl
                dec     cl
                add     cl, 0Ch
                dec     cl
                dec     cl
                dec     cl
                jo      short loc_480092
                jl      short loc_480090

loc_48008D:

                jmp     short loc_480094
```

```
loc_480090:
                jz      short loc_48008D

loc_480092:
                jmp     short loc_48008D

loc_480094:
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 10h
                sub     cl, 1
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                and     al, cl
                mov     eax, eax
                pop     ecx
                pop     ebx
                test    eax, eax
                jnz     loc_4801A8
                mov     eax, [ebp-4]
                jo      short loc_4800D9
                jl      short loc_4800D7

loc_4800D4:

                jmp     short loc_4800DB

loc_4800D7:
                jz      short loc_4800D4

loc_4800D9:
                jmp     short loc_4800D4

loc_4800DB:
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 0C6h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                and     eax, 40h
```

```
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1200h
                dec     dh
                sub     dh, 1
                dec     dh
                sub     dh, 7
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_48017F
                jl      short loc_48017D

loc_48017A:

                jmp     short loc_480181

loc_48017D:
                jz      short loc_48017A

loc_48017F:
                jmp     short loc_48017A

loc_480181:
                inc     eax
                dec     eax
                jo      short loc_48018C
                jl      short loc_48018A

loc_480187:

                jmp     short loc_48018E

loc_48018A:
                jz      short loc_480187

loc_48018C:
                jmp     short loc_480187

loc_48018E:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_48019B
                jl      short loc_480199

loc_480196:

                jmp     short loc_48019D

loc_480199:
                jz      short loc_480196

loc_48019B:
                jmp     short loc_480196

loc_48019D:
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_4801A8
                and     eax, 0
                jmp     short loc_4801AC

loc_4801A8:

                and     eax, 0
                inc     eax

loc_4801AC:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9378
                xor     ecx, ds:dword_4D937C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4801CF
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_4801CF:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}
```

```
__declspec(naked) void sub_487043(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE0
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, [ebp+0Ch]
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Fh
                dec     ch
                dec     ch
                dec     ch
                sub     ch, 1
                sub     ch, 15h
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                sub     ch, 7
                dec     ch
                dec     ch
                and     ah, ch
                mov     cl, 0BDh
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     cl
                bswap   edx
                not     cl
                bswap   edx
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                push    eax
                dec     cl
                dec     cl
                sub     cl, 13h
                dec     cl
                sub     cl, 3
                dec     cl
                and     eax, 41h
                dec     cl
                dec     cl
                dec     cl
                add     cl, 0Eh
                dec     cl
                dec     cl
                and     eax, 80h
                sub     cl, 22h
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                pop     eax
                inc     cl
                inc     cl
                inc     cl
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9374
                xor     ecx, ds:dword_4D9378
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_487123
                mov     edx, [ebp-8]
                or      edx, 1
```

```
                mov     [ebp-8], edx

 loc_487123:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_487383(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF0
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 80h
                jmp     short loc_4873AC
                mov     ebx, 4

 loc_4873AC:
                mov     ebx, 30h
                dec     esi
                xor     ebx, 41h
                dec     edi
                add     esi, 23h
                add     ebx, 2
                not     ebx
                bswap   eax
                not     ebx
                add     esi, 2
                dec     esi
                dec     esi
                inc     ebx
                dec     esi
                inc     ebx
                dec     esi
                inc     ebx
                inc     ebx
                dec     esi
                inc     ebx
                dec     esi
                inc     ebx
                dec     esi
                inc     ebx
                inc     ebx
                dec     esi
                inc     ebx
                inc     ebx
                dec     esi
                dec     ebx
                dec     esi
                dec     ebx
                push    ecx
                dec     esi
                mov     ecx, 4
                add     ebx, ecx
                inc     ebx
                dec     esi
                pop     ecx
                dec     esi
                bswap   eax
                sub     ebx, 10h
                dec     ebx
                dec     ebx
                dec     ebx
                dec     ebx
                dec     ebx
                dec     ebx
                dec     ebx
                dec     ebx
                dec     ebx
                dec     ebx
                dec     ebx
                dec     ebx
                dec     ebx
                dec     ebx
                sub     ebx, 1Ch
                dec     ebx
                dec     ebx
                dec     ebx
                dec     ebx
                dec     ebx
                dec     ebx
```

```
                        and     eax, ebx
                        pop     ebx
                        dec     esi
                        neg     eax
                        sbb     eax, eax
                        inc     eax
                        pop     edx
                        mov     [ebp-0Ch], eax
                        mov     ecx, ds:dword_4D9384
                        xor     ecx, ds:dword_4D9388
                        shl     ecx, 1
                        mov     [ebp-8], ecx
                        cmp     dword ptr [ebp-0Ch], 0
                        jz      short loc_487431
                        mov     edx, [ebp-8]
                        or      edx, 1
                        mov     [ebp-8], edx

 loc_487431:
                        mov     eax, [ebp-8]
                        push    eax
                        call    ds:off_4DDCB0
                        add     esp, 4
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
     }
}




__declspec(naked) void sub_488C12(void)
{
     __asm
     {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDD14
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    edx
```

```
                    mov     edx, [ebp+0Ch]
                    mov     edx, 0FFFFh
                    and     eax, edx
                    push    ebx
                    push    eax
                    dec     bh
//                    ja      short $+2
                    dec     bh
                    dec     bh
                    dec     bh
                    and     eax, 41h
                    bswap   ecx
                    and     eax, 0
//                    jno     short $+2
                    mov     bl, 85h
                    sub     bl, 20h
                    dec     bl
                    dec     bl
                    sub     bl, 1Ah
                    dec     bl
                    sub     bl, 1Fh
                    not     bx
                    inc     eax
                    dec     bl
                    dec     bl
                    dec     bl
                    pop     ebx
                    mov     [ebp-0Ch], eax
                    mov     ecx, ds:dword_4D93A8
                    xor     ecx, ds:dword_4D93AC
                    shl     ecx, 1
                    mov     [ebp-8], ecx
                    cmp     dword ptr [ebp-0Ch], 0
                    jz      short loc_488C8E
                    mov     edx, [ebp-8]
                    or      edx, 1
                    mov     [ebp-8], edx

 loc_488C8E:
                    mov     eax, [ebp-8]
                    push    eax
                    call    ds:off_4DDCD4
                    add     esp, 4
                    pop     edi
                    pop     esi
                    pop     ebx
                    mov     esp, ebp
                    pop     ebp
                    retn
    }
}
```

```
__declspec(naked) void sub_489798(void)
{
     __asm
     {
                    push    ebp
                    mov     ebp, esp
                    sub     esp, 0Ch
                    push    ebx
                    push    esi
                    push    edi
                    mov     eax, [ebp+8]
                    push    eax
                    call    ds:off_4DDCE8
                    add     esp, 4
                    mov     [ebp-4], eax
                    mov     eax, [ebp-4]
                    push    edx
                    mov     dh, 2
                    dec     dh
                    dec     dh
                    and     ah, dh
                    mov     dl, 0Eh
                    sub     dl, 0FFh
                    jo      short loc_4897CB
                    jl      short loc_4897C9

 loc_4897C6:
                    jmp     short loc_4897CD

 loc_4897C9:
                    jz      short loc_4897C6

 loc_4897CB:
                    jmp     short loc_4897C6

 loc_4897CD:
                    sub     dl, 0FFh
                    sub     dl, 0FFh
                    sub     dl, 0Ah
                    sub     dl, 0FFh
                    sub     dl, 0FFh
                    sub     dl, 5
                    dec     dl
                    dec     dl
                    dec     dl
                    sub     dl, 3
                    sub     dl, 0FFh
                    dec     dl
                    inc     dl
```

```
                inc     dl
                inc     dl
                and     al, dl
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D937C
                xor     ecx, ds:dword_4D9380
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48981E
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48981E:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_4861C8(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCDC_2
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Dh
                dec     ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 7
                dec     ch
                dec     ch
                and     ah, ch
                mov     cl, 79h
                sub     cl, 4
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     cl
                bswap   edx
                not     cl
                bswap   edx
                dec     cl
                dec     cl
                push    eax
                dec     cl
                dec     cl
                sub     cl, 13h
                jo      short loc_48622E
                jl      short loc_48622C

loc_486229:
                jmp     short loc_486230

loc_48622C:
                jz      short loc_486229

loc_48622E:
                jmp     short loc_486229

loc_486230:
                dec     cl
                and     eax, 40h
                dec     cl
                dec     cl
                dec     cl
                add     cl, 0Eh
                dec     cl
                dec     cl
                and     eax, 40h
                sub     cl, 1Fh
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                pop     eax
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9370
                xor     ecx, ds:dword_4D9374
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_486285
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_486285:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDC9C
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_47FCB2(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD04
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 3
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 95h
                sub     bl, 9
                sub     bl, 0Ch
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_47FD2C
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_47FD2C:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
                add     esp, 4
                pop     edi
```

```
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
            }
}




__declspec(naked) void sub_48A0FB(void)
{
            __asm
            {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDCF4
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    ebx
                        mov     ebx, 0FFFFh
                        and     eax, ebx
                        push    ecx
                        push    2
                        jo      short loc_48A12D
                        jl      short loc_48A12B

 loc_48A126:
                        jmp     short loc_48A12F

 loc_48A12B:
                        jz      short loc_48A126

 loc_48A12D:
                        jmp     short loc_48A126

 loc_48A12F:
                        pop     ecx
                        dec     cl
                        dec     cl
                        jo      short loc_48A13F
                        jl      short loc_48A13D

 loc_48A138:
```

```
                jmp     short loc_48A141

loc_48A13D:
                jz      short loc_48A138

loc_48A13F:
                jmp     short loc_48A138

loc_48A141:
                and     al, cl
                mov     bh, 0Fh
                and     bl, 0
                dec     bh
                sub     bh, 3
                dec     bh
                sub     bh, 1
                dec     bh
                jo      short loc_48A15F
                jl      short loc_48A15D

loc_48A158:
                jmp     short loc_48A161

loc_48A15D:
                jz      short loc_48A158

loc_48A15F:
                jmp     short loc_48A158

loc_48A161:
                and     ah, bh
                pop     ecx
                pop     ebx
                test    eax, eax
                jz      short loc_48A171
                not     eax
                add     eax, 1
                stc
                jmp     short loc_48A177

loc_48A171:
                not     eax
                add     eax, 1
                clc

loc_48A177:
                sbb     eax, eax
                add     eax, 1
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9388
                xor     ecx, ds:dword_4D938C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48A19F
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48A19F:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_4889F5(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE8
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 38h
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 87h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                and     eax, 0
                dec     bl
                dec     bl
                sub     bl, 10h
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                inc     eax
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                mov     eax, eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D937C
                xor     ecx, ds:dword_4D9380
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_488A92
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_488A92:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48A60F(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCFC
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 800h
                jmp     short loc_48A638
                mov     ebx, 80h

 loc_48A638:
                mov     ebx, 71h
                not     ebx
                bswap   eax
                not     ebx
                inc     ebx
                inc     ebx
                inc     ebx
                add     ebx, 7
                push    ecx
                mov     ecx, 4
                add     ebx, ecx
                inc     ebx
                pop     ecx
                bswap   eax
                and     eax, ebx
                pop     ebx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9390
                xor     ecx, ds:dword_4D9394
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48A682
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48A682:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCBC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48CFE9(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCFC
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1F00h
                pop     ebx
                jo      short loc_48D01F
                jl      short loc_48D01D

 loc_48D018:
```

```
                jmp     short loc_48D021

loc_48D01D:
                jz      short loc_48D018

loc_48D01F:
                jmp     short loc_48D018

loc_48D021:
                sub     bh, 3
                sub     bh, 3
                push    eax
                dec     bh
                dec     bh
                and     eax, 80h
                dec     bh
                sub     bh, 10h
                sub     bh, 5
                pop     eax
                dec     bh
                and     ah, bh
                mov     edx, 16h
                dec     dl
                sub     dl, 3
                dec     dl
                sub     dl, 8
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                and     al, dl
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9390
                xor     ecx, ds:dword_4D9394
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48D084
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48D084:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCBC
```

```
                        add     esp, 4
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
          }
}



__declspec(naked) void sub_483F62(void)
{
     __asm
     {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDD18
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    edx
                        mov     edx, [ebp+0Ch]
                        mov     edx, 0FFFFh
                        and     eax, edx
                        push    ebx
                        push    410h
                        pop     ebx
                        dec     bh
                        dec     bh
                        sub     bh, 0FFh
                        sub     bh, 2
                        dec     bh
                        and     ah, bh
                        mov     bl, 0Dh
                        sub     bl, 4
                        sub     bl, 1
                        sub     bl, 1
                        sub     bl, 1
                        sub     bl, 1
                        sub     bl, 1
                        and     al, bl
                        pop     ebx
                        pop     edx
                        test    eax, eax
```

```
                jz      short loc_483FC2
                not     eax
                add     eax, 1
                stc
                jmp     short loc_483FC8

loc_483FC2:
                not     eax
                add     eax, 1
                clc

loc_483FC8:
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93AC
                xor     ecx, ds:dword_4D93B0
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_483FEF
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_483FEF:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_47FBF8(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 8
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    ebx
                jo      short loc_47FC0E
                jl      short loc_47FC0C

loc_47FC09:
                jmp     short loc_47FC10

loc_47FC0C:
                jz      short loc_47FC09

loc_47FC0E:
                jmp     short loc_47FC09

loc_47FC10:
                mov     ebx, 4
                and     eax, ebx
                push    ecx
                mov     ch, 10h
                sub     ch, 1
                dec     ch
                dec     ch
                dec     ch
                jo      short loc_47FC2C
                jl      short loc_47FC2A

loc_47FC27:
                jmp     short loc_47FC2E

loc_47FC2A:
                jz      short loc_47FC27

loc_47FC2C:
                jmp     short loc_47FC27

loc_47FC2E:
                dec     ch
                dec     ch
                mov     ebx, [ebp+0Ch]
                dec     esi
                dec     edi
                dec     edi
                xor     edx, edx
                or      ebx, edx
                jo      short loc_47FC45
                jl      short loc_47FC43

loc_47FC40:
                jmp     short loc_47FC47

loc_47FC43:
                jz      short loc_47FC40
```

```
loc_47FC45:
                jmp     short loc_47FC40

loc_47FC47:
                jz      short loc_47FC65
                dec     edi
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                sub     ch, 8
                and     eax, 0
                jo      short loc_47FC61
                jl      short loc_47FC5F

loc_47FC5C:
                jmp     short loc_47FC63

loc_47FC5F:
                jz      short loc_47FC5C

loc_47FC61:
                jmp     short loc_47FC5C

loc_47FC63:
                jmp     short loc_47FC7C

loc_47FC65:
                dec     edi
                dec     ecx
                sub     ch, 2
                dec     ch
                dec     ch
                sub     ch, 8
                and     eax, 0
                dec     ecx
                sub     ch, 2
                dec     ch
                dec     edx
                inc     eax

loc_47FC7C:
                mov     [ebp-8], eax
                mov     eax, ds:dword_4D9390
                xor     eax, ds:dword_4D9394
                shl     eax, 1
                mov     [ebp-4], eax
                cmp     dword ptr [ebp-8], 0
                jz      short loc_47FC9E
                mov     ecx, [ebp-4]
                or      ecx, 1
                mov     [ebp-4], ecx
```

```
 loc_47FC9E:
                mov     edx, [ebp-4]
                push    edx
                call    ds:off_4DDCBC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_48493A(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF4
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 4
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 80h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 86h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                jo      short loc_48498C
                jl      short loc_48498A

loc_484985:
                jmp     short loc_48498E

loc_48498A:
                jz      short loc_484985

loc_48498C:
                jmp     short loc_484985

loc_48498E:
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                pop     ebx
                pop     edx
                test    eax, eax
                jnz     loc_484A5F
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 1
                dec     bh
                and     eax, 41h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 93h
                sub     bl, 0Bh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_484A06
                jl      short loc_484A04

loc_4849FF:
                jmp     short loc_484A08

loc_484A04:
                jz      short loc_4849FF

loc_484A06:
                jmp     short loc_4849FF

loc_484A08:
                sub     bl, 6
                push    eax
                and     eax, 40h
                dec     bl
                sub     bl, 14h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1200h
                dec     dh
                sub     dh, 1
                dec     dh
                sub     dh, 7
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_484A40
                jl      short loc_484A3E

loc_484A39:
```

```
                jmp     short loc_484A42

loc_484A3E:
                jz      short loc_484A39

loc_484A40:
                jmp     short loc_484A39

loc_484A42:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_484A51
                jl      short loc_484A4F

loc_484A4C:
                jmp     short loc_484A53

loc_484A4F:
                jz      short loc_484A4C

loc_484A51:
                jmp     short loc_484A4C

loc_484A53:
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_484A5F
                and     eax, 0
                inc     eax
                jmp     short loc_484A62

loc_484A5F:
                and     eax, 0

loc_484A62:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9388
                xor     ecx, ds:dword_4D938C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_484A85
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_484A85:
```

```
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48C274(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCDC_2
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    0D00h
                pop     ebx
                jo      short loc_48C2AA
                jl      short loc_48C2A8

 loc_48C2A3:
                jmp     short loc_48C2AC

 loc_48C2A8:
                jz      short loc_48C2A3

 loc_48C2AA:
                jmp     short loc_48C2A3

 loc_48C2AC:
                sub     bh, 5
                dec     bh
                push    eax
                dec     bh
                dec     bh
                and     eax, 41h
                dec     bh
                sub     bh, 3
                pop     eax
                dec     bh
                and     ah, bh
                mov     edx, 28h
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                sub     dl, 1Bh
                dec     dl
                dec     dl
                inc     dl
                dec     dl
                inc     dl
                jo      short loc_48C2F0
                jl      short loc_48C2EE

loc_48C2E9:
                jmp     short loc_48C2F2

loc_48C2EE:
                jz      short loc_48C2E9

loc_48C2F0:
                jmp     short loc_48C2E9

loc_48C2F2:
                and     al, dl
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9370
                xor     ecx, ds:dword_4D9374
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48C31E
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48C31E:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDC9C
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_4899A6(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD0C
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, [ebp+0Ch]
                mov     ebx, 800h
                jmp     short loc_4899D2
                mov     ebx, 80h

 loc_4899D2:
                mov     ebx, 70h
                not     ebx
                bswap   eax
                not     ebx
                inc     ebx
                inc     ebx
                inc     ebx
                inc     ebx
                add     ebx, 4
                inc     ebx
                inc     ebx
                inc     ebx
                inc     ebx
                dec     ebx
                push    ecx
                mov     ecx, 4
                add     ebx, ecx
                inc     ebx
                pop     ecx
                bswap   eax
                and     eax, ebx
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A0
                xor     ecx, ds:dword_4D93A4
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_489A21
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_489A21:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCCC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48BE35(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
```

```
push    ebx
push    esi
push    edi
mov     eax, [ebp+8]
push    eax
call    ds:off_4DDCDC_2
add     esp, 4
mov     [ebp-4], eax
mov     eax, [ebp-4]
push    ecx
mov     ecx, 800h
mov     ecx, 0Ch
not     ecx
bswap   eax
not     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
dec     ecx
inc     ecx
inc     cl
inc     cl
inc     cl
add     ecx, 0Dh
inc     cl
inc     cl
inc     cl
inc     cl
inc     cl
add     ecx, 0Ah
dec     ecx
push    edx
mov     edx, 4
add     ecx, edx
inc     ecx
pop     edx
bswap   eax
and     eax, ecx
```

```
                pop     ecx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9370
                xor     ecx, ds:dword_4D9374
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48BEC5
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48BEC5:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDC9C
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_4885F8(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 8
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
```

```
                        dec     ch
                        sub     ch, 4
                        dec     ch
                        sub     ch, 3
                        dec     ch
                        mov     ebx, [ebp+0Ch]
                        dec     esi
                        dec     edi
                        xor     edx, edx
                        or      ebx, edx
                        jz      short loc_488634
                        dec     edi
                        and     eax, 0
                        jmp     short loc_48863B

         loc_488634:
                        dec     edi
                        and     eax, 0
                        dec     edi
                        dec     edi
                        inc     eax

         loc_48863B:
                        mov     [ebp-8], eax
                        mov     eax, ds:dword_4D9378
                        xor     eax, ds:dword_4D937C
                        shl     eax, 1
                        mov     [ebp-4], eax
                        cmp     dword ptr [ebp-8], 0
                        jz      short loc_48865D
                        mov     ecx, [ebp-4]
                        or      ecx, 1
                        mov     [ebp-4], ecx

         loc_48865D:
                        mov     edx, [ebp-4]
                        push    edx
                        call    ds:off_4DDCA4
                        add     esp, 4
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
            }
        }



        __declspec(naked) void sub_4836E0(void)
        {
```

```
__asm
{
            push    ebp
            mov     ebp, esp
            sub     esp, 0Ch
            push    ebx
            push    esi
            push    edi
            mov     eax, [ebp+8]
            push    eax
            call    ds:off_4DDCDC_2
            add     esp, 4
            mov     [ebp-4], eax
            mov     eax, [ebp-4]
            push    ecx
            mov     ecx, 800h
            mov     ecx, 4Bh
            not     ecx
            bswap   eax
            not     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            inc     ecx
            dec     ecx
            inc     ecx
            inc     cl
            inc     cl
            inc     cl
            add     ecx, 0Dh
            inc     cl
            inc     cl
            inc     cl
            inc     cl
            inc     cl
            add     ecx, 0Ah
            dec     ecx
            push    edx
            mov     edx, 4
            add     ecx, edx
            inc     ecx
```

```
                pop     edx
                bswap   eax
                add     ecx, 3
                and     eax, ecx
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9370
                xor     ecx, ds:dword_4D9374
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_483770
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_483770:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDC9C
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_487916(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD18
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 87h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93AC
                xor     ecx, ds:dword_4D93B0
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4879AE
                mov     edx, [ebp-8]
                or      edx, 1
```

```
                mov     [ebp-8], edx

 loc_4879AE:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_487445(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE4
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                dec     edi
                inc     esi
                dec     bh
                jz      short $+2
                dec     bh
                dec     bh
                dec     edi
                inc     esi
                dec     bh
                and     eax, 800h
                bswap   ecx
                jo      short $+2
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 86h
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     edi
                inc     esi
                dec     bl
                dec     bl
                dec     edi
                inc     esi
                dec     bl
                dec     bl
                dec     edi
                inc     esi
                sub     bl, 1Ah
                dec     bl
                dec     edi
                inc     esi
                sub     bl, 1Fh
                not     bx
                dec     edi
                inc     esi
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                and     eax, 0
                inc     eax
                dec     edi
                inc     esi
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9378
                xor     ecx, ds:dword_4D937C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4874E1
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_4874E1:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48C973(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE0
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                mov     ecx, 800h
                mov     ecx, 0Ch
                not     ecx
                bswap   eax
                not     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Dh
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Ah
                dec     ecx
                push    edx
                mov     edx, 4
                add     ecx, edx
                inc     ecx
                pop     edx
                bswap   eax
                and     eax, ecx
                pop     ecx
                pop     edx
                test    eax, eax
                jnz     loc_48CABE
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 4
                dec     ch
                sub     ch, 3
                dec     ch
                and     ah, ch
                mov     cl, 0AEh
                sub     cl, 2
                dec     cl
                dec     cl
                sub     cl, 6
                not     al
                bswap   ecx
                not     al
                bswap   ecx
                dec     cl
                dec     cl
                sub     cl, 10h
                dec     cl
                dec     cl
                add     cl, 0Ch
```

```
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 10h
                sub     cl, 1
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                and     al, cl
                pop     ecx
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_48CA7D
                jl      short loc_48CA7B

loc_48CA76:
                jmp     short loc_48CA7F

loc_48CA7B:
                jz      short loc_48CA76

loc_48CA7D:
                jmp     short loc_48CA76

loc_48CA7F:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                and     eax, 41h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1500h
                dec     dh
                sub     dh, 3
                dec     dh
                sub     dh, 7
                dec     dh
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_48CABE
                and     eax, 0
                inc     eax
                jmp     short loc_48CAC1

loc_48CABE:
                and     eax, 0

loc_48CAC1:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9374
                xor     ecx, ds:dword_4D9378
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48CAE4
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48CAE4:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_48640A(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF0
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     dh, 2
                dec     dh
                dec     dh
                and     ah, dh
                mov     dl, 0Eh
                sub     dl, 0FFh
                jo      short loc_48643D
                jl      short loc_48643B

 loc_486438:
                jmp     short loc_48643F

 loc_48643B:
                jz      short loc_486438

 loc_48643D:
                jmp     short loc_486438

 loc_48643F:
                sub     dl, 0FFh
                sub     dl, 0FFh
                sub     dl, 0Ah
                sub     dl, 0FFh
                sub     dl, 0FFh
                sub     dl, 5
                dec     dl
                dec     dl
                dec     dl
                sub     dl, 3
                sub     dl, 0FFh
                dec     dl
                inc     dl
                inc     dl
                inc     dl
                and     al, dl
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9384
                xor     ecx, ds:dword_4D9388
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48648B
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48648B:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_48B2BC(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD00
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
```

```
push    edx
mov     edx, 0FFFFh
and     eax, edx
push    ebx
push    eax
mov     bh, 7
dec     bh
dec     bh
dec     bh
xor     bh, 4
and     eax, 800h
bswap   ecx
pop     eax
bswap   ecx
and     ah, bh
mov     bl, 98h
sub     bl, 9
dec     bl
dec     bl
dec     bl
sub     bl, 0Ch
not     bx
bswap   eax
not     bx
bswap   eax
and     al, bl
mov     eax, eax
pop     ebx
neg     eax
sbb     eax, eax
inc     eax
pop     edx
push    eax
mov     eax, [ebp-4]
mov     edx, 0F00h
sub     dh, 1
dec     dh
dec     dh
sub     dh, 4
and     eax, edx
neg     eax
sbb     eax, eax
inc     eax
mov     edx, eax
pop     eax
xor     ecx, ecx
cmp     eax, edx
setz    cl
mov     al, cl
mov     [ebp-0Ch], eax
mov     ecx, ds:dword_4D9394
xor     ecx, ds:dword_4D9398
```

```
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48B365
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48B365:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_488F0A(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF8
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     dh, 6
                dec     dh
                jo      short loc_488F34
                jl      short loc_488F32

 loc_488F2F:
                jmp     short loc_488F36
```

```
  loc_488F32:
                  jz      short loc_488F2F

  loc_488F34:
                  jmp     short loc_488F2F

  loc_488F36:
                  sub     dh, 2
                  push    eax
                  mov     eax, 800h
                  bswap   eax
                  not     eax
                  pop     eax
                  sub     dh, 3
                  and     ah, dh
                  mov     dl, 4
                  dec     dl
                  sub     dl, 2
                  dec     dl
                  sub     dl, 0FFh
                  and     al, dl
                  not     ah
                  bswap   eax
                  bswap   eax
                  not     ah
                  pop     edx
                  mov     [ebp-0Ch], eax
                  mov     ecx, ds:dword_4D938C
                  xor     ecx, ds:dword_4D9390
                  shl     ecx, 1
                  mov     [ebp-8], ecx
                  cmp     dword ptr [ebp-0Ch], 0
                  jz      short loc_488F83
                  mov     edx, [ebp-8]
                  or      edx, 1
                  mov     [ebp-8], edx

  loc_488F83:
                  mov     eax, [ebp-8]
                  push    eax
                  call    ds:off_4DDCB8
                  add     esp, 4
                  pop     edi
                  pop     esi
                  pop     ebx
                  mov     esp, ebp
                  pop     ebp
                  retn
    }
}
```

```
__declspec(naked) void sub_48A1B3(void)
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 sub     esp, 0Ch
                 push    ebx
                 push    esi
                 push    edi
                 mov     eax, [ebp+8]
                 push    eax
                 call    ds:off_4DDCDC_2
                 add     esp, 4
                 mov     [ebp-4], eax
                 mov     eax, [ebp-4]
                 push    ebx
                 mov     ebx, [ebp+0Ch]
                 mov     ebx, 0FFFFh
                 and     eax, ebx
                 push    ecx
                 mov     ch, 2Dh
                 dec     ch
                 sub     ch, 1
                 sub     ch, 20h
                 dec     ch
                 dec     ch
                 sub     ch, 7
                 dec     ch
                 dec     ch
                 and     ah, ch
                 mov     cl, 0C0h
                 sub     cl, 9
                 dec     cl
                 dec     cl
                 dec     cl
                 not     cl
                 bswap   edx
                 not     cl
                 bswap   edx
                 dec     cl
                 dec     cl
                 dec     cl
                 dec     cl
                 push    eax
                 dec     cl
                 dec     cl
                 sub     cl, 12h
                 dec     cl
                 dec     cl
                 sub     cl, 3
                 dec     cl
                 and     eax, 40h
                 dec     cl
                 dec     cl
                 dec     cl
                 add     cl, 0Eh
                 dec     cl
                 dec     cl
                 and     eax, 80h
                 sub     cl, 1Fh
                 dec     cl
                 dec     cl
                 dec     cl
                 dec     cl
                 dec     cl
                 dec     cl
                 not     ecx
                 bswap   eax
                 not     ecx
                 bswap   eax
                 pop     eax
                 inc     cl
                 inc     cl
                 inc     cl
                 inc     cl
                 inc     cl
                 inc     cl
                 and     al, cl
                 mov     eax, eax
                 pop     ecx
                 neg     eax
                 sbb     eax, eax
                 inc     eax
                 pop     ebx
                 push    eax
                 mov     eax, [ebp-4]
                 mov     edx, 0C00h
                 sub     dh, 1
                 dec     dh
                 dec     dh
                 dec     dh
                 and     eax, edx
                 neg     eax
                 sbb     eax, eax
                 inc     eax
                 mov     edx, eax
                 pop     eax
                 xor     ecx, ecx
                 cmp     eax, edx
                 setz    cl
```

```
                mov     al, cl
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9370
                xor     ecx, ds:dword_4D9374
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48A2AA
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48A2AA:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDC9C
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_48C180(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF4
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                mov     ecx, 800h
                mov     ecx, 4Bh
                not     ecx
                bswap   eax
                not     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Dh
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Ah
                dec     ecx
                push    edx
                mov     edx, 4
                add     ecx, edx
                inc     ecx
                pop     edx
                bswap   eax
                add     ecx, 3
                and     eax, ecx
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                push    eax
                mov     eax, [ebp-4]
                mov     edx, 0E00h
                sub     dh, 1
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                and     eax, edx
                neg     eax
                sbb     eax, eax
```

```
                inc     eax
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                jo      short loc_48C218
                jl      short loc_48C216

loc_48C213:
                jmp     short loc_48C21A

loc_48C216:
                jz      short loc_48C213

loc_48C218:
                jmp     short loc_48C213

loc_48C21A:
                cmp     eax, edx
                jo      short loc_48C225
                jl      short loc_48C223

loc_48C220:
                jmp     short loc_48C227

loc_48C223:
                jz      short loc_48C220

loc_48C225:
                jmp     short loc_48C220

loc_48C227:
                jz      short loc_48C23A
                and     eax, 0
                jo      short loc_48C235
                jl      short loc_48C233

loc_48C230:
                jmp     short loc_48C237

loc_48C233:
                jz      short loc_48C230

loc_48C235:
                jmp     short loc_48C230

loc_48C237:
                inc     eax
                jmp     short loc_48C23D

loc_48C23A:
                and     eax, 0
```

```
 loc_48C23D:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9388
                xor     ecx, ds:dword_4D938C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48C260
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48C260:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_483349(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD14
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     dh, 2
                sub     dh, 0FFh
                dec     dh
                sub     dh, 0FFh
                dec     dh
                sub     dh, 0FFh
                sub     dh, 1
                sub     dh, 1
                dec     dh
                and     ah, dh
                mov     edx, 800h
                mov     dl, 0Fh
                sub     dl, 0FFh
                sub     dl, 0FFh
                sub     dl, 0FFh
                sub     dl, 0Ah
                sub     dl, 0FFh
                sub     dl, 0FFh
                sub     dl, 5
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                sub     dl, 0FFh
                dec     dl
                inc     dl
                inc     dl
                and     al, dl
                not     ah
                not     ah
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A8
                xor     ecx, ds:dword_4D93AC
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4833DA
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_4833DA:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}
```

```
__declspec(naked) void sub_48AF5D(void)
{
     __asm
     {
                  push    ebp
                  mov     ebp, esp
                  sub     esp, 0Ch
                  push    ebx
                  push    esi
                  push    edi
                  mov     eax, [ebp+8]
                  push    eax
                  call    ds:off_4DDD04
                  add     esp, 4
                  mov     [ebp-4], eax
                  mov     eax, [ebp-4]
                  jo      short loc_48AF82
                  jl      short loc_48AF80

 loc_48AF7D:
                  jmp     short loc_48AF84

 loc_48AF80:
                  jz      short loc_48AF7D

 loc_48AF82:
                  jmp     short loc_48AF7D

 loc_48AF84:
                  push    ebx
                  mov     ebx, 0FFFFh
                  and     eax, ebx
                  push    ecx
                  mov     ch, 2Ch
                  sub     ch, 1
                  sub     ch, 20h
                  dec     ch
                  dec     ch
                  sub     ch, 4
                  dec     ch
                  sub     ch, 3
                  dec     ch
                  and     ah, ch
                  mov     cl, 70h
                  sub     cl, 2
                  dec     cl
                  dec     cl
                  dec     cl
```

```
                sub     cl, 6
                not     al
                bswap   ecx
                not     al
                bswap   ecx
                dec     cl
                dec     cl
                sub     cl, 10h
                dec     cl
                dec     cl
                add     cl, 0Ch
                dec     cl
                dec     cl
                dec     cl
                jo      short loc_48AFD8
                jl      short loc_48AFD6

loc_48AFD3:
                jmp     short loc_48AFDA

loc_48AFD6:
                jz      short loc_48AFD3

loc_48AFD8:
                jmp     short loc_48AFD3

loc_48AFDA:
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 10h
                sub     cl, 1
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48B034
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48B034:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_483976(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD08
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                mov     ecx, 800h
                mov     ecx, 0Ch
                not     ecx
                bswap   eax
                jo      short loc_4839AC
                jl      short loc_4839AA

loc_4839A5:
                jmp     short loc_4839AE

loc_4839AA:
                jz      short loc_4839A5

loc_4839AC:
                jmp     short loc_4839A5

loc_4839AE:
                not     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                not     ecx
                not     ecx
                inc     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Dh
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Ah
                dec     ecx
                push    edx
                mov     edx, 4
                add     ecx, edx
                inc     ecx
                pop     edx
                bswap   eax
                jo      short loc_4839FD
                jl      short loc_4839FB

loc_4839F6:
                jmp     short loc_4839FF

loc_4839FB:
                jz      short loc_4839F6

loc_4839FD:
                jmp     short loc_4839F6

loc_4839FF:
                and     eax, ecx
                pop     ecx
                pop     edx
                test    eax, eax
                jnz     loc_483B28
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 4
                dec     ch
                sub     ch, 3
                dec     ch
                and     ah, ch
                mov     cl, 0AEh
                sub     cl, 2
                dec     cl
                dec     cl
                sub     cl, 6
                not     al
                bswap   ecx
                not     al
                bswap   ecx
                dec     cl
                dec     cl
                sub     cl, 10h
                dec     cl
                dec     cl
                add     cl, 0Ch
                dec     cl
                dec     cl
                jo      short loc_483A60
                jl      short loc_483A5E

loc_483A59:
                jmp     short loc_483A62

loc_483A5E:
                jz      short loc_483A59

loc_483A60:
                jmp     short loc_483A59

loc_483A62:
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                jo      short loc_483A75
                jl      short loc_483A73

loc_483A6E:
                jmp     short loc_483A77

loc_483A73:
                jz      short loc_483A6E

loc_483A75:
                jmp     short loc_483A6E

loc_483A77:
                sub     cl, 10h
                sub     cl, 1
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                jo      short loc_483AA5
                jl      short loc_483AA3

loc_483A9E:
                jmp     short loc_483AA7

loc_483AA3:
                jz      short loc_483A9E

loc_483AA5:
                jmp     short loc_483A9E

loc_483AA7:
                and     al, cl
                jo      short loc_483AB4
                jl      short loc_483AB2

loc_483AAD:
                jmp     short loc_483AB6

loc_483AB2:
                jz      short loc_483AAD

loc_483AB4:
                jmp     short loc_483AAD

loc_483AB6:
                pop     ecx
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_483ADA
                jl      short loc_483AD8

loc_483AD3:
                jmp     short loc_483ADC

loc_483AD8:
                jz      short loc_483AD3

loc_483ADA:
                jmp     short loc_483AD3
```

```
loc_483ADC:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                and     eax, 41h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1500h
                dec     dh
                sub     dh, 7
                dec     dh
                sub     dh, 3
                dec     dh
                jo      short loc_483B12
                jl      short loc_483B10

loc_483B0B:
                jmp     short loc_483B14

loc_483B10:
                jz      short loc_483B0B

loc_483B12:
                jmp     short loc_483B0B

loc_483B14:
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_483B28
                and     eax, 0
                inc     eax
                jmp     short loc_483B2B

loc_483B28:
                and     eax, 0

loc_483B2B:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D939C
                xor     ecx, ds:dword_4D93A0
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_483B4E
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_483B4E:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_48CF27(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD00
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    0Ah
                pop     ebx
                dec     bl
                dec     bl
                dec     bl
                add     bl, 0FFh
                add     bl, 0FFh
                dec     bl
                jo      short loc_48CF66
                jl      short loc_48CF64

loc_48CF61:
                jmp     short loc_48CF68

loc_48CF64:
                jz      short loc_48CF61

loc_48CF66:
                jmp     short loc_48CF61

loc_48CF68:
                add     bl, 0FFh
                add     bl, 0FFh
                add     bl, 0FFh
                add     bl, 0FFh
                and     al, bl
                jo      short loc_48CF7F
                jl      short loc_48CF7D

loc_48CF7A:
                jmp     short loc_48CF81

loc_48CF7D:
                jz      short loc_48CF7A

loc_48CF7F:
                jmp     short loc_48CF7A

loc_48CF81:
                mov     dh, 15h
                and     dl, 0
                dec     dh
                sub     dh, 6
                dec     dh
                dec     dh
                dec     dh
                sub     dh, 1
                dec     dh
                dec     dh
                and     ah, dh
                pop     ebx
                pop     edx
                test    eax, eax
                jz      short loc_48CFA8
                not     eax
                add     eax, 1
                stc
                jmp     short loc_48CFAE
```

```
 loc_48CFA8:
                not     eax
                add     eax, 1
                clc

 loc_48CFAE:
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9394
                xor     ecx, ds:dword_4D9398
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48CFD5
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48CFD5:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}


__declspec(naked) void sub_489832(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE4
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 8
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 98h
                sub     bl, 4
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 0Ch
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9378
                xor     ecx, ds:dword_4D937C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4898C7
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_4898C7:
                mov     eax, [ebp-8]
```

```
                        push    eax
                        call    ds:off_4DDCA4
                        add     esp, 4
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
            }
}


__declspec(naked) void sub_48BB6B(void)
{
            __asm
            {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDCF8
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    edx
                        mov     dh, 3
                        dec     dh
                        jo      short loc_48BB95
                        jl      short loc_48BB93

  loc_48BB90:
                        jmp     short loc_48BB97

  loc_48BB93:
                        jz      short loc_48BB90

  loc_48BB95:
                        jmp     short loc_48BB90

  loc_48BB97:
                        push    eax
                        and     eax, 80h
                        bswap   eax
                        not     eax
                        pop     eax
                        sub     dh, 2
                        jo      short loc_48BBAE
```

```
                jl      short loc_48BBAC

loc_48BBA9:
                jmp     short loc_48BBB0

loc_48BBAC:
                jz      short loc_48BBA9

loc_48BBAE:
                jmp     short loc_48BBA9

loc_48BBB0:
                and     ah, dh
                mov     dl, 4
                dec     dl
                sub     dl, 2
                dec     dl
                sub     dl, 0FFh
                and     al, dl
                not     ah
                bswap   eax
                bswap   eax
                not     ah
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D938C
                xor     ecx, ds:dword_4D9390
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48BBF1
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48BBF1:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}
```

```
__declspec(naked) void sub_48D1F8(void)
{
     __asm
     {
                  push    ebp
                  mov     ebp, esp
                  sub     esp, 0Ch
                  push    ebx
                  push    esi
                  push    edi
                  mov     eax, [ebp+8]
                  push    eax
                  call    ds:off_4DDCEC
                  add     esp, 4
                  mov     [ebp-4], eax
                  mov     eax, [ebp-4]
                  push    ecx
                  mov     ecx, 800h
                  mov     ecx, 52h
                  not     ecx
                  bswap   eax
                  not     ecx
                  inc     ecx
                  inc     ecx
                  inc     ecx
                  inc     ecx
                  inc     ecx
                  inc     ecx
                  sub     ecx, 6
                  inc     ecx
                  inc     ecx
                  inc     ecx
                  inc     ecx
                  inc     ecx
                  inc     ecx
                  inc     ecx
                  inc     ecx
                  dec     ecx
                  inc     ecx
                  inc     cl
                  inc     cl
                  inc     cl
                  add     ecx, 0Ch
                  inc     ecx
                  inc     cl
                  inc     cl
                  add     ecx, 0Fh
                  inc     cl
                  inc     cl
```

```
                add     cl, 2
                add     ecx, 0Ah
                dec     ecx
                push    edx
                mov     edx, 4
                sub     ecx, edx
                dec     ecx
                pop     edx
                bswap   eax
                sub     ecx, 3
                and     eax, ecx
                pop     ecx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9380
                xor     ecx, ds:dword_4D9384
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48D290
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48D290:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCAC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_4832AE(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD0C
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_4832E1
                jl      short loc_4832DF

loc_4832DA:
                jmp     short loc_4832E3

loc_4832DF:
                jz      short loc_4832DA

loc_4832E1:
                jmp     short loc_4832DA

loc_4832E3:
                sub     bl, 6
                push    eax
                dec     bl
                dec     bl
                and     eax, 41h
                sub     bl, 15h
                pop     eax
                dec     bl
                dec     bl
                and     al, bl
                mov     edx, 1500h
                dec     dh
                sub     dh, 3
                dec     dh
                sub     dh, 7
                dec     dh
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A0
                xor     ecx, ds:dword_4D93A4
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_483335
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_483335:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCCC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_48B761(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD0C
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, [ebp+0Ch]
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Fh
                dec     ch
                dec     ch
                dec     ch
                sub     ch, 1
                sub     ch, 15h
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                dec     ch
                sub     ch, 7
                dec     ch
                dec     ch
                and     ah, ch
                mov     cl, 0BDh
                sub     cl, 2
                dec     cl
                inc     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                inc     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                inc     cl
                dec     cl
                not     cl
                bswap   edx
                not     cl
                bswap   edx
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                push    eax
                dec     cl
                dec     cl
                sub     cl, 13h
                dec     cl
                sub     cl, 3
                dec     cl
                and     eax, 41h
                dec     cl
                dec     cl
                dec     cl
                add     cl, 0Dh
                dec     cl
                and     eax, 80h
```

```
                sub     cl, 22h
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                pop     eax
                inc     cl
                inc     cl
                inc     cl
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A0
                xor     ecx, ds:dword_4D93A4
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48B84B
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48B84B:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCCC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48D098(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD18
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 800h
                jmp     short loc_48D0C1
                mov     ebx, 80h

loc_48D0C1:
                mov     ebx, 72h
                not     ebx
                bswap   eax
                not     ebx
                inc     ebx
                inc     ebx
                add     ebx, 8
                dec     ebx
                push    ecx
                mov     ecx, 4
                add     ebx, ecx
                inc     ebx
                pop     ecx
                bswap   eax
                and     eax, ebx
                pop     ebx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93AC
                xor     ecx, ds:dword_4D93B0
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48D10B
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48D10B:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD8
                add     esp, 4
                pop     edi
                pop     esi
```

```
                    pop     ebx
                    mov     esp, ebp
                    pop     ebp
                    retn
        }
}



__declspec(naked) void sub_47F9AE(void)
{
        __asm
        {
                    push    ebp
                    mov     ebp, esp
                    sub     esp, 0Ch
                    push    ebx
                    push    esi
                    push    edi
                    mov     eax, [ebp+8]
                    push    eax
                    call    ds:off_4DDCF8
                    add     esp, 4
                    mov     [ebp-4], eax
                    mov     eax, [ebp-4]
                    push    ebx
                    mov     ebx, 0FFFFh
                    and     eax, ebx
                    push    ecx
                    mov     ch, 2Ch
                    sub     ch, 1
                    sub     ch, 20h
                    dec     ch
                    dec     ch
                    sub     ch, 4
                    dec     ch
                    sub     ch, 3
                    dec     ch
                    and     ah, ch
                    mov     cl, 0AEh
                    sub     cl, 2
                    dec     cl
                    dec     cl
                    sub     cl, 6
                    not     al
                    bswap   ecx
                    not     al
                    bswap   ecx
                    dec     cl
                    dec     cl
                    sub     cl, 10h
                    dec     cl
```

```
                dec     cl
                add     cl, 0Ch
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 10h
                sub     cl, 1
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D938C
                xor     ecx, ds:dword_4D9390
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_47FA6B
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_47FA6B:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
```

```
                        retn
        }
}





__declspec(naked) void sub_484A99(void)
{
        __asm
        {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDCE4
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    ebx
                        mov     ebx, 80h
                        jmp     short loc_484AC2
                        mov     ebx, 40h

 loc_484AC2:
                        mov     ebx, 71h
                        not     ebx
                        bswap   eax
                        not     ebx
                        inc     ebx
                        inc     ebx
                        inc     ebx
                        add     ebx, 7
                        push    ecx
                        mov     ecx, 4
                        add     ebx, ecx
                        inc     ebx
                        pop     ecx
                        bswap   eax
                        and     eax, ebx
                        pop     ebx
                        neg     eax
                        sbb     eax, eax
                        neg     eax
                        pop     edx
                        mov     [ebp-0Ch], eax
                        mov     ecx, ds:dword_4D9378
```

```
                xor     ecx, ds:dword_4D937C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_484B0C
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_484B0C:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_48B9EF(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD08
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                bswap   ecx
                not     ecx
                push    eax
                not     eax
                mov     eax, 80h
                xchg    eax, ecx
                mov     ecx, 1
                xchg    eax, ecx
                not     eax
                and     eax, 41h
                pop     eax
                not     ecx
                pop     ecx
                push    edx
                mov     dh, 18h
                dec     dh
                dec     dh
                not     ecx
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                bswap   eax
                dec     dh
                dec     dh
                sub     dh, 0Dh
                dec     dh
                dec     dh
                dec     dh
                bswap   eax
                jo      short loc_48BA53
                jl      short loc_48BA51

loc_48BA4E:
                jmp     short loc_48BA55

loc_48BA51:
                jz      short loc_48BA4E

loc_48BA53:
                jmp     short loc_48BA4E

loc_48BA55:
                and     ah, dh
                mov     dl, 9
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                add     dl, 4
                sub     dl, 3
                dec     dl
                jo      short loc_48BA7A
                jl      short loc_48BA78

loc_48BA75:
                jmp     short loc_48BA7C
```

```
 loc_48BA78:
                jz      short loc_48BA75

 loc_48BA7A:
                jmp     short loc_48BA75

 loc_48BA7C:
                and     al, dl
                jo      short loc_48BA87
                jl      short loc_48BA85

 loc_48BA82:
                jmp     short loc_48BA89

 loc_48BA85:
                jz      short loc_48BA82

 loc_48BA87:
                jmp     short loc_48BA82

 loc_48BA89:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D939C
                xor     ecx, ds:dword_4D93A0
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48BAAC
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48BAAC:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_488B2B(void)
{
     __asm
```

```
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD04
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                jo      short loc_488B50
                jl      short loc_488B4E

loc_488B4B:
                jmp     short loc_488B52

loc_488B4E:
                jz      short loc_488B4B

loc_488B50:
                jmp     short loc_488B4B

loc_488B52:
                push    ebx
                mov     ebx, [ebp+0Ch]
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 10h
                dec     ch
                dec     ch
                sub     ch, 4
                dec     ch
                sub     ch, 3
                dec     ch
                and     ah, ch
                mov     cl, 70h
                sub     cl, 5
                sub     cl, 6
                not     al
                bswap   ecx
                not     al
                bswap   ecx
                dec     cl
                dec     cl
                sub     cl, 10h
                dec     cl
                dec     cl
                add     cl, 0Ch
                dec     cl
                dec     cl
                dec     cl
                jo      short loc_488BA3
                jl      short loc_488BA1

loc_488B9E:
                jmp     short loc_488BA5

loc_488BA1:
                jz      short loc_488B9E

loc_488BA3:
                jmp     short loc_488B9E

loc_488BA5:
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 10h
                sub     cl, 1
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_488BFE
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_488BFE:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48CDAE(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD08
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 98h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 0Ch
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                push    eax
                mov     eax, [ebp-4]
                mov     edx, 0F00h
                sub     dh, 1
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                dec     dh
                and     eax, edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                jo      short loc_48CE48
                jl      short loc_48CE46

loc_48CE43:
                jmp     short loc_48CE4A

loc_48CE46:
                jz      short loc_48CE43

loc_48CE48:
                jmp     short loc_48CE43
```

```
 loc_48CE4A:
                cmp     eax, edx
                jo      short loc_48CE55
                jl      short loc_48CE53

 loc_48CE50:
                jmp     short loc_48CE57

 loc_48CE53:
                jz      short loc_48CE50

 loc_48CE55:
                jmp     short loc_48CE50

 loc_48CE57:
                setnz   cl
                mov     al, cl
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D939C
                xor     ecx, ds:dword_4D93A0
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48CE7F
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48CE7F:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_47FF6D(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD10
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1E00h
                pop     ebx
                jo      short loc_47FFA3
                jl      short loc_47FFA1

loc_47FF9C:
                jmp     short loc_47FFA5

loc_47FFA1:
                jz      short loc_47FF9C

loc_47FFA3:
                jmp     short loc_47FF9C

loc_47FFA5:
                sub     bh, 4
                dec     bh
                push    eax
                dec     bh
                dec     bh
                jo      short loc_47FFB8
                jl      short loc_47FFB6

loc_47FFB3:
                jmp     short loc_47FFBA

loc_47FFB6:
                jz      short loc_47FFB3

loc_47FFB8:
                jmp     short loc_47FFB3

loc_47FFBA:
                and     eax, 40h
                dec     bh
                sub     bh, 12h
                sub     bh, 3
                pop     eax
                dec     bh
                and     ah, bh
                mov     edx, 12h
                dec     dl
                sub     dl, 1
                dec     dl
                sub     dl, 7
                dec     dl
                dec     dl
                dec     dl
                dec     dl
                and     al, dl
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A4
                xor     ecx, ds:dword_4D93A8
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48000E
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48000E:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48BC05(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
```

```
push    ebx
push    esi
push    edi
mov     eax, [ebp+8]
push    eax
call    ds:off_4DDD18
add     esp, 4
mov     [ebp-4], eax
mov     eax, [ebp-4]
push    ecx
mov     ecx, 800h
mov     ecx, 0Dh
not     ecx
bswap   eax
not     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
dec     ecx
inc     ecx
inc     cl
inc     cl
inc     cl
add     ecx, 0Dh
inc     cl
inc     cl
inc     cl
inc     cl
inc     cl
add     ecx, 0Ah
dec     ecx
push    edx
mov     edx, 4
add     ecx, edx
inc     ecx
pop     edx
bswap   eax
and     eax, ecx
```

```
                    pop     ecx
                    neg     eax
                    sbb     eax, eax
                    inc     eax
                    pop     edx
                    mov     [ebp-0Ch], eax
                    mov     ecx, ds:dword_4D93AC
                    xor     ecx, ds:dword_4D93B0
                    shl     ecx, 1
                    mov     [ebp-8], ecx
                    cmp     dword ptr [ebp-0Ch], 0
                    jz      short loc_48BC94
                    mov     edx, [ebp-8]
                    or      edx, 1
                    mov     [ebp-8], edx

 loc_48BC94:
                    mov     eax, [ebp-8]
                    push    eax
                    call    ds:off_4DDCD8
                    add     esp, 4
                    pop     edi
                    pop     esi
                    pop     ebx
                    mov     esp, ebp
                    pop     ebp
                    retn
     }
}



__declspec(naked) void sub_483B62(void)
{
     __asm
     {
                    push    ebp
                    mov     ebp, esp
                    sub     esp, 0Ch
                    push    ebx
                    push    esi
                    push    edi
                    mov     eax, [ebp+8]
                    push    eax
                    call    ds:off_4DDD14
                    add     esp, 4
                    mov     [ebp-4], eax
                    mov     eax, [ebp-4]
                    push    eax
                    mov     eax, 4
                    bswap   eax
```

```
                not     eax
                pop     eax
                push    edx
                mov     dh, 80h
                mov     dh, 0
                inc     dh
                mov     ecx, ecx
                inc     dh
                inc     dh
                dec     edi
                inc     dh
                dec     edi
                inc     dh
                jo      short loc_483BA5
                jl      short loc_483BA3

loc_483BA0:
                jmp     short loc_483BA7

loc_483BA3:
                jz      short loc_483BA0

loc_483BA5:
                jmp     short loc_483BA0

loc_483BA7:
                inc     dh
                push    ecx
                bswap   ecx
                not     ecx
                push    eax
                not     eax
                mov     eax, 80h
                xchg    eax, ecx
                mov     ecx, 41h
                xchg    eax, ecx
                not     eax
                pop     eax
                dec     edi
                not     ecx
                pop     ecx
                inc     dh
                dec     edi
                inc     dh
                and     ebx, 800h
                inc     dh
                dec     edi
                inc     dh
                dec     edi
                inc     dh
                dec     edi
                inc     dh
                dec     edi
                and     ebx, 10h
                inc     dh
                inc     dh
                jo      short loc_483BEB
                jl      short loc_483BE9

loc_483BE6:
                jmp     short loc_483BED

loc_483BE9:
                jz      short loc_483BE6

loc_483BEB:
                jmp     short loc_483BE6

loc_483BED:
                sub     dh, 0Dh
                dec     dh
                and     ah, dh
                mov     dl, 5
                sub     dl, 0FFh
                dec     dl
                dec     edi
                dec     dl
                dec     dl
                dec     edi
                sub     dl, 0FFh
                dec     dl
                dec     dl
                dec     dl
                and     al, dl
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A8
                xor     ecx, ds:dword_4D93AC
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_483C30
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_483C30:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48854E(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF4
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 86h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9388
                xor     ecx, ds:dword_4D938C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4885E4
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_4885E4:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48912A(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE8
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                jo      short loc_48914F
                jl      short loc_48914D

loc_48914A:
                jmp     short loc_489151

loc_48914D:
                jz      short loc_48914A

loc_48914F:
                jmp     short loc_48914A

loc_489151:
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 4
                dec     ch
                sub     ch, 3
                dec     ch
                and     ah, ch
                mov     cl, 70h
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 6
                not     al
                bswap   ecx
                not     al
                bswap   ecx
                dec     cl
                dec     cl
                sub     cl, 10h
                dec     cl
                dec     cl
                add     cl, 0Ch
                dec     cl
                dec     cl
                dec     cl
                jo      short loc_4891A5
                jl      short loc_4891A3

loc_4891A0:
                jmp     short loc_4891A7

loc_4891A3:
                jz      short loc_4891A0

loc_4891A5:
                jmp     short loc_4891A0

loc_4891A7:
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 10h
                sub     cl, 1
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D937C
                xor     ecx, ds:dword_4D9380
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_489200
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_489200:
```

```
                        mov     eax, [ebp-8]
                        push    eax
                        call    ds:off_4DDCA8
                        add     esp, 4
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
    }
}




__declspec(naked) void sub_4888F0(void)
{
    __asm
    {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDD04
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    ebx
                        mov     ebx, 0FFFFh
                        and     eax, ebx
                        push    ecx
                        mov     ch, 2Dh
                        dec     ch
                        sub     ch, 1
                        sub     ch, 20h
                        dec     ch
                        dec     ch
                        sub     ch, 8
                        dec     ch
                        and     ah, ch
                        mov     cl, 0BEh
                        sub     cl, 2
                        dec     cl
                        dec     cl
                        dec     cl
                        dec     cl
                        dec     cl
                        dec     cl
                        dec     cl
                        dec     cl
                        not     cl
                        bswap   edx
                        not     cl
                        bswap   edx
                        dec     cl
                        dec     cl
                        dec     cl
                        dec     cl
                        push    eax
                        dec     cl
                        dec     cl
                        sub     cl, 12h
                        dec     cl
                        dec     cl
                        sub     cl, 3
                        dec     cl
                        and     eax, 80h
                        dec     cl
                        dec     cl
                        dec     cl
                        add     cl, 0Eh
                        dec     cl
                        dec     cl
                        and     eax, 800h
                        sub     cl, 1Fh
                        not     ecx
                        bswap   eax
                        not     ecx
                        bswap   eax
                        pop     eax
                        and     al, cl
                        mov     eax, eax
                        pop     ecx
                        neg     eax
                        sbb     eax, eax
                        inc     eax
                        pop     ebx
                        push    eax
                        mov     eax, [ebp-4]
                        mov     edx, 0C00h
                        sub     dh, 1
                        dec     dh
                        dec     dh
                        dec     dh
                        and     eax, edx
                        neg     eax
                        sbb     eax, eax
                        inc     eax
```

```
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                cmp     eax, edx
                jo      short loc_4889B7
                jl      short loc_4889B5

 loc_4889B2:
                jmp     short loc_4889B9

 loc_4889B5:
                jz      short loc_4889B2

 loc_4889B7:
                jmp     short loc_4889B2

 loc_4889B9:
                setnz   cl
                mov     al, cl
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4889E1
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_4889E1:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_482B40(void)
{
    __asm
    {
```

```
push    ebp
mov     ebp, esp
sub     esp, 0Ch
push    ebx
push    esi
push    edi
mov     eax, [ebp+8]
push    eax
call    ds:off_4DDCEC
add     esp, 4
mov     [ebp-4], eax
mov     eax, [ebp-4]
push    edx
mov     edx, 0FFFFh
and     eax, edx
push    ebx
push    eax
mov     bh, 7
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
and     eax, 800h
bswap   ecx
pop     eax
bswap   ecx
and     ah, bh
mov     bl, 86h
sub     bl, 5
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
sub     bl, 1Ah
dec     bl
sub     bl, 1Fh
not     bx
bswap   eax
not     bx
bswap   eax
and     al, bl
mov     eax, eax
test    eax, eax
jnz     loc_482C64
pop     ebx
pop     edx
```

```
mov     eax, [ebp-4]
push    edx
mov     edx, 0FFFFh
and     eax, edx
push    ebx
push    eax
mov     bh, 7
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
and     eax, 800h
bswap   ecx
pop     eax
bswap   ecx
and     ah, bh
mov     bl, 98h
sub     bl, 5
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
sub     bl, 0Ch
not     bx
bswap   eax
not     bx
bswap   eax
and     al, bl
mov     eax, eax
pop     ebx
neg     eax
sbb     eax, eax
inc     eax
pop     edx
mov     ecx, eax
push    ecx
mov     eax, [ebp-4]
push    edx
mov     edx, 0FFFFh
and     eax, edx
push    ebx
push    0Dh
pop     ebx
jo      short loc_482C26
jl      short loc_482C24
```

```
loc_482C1F:
                jmp     short loc_482C28

loc_482C24:
                jz      short loc_482C1F

loc_482C26:
                jmp     short loc_482C1F

loc_482C28:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                and     eax, 41h
                dec     bl
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 2500h
                dec     dh
                sub     dh, 3
                dec     dh
                sub     dh, 17h
                dec     dh
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_482C64
                and     eax, 0
                inc     eax
                jmp     short loc_482C67

loc_482C64:
                and     eax, 0

loc_482C67:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9380
                xor     ecx, ds:dword_4D9384
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_482C8A
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_482C8A:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCAC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_485E46(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF8
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                mov     ecx, 41h
                mov     ecx, 0Dh
                not     ecx
                bswap   eax
                not     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                jo      short loc_485E81
                jl      short loc_485E7F

 loc_485E7C:
                jmp     short loc_485E83
```

```
loc_485E7F:
                jz      short loc_485E7C

loc_485E81:
                jmp     short loc_485E7C

loc_485E83:
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                jo      short loc_485E98
                jl      short loc_485E96

loc_485E93:
                jmp     short loc_485E9A

loc_485E96:
                jz      short loc_485E93

loc_485E98:
                jmp     short loc_485E93

loc_485E9A:
                dec     ecx
                inc     ecx
                add     cl, 3
                add     ecx, 0Dh
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 9
                push    edx
                mov     edx, 4
                add     ecx, edx
                inc     ecx
                pop     edx
                bswap   eax
                and     eax, ecx
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D938C
                xor     ecx, ds:dword_4D9390
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_485EE7
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_485EE7:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_47F8EF(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF0
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                jo      short loc_47F914
                jl      short loc_47F912

 loc_47F90F:
```

```
                jmp     short loc_47F916

loc_47F912:
                jz      short loc_47F90F

loc_47F914:
                jmp     short loc_47F90F

loc_47F916:
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                jo      short loc_47F945
                jl      short loc_47F943

loc_47F940:
                jmp     short loc_47F947

loc_47F943:
                jz      short loc_47F940

loc_47F945:
                jmp     short loc_47F940

loc_47F947:
                mov     bl, 0C6h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9384
                xor     ecx, ds:dword_4D9388
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_47F99A
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_47F99A:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_487D4D(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD0C
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    0Dh
                pop     ebx
                jo      short loc_487D80
                jl      short loc_487D7E

loc_487D79:
                jmp     short loc_487D82

loc_487D7E:
                jz      short loc_487D79

loc_487D80:
                jmp     short loc_487D79

loc_487D82:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                and     eax, 41h
                dec     bl
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 2500h
                dec     dh
                sub     dh, 3
                dec     dh
                sub     dh, 17h
                dec     dh
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A0
                xor     ecx, ds:dword_4D93A4
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_487DD6
                mov     edx, [ebp-8]
```

```
                    or      edx, 1
                    mov     [ebp-8], edx

  loc_487DD6:
                    mov     eax, [ebp-8]
                    push    eax
                    call    ds:off_4DDCCC
                    add     esp, 4
                    pop     edi
                    pop     esi
                    pop     ebx
                    mov     esp, ebp
                    pop     ebp
                    retn
     }
}




__declspec(naked) void sub_4875A3(void)
{
     __asm
     {
                    push    ebp
                    mov     ebp, esp
                    sub     esp, 0Ch
                    push    ebx
                    push    esi
                    push    edi
                    mov     eax, [ebp+8]
                    push    eax
                    call    ds:off_4DDD18
                    add     esp, 4
                    mov     [ebp-4], eax
                    mov     eax, [ebp-4]
                    push    edx
                    mov     edx, [ebp+0Ch]
                    mov     edx, 0FFFFh
                    and     eax, edx
                    push    ebx
                    push    100h
                    pop     ebx
                    dec     bh
                    jo      short loc_4875DC
                    jl      short loc_4875DA

  loc_4875D7:
                    jmp     short loc_4875DE

  loc_4875DA:
                    jz      short loc_4875D7
```

```
loc_4875DC:
                jmp     short loc_4875D7

loc_4875DE:
                add     bh, 0FFh
                add     bh, 0FFh
                add     bh, 0FFh
                add     bh, 0FFh
                inc     bh
                inc     bh
                inc     bh
                inc     bh
                and     ah, bh
                jo      short loc_4875FD
                jl      short loc_4875FB

loc_4875F8:
                jmp     short loc_4875FF

loc_4875FB:
                jz      short loc_4875F8

loc_4875FD:
                jmp     short loc_4875F8

loc_4875FF:
                mov     bl, 17h
                sub     bl, 9
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 4
                dec     bl
                dec     bl
                dec     bl
                and     al, bl
                pop     ebx
                pop     edx
                test    eax, eax
                jz      short loc_487623
                not     eax
                add     eax, 1
                stc
                jmp     short loc_487629

loc_487623:
                not     eax
                add     eax, 1
                clc

loc_487629:
```

```
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_487638
                jl      short loc_487636

loc_487631:
                jmp     short loc_48763A

loc_487636:
                jz      short loc_487631

loc_487638:
                jmp     short loc_487631

loc_48763A:
                inc     eax
                dec     eax
                jo      short loc_487647
                jl      short loc_487645

loc_487640:
                jmp     short loc_487649

loc_487645:
                jz      short loc_487640

loc_487647:
                jmp     short loc_487640

loc_487649:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_487658
                jl      short loc_487656

loc_487651:
                jmp     short loc_48765A

loc_487656:
                jz      short loc_487651

loc_487658:
                jmp     short loc_487651

loc_48765A:
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93AC
                xor     ecx, ds:dword_4D93B0
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48767E
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48767E:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48C3D7(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF0
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     dh, 2
                sub     dh, 0FFh
                dec     dh
                sub     dh, 0FFh
                dec     dh
                sub     dh, 0FFh
                sub     dh, 1
                sub     dh, 1
                dec     dh
                and     ah, dh
                mov     edx, 800h
                mov     dl, 0Fh
                sub     dl, 0FFh
                sub     dl, 0FFh
                sub     dl, 0FFh
                sub     dl, 0Ah
                sub     dl, 0FFh
                sub     dl, 0FFh
                sub     dl, 5
                dec     dl
                dec     dl
                dec     dl
                sub     dl, 3
                sub     dl, 0FFh
                dec     dl
                inc     dl
                inc     dl
                and     al, dl
                not     ah
                not     ah
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9384
                xor     ecx, ds:dword_4D9388
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48C465
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48C465:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_48D7F5(void)
{
     __asm
     {
```

```
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD04
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    2
                pop     ebx
                dec     bl
                dec     bl
                and     al, bl
                mov     dh, 0Eh
                and     dl, 0
                sub     dh, 4
                dec     dh
                sub     dh, 1
                and     ah, dh
                pop     ebx
                pop     edx
                test    eax, eax
                jz      short loc_48D840
                not     eax
                add     eax, 1
                stc
                jmp     short loc_48D846

loc_48D840:
                not     eax
                add     eax, 1
                clc

loc_48D846:
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_48D855
                jl      short loc_48D853

loc_48D84E:
                jmp     short loc_48D857

loc_48D853:
```

```
                jz      short loc_48D84E

loc_48D855:
                jmp     short loc_48D84E

loc_48D857:
                inc     eax
                dec     eax
                jo      short loc_48D864
                jl      short loc_48D862

loc_48D85D:
                jmp     short loc_48D866

loc_48D862:
                jz      short loc_48D85D

loc_48D864:
                jmp     short loc_48D85D

loc_48D866:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_48D875
                jl      short loc_48D873

loc_48D86E:
                jmp     short loc_48D877

loc_48D873:
                jz      short loc_48D86E

loc_48D875:
                jmp     short loc_48D86E

loc_48D877:
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48D89B
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48D89B:
                mov     eax, [ebp-8]
```

```
                 push    eax
                 call    ds:off_4DDCC4
                 add     esp, 4
                 pop     edi
                 pop     esi
                 pop     ebx
                 mov     esp, ebp
                 pop     ebp
                 retn
     }
}



__declspec(naked) void sub_4874F5(void)
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 sub     esp, 0Ch
                 push    ebx
                 push    esi
                 push    edi
                 mov     eax, [ebp+8]
                 push    eax
                 call    ds:off_4DDD04
                 add     esp, 4
                 mov     [ebp-4], eax
                 mov     eax, [ebp-4]
                 push    ecx
                 mov     ecx, 800h
                 mov     ecx, 4
                 not     ecx
                 bswap   eax
                 not     ecx
                 inc     ecx
                 inc     ecx
                 inc     ecx
                 inc     ecx
                 inc     ecx
                 inc     ecx
                 inc     ecx
                 inc     ecx
                 inc     ecx
                 inc     ecx
                 inc     ecx
                 inc     ecx
                 inc     ecx
                 inc     ecx
                 inc     ecx
```

```
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 3
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Ah
                inc     cl
                inc     cl
                add     ecx, 0Ah
                dec     ecx
                push    edx
                mov     edx, 4
                add     ecx, edx
                inc     ecx
                pop     edx
                bswap   eax
                and     eax, ecx
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48758F
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48758F:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_486A5F(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF4
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                jo      short loc_486A84
                jl      short loc_486A82

 loc_486A7F:
                jmp     short loc_486A86

 loc_486A82:
                jz      short loc_486A7F

 loc_486A84:
                jmp     short loc_486A7F

 loc_486A86:
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 4
                dec     ch
                sub     ch, 3
                dec     ch
                and     ah, ch
                mov     cl, 70h
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 6
                not     al
                bswap   ecx
                not     al
                bswap   ecx
                dec     cl
                dec     cl
                sub     cl, 10h
                dec     cl
                dec     cl
                add     cl, 0Ch
                dec     cl
                dec     cl
                dec     cl
                jo      short loc_486ADA
                jl      short loc_486AD8

loc_486AD5:
                jmp     short loc_486ADC

loc_486AD8:
                jz      short loc_486AD5

loc_486ADA:
                jmp     short loc_486AD5

loc_486ADC:
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 10h
                sub     cl, 1
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                and     al, cl
                mov     eax, eax
                pop     ecx
                pop     ebx
                test    eax, eax
                jnz     loc_486C18
                mov     eax, [ebp-4]
                jo      short loc_486B21
                jl      short loc_486B1F

loc_486B1C:
                jmp     short loc_486B23

loc_486B1F:
                jz      short loc_486B1C

loc_486B21:
                jmp     short loc_486B1C

loc_486B23:
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 7
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                jo      short loc_486B52
                jl      short loc_486B50

loc_486B4D:
                jmp     short loc_486B54

loc_486B50:
                jz      short loc_486B4D

loc_486B52:
                jmp     short loc_486B4D
```

```
loc_486B54:
                mov     bl, 0C0h
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     ecx, eax
                push    ecx
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_486B9C
                jl      short loc_486B9A

loc_486B95:
                jmp     short loc_486B9E

loc_486B9A:
                jz      short loc_486B95

loc_486B9C:
                jmp     short loc_486B95

loc_486B9E:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                jo      short loc_486BB1
                jl      short loc_486BAF
```

```
loc_486BAC:
                jmp     short loc_486BB3

loc_486BAF:
                jz      short loc_486BAC

loc_486BB1:
                jmp     short loc_486BAC

loc_486BB3:
                and     eax, 41h
                dec     bl
                sub     bl, 12h
                and     eax, 800h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1200h
                dec     dh
                sub     dh, 1
                dec     dh
                sub     dh, 7
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_486BEC
                jl      short loc_486BEA

loc_486BE5:
                jmp     short loc_486BEE

loc_486BEA:
                jz      short loc_486BE5

loc_486BEC:
                jmp     short loc_486BE5

loc_486BEE:
                inc     eax
                dec     eax
                jo      short loc_486BFB
                jl      short loc_486BF9

loc_486BF4:
                jmp     short loc_486BFD
```

```
loc_486BF9:
                jz      short loc_486BF4

loc_486BFB:
                jmp     short loc_486BF4

loc_486BFD:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_486C0A
                jl      short loc_486C08

loc_486C05:
                jmp     short loc_486C0C

loc_486C08:
                jz      short loc_486C05

loc_486C0A:
                jmp     short loc_486C05

loc_486C0C:
                inc     eax
                pop     ecx
                cmp     ecx, eax
                jnz     short loc_486C18
                and     eax, 0
                inc     eax
                jmp     short loc_486C1B

loc_486C18:
                and     eax, 0

loc_486C1B:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9388
                xor     ecx, ds:dword_4D938C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_486C3E
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_486C3E:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_48C776(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD04
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 5
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 41h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                and     eax, 0
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                inc     eax
                dec     bl
                dec     bl
                not     bx
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9398
                xor     ecx, ds:dword_4D939C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48C7F5
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48C7F5:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_48D11F(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCFC
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 4
                dec     ch
                sub     ch, 3
                dec     ch
                and     ah, ch
                mov     cl, 0AFh
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 5
                not     al
                bswap   ecx
                not     al
                bswap   ecx
                dec     cl
                dec     cl
                sub     cl, 10h
                dec     cl
                dec     cl
                dec     cl
                add     cl, 12h
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 13h
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
```

```
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9390
                xor     ecx, ds:dword_4D9394
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48D1E4
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48D1E4:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCBC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_48B1EC(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF0
```

```
add     esp, 4
mov     [ebp-4], eax
mov     eax, [ebp-4]
push    ebx
mov     ebx, 0FFFFh
and     eax, ebx
push    ecx
mov     ch, 2Ch
sub     ch, 1
sub     ch, 20h
dec     ch
dec     ch
sub     ch, 4
dec     ch
sub     ch, 3
dec     ch
and     ah, ch
mov     cl, 0AEh
sub     cl, 2
dec     cl
dec     cl
sub     cl, 6
not     al
bswap   ecx
not     al
bswap   ecx
dec     cl
dec     cl
sub     cl, 10h
dec     cl
dec     cl
add     cl, 0Ch
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
sub     cl, 10h
sub     cl, 1
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
not     ecx
bswap   eax
not     ecx
bswap   eax
```

```
                inc     cl
                add     cl, 2
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9384
                xor     ecx, ds:dword_4D9388
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48B2A8
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48B2A8:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_48C332(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD00
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
```

```
push    ecx
mov     ecx, 800h
mov     ecx, 4Bh
not     ecx
bswap   eax
not     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
inc     ecx
dec     ecx
inc     ecx
inc     cl
inc     cl
inc     cl
add     ecx, 0Dh
inc     cl
inc     cl
inc     cl
inc     cl
inc     cl
add     ecx, 0Ah
dec     ecx
push    edx
mov     edx, 4
add     ecx, edx
inc     ecx
pop     edx
bswap   eax
add     ecx, 3
and     eax, ecx
pop     ecx
neg     eax
sbb     eax, eax
neg     eax
pop     edx
mov     [ebp-0Ch], eax
mov     ecx, ds:dword_4D9394
xor     ecx, ds:dword_4D9398
shl     ecx, 1
mov     [ebp-8], ecx
```

```
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48C3C3
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48C3C3:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_484752(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE8
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 80h
                jmp     short loc_48477B
                mov     ebx, 4

 loc_48477B:
                mov     ebx, 0A4h
                xor     ebx, 96h
                not     ebx
                bswap   eax
                not     ebx
                inc     ebx
                inc     ebx
                inc     ebx
                inc     ebx
                inc     ebx
                add     ebx, 5
                dec     ebx
                push    ecx
                mov     ecx, 4
                add     ebx, ecx
                inc     ebx
                pop     ecx
                bswap   eax
                and     eax, ebx
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D937C
                xor     ecx, ds:dword_4D9380
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4847CD
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_4847CD:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_48CC9F(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE4
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Dh
                dec     ch
                sub     ch, 1
                sub     ch, 20h
                dec     ch
                dec     ch
                sub     ch, 7
                dec     ch
                dec     ch
                and     ah, ch
                mov     cl, 0BDh
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     cl
                bswap   edx
                not     cl
                bswap   edx
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                push    eax
                dec     cl
                dec     cl
                sub     cl, 12h
                dec     cl
                dec     cl
                sub     cl, 3
                dec     cl
                and     eax, 40h
                dec     cl
                dec     cl
                dec     cl
                add     cl, 0Eh
```

```
                dec     cl
                dec     cl
                and     eax, 80h
                sub     cl, 1Fh
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                pop     eax
                inc     cl
                inc     cl
                inc     cl
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ebx
                push    eax
                mov     eax, [ebp-4]
                mov     edx, 0C00h
                sub     dh, 1
                dec     dh
                dec     dh
                dec     dh
                and     eax, edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                cmp     eax, edx
                jo      short loc_48CD70
                jl      short loc_48CD6E

loc_48CD6B:
                jmp     short loc_48CD72

loc_48CD6E:
                jz      short loc_48CD6B

loc_48CD70:
                jmp     short loc_48CD6B

loc_48CD72:
                setnz   cl
                mov     al, cl
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9378
                xor     ecx, ds:dword_4D937C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48CD9A
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48CD9A:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_48B5B5(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCEC
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                mov     ecx, 800h
                mov     ecx, 4Ch
                not     ecx
                bswap   eax
                not     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                inc     ecx
                dec     ecx
                inc     ecx
                inc     cl
                inc     cl
                inc     cl
                add     ecx, 0Bh
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                inc     cl
                add     cl, 2
                add     ecx, 0Ah
                dec     ecx
                push    edx
                mov     edx, 4
                add     ecx, edx
                inc     ecx
                pop     edx
                bswap   eax
                add     ecx, 3
                and     eax, ecx
                pop     ecx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9380
                xor     ecx, ds:dword_4D9384
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48B648
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48B648:
                mov     eax, [ebp-8]
```

```
                        push    eax
                        call    ds:off_4DDCAC
                        add     esp, 4
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
    }
}




__declspec(naked) void sub_48AA4C(void)
{
    __asm
    {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDCDC_2
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    edx
                        mov     edx, 0FFFFh
                        and     eax, edx
                        push    ebx
                        push    0Fh
                        pop     ebx
                        jo      short loc_48AA7F
                        jl      short loc_48AA7D

 loc_48AA78:
                        jmp     short loc_48AA81

 loc_48AA7D:
                        jz      short loc_48AA78

 loc_48AA7F:
                        jmp     short loc_48AA78

 loc_48AA81:
                        sub     bl, 5
                        dec     bl
```

```
                push    eax
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                and     eax, 41h
                dec     bl
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 2700h
                dec     dh
                sub     dh, 5
                dec     dh
                sub     dh, 17h
                dec     dh
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9370
                xor     ecx, ds:dword_4D9374
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48AAD9
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48AAD9:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDC9C
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_4865F8(void)
{
```

```
__asm
{
            push    ebp
            mov     ebp, esp
            sub     esp, 0Ch
            push    ebx
            push    esi
            push    edi
            mov     eax, [ebp+8]
            push    eax
            call    ds:off_4DDD08
            add     esp, 4
            mov     [ebp-4], eax
            mov     eax, [ebp-4]
            push    edx
            mov     edx, 0FFFFh
            and     eax, edx
            push    ebx
            push    eax
            mov     bh, 2
            dec     bh
            dec     bh
            mov     edi, 80h
            and     eax, 800h
            bswap   ecx
            pop     eax
            bswap   ecx
            and     ah, bh
            mov     bl, 83h
            dec     edi
            sub     bl, 8
            dec     bl
            dec     edi
            dec     bl
            dec     edi
            dec     bl
            dec     bl
            dec     edi
            dec     bl
            dec     bl
            dec     edi
            and     edi, ebx
            dec     bl
            sub     bl, 7
            dec     edi
            sub     bl, 10h
            dec     edi
            dec     bl
            dec     edi
            sub     bl, 1Ch
            not     bx
            bswap   eax
```

```
                dec     edi
                not     bx
                bswap   eax
                dec     edi
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D939C
                xor     ecx, ds:dword_4D93A0
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_486698
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_486698:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_4807FD(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE4
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    0Ch
                pop     ebx
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                add     bl, 0FFh
                add     bl, 0FFh
                dec     bl
                jo      short loc_480842
                jl      short loc_480840

loc_48083B:
                jmp     short loc_480844

loc_480840:
                jz      short loc_48083B

loc_480842:
                jmp     short loc_48083B

loc_480844:
                add     bl, 0FFh
                add     bl, 0FFh
                add     bl, 0FFh
                add     bl, 0FFh
                and     al, bl
                jo      short loc_48085B
                jl      short loc_480859

loc_480856:
                jmp     short loc_48085D

loc_480859:
                jz      short loc_480856

loc_48085B:
                jmp     short loc_480856

loc_48085D:
                mov     dh, 15h
                and     dl, 0
                dec     dh
                sub     dh, 9
                sub     dh, 1
                dec     dh
                dec     dh
                and     ah, dh
                pop     ebx
                pop     edx
                test    eax, eax
                jz      short loc_48087E
                not     eax
                add     eax, 1
                stc
                jmp     short loc_480884

loc_48087E:
                not     eax
                add     eax, 1
                clc

loc_480884:
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_480893
                jl      short loc_480891

loc_48088C:
                jmp     short loc_480895

loc_480891:
                jz      short loc_48088C

loc_480893:
                jmp     short loc_48088C

loc_480895:
                inc     eax
                dec     eax
                jo      short loc_4808A2
                jl      short loc_4808A0

loc_48089B:
                jmp     short loc_4808A4

loc_4808A0:
                jz      short loc_48089B

loc_4808A2:
                jmp     short loc_48089B

loc_4808A4:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_4808B3
                jl      short loc_4808B1

 loc_4808AC:
                jmp     short loc_4808B5

 loc_4808B1:
                jz      short loc_4808AC

 loc_4808B3:
                jmp     short loc_4808AC

 loc_4808B5:
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9378
                xor     ecx, ds:dword_4D937C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4808D9
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_4808D9:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_488AA6(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF4
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 80h
                jmp     short loc_488ACF
                mov     ebx, 4

loc_488ACF:
                mov     ebx, 41h
                not     ebx
                bswap   eax
                not     ebx
                inc     ebx
                inc     ebx
                and     eax, 0
                and     ebx, 800h
                dec     ebx
                push    ecx
                mov     ecx, 4
                add     ebx, ecx
                inc     ebx
                pop     ecx
                bswap   eax
                inc     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9388
                xor     ecx, ds:dword_4D938C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_488B17
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_488B17:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
```

```
    }
}



__declspec(naked) void sub_480A1B(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE4
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1Fh
                pop     ebx
                jo      short loc_480A4E
                jl      short loc_480A4C

 loc_480A47:
                jmp     short loc_480A50

 loc_480A4C:
                jz      short loc_480A47

 loc_480A4E:
                jmp     short loc_480A47

 loc_480A50:
                sub     bl, 5
                dec     bl
                push    eax
                dec     bl
                dec     bl
                and     eax, 41h
                dec     bl
                sub     bl, 12h
                sub     bl, 3
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 1500h
                dec     dh
                sub     dh, 3
                dec     dh
                sub     dh, 7
                dec     dh
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9378
                xor     ecx, ds:dword_4D937C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_480AA8
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_480AA8:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_4801E3(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF0
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 80h
                jmp     short loc_48020C
                mov     ebx, 4

loc_48020C:
                mov     ebx, 27h
                xor     ebx, 15h
                not     ebx
                bswap   eax
                not     ebx
                inc     ebx
                inc     ebx
                sub     ebx, 0FFFFFFFFh
                inc     ebx
                inc     ebx
                inc     ebx
                sub     ebx, 0FFFFFFFFh
                inc     ebx
                inc     ebx
                sub     ebx, 0FFFFFFFFh
                add     ebx, 0FFFFFFFFh
                push    ecx
                mov     ecx, 3
                inc     ecx
                add     ebx, ecx
                inc     ebx
                pop     ecx
                bswap   eax
                and     eax, ebx
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9384
                xor     ecx, ds:dword_4D9388
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_480266
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_480266:
```

```
                    mov     eax, [ebp-8]
                    push    eax
                    call    ds:off_4DDCB0
                    add     esp, 4
                    pop     edi
                    pop     esi
                    pop     ebx
                    mov     esp, ebp
                    pop     ebp
                    retn
     }
}




__declspec(naked) void sub_484255(void)
{
     __asm
     {
                    push    ebp
                    mov     ebp, esp
                    sub     esp, 0Ch
                    push    ebx
                    push    esi
                    push    edi
                    mov     eax, [ebp+8]
                    push    eax
                    call    ds:off_4DDD08
                    add     esp, 4
                    mov     [ebp-4], eax
                    mov     eax, [ebp-4]
                    jo      short loc_48427A
                    jl      short loc_484278

 loc_484275:
                    jmp     short loc_48427C

 loc_484278:
                    jz      short loc_484275

 loc_48427A:
                    jmp     short loc_484275

 loc_48427C:
                    push    edx
                    mov     edx, 0FFFFh
                    and     eax, edx
                    push    ebx
                    push    eax
                    mov     bh, 7
                    dec     bh
```

```
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                jo      short loc_4842AB
                jl      short loc_4842A9

loc_4842A6:
                jmp     short loc_4842AD

loc_4842A9:
                jz      short loc_4842A6

loc_4842AB:
                jmp     short loc_4842A6

loc_4842AD:
                mov     bl, 0C6h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D939C
                xor     ecx, ds:dword_4D93A0
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_484301
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_484301:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_4843C4(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE0
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    eax
                mov     eax, 4
                bswap   eax
                not     eax
                pop     eax
                push    edx
                mov     dh, 80h
                mov     dh, 0
                inc     dh
                mov     ecx, ecx
                inc     dh
                inc     dh
                inc     dh
                inc     dh
                push    ebx
                inc     dh
                push    ecx
                bswap   ecx
                not     ecx
                push    eax
                not     eax
                mov     eax, 800h
                xchg    eax, ecx
                mov     ecx, 40h
                xchg    eax, ecx
                not     eax
                pop     eax
                not     ecx
                pop     ecx
                inc     dh
                inc     dh
                and     ebx, 800h
                inc     dh
                inc     dh
                inc     dh
                inc     dh
                and     ebx, 10h
                inc     dh
                inc     dh
                pop     ebx
                sub     dh, 0Dh
                dec     dh
                and     ah, dh
                mov     dl, 5
                sub     dl, 0FFh
                dec     dl
                dec     dl
                dec     dl
                sub     dl, 0FFh
                dec     dl
                dec     dl
                dec     dl
                and     al, dl
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9374
                xor     ecx, ds:dword_4D9378
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_484474
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_484474:
```

```
                        mov     eax, [ebp-8]
                        push    eax
                        call    ds:off_4DDCA0
                        add     esp, 4
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
        }
}



__declspec(naked) void sub_48A7DD(void)
{
        __asm
        {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDCEC
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    edx
                        mov     edx, 0FFFFh
                        and     eax, edx
                        push    ebx
                        push    410h
                        pop     ebx
                        dec     bh
                        dec     bh
                        sub     bh, 0FFh
                        sub     bh, 2
                        dec     bh
                        and     ah, bh
                        mov     bl, 0Eh
                        sub     bl, 4
                        dec     bl
                        sub     bl, 1
                        sub     bl, 1
                        sub     bl, 1
                        sub     bl, 1
                        sub     bl, 1
```

```
                and     al, bl
                pop     ebx
                pop     edx
                test    eax, eax
                jz      short loc_48A83C
                not     eax
                add     eax, 1
                stc
                jmp     short loc_48A842

loc_48A83C:
                not     eax
                add     eax, 1
                clc

loc_48A842:
                sbb     eax, eax
                inc     eax
                dec     eax
                jo      short loc_48A851
                jl      short loc_48A84F

loc_48A84A:
                jmp     short loc_48A853

loc_48A84F:
                jz      short loc_48A84A

loc_48A851:
                jmp     short loc_48A84A

loc_48A853:
                inc     eax
                dec     eax
                jo      short loc_48A860
                jl      short loc_48A85E

loc_48A859:
                jmp     short loc_48A862

loc_48A85E:
                jz      short loc_48A859

loc_48A860:
                jmp     short loc_48A859

loc_48A862:
                inc     eax
                dec     eax
                inc     eax
                dec     eax
                jo      short loc_48A86F
                jl      short loc_48A86D

loc_48A86A:
                jmp     short loc_48A871

loc_48A86D:
                jz      short loc_48A86A

loc_48A86F:
                jmp     short loc_48A86A

loc_48A871:
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9380
                xor     ecx, ds:dword_4D9384
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48A895
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_48A895:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCAC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_48C6A1(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
```

```
call    ds:off_4DDD00
add     esp, 4
mov     [ebp-4], eax
mov     eax, [ebp-4]
push    edx
mov     edx, 0FFFFh
and     eax, edx
push    ebx
push    eax
mov     bh, 10h
dec     bh
dec     bh
dec     al
dec     bh
dec     bh
dec     al
dec     bh
dec     al
dec     al
dec     bh
dec     bh
dec     bh
dec     bh
dec     bh
dec     al
dec     bh
dec     bh
dec     al
dec     al
dec     bh
dec     bh
dec     al
dec     bh
dec     bh
and     eax, 800h
bswap   ecx
pop     eax
bswap   ecx
and     ah, bh
mov     bl, 8Ch
sub     bl, 5
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
dec     bl
sub     bl, 3
dec     bl
dec     bl
dec     bl
dec     bl
```

```
                sub     bl, 1Ah
                dec     bl
                sub     bl, 1Fh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                neg     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9394
                xor     ecx, ds:dword_4D9398
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48C762
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48C762:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCC0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_485BB2(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCF4
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 80h
                jmp     short loc_485BDB
                mov     ebx, 4

loc_485BDB:
                mov     ebx, 32h
                not     ebx
                bswap   eax
                not     ebx
                inc     ebx
                jo      short loc_485BF0
                jl      short loc_485BEE

loc_485BEB:
                jmp     short loc_485BF2

loc_485BEE:
                jz      short loc_485BEB

loc_485BF0:
                jmp     short loc_485BEB

loc_485BF2:
                inc     ebx
                inc     ebx
                add     ebx, 7
                push    ecx
                mov     ecx, 4
                add     ebx, ecx
                inc     ebx
                pop     ecx
                bswap   eax
                jo      short loc_485C0C
                jl      short loc_485C0A

loc_485C07:
                jmp     short loc_485C0E

loc_485C0A:
                jz      short loc_485C07

loc_485C0C:
                jmp     short loc_485C07

loc_485C0E:
```

```
                and     eax, ebx
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9388
                xor     ecx, ds:dword_4D938C
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_485C39
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_485C39:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB4
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_480491(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD0C
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ebx
                mov     ebx, 80h
                jmp     short loc_4804BA
                mov     ebx, 4

 loc_4804BA:
                mov     ebx, 32h
                not     ebx
                bswap   eax
                not     ebx
                inc     ebx
                inc     ebx
                and     eax, 0
                add     ebx, 8
                dec     ebx
                push    ecx
                mov     ecx, 4
                add     ebx, ecx
                inc     ebx
                pop     ecx
                bswap   eax
                inc     eax
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A0
                xor     ecx, ds:dword_4D93A4
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4804FF
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_4804FF:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCCC
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_484C97(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE0
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    0D00h
                pop     ebx
                jo      short loc_484CCD
                jl      short loc_484CCB

 loc_484CC6:
                jmp     short loc_484CCF

 loc_484CCB:
                jz      short loc_484CC6

 loc_484CCD:
                jmp     short loc_484CC6

 loc_484CCF:
                sub     bh, 5
                dec     bh
                push    eax
                dec     bh
                dec     bh
                and     eax, 41h
                dec     bh
                sub     bh, 3
                pop     eax
                dec     bh
                and     ah, bh
                mov     edx, 20h
                sub     dl, 19h
                dec     dl
                dec     dl
                dec     dl
                and     al, dl
                pop     ebx
                pop     edx
```

```
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9374
                xor     ecx, ds:dword_4D9378
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_484D20
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_484D20:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}




__declspec(naked) void sub_48259B(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE8
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    eax
                mov     bh, 3
                jo      short loc_4825CC
                jl      short loc_4825CA

loc_4825C7:
                jmp     short loc_4825CE

loc_4825CA:
                jz      short loc_4825C7

loc_4825CC:
                jmp     short loc_4825C7

loc_4825CE:
                dec     bh
                dec     bh
                dec     bh
                and     eax, 800h
                bswap   ecx
                pop     eax
                bswap   ecx
                and     ah, bh
                mov     bl, 87h
                sub     bl, 5
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1Ah
                sub     bl, 1Eh
                not     bx
                bswap   eax
                not     bx
                bswap   eax
                jo      short loc_482610
                jl      short loc_48260E

loc_48260B:
                jmp     short loc_482612

loc_48260E:
                jz      short loc_48260B

loc_482610:
                jmp     short loc_48260B

loc_482612:
```

```
                and     al, bl
                mov     eax, eax
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D937C
                xor     ecx, ds:dword_4D9380
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_482640
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_482640:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_484F93(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCDC_2
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    1E00h
                pop     ebx
                jo      short loc_484FC7
                jl      short loc_484FC5

loc_484FC2:
                jmp     short loc_484FC9

loc_484FC5:
                jz      short loc_484FC2

loc_484FC7:
                jmp     short loc_484FC2

loc_484FC9:
                sub     bh, 4
                inc     bh
                dec     bh
                dec     bh
                push    eax
                dec     bh
                dec     bh
                inc     bh
                dec     bh
                jo      short loc_484FE4
                jl      short loc_484FE2

loc_484FDF:
                jmp     short loc_484FE6

loc_484FE2:
                jz      short loc_484FDF

loc_484FE4:
                jmp     short loc_484FDF

loc_484FE6:
                and     eax, 40h
                dec     bh
                sub     bh, 12h
                sub     bh, 3
                pop     eax
                dec     bh
                and     ah, bh
                mov     edx, 12h
                dec     dl
                sub     dl, 1
                dec     dl
                sub     dl, 7
                dec     dl
                dec     dl
                inc     dl
                dec     dl
                dec     dl
                dec     dl
                and     al, dl
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D9370
                xor     ecx, ds:dword_4D9374
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_48503D
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_48503D:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDC9C
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_488803(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD0C
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                jo      short loc_488828
                jl      short loc_488826

loc_488823:
                jmp     short loc_48882A

loc_488826:
                jz      short loc_488823

loc_488828:
                jmp     short loc_488823

loc_48882A:
                push    ebx
                mov     ebx, 0FFFFh
                and     eax, ebx
                push    ecx
                mov     ch, 2Ch
                sub     ch, 1
                dec     edi
                inc     esi
                sub     ch, 20h
                dec     edi
                inc     esi
                dec     ch
                dec     ch
                dec     edi
                inc     esi
                sub     ch, 4
                dec     ch
                sub     ch, 3
                dec     ch
                and     ah, ch
                mov     cl, 70h
                sub     cl, 2
                dec     cl
                dec     cl
                dec     cl
                dec     edi
                inc     esi
                sub     cl, 6
                not     al
                dec     edi
                inc     esi
                bswap   ecx
                not     al
                bswap   ecx
                dec     cl
                dec     cl
                sub     cl, 12h
                add     cl, 0Bh
                dec     cl
                dec     cl
                jo      short loc_488882
                jl      short loc_488880

loc_48887D:
                jmp     short loc_488884

loc_488880:
                jz      short loc_48887D

loc_488882:
                jmp     short loc_48887D

loc_488884:
                dec     cl
                dec     cl
                dec     edi
                inc     esi
                dec     cl
                dec     cl
                sub     cl, 10h
                sub     cl, 1
                dec     edi
                inc     esi
                dec     cl
                dec     cl
                dec     cl
                and     eax, 0
                inc     eax
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                dec     cl
                not     ecx
                bswap   eax
                not     ecx
                bswap   eax
                inc     cl
                add     cl, 2
                pop     ecx
                pop     ebx
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A0
                xor     ecx, ds:dword_4D93A4
                shl     ecx, 1
```

```
                    mov     [ebp-8], ecx
                    cmp     dword ptr [ebp-0Ch], 0
                    jz      short loc_4888DC
                    mov     edx, [ebp-8]
                    or      edx, 1
                    mov     [ebp-8], edx

  loc_4888DC:
                    mov     eax, [ebp-8]
                    push    eax
                    call    ds:off_4DDCCC
                    add     esp, 4
                    pop     edi
                    pop     esi
                    pop     ebx
                    mov     esp, ebp
                    pop     ebp
                    retn
     }
}




__declspec(naked) void sub_482D75(void)
{
     __asm
     {
                    push    ebp
                    mov     ebp, esp
                    sub     esp, 0Ch
                    push    ebx
                    push    esi
                    push    edi
                    mov     eax, [ebp+8]
                    push    eax
                    call    ds:off_4DDCF8
                    add     esp, 4
                    mov     [ebp-4], eax
                    mov     eax, [ebp-4]
                    push    edx
                    mov     edx, 0FFFFh
                    and     eax, edx
                    push    ebx
                    push    100h
                    pop     ebx
                    dec     bh
                    jo      short loc_482DAB
                    jl      short loc_482DA9

  loc_482DA6:
                    jmp     short loc_482DAD
```

```
loc_482DA9:
                jz      short loc_482DA6

loc_482DAB:
                jmp     short loc_482DA6

loc_482DAD:
                add     bh, 0FFh
                add     bh, 0FFh
                add     bh, 0FFh
                add     bh, 0FFh
                inc     bh
                inc     bh
                inc     bh
                inc     bh
                and     ah, bh
                jo      short loc_482DCC
                jl      short loc_482DCA

loc_482DC7:
                jmp     short loc_482DCE

loc_482DCA:
                jz      short loc_482DC7

loc_482DCC:
                jmp     short loc_482DC7

loc_482DCE:
                mov     bl, 15h
                dec     bl
                sub     bl, 6
                dec     bl
                dec     bl
                dec     bl
                sub     bl, 1
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                dec     bl
                and     al, bl
                pop     ebx
                pop     edx
                test    eax, eax
                jz      short loc_482DFA
                not     eax
                add     eax, 1
                stc
                jmp     short loc_482E00
```

```
 loc_482DFA:
                not     eax
                add     eax, 1
                clc

 loc_482E00:
                sbb     eax, eax
                neg     eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D938C
                xor     ecx, ds:dword_4D9390
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_482E29
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_482E29:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_4866AC(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD10
                add     esp, 4
```

```
mov     [ebp-4], eax
mov     eax, [ebp-4]
push    ebx
mov     ebx, 0FFFFh
and     eax, ebx
push    ecx
mov     ch, 2Ch
sub     ch, 1
sub     ch, 20h
dec     ch
dec     ch
sub     ch, 4
dec     ch
sub     ch, 3
dec     ch
and     ah, ch
mov     cl, 0AEh
sub     cl, 2
dec     cl
dec     cl
sub     cl, 6
not     al
bswap   ecx
not     al
bswap   ecx
dec     cl
dec     cl
sub     cl, 10h
dec     cl
dec     cl
add     cl, 0Ch
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
sub     cl, 10h
sub     cl, 1
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
dec     cl
not     ecx
bswap   eax
not     ecx
bswap   eax
inc     cl
```

```
                add     cl, 2
                and     al, cl
                mov     eax, eax
                pop     ecx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     ebx
                push    eax
                mov     eax, [ebp-4]
                mov     edx, 200h
                inc     dh
                inc     dh
                dec     dh
                inc     dh
                inc     dh
                inc     dh
                inc     dh
                inc     dh
                and     eax, edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                cmp     eax, edx
                setz    cl
                mov     al, cl
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D93A4
                xor     ecx, ds:dword_4D93A8
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_486794
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

loc_486794:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCD0
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
```

```
}



__declspec(naked) void sub_4822B2(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDCE8
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    edx
                mov     edx, 0FFFFh
                and     eax, edx
                push    ebx
                push    0Eh
                pop     ebx
                sub     bl, 6
                dec     bl
                push    eax
                dec     bl
                dec     bl
                and     eax, 80h
                dec     bl
                sub     bl, 2
                dec     bl
                pop     eax
                dec     bl
                and     al, bl
                mov     edx, 2400h
                dec     dh
                sub     dh, 3
                dec     dh
                sub     dh, 16h
                dec     dh
                and     ah, dh
                pop     ebx
                pop     edx
                neg     eax
                sbb     eax, eax
                neg     eax
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D937C
                xor     ecx, ds:dword_4D9380
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_482333
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_482333:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCA8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}




__declspec(naked) void sub_47FB2E(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    ebx
                push    esi
                push    edi
                mov     eax, [ebp+8]
                push    eax
                call    ds:off_4DDD14
                add     esp, 4
                mov     [ebp-4], eax
                mov     eax, [ebp-4]
                push    ecx
                bswap   ecx
                not     ecx
                push    eax
                not     eax
                mov     eax, 80h
                xchg    eax, ecx
                mov     ecx, 1
```

```
xchg    eax, ecx
not     eax
pop     eax
not     ecx
pop     ecx
push    edx
mov     dh, 16h
sub     dh, 6
not     ecx
dec     dh
dec     dh
dec     dh
dec     dh
bswap   eax
dec     dh
dec     dh
sub     dh, 5
sub     dh, 3
dec     dh
dec     dh
bswap   eax
and     ah, dh
mov     dl, 9
dec     dl
dec     dl
dec     dl
dec     dl
not     ecx
dec     dl
dec     dl
dec     dl
inc     dl
dec     dl
inc     dl
dec     dl
inc     dl
inc     dl
dec     dl
dec     dl
dec     dl
dec     dl
add     dl, 1
and     al, dl
not     ah
bswap   eax
bswap   eax
not     ah
pop     edx
neg     eax
sbb     eax, eax
inc     eax
mov     [ebp-0Ch], eax
```

```
                        mov     ecx, ds:dword_4D93A8
                        xor     ecx, ds:dword_4D93AC
                        shl     ecx, 1
                        mov     [ebp-8], ecx
                        cmp     dword ptr [ebp-0Ch], 0
                        jz      short loc_47FBE4
                        mov     edx, [ebp-8]
                        or      edx, 1
                        mov     [ebp-8], edx

 loc_47FBE4:
                        mov     eax, [ebp-8]
                        push    eax
                        call    ds:off_4DDCD4
                        add     esp, 4
                        pop     edi
                        pop     esi
                        pop     ebx
                        mov     esp, ebp
                        pop     ebp
                        retn
    }
}




__declspec(naked) void sub_489419(void)
{
    __asm
    {
                        push    ebp
                        mov     ebp, esp
                        sub     esp, 0Ch
                        push    ebx
                        push    esi
                        push    edi
                        mov     eax, [ebp+8]
                        push    eax
                        call    ds:off_4DDCF8
                        add     esp, 4
                        mov     [ebp-4], eax
                        mov     eax, [ebp-4]
                        push    ebx
                        mov     ebx, 800h
                        jmp     short loc_489442
                        mov     ebx, 80h

 loc_489442:
                        mov     ebx, 6Eh
                        not     ebx
                        bswap   eax
```

```
                not     ebx
                inc     ebx
                inc     ebx
                inc     ebx
                inc     ebx
                inc     ebx
                inc     ebx
                add     ebx, 8
                dec     ebx
                push    ecx
                mov     ecx, 5
                add     ebx, ecx
                pop     ecx
                bswap   eax
                and     eax, ebx
                pop     ebx
                neg     eax
                sbb     eax, eax
                inc     eax
                pop     edx
                push    eax
                mov     eax, [ebp-4]
                mov     edx, 0D00h
                sub     dh, 1
                dec     dh
                dec     dh
                dec     dh
                sub     dh, 0FFh
                dec     dh
                dec     dh
                and     eax, edx
                neg     eax
                sbb     eax, eax
                inc     eax
                mov     edx, eax
                pop     eax
                xor     ecx, ecx
                jo      short loc_489499
                jl      short loc_489497

loc_489494:
                jmp     short loc_48949B

loc_489497:
                jz      short loc_489494

loc_489499:
                jmp     short loc_489494

loc_48949B:
                cmp     eax, edx
                jo      short loc_4894A6
                jl      short loc_4894A4

loc_4894A1:
                jmp     short loc_4894A8

loc_4894A4:
                jz      short loc_4894A1

loc_4894A6:
                jmp     short loc_4894A1

loc_4894A8:
                jnz     short loc_4894BA
                jo      short loc_4894B3
                jl      short loc_4894B1

loc_4894AE:
                jmp     short loc_4894B5

loc_4894B1:
                jz      short loc_4894AE

loc_4894B3:
                jmp     short loc_4894AE

loc_4894B5:
                and     eax, 0
                jmp     short loc_4894C9

loc_4894BA:
                and     eax, 0
                jo      short loc_4894C6
                jl      short loc_4894C4

loc_4894C1:
                jmp     short loc_4894C8

loc_4894C4:
                jz      short loc_4894C1

loc_4894C6:
                jmp     short loc_4894C1

loc_4894C8:
                inc     eax

loc_4894C9:
                mov     [ebp-0Ch], eax
                mov     ecx, ds:dword_4D938C
                xor     ecx, ds:dword_4D9390
                shl     ecx, 1
                mov     [ebp-8], ecx
                cmp     dword ptr [ebp-0Ch], 0
                jz      short loc_4894EC
                mov     edx, [ebp-8]
                or      edx, 1
                mov     [ebp-8], edx

 loc_4894EC:
                mov     eax, [ebp-8]
                push    eax
                call    ds:off_4DDCB8
                add     esp, 4
                pop     edi
                pop     esi
                pop     ebx
                mov     esp, ebp
                pop     ebp
                retn
    }
}


//block 6 dynamic functions sub functions layer 2 (internal)

__declspec(naked) void B6sublayer2_1(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 14h
                mov     eax, 1
                mov     ecx, [ebp+0Ch]
                shl     eax, cl
                sub     eax, 1
                mov     ecx, [ebp+10h]
                shl     eax, cl
                mov     [ebp-10h], eax
                mov     ecx, [ebp-10h]
                not     ecx
                mov     edx, [ebp+8]
                and     edx, ecx
                mov     [ebp-4], edx
                mov     eax, 1
                mov     ecx, [ebp+10h]
                shl     eax, cl
                mov     [ebp-0Ch], eax
                mov     ecx, [ebp+0Ch]
                mov     edx, [ebp+10h]
                lea     ecx, [edx+ecx-1]
                mov     eax, 1
                shl     eax, cl
                mov     [ebp-14h], eax
                mov     dword ptr [ebp-8], 0
                jmp     short loc_47F51F

loc_47F516:
                mov     ecx, [ebp-8]
                add     ecx, 1
                mov     [ebp-8], ecx

loc_47F51F:
                mov     edx, [ebp-8]
                cmp     edx, [ebp+0Ch]
                jge     short loc_47F54C
                mov     eax, [ebp+8]
                and     eax, [ebp-0Ch]
                test    eax, eax
                jz      short loc_47F53A
                mov     ecx, [ebp-4]
                or      ecx, [ebp-14h]
                mov     [ebp-4], ecx

loc_47F53A:
                mov     edx, [ebp-0Ch]
                shl     edx, 1
                mov     [ebp-0Ch], edx
                mov     eax, [ebp-14h]
                shr     eax, 1
                mov     [ebp-14h], eax
                jmp     short loc_47F516

loc_47F54C:
                mov     eax, [ebp-4]
                mov     esp, ebp
                pop     ebp
                retn
    }
}


__declspec(naked) void B6Sublayer2_2(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                sub     esp, 10h
                mov     eax, 1
                mov     ecx, [ebp+0Ch]
                shl     eax, cl
                sub     eax, 1
                mov     [ebp-4], eax
                mov     edx, [ebp-4]
                mov     ecx, [ebp+10h]
```

```
                  shl     edx, cl
                  mov     [ebp-0Ch], edx
                  mov     eax, [ebp+8]
                  and     eax, [ebp-4]
                  mov     [ebp-10h], eax
                  mov     edx, [ebp+8]
                  and     edx, [ebp-0Ch]
                  mov     ecx, [ebp+10h]
                  shr     edx, cl
                  mov     [ebp-8], edx
                  mov     eax, [ebp-4]
                  or      eax, [ebp-0Ch]
                  not     eax
                  mov     ecx, [ebp+8]
                  and     ecx, eax
                  mov     [ebp+8], ecx
                  mov     edx, [ebp-10h]
                  mov     ecx, [ebp+10h]
                  shl     edx, cl
                  or      edx, [ebp-8]
                  mov     eax, [ebp+8]
                  or      eax, edx
                  mov     [ebp+8], eax
                  mov     eax, [ebp+8]
                  mov     esp, ebp
                  pop     ebp
                  retn
     }
}



__declspec(naked) void B6_Sublayer2_3(void)
{
     __asm
     {
                  push    ebp
                  mov     ebp, esp
                  sub     esp, 8
                  cmp     dword ptr [ebp+14h], 0
                  jge     short loc_47F400
                  mov     eax, [ebp+14h]
                  add     eax, [ebp+0Ch]
                  mov     [ebp+14h], eax

 loc_47F400:
                  mov     edx, 1
                  mov     ecx, [ebp+0Ch]
                  shl     edx, cl
                  sub     edx, 1
                  mov     [ebp-8], edx
                  mov     eax, [ebp+8]
```

```
                        mov     ecx, [ebp+10h]
                        shr     eax, cl
                        and     eax, [ebp-8]
                        mov     [ebp-4], eax
                        mov     edx, [ebp-4]
                        mov     ecx, [ebp+14h]
                        shr     edx, cl
                        mov     ecx, [ebp+0Ch]
                        sub     ecx, [ebp+14h]
                        mov     eax, [ebp-4]
                        shl     eax, cl
                        or      edx, eax
                        and     edx, [ebp-8]
                        mov     [ebp-4], edx
                        mov     edx, [ebp-8]
                        mov     ecx, [ebp+10h]
                        shl     edx, cl
                        not     edx
                        mov     eax, [ebp+8]
                        and     eax, edx
                        mov     [ebp+8], eax
                        mov     edx, [ebp-4]
                        mov     ecx, [ebp+10h]
                        shl     edx, cl
                        mov     eax, [ebp+8]
                        or      eax, edx
                        mov     [ebp+8], eax
                        mov     eax, [ebp+8]
                        mov     esp, ebp
                        pop     ebp
                        retn
        }
}


__declspec(naked) void B6_Sublayer2_4(void)
{
        __asm
        {
                        push    ebp
                        mov     ebp, esp
                        cmp     dword ptr [ebp+0Ch], 0
                        jge     short loc_47F3D2
                        mov     eax, [ebp+0Ch]
                        add     eax, 20h
                        mov     [ebp+0Ch], eax

 loc_47F3D2:
                        mov     eax, [ebp+8]
                        mov     ecx, [ebp+0Ch]
                        shr     eax, cl
                        mov     ecx, 20h
                        sub     ecx, [ebp+0Ch]
                        mov     edx, [ebp+8]
                        shl     edx, cl
                        or      eax, edx
                        pop     ebp
                        retn
        }
}

//block 6 dynamic functions sub functions layer 1


__declspec(naked) void sub_47CFB0(void)
{
        __asm
        {
                        push    ebp
                        mov     ebp, esp
                        mov     eax, [ebp+8]
                        xor     eax, 6E4957A8h
                        mov     [ebp+8], eax
                        push    5
                        push    15h
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    B6sublayer2_1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    0Dh
                        push    2
                        mov     edx, [ebp+8]
                        push    edx
                        call    B6Sublayer2_2
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    0Ch
                        push    9
                        mov     eax, [ebp+8]
                        push    eax
                        call    B6sublayer2_1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        mov     ecx, [ebp+8]
                        xor     ecx, 4B487412h
                        mov     [ebp+8], ecx
                        push    5
                        push    0Eh
                        push    7
                        mov     edx, [ebp+8]
                        push    edx
                        call    B6_Sublayer2_3
                        add     esp, 10h
```

```
                mov     [ebp+8], eax
                push    13h
                push    8
                mov     eax, [ebp+8]
                push    eax
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     ecx, [ebp+8]
                xor     ecx, 0D972B853h
                mov     [ebp+8], ecx
                push    7
                push    16h
                mov     edx, [ebp+8]
                push    edx
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_47D0EF(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                mov     eax, [ebp+8]
                xor     eax, 0FB1E52AFh
                mov     [ebp+8], eax
                push    1Ah
                push    5
                mov     ecx, [ebp+8]
                push    ecx
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    16h
                push    0
                push    1Ch
                mov     edx, [ebp+8]
                push    edx
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                push    0Ah
                push    0Ah
```

```
mov     eax, [ebp+8]
push    eax
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    0Dh
push    2
push    14h
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 8B9D36E9h
mov     [ebp+8], edx
push    1Ah
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    7
push    14h
push    0Bh
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 0A52B3D68h
mov     [ebp+8], edx
push    14h
push    1
mov     eax, [ebp+8]
push    eax
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    0
push    1Eh
mov     ecx, [ebp+8]
push    ecx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    0Ch
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_4
add     esp, 8
```

```
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                xor     eax, 40174D5Bh
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                pop     ebp
                retn
    }
}


__declspec(naked) void sub_47D2B5(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                mov     eax, [ebp+8]
                xor     eax, 0FBBD38E7h
                mov     [ebp+8], eax
                push    17h
                push    4
                mov     ecx, [ebp+8]
                push    ecx
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    11h
                mov     edx, [ebp+8]
                push    edx
                call    B6_Sublayer2_4
                add     esp, 8
                mov     [ebp+8], eax
                push    6
                push    9
                push    0Eh
                mov     eax, [ebp+8]
                push    eax
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                push    1Ah
                mov     ecx, [ebp+8]
                push    ecx
                call    B6_Sublayer2_4
                add     esp, 8
                mov     [ebp+8], eax
                mov     edx, [ebp+8]
                xor     edx, 81A5E699h
                mov     [ebp+8], edx
                push    0Bh
                push    0
                push    18h
                mov     eax, [ebp+8]
                push    eax
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                push    0
                push    11h
                mov     ecx, [ebp+8]
                push    ecx
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0Eh
                push    8
                mov     edx, [ebp+8]
                push    edx
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    9
                push    1
                push    1Eh
                mov     eax, [ebp+8]
                push    eax
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                push    9
                push    2
                mov     ecx, [ebp+8]
                push    ecx
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    4
                push    17h
                mov     edx, [ebp+8]
                push    edx
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0Fh
                mov     eax, [ebp+8]
                push    eax
                call    B6_Sublayer2_4
                add     esp, 8
                mov     [ebp+8], eax
                push    2
                push    0Fh
                push    9
                mov     ecx, [ebp+8]
```

```
                push    ecx
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                mov     edx, [ebp+8]
                xor     edx, 0F185A47Ch
                mov     [ebp+8], edx
                push    0Fh
                mov     eax, [ebp+8]
                push    eax
                call    B6_Sublayer2_4
                add     esp, 8
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_47D4F9(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                mov     eax, [ebp+8]
                xor     eax, 5C2ACD4Dh
                mov     [ebp+8], eax
                push    0Ah
                push    3
                mov     ecx, [ebp+8]
                push    ecx
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    6
                push    0Ah
                mov     edx, [ebp+8]
                push    edx
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    5
                push    7
                push    11h
                mov     eax, [ebp+8]
                push    eax
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
```

```
push    0Bh
push    7
mov     ecx, [ebp+8]
push    ecx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    7
push    8
mov     edx, [ebp+8]
push    edx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    7
push    0Ah
push    0Bh
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    16h
push    2
mov     ecx, [ebp+8]
push    ecx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    5
push    7
push    8
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    0Fh
push    5
mov     eax, [ebp+8]
push    eax
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    12h
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    6
push    3
```

```
                 push    0Dh
                 mov     edx, [ebp+8]
                 push    edx
                 call    B6_Sublayer2_3
                 add     esp, 10h
                 mov     [ebp+8], eax
                 mov     eax, [ebp+8]
                 pop     ebp
                 retn
     }
}



__declspec(naked) void sub_47D6CE(void)
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 mov     eax, [ebp+8]
                 xor     eax, 47D8FFBDh
                 mov     [ebp+8], eax
                 push    4
                 mov     ecx, [ebp+8]
                 push    ecx
                 call    B6_Sublayer2_4
                 add     esp, 8
                 mov     [ebp+8], eax
                 push    17h
                 push    0
                 push    1Ch
                 mov     edx, [ebp+8]
                 push    edx
                 call    B6_Sublayer2_3
                 add     esp, 10h
                 mov     [ebp+8], eax
                 push    7
                 push    0Fh
                 mov     eax, [ebp+8]
                 push    eax
                 call    B6sublayer2_1
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 push    13h
                 push    5
                 mov     ecx, [ebp+8]
                 push    ecx
                 call    B6Sublayer2_2
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 push    1Bh
```

```
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    3
push    0Dh
push    8
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    0Ah
push    7
mov     ecx, [ebp+8]
push    ecx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    0
push    1Fh
mov     edx, [ebp+8]
push    edx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    14h
push    1
mov     eax, [ebp+8]
push    eax
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    6
push    12h
mov     ecx, [ebp+8]
push    ecx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    12h
push    5
mov     edx, [ebp+8]
push    edx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    15h
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_4
```

```
add     esp, 8
mov     [ebp+8], eax
push    15h
push    4
mov     ecx, [ebp+8]
push    ecx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    3
push    0Fh
mov     edx, [ebp+8]
push    edx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    9
push    8
mov     eax, [ebp+8]
push    eax
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 89C7C9A6h
mov     [ebp+8], ecx
push    0Eh
push    1
mov     edx, [ebp+8]
push    edx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    0Ch
push    0Fh
mov     eax, [ebp+8]
push    eax
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    13h
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    5
push    7
push    17h
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_3
```

```
                    add     esp, 10h
                    mov     [ebp+8], eax
                    mov     eax, [ebp+8]
                    pop     ebp
                    retn
        }
}


__declspec(naked) void sub_47D9DB(void)
{
        __asm
        {
                    push    ebp
                    mov     ebp, esp
                    mov     eax, [ebp+8]
                    xor     eax, 24A4A3B5h
                    mov     [ebp+8], eax
                    push    17h
                    push    8
                    mov     ecx, [ebp+8]
                    push    ecx
                    call    B6sublayer2_1
                    add     esp, 0Ch
                    mov     [ebp+8], eax
                    push    10h
                    push    2
                    mov     edx, [ebp+8]
                    push    edx
                    call    B6Sublayer2_2
                    add     esp, 0Ch
                    mov     [ebp+8], eax
                    push    13h
                    push    0
                    push    1Fh
                    mov     eax, [ebp+8]
                    push    eax
                    call    B6_Sublayer2_3
                    add     esp, 10h
                    mov     [ebp+8], eax
                    mov     ecx, [ebp+8]
                    xor     ecx, 8B465658h
                    mov     [ebp+8], ecx
                    push    0Eh
                    mov     edx, [ebp+8]
                    push    edx
                    call    B6_Sublayer2_4
                    add     esp, 8
                    mov     [ebp+8], eax
                    mov     eax, [ebp+8]
                    xor     eax, 5BFB6B28h
                    mov     [ebp+8], eax
```

```
push    0Eh
push    1
mov     ecx, [ebp+8]
push    ecx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    5
push    1
push    1Eh
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    5
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    1Bh
push    0
push    1Ch
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    0Bh
push    1
mov     edx, [ebp+8]
push    edx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    9
push    0
push    0Dh
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    14h
push    4
mov     ecx, [ebp+8]
push    ecx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    15h
```

```
                        mov     edx, [ebp+8]
                        push    edx
                        call    B6_Sublayer2_4
                        add     esp, 8
                        mov     [ebp+8], eax
                        mov     eax, [ebp+8]
                        xor     eax, 7C3547F7h
                        mov     [ebp+8], eax
                        push    0Bh
                        push    4
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    B6Sublayer2_2
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    4
                        push    0Bh
                        push    6
                        mov     edx, [ebp+8]
                        push    edx
                        call    B6_Sublayer2_3
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    0Ch
                        push    1
                        mov     eax, [ebp+8]
                        push    eax
                        call    B6Sublayer2_2
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    6
                        push    3
                        push    16h
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    B6_Sublayer2_3
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    0
                        push    1Fh
                        mov     edx, [ebp+8]
                        push    edx
                        call    B6sublayer2_1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        mov     eax, [ebp+8]
                        pop     ebp
                        retn
            }
        }
```

```
__declspec(naked) void sub_47DCDA(void)
{
     __asm
     {
                 push    ebp
                 mov     ebp, esp
                 mov     eax, [ebp+8]
                 xor     eax, 34473F96h
                 mov     [ebp+8], eax
                 push    11h
                 mov     ecx, [ebp+8]
                 push    ecx
                 call    B6_Sublayer2_4
                 add     esp, 8
                 mov     [ebp+8], eax
                 mov     edx, [ebp+8]
                 xor     edx, 5ED8936Ch
                 mov     [ebp+8], edx
                 push    0Ch
                 push    8
                 mov     eax, [ebp+8]
                 push    eax
                 call    B6Sublayer2_2
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 push    0Ah
                 push    12h
                 mov     ecx, [ebp+8]
                 push    ecx
                 call    B6sublayer2_1
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 mov     edx, [ebp+8]
                 xor     edx, 6BA330BFh
                 mov     [ebp+8], edx
                 push    4
                 push    8
                 push    13h
                 mov     eax, [ebp+8]
                 push    eax
                 call    B6_Sublayer2_3
                 add     esp, 10h
                 mov     [ebp+8], eax
                 push    0Eh
                 mov     ecx, [ebp+8]
                 push    ecx
                 call    B6_Sublayer2_4
                 add     esp, 8
                 mov     [ebp+8], eax
                 push    0Bh
                 push    4
```

```
                        push    19h
                        mov     edx, [ebp+8]
                        push    edx
                        call    B6_Sublayer2_3
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    3
                        mov     eax, [ebp+8]
                        push    eax
                        call    B6_Sublayer2_4
                        add     esp, 8
                        mov     [ebp+8], eax
                        mov     ecx, [ebp+8]
                        xor     ecx, 251AFCFEh
                        mov     [ebp+8], ecx
                        push    0Fh
                        mov     edx, [ebp+8]
                        push    edx
                        call    B6_Sublayer2_4
                        add     esp, 8
                        mov     [ebp+8], eax
                        push    0Dh
                        push    0Bh
                        mov     eax, [ebp+8]
                        push    eax
                        call    B6sublayer2_1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        mov     eax, [ebp+8]
                        pop     ebp
                        retn
            }
        }



        __declspec(naked) void sub_47DE96(void)
        {
            __asm
            {
                        push    ebp
                        mov     ebp, esp
                        mov     eax, [ebp+8]
                        xor     eax, 5A07B2A1h
                        mov     [ebp+8], eax
                        push    10h
                        push    0Bh
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    B6sublayer2_1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
```

```
push    0Bh
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    0Dh
push    6
mov     eax, [ebp+8]
push    eax
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 70648B0Dh
mov     [ebp+8], ecx
push    3
push    11h
mov     edx, [ebp+8]
push    edx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
mov     eax, [ebp+8]
xor     eax, 3773D297h
mov     [ebp+8], eax
push    0
push    1Eh
mov     ecx, [ebp+8]
push    ecx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 49253F31h
mov     [ebp+8], edx
push    0Fh
push    2
mov     eax, [ebp+8]
push    eax
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    10h
push    0
push    1Fh
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    11h
push    5
mov     edx, [ebp+8]
push    edx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    11h
push    9
mov     eax, [ebp+8]
push    eax
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 0ABA4CE9Ah
mov     [ebp+8], ecx
push    0Dh
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
mov     eax, [ebp+8]
xor     eax, 0F6547803h
mov     [ebp+8], eax
push    13h
push    4
mov     ecx, [ebp+8]
push    ecx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 3A22CD09h
mov     [ebp+8], edx
push    4
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    5
push    0Ah
push    0Fh
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    1Fh
mov     edx, [ebp+8]
push    edx
```

```
                call    B6_Sublayer2_4
                add     esp, 8
                mov     [ebp+8], eax
                push    0
                push    7
                mov     eax, [ebp+8]
                push    eax
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_47E17A(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                mov     eax, [ebp+8]
                xor     eax, 949D26F0h
                mov     [ebp+8], eax
                push    7
                push    9
                push    10h
                mov     ecx, [ebp+8]
                push    ecx
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                mov     edx, [ebp+8]
                xor     edx, 58887477h
                mov     [ebp+8], edx
                push    5
                push    5
                mov     eax, [ebp+8]
                push    eax
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    1Dh
                mov     ecx, [ebp+8]
                push    ecx
                call    B6_Sublayer2_4
                add     esp, 8
                mov     [ebp+8], eax
                push    0Ch
                push    6
                push    0Fh
                mov     edx, [ebp+8]
                push    edx
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                push    15h
                mov     eax, [ebp+8]
                push    eax
                call    B6_Sublayer2_4
                add     esp, 8
                mov     [ebp+8], eax
                mov     ecx, [ebp+8]
                xor     ecx, 0A8DB534Eh
                mov     [ebp+8], ecx
                push    10h
                push    5
                mov     edx, [ebp+8]
                push    edx
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    5
                push    19h
                mov     eax, [ebp+8]
                push    eax
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0Bh
                mov     ecx, [ebp+8]
                push    ecx
                call    B6_Sublayer2_4
                add     esp, 8
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                pop     ebp
                retn
     }
}


__declspec(naked) void sub_47E2FC(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                mov     eax, [ebp+8]
                xor     eax, 0EECED199h
                mov     [ebp+8], eax
```

```
push    6
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    2
push    17h
push    3
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    0Eh
push    0Fh
mov     eax, [ebp+8]
push    eax
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    0Eh
push    2
mov     ecx, [ebp+8]
push    ecx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 0F097965Dh
mov     [ebp+8], edx
push    15h
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 247C11F6h
mov     [ebp+8], ecx
push    14h
push    1
mov     edx, [ebp+8]
push    edx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    0Ah
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_4
add     esp, 8
```

```
                mov     [ebp+8], eax
                push    19h
                push    1
                push    1Eh
                mov     ecx, [ebp+8]
                push    ecx
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                push    0Bh
                push    8
                mov     edx, [ebp+8]
                push    edx
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    2
                push    1Dh
                mov     eax, [ebp+8]
                push    eax
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     ecx, [ebp+8]
                xor     ecx, 71B455A8h
                mov     [ebp+8], ecx
                push    0Eh
                push    2
                mov     edx, [ebp+8]
                push    edx
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_47E508(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                mov     eax, [ebp+8]
                xor     eax, 0FDEB38C7h
                mov     [ebp+8], eax
                push    1
                mov     ecx, [ebp+8]
```

```
push    ecx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    2
push    8
push    4
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    0Ch
push    7
mov     eax, [ebp+8]
push    eax
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    10h
push    7
mov     ecx, [ebp+8]
push    ecx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 0DCA20F48h
mov     [ebp+8], edx
push    16h
push    8
mov     eax, [ebp+8]
push    eax
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    7
push    4
push    1Ah
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    0Fh
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    2
push    7
```

```
                        push    5
                        mov     eax, [ebp+8]
                        push    eax
                        call    B6_Sublayer2_3
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    3
                        push    1Bh
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    B6sublayer2_1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        mov     edx, [ebp+8]
                        xor     edx, 4A7BFE98h
                        mov     [ebp+8], edx
                        push    8
                        push    15h
                        mov     eax, [ebp+8]
                        push    eax
                        call    B6sublayer2_1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        mov     ecx, [ebp+8]
                        xor     ecx, 71912DB8h
                        mov     [ebp+8], ecx
                        push    9
                        push    1
                        mov     edx, [ebp+8]
                        push    edx
                        call    B6Sublayer2_2
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    9
                        push    8
                        push    17h
                        mov     eax, [ebp+8]
                        push    eax
                        call    B6_Sublayer2_3
                        add     esp, 10h
                        mov     [ebp+8], eax
                        mov     eax, [ebp+8]
                        pop     ebp
                        retn
     }
}



__declspec(naked) void sub_47E746(void)
{
     __asm
```

```
{
            push    ebp
            mov     ebp, esp
            mov     eax, [ebp+8]
            xor     eax, 3F4C93CAh
            mov     [ebp+8], eax
            push    12h
            mov     ecx, [ebp+8]
            push    ecx
            call    B6_Sublayer2_4
            add     esp, 8
            mov     [ebp+8], eax
            push    0Ah
            push    9
            mov     edx, [ebp+8]
            push    edx
            call    B6sublayer2_1
            add     esp, 0Ch
            mov     [ebp+8], eax
            mov     eax, [ebp+8]
            xor     eax, 16F12999h
            mov     [ebp+8], eax
            push    3
            push    4
            mov     ecx, [ebp+8]
            push    ecx
            call    B6sublayer2_1
            add     esp, 0Ch
            mov     [ebp+8], eax
            push    4
            push    5
            push    0Bh
            mov     edx, [ebp+8]
            push    edx
            call    B6_Sublayer2_3
            add     esp, 10h
            mov     [ebp+8], eax
            push    1
            push    18h
            mov     eax, [ebp+8]
            push    eax
            call    B6sublayer2_1
            add     esp, 0Ch
            mov     [ebp+8], eax
            push    15h
            push    2
            mov     ecx, [ebp+8]
            push    ecx
            call    B6Sublayer2_2
            add     esp, 0Ch
            mov     [ebp+8], eax
            push    9
            push    5
            push    18h
            mov     edx, [ebp+8]
            push    edx
            call    B6_Sublayer2_3
            add     esp, 10h
            mov     [ebp+8], eax
            push    3
            push    12h
            mov     eax, [ebp+8]
            push    eax
            call    B6sublayer2_1
            add     esp, 0Ch
            mov     [ebp+8], eax
            push    0Bh
            mov     ecx, [ebp+8]
            push    ecx
            call    B6_Sublayer2_4
            add     esp, 8
            mov     [ebp+8], eax
            push    0Eh
            push    8
            mov     edx, [ebp+8]
            push    edx
            call    B6Sublayer2_2
            add     esp, 0Ch
            mov     [ebp+8], eax
            push    1
            push    1Bh
            mov     eax, [ebp+8]
            push    eax
            call    B6sublayer2_1
            add     esp, 0Ch
            mov     [ebp+8], eax
            push    8
            push    5
            push    16h
            mov     ecx, [ebp+8]
            push    ecx
            call    B6_Sublayer2_3
            add     esp, 10h
            mov     [ebp+8], eax
            mov     edx, [ebp+8]
            xor     edx, 0A59EEE9Ah
            mov     [ebp+8], edx
            push    0Eh
            mov     eax, [ebp+8]
            push    eax
            call    B6_Sublayer2_4
            add     esp, 8
            mov     [ebp+8], eax
            push    19h
```

```
                        push    3
                        push    1Bh
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    B6_Sublayer2_3
                        add     esp, 10h
                        mov     [ebp+8], eax
                        mov     eax, [ebp+8]
                        pop     ebp
                        retn
        }
}



__declspec(naked) void sub_47E9B4(void)
{
        __asm
        {
                        push    ebp
                        mov     ebp, esp
                        mov     eax, [ebp+8]
                        xor     eax, 10AFC3E4h
                        mov     [ebp+8], eax
                        push    8
                        push    6
                        push    0Bh
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    B6_Sublayer2_3
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    4
                        mov     edx, [ebp+8]
                        push    edx
                        call    B6_Sublayer2_4
                        add     esp, 8
                        mov     [ebp+8], eax
                        push    2
                        push    0Ah
                        push    8
                        mov     eax, [ebp+8]
                        push    eax
                        call    B6_Sublayer2_3
                        add     esp, 10h
                        mov     [ebp+8], eax
                        mov     ecx, [ebp+8]
                        xor     ecx, 8C6C2052h
                        mov     [ebp+8], ecx
                        push    0Bh
                        push    3
                        mov     edx, [ebp+8]
```

```
push    edx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    2
push    3
push    7
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    0Eh
push    6
mov     ecx, [ebp+8]
push    ecx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    19h
push    2
mov     edx, [ebp+8]
push    edx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
mov     eax, [ebp+8]
xor     eax, 2ACA701Ah
mov     [ebp+8], eax
push    0Ah
push    4
mov     ecx, [ebp+8]
push    ecx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    9
push    2
push    0Eh
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    0Ch
push    6
mov     eax, [ebp+8]
push    eax
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
mov     eax, [ebp+8]
```

```
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_47EB90(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                mov     eax, [ebp+8]
                xor     eax, 96174FB5h
                mov     [ebp+8], eax
                push    0Ch
                push    7
                mov     ecx, [ebp+8]
                push    ecx
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0Ch
                push    0Dh
                mov     edx, [ebp+8]
                push    edx
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    14h
                mov     eax, [ebp+8]
                push    eax
                call    B6_Sublayer2_4
                add     esp, 8
                mov     [ebp+8], eax
                push    0
                push    0Fh
                mov     ecx, [ebp+8]
                push    ecx
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    17h
                push    4
                mov     edx, [ebp+8]
                push    edx
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    6
                push    1
                push    14h
                mov     eax, [ebp+8]
                push    eax
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                mov     ecx, [ebp+8]
                xor     ecx, 0B4324752h
                mov     [ebp+8], ecx
                push    0Ch
                push    3
                mov     edx, [ebp+8]
                push    edx
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    1Fh
                mov     eax, [ebp+8]
                push    eax
                call    B6_Sublayer2_4
                add     esp, 8
                mov     [ebp+8], eax
                push    6
                push    19h
                mov     ecx, [ebp+8]
                push    ecx
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    5
                push    8
                push    10h
                mov     edx, [ebp+8]
                push    edx
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                push    0Dh
                push    7
                mov     eax, [ebp+8]
                push    eax
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     ecx, [ebp+8]
                xor     ecx, 414B8E93h
                mov     [ebp+8], ecx
                push    16h
                push    0
                push    1Dh
                mov     edx, [ebp+8]
                push    edx
```

```
                        call    B6_Sublayer2_3
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    12h
                        push    0Bh
                        mov     eax, [ebp+8]
                        push    eax
                        call    B6sublayer2_1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    14h
                        push    3
                        push    1Bh
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    B6_Sublayer2_3
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    18h
                        mov     edx, [ebp+8]
                        push    edx
                        call    B6_Sublayer2_4
                        add     esp, 8
                        mov     [ebp+8], eax
                        push    7
                        push    0Eh
                        mov     eax, [ebp+8]
                        push    eax
                        call    B6sublayer2_1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    1Bh
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    B6_Sublayer2_4
                        add     esp, 8
                        mov     [ebp+8], eax
                        mov     eax, [ebp+8]
                        pop     ebp
                        retn
            }
}



__declspec(naked) void sub_47EE6B(void)
{
            __asm
            {
                        push    ebp
                        mov     ebp, esp
                        mov     eax, [ebp+8]
```

```
xor     eax, 8B8BAF82h
mov     [ebp+8], eax
push    16h
push    7
mov     ecx, [ebp+8]
push    ecx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    11h
push    0Bh
mov     edx, [ebp+8]
push    edx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    14h
push    5
mov     eax, [ebp+8]
push    eax
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    4
push    13h
push    0Bh
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 0DDF185A2h
mov     [ebp+8], edx
push    5
push    9
push    12h
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    0Eh
push    10h
mov     ecx, [ebp+8]
push    ecx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    16h
push    8
mov     edx, [ebp+8]
```

```
                    push    edx
                    call    B6Sublayer2_2
                    add     esp, 0Ch
                    mov     [ebp+8], eax
                    push    2
                    push    1
                    push    7
                    mov     eax, [ebp+8]
                    push    eax
                    call    B6_Sublayer2_3
                    add     esp, 10h
                    mov     [ebp+8], eax
                    push    14h
                    mov     ecx, [ebp+8]
                    push    ecx
                    call    B6_Sublayer2_4
                    add     esp, 8
                    mov     [ebp+8], eax
                    push    2
                    push    0Bh
                    mov     edx, [ebp+8]
                    push    edx
                    call    B6sublayer2_1
                    add     esp, 0Ch
                    mov     [ebp+8], eax
                    push    0Bh
                    mov     eax, [ebp+8]
                    push    eax
                    call    B6_Sublayer2_4
                    add     esp, 8
                    mov     [ebp+8], eax
                    push    3
                    push    4
                    push    15h
                    mov     ecx, [ebp+8]
                    push    ecx
                    call    B6_Sublayer2_3
                    add     esp, 10h
                    mov     [ebp+8], eax
                    push    15h
                    push    7
                    mov     edx, [ebp+8]
                    push    edx
                    call    B6Sublayer2_2
                    add     esp, 0Ch
                    mov     [ebp+8], eax
                    mov     eax, [ebp+8]
                    pop     ebp
                    retn
        }
    }
```

```
__declspec(naked) void sub_47F09F(void)
{
     __asm
     {
                  push    ebp
                  mov     ebp, esp
                  mov     eax, [ebp+8]
                  xor     eax, 6760502h
                  mov     [ebp+8], eax
                  push    13h
                  push    7
                  mov     ecx, [ebp+8]
                  push    ecx
                  call    B6Sublayer2_2
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  mov     edx, [ebp+8]
                  xor     edx, 17019638h
                  mov     [ebp+8], edx
                  push    4
                  push    2
                  push    0Eh
                  mov     eax, [ebp+8]
                  push    eax
                  call    B6_Sublayer2_3
                  add     esp, 10h
                  mov     [ebp+8], eax
                  push    5
                  mov     ecx, [ebp+8]
                  push    ecx
                  call    B6_Sublayer2_4
                  add     esp, 8
                  mov     [ebp+8], eax
                  push    1Ah
                  push    4
                  mov     edx, [ebp+8]
                  push    edx
                  call    B6sublayer2_1
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    16h
                  push    6
                  mov     eax, [ebp+8]
                  push    eax
                  call    B6Sublayer2_2
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    7
                  push    8
                  push    17h
```

```
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    1
push    1Dh
mov     edx, [ebp+8]
push    edx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    4
push    5
push    17h
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    12h
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 87E29F3Ch
mov     [ebp+8], edx
push    1
push    1Ah
mov     eax, [ebp+8]
push    eax
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    9
push    2
mov     ecx, [ebp+8]
push    ecx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    0Bh
push    0
push    1Fh
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    3
```

```
                push    15h
                mov     eax, [ebp+8]
                push    eax
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     ecx, [ebp+8]
                xor     ecx, 1AD7F4D0h
                mov     [ebp+8], ecx
                push    0Eh
                push    6
                mov     edx, [ebp+8]
                push    edx
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    3
                push    18h
                mov     eax, [ebp+8]
                push    eax
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     ecx, [ebp+8]
                xor     ecx, offset unk_552CA9
                mov     [ebp+8], ecx
                push    0Ah
                push    0
                push    12h
                mov     edx, [ebp+8]
                push    edx
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                push    2
                push    1Bh
                mov     eax, [ebp+8]
                push    eax
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_47D04F(void)
{
    __asm
```

```
{
            push    ebp
            mov     ebp, esp
            push    7
            push    16h
            mov     eax, [ebp+8]
            push    eax
            call    B6sublayer2_1
            add     esp, 0Ch
            mov     [ebp+8], eax
            mov     ecx, [ebp+8]
            xor     ecx, 0D972B853h
            mov     [ebp+8], ecx
            push    13h
            push    8
            mov     edx, [ebp+8]
            push    edx
            call    B6Sublayer2_2
            add     esp, 0Ch
            mov     [ebp+8], eax
            push    0FFFFFFFBh
            push    0Eh
            push    7
            mov     eax, [ebp+8]
            push    eax
            call    B6_Sublayer2_3
            add     esp, 10h
            mov     [ebp+8], eax
            mov     ecx, [ebp+8]
            xor     ecx, 4B487412h
            mov     [ebp+8], ecx
            push    0Ch
            push    9
            mov     edx, [ebp+8]
            push    edx
            call    B6sublayer2_1
            add     esp, 0Ch
            mov     [ebp+8], eax
            push    0Dh
            push    2
            mov     eax, [ebp+8]
            push    eax
            call    B6Sublayer2_2
            add     esp, 0Ch
            mov     [ebp+8], eax
            push    5
            push    15h
            mov     ecx, [ebp+8]
            push    ecx
            call    B6sublayer2_1
            add     esp, 0Ch
            mov     [ebp+8], eax
```

```
                        mov     edx, [ebp+8]
                        xor     edx, 6E4957A8h
                        mov     [ebp+8], edx
                        mov     eax, [ebp+8]
                        pop     ebp
                        retn
        }
}



__declspec(naked) void sub_47D1D2(void)
{
        __asm
        {
                        push    ebp
                        mov     ebp, esp
                        mov     eax, [ebp+8]
                        xor     eax, 40174D5Bh
                        mov     [ebp+8], eax
                        push    0FFFFFFF4h
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    B6_Sublayer2_4
                        add     esp, 8
                        mov     [ebp+8], eax
                        push    0
                        push    1Eh
                        mov     edx, [ebp+8]
                        push    edx
                        call    B6sublayer2_1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    14h
                        push    1
                        mov     eax, [ebp+8]
                        push    eax
                        call    B6Sublayer2_2
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        mov     ecx, [ebp+8]
                        xor     ecx, 0A52B3D68h
                        mov     [ebp+8], ecx
                        push    0FFFFFFF9h
                        push    14h
                        push    0Bh
                        mov     edx, [ebp+8]
                        push    edx
                        call    B6_Sublayer2_3
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    0FFFFFFE6h
```

```
                 mov     eax, [ebp+8]
                 push    eax
                 call    B6_Sublayer2_4
                 add     esp, 8
                 mov     [ebp+8], eax
                 mov     ecx, [ebp+8]
                 xor     ecx, 8B9D36E9h
                 mov     [ebp+8], ecx
                 push    0FFFFFFF3h
                 push    2
                 push    14h
                 mov     edx, [ebp+8]
                 push    edx
                 call    B6_Sublayer2_3
                 add     esp, 10h
                 mov     [ebp+8], eax
                 push    0Ah
                 push    0Ah
                 mov     eax, [ebp+8]
                 push    eax
                 call    B6sublayer2_1
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 push    0FFFFFFEAh
                 push    0
                 push    1Ch
                 mov     ecx, [ebp+8]
                 push    ecx
                 call    B6_Sublayer2_3
                 add     esp, 10h
                 mov     [ebp+8], eax
                 push    1Ah
                 push    5
                 mov     edx, [ebp+8]
                 push    edx
                 call    B6sublayer2_1
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 mov     eax, [ebp+8]
                 xor     eax, 0FB1E52AFh
                 mov     [ebp+8], eax
                 mov     eax, [ebp+8]
                 pop     ebp
                 retn
     }
}



__declspec(naked) void sub_47D3D7(void)
{
     __asm
```

```
{
            push    ebp
            mov     ebp, esp
            push    0FFFFFFF1h
            mov     eax, [ebp+8]
            push    eax
            call    B6_Sublayer2_4
            add     esp, 8
            mov     [ebp+8], eax
            mov     ecx, [ebp+8]
            xor     ecx, 0F185A47Ch
            mov     [ebp+8], ecx
            push    0FFFFFFFEh
            push    0Fh
            push    9
            mov     edx, [ebp+8]
            push    edx
            call    B6_Sublayer2_3
            add     esp, 10h
            mov     [ebp+8], eax
            push    0FFFFFFF1h
            mov     eax, [ebp+8]
            push    eax
            call    B6_Sublayer2_4
            add     esp, 8
            mov     [ebp+8], eax
            push    4
            push    17h
            mov     ecx, [ebp+8]
            push    ecx
            call    B6sublayer2_1
            add     esp, 0Ch
            mov     [ebp+8], eax
            push    9
            push    2
            mov     edx, [ebp+8]
            push    edx
            call    B6Sublayer2_2
            add     esp, 0Ch
            mov     [ebp+8], eax
            push    0FFFFFFF7h
            push    1
            push    1Eh
            mov     eax, [ebp+8]
            push    eax
            call    B6_Sublayer2_3
            add     esp, 10h
            mov     [ebp+8], eax
            push    0Eh
            push    8
            mov     ecx, [ebp+8]
            push    ecx
```

```
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    0
push    11h
mov     edx, [ebp+8]
push    edx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFF5h
push    0
push    18h
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 81A5E699h
mov     [ebp+8], ecx
push    0FFFFFFE6h
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    0FFFFFFFAh
push    9
push    0Eh
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    0FFFFFFEFh
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    17h
push    4
mov     edx, [ebp+8]
push    edx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
mov     eax, [ebp+8]
xor     eax, 0FBBD38E7h
mov     [ebp+8], eax
mov     eax, [ebp+8]
```

```
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_47D5E3(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                push    0FFFFFFFAh
                push    3
                push    0Dh
                mov     eax, [ebp+8]
                push    eax
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                push    0FFFFFFEEh
                mov     ecx, [ebp+8]
                push    ecx
                call    B6_Sublayer2_4
                add     esp, 8
                mov     [ebp+8], eax
                push    0Fh
                push    5
                mov     edx, [ebp+8]
                push    edx
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0FFFFFFFBh
                push    7
                push    8
                mov     eax, [ebp+8]
                push    eax
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                push    16h
                push    2
                mov     ecx, [ebp+8]
                push    ecx
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0FFFFFFF9h
                push    0Ah
                push    0Bh
```

```
                        mov     edx, [ebp+8]
                        push    edx
                        call    B6_Sublayer2_3
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    7
                        push    8
                        mov     eax, [ebp+8]
                        push    eax
                        call    B6sublayer2_1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    0Bh
                        push    7
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    B6Sublayer2_2
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    0FFFFFFFBh
                        push    7
                        push    11h
                        mov     edx, [ebp+8]
                        push    edx
                        call    B6_Sublayer2_3
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    6
                        push    0Ah
                        mov     eax, [ebp+8]
                        push    eax
                        call    B6sublayer2_1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    0Ah
                        push    3
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    B6Sublayer2_2
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        mov     edx, [ebp+8]
                        xor     edx, 5C2ACD4Dh
                        mov     [ebp+8], edx
                        mov     eax, [ebp+8]
                        pop     ebp
                        retn
        }
    }
```

```
__declspec(naked) void sub_47D854(void)
{
     __asm
     {
                  push    ebp
                  mov     ebp, esp
                  push    0FFFFFFFBh
                  push    7
                  push    17h
                  mov     eax, [ebp+8]
                  push    eax
                  call    B6_Sublayer2_3
                  add     esp, 10h
                  mov     [ebp+8], eax
                  push    0FFFFFEDh
                  mov     ecx, [ebp+8]
                  push    ecx
                  call    B6_Sublayer2_4
                  add     esp, 8
                  mov     [ebp+8], eax
                  push    0Ch
                  push    0Fh
                  mov     edx, [ebp+8]
                  push    edx
                  call    B6sublayer2_1
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    0Eh
                  push    1
                  mov     eax, [ebp+8]
                  push    eax
                  call    B6Sublayer2_2
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  mov     ecx, [ebp+8]
                  xor     ecx, 89C7C9A6h
                  mov     [ebp+8], ecx
                  push    9
                  push    8
                  mov     edx, [ebp+8]
                  push    edx
                  call    B6Sublayer2_2
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    3
                  push    0Fh
                  mov     eax, [ebp+8]
                  push    eax
                  call    B6sublayer2_1
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    15h
                  push    4
                  mov     ecx, [ebp+8]
                  push    ecx
                  call    B6Sublayer2_2
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    0FFFFFFEBh
                  mov     edx, [ebp+8]
                  push    edx
                  call    B6_Sublayer2_4
                  add     esp, 8
                  mov     [ebp+8], eax
                  push    12h
                  push    5
                  mov     eax, [ebp+8]
                  push    eax
                  call    B6Sublayer2_2
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    6
                  push    12h
                  mov     ecx, [ebp+8]
                  push    ecx
                  call    B6sublayer2_1
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    14h
                  push    1
                  mov     edx, [ebp+8]
                  push    edx
                  call    B6Sublayer2_2
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    0
                  push    1Fh
                  mov     eax, [ebp+8]
                  push    eax
                  call    B6sublayer2_1
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    0Ah
                  push    7
                  mov     ecx, [ebp+8]
                  push    ecx
                  call    B6Sublayer2_2
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    0FFFFFFFDh
                  push    0Dh
                  push    8
                  mov     edx, [ebp+8]
                  push    edx
```

```
                 call    B6_Sublayer2_3
                 add     esp, 10h
                 mov     [ebp+8], eax
                 push    0FFFFFFE5h
                 mov     eax, [ebp+8]
                 push    eax
                 call    B6_Sublayer2_4
                 add     esp, 8
                 mov     [ebp+8], eax
                 push    13h
                 push    5
                 mov     ecx, [ebp+8]
                 push    ecx
                 call    B6Sublayer2_2
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 push    7
                 push    0Fh
                 mov     edx, [ebp+8]
                 push    edx
                 call    B6sublayer2_1
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 push    0FFFFFFE9h
                 push    0
                 push    1Ch
                 mov     eax, [ebp+8]
                 push    eax
                 call    B6_Sublayer2_3
                 add     esp, 10h
                 mov     [ebp+8], eax
                 push    0FFFFFFFCh
                 mov     ecx, [ebp+8]
                 push    ecx
                 call    B6_Sublayer2_4
                 add     esp, 8
                 mov     [ebp+8], eax
                 mov     edx, [ebp+8]
                 xor     edx, 47D8FFBDh
                 mov     [ebp+8], edx
                 mov     eax, [ebp+8]
                 pop     ebp
                 retn
     }
}



__declspec(naked) void sub_47DB59(void)
{
     __asm
     {
```

```
push    ebp
mov     ebp, esp
push    0
push    1Fh
mov     eax, [ebp+8]
push    eax
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFFAh
push    3
push    16h
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    0Ch
push    1
mov     edx, [ebp+8]
push    edx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFFCh
push    0Bh
push    6
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    0Bh
push    4
mov     ecx, [ebp+8]
push    ecx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 7C3547F7h
mov     [ebp+8], edx
push    0FFFFFFEBh
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    14h
push    4
mov     ecx, [ebp+8]
push    ecx
```

```
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFF7h
push    0
push    0Dh
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    0Bh
push    1
mov     eax, [ebp+8]
push    eax
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFE5h
push    0
push    1Ch
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    0FFFFFFFBh
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    0FFFFFFFBh
push    1
push    1Eh
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    0Eh
push    1
mov     ecx, [ebp+8]
push    ecx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 5BFB6B28h
mov     [ebp+8], edx
push    0FFFFFFF2h
mov     eax, [ebp+8]
```

```
                push    eax
                call    B6_Sublayer2_4
                add     esp, 8
                mov     [ebp+8], eax
                mov     ecx, [ebp+8]
                xor     ecx, 8B465658h
                mov     [ebp+8], ecx
                push    0FFFFFFEDh
                push    0
                push    1Fh
                mov     edx, [ebp+8]
                push    edx
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                push    10h
                push    2
                mov     eax, [ebp+8]
                push    eax
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    17h
                push    8
                mov     ecx, [ebp+8]
                push    ecx
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     edx, [ebp+8]
                xor     edx, 24A4A3B5h
                mov     [ebp+8], edx
                mov     eax, [ebp+8]
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_47DDB8(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                push    0Dh
                push    0Bh
                mov     eax, [ebp+8]
                push    eax
                call    B6sublayer2_1
                add     esp, 0Ch
```

```
mov     [ebp+8], eax
push    0FFFFFFF1h
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 251AFCFEh
mov     [ebp+8], edx
push    0FFFFFFFDh
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    0FFFFFFF5h
push    4
push    19h
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    0FFFFFFF2h
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    0FFFFFFFCh
push    8
push    13h
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 6BA330BFh
mov     [ebp+8], ecx
push    0Ah
push    12h
mov     edx, [ebp+8]
push    edx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    0Ch
push    8
mov     eax, [ebp+8]
push    eax
```

```
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     ecx, [ebp+8]
                xor     ecx, 5ED8936Ch
                mov     [ebp+8], ecx
                push    0FFFFFFEFh
                mov     edx, [ebp+8]
                push    edx
                call    B6_Sublayer2_4
                add     esp, 8
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                xor     eax, 34473F96h
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_47E008(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                push    0
                push    7
                mov     eax, [ebp+8]
                push    eax
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0FFFFFFE1h
                mov     ecx, [ebp+8]
                push    ecx
                call    B6_Sublayer2_4
                add     esp, 8
                mov     [ebp+8], eax
                push    0FFFFFFFBh
                push    0Ah
                push    0Fh
                mov     edx, [ebp+8]
                push    edx
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                push    0FFFFFFFCh
                mov     eax, [ebp+8]
```

```
push    eax
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 3A22CD09h
mov     [ebp+8], ecx
push    13h
push    4
mov     edx, [ebp+8]
push    edx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
mov     eax, [ebp+8]
xor     eax, 0F6547803h
mov     [ebp+8], eax
push    0FFFFFFF3h
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 0ABA4CE9Ah
mov     [ebp+8], edx
push    11h
push    9
mov     eax, [ebp+8]
push    eax
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    11h
push    5
mov     ecx, [ebp+8]
push    ecx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFF0h
push    0
push    1Fh
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    0Fh
push    2
mov     eax, [ebp+8]
push    eax
```

```
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 49253F31h
mov     [ebp+8], ecx
push    0
push    1Eh
mov     edx, [ebp+8]
push    edx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
mov     eax, [ebp+8]
xor     eax, 3773D297h
mov     [ebp+8], eax
push    3
push    11h
mov     ecx, [ebp+8]
push    ecx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 70648B0Dh
mov     [ebp+8], edx
push    0Dh
push    6
mov     eax, [ebp+8]
push    eax
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFF5h
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    10h
push    0Bh
mov     edx, [ebp+8]
push    edx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
mov     eax, [ebp+8]
xor     eax, 5A07B2A1h
mov     [ebp+8], eax
mov     eax, [ebp+8]
pop     ebp
retn
```

```
    }
}



__declspec(naked) void sub_47E23B(void)
{
    __asm
    {
            push    ebp
            mov     ebp, esp
            push    0FFFFFFF5h
            mov     eax, [ebp+8]
            push    eax
            call    B6_Sublayer2_4
            add     esp, 8
            mov     [ebp+8], eax
            push    5
            push    19h
            mov     ecx, [ebp+8]
            push    ecx
            call    B6sublayer2_1
            add     esp, 0Ch
            mov     [ebp+8], eax
            push    10h
            push    5
            mov     edx, [ebp+8]
            push    edx
            call    B6Sublayer2_2
            add     esp, 0Ch
            mov     [ebp+8], eax
            mov     eax, [ebp+8]
            xor     eax, 0A8DB534Eh
            mov     [ebp+8], eax
            push    0FFFFFFEBh
            mov     ecx, [ebp+8]
            push    ecx
            call    B6_Sublayer2_4
            add     esp, 8
            mov     [ebp+8], eax
            push    0FFFFFFF4h
            push    6
            push    0Fh
            mov     edx, [ebp+8]
            push    edx
            call    B6_Sublayer2_3
            add     esp, 10h
            mov     [ebp+8], eax
            push    0FFFFFFE3h
            mov     eax, [ebp+8]
            push    eax
            call    B6_Sublayer2_4
```

```
                add     esp, 8
                mov     [ebp+8], eax
                push    5
                push    5
                mov     ecx, [ebp+8]
                push    ecx
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     edx, [ebp+8]
                xor     edx, 58887477h
                mov     [ebp+8], edx
                push    0FFFFFFF9h
                push    9
                push    10h
                mov     eax, [ebp+8]
                push    eax
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                mov     ecx, [ebp+8]
                xor     ecx, 949D26F0h
                mov     [ebp+8], ecx
                mov     eax, [ebp+8]
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_47E402(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                push    0Eh
                push    2
                mov     eax, [ebp+8]
                push    eax
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     ecx, [ebp+8]
                xor     ecx, 71B455A8h
                mov     [ebp+8], ecx
                push    2
                push    1Dh
                mov     edx, [ebp+8]
                push    edx
                call    B6sublayer2_1
```

```
add     esp, 0Ch
mov     [ebp+8], eax
push    0Bh
push    8
mov     eax, [ebp+8]
push    eax
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFE7h
push    1
push    1Eh
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    0FFFFFFF6h
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    14h
push    1
mov     eax, [ebp+8]
push    eax
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 247C11F6h
mov     [ebp+8], ecx
push    0FFFFFFEBh
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
mov     eax, [ebp+8]
xor     eax, 0F097965Dh
mov     [ebp+8], eax
push    0Eh
push    2
mov     ecx, [ebp+8]
push    ecx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    0Eh
push    0Fh
mov     edx, [ebp+8]
```

```
                push    edx
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0FFFFFFFEh
                push    17h
                push    3
                mov     eax, [ebp+8]
                push    eax
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                push    0FFFFFFFAh
                mov     ecx, [ebp+8]
                push    ecx
                call    B6_Sublayer2_4
                add     esp, 8
                mov     [ebp+8], eax
                mov     edx, [ebp+8]
                xor     edx, 0EECED199h
                mov     [ebp+8], edx
                mov     eax, [ebp+8]
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_47E627(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                push    0FFFFFFF7h
                push    8
                push    17h
                mov     eax, [ebp+8]
                push    eax
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                push    9
                push    1
                mov     ecx, [ebp+8]
                push    ecx
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     edx, [ebp+8]
                xor     edx, 71912DB8h
```

```
mov     [ebp+8], edx
push    8
push    15h
mov     eax, [ebp+8]
push    eax
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 4A7BFE98h
mov     [ebp+8], ecx
push    3
push    1Bh
mov     edx, [ebp+8]
push    edx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFFEh
push    7
push    5
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    0FFFFFFF1h
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    0FFFFFFF9h
push    4
push    1Ah
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    16h
push    8
mov     eax, [ebp+8]
push    eax
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 0DCA20F48h
mov     [ebp+8], ecx
push    10h
push    7
```

```
                mov     edx, [ebp+8]
                push    edx
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0Ch
                push    7
                mov     eax, [ebp+8]
                push    eax
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0FFFFFFFEh
                push    8
                push    4
                mov     ecx, [ebp+8]
                push    ecx
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                push    0FFFFFFFFh
                mov     edx, [ebp+8]
                push    edx
                call    B6_Sublayer2_4
                add     esp, 8
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                xor     eax, 0FDEB38C7h
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_47E87C(void)
{
     __asm
     {
                push    ebp
                mov     ebp, esp
                push    0FFFFFFE7h
                push    3
                push    1Bh
                mov     eax, [ebp+8]
                push    eax
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                push    0FFFFFFF2h
```

```
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 0A59EEE9Ah
mov     [ebp+8], edx
push    0FFFFFFF8h
push    5
push    16h
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    1
push    1Bh
mov     ecx, [ebp+8]
push    ecx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    0Eh
push    8
mov     edx, [ebp+8]
push    edx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFF5h
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    3
push    12h
mov     ecx, [ebp+8]
push    ecx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFF7h
push    5
push    18h
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    15h
```

```
                        push    2
                        mov     eax, [ebp+8]
                        push    eax
                        call    B6Sublayer2_2
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    1
                        push    18h
                        mov     ecx, [ebp+8]
                        push    ecx
                        call    B6sublayer2_1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    0FFFFFFFCh
                        push    5
                        push    0Bh
                        mov     edx, [ebp+8]
                        push    edx
                        call    B6_Sublayer2_3
                        add     esp, 10h
                        mov     [ebp+8], eax
                        push    3
                        push    4
                        mov     eax, [ebp+8]
                        push    eax
                        call    B6sublayer2_1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        mov     ecx, [ebp+8]
                        xor     ecx, 16F12999h
                        mov     [ebp+8], ecx
                        push    0Ah
                        push    9
                        mov     edx, [ebp+8]
                        push    edx
                        call    B6sublayer2_1
                        add     esp, 0Ch
                        mov     [ebp+8], eax
                        push    0FFFFFFEEh
                        mov     eax, [ebp+8]
                        push    eax
                        call    B6_Sublayer2_4
                        add     esp, 8
                        mov     [ebp+8], eax
                        mov     ecx, [ebp+8]
                        xor     ecx, 3F4C93CAh
                        mov     [ebp+8], ecx
                        mov     eax, [ebp+8]
                        pop     ebp
                        retn
    }
}
```

```
__declspec(naked) void sub_47EAA2(void)
{
     __asm
     {
                  push    ebp
                  mov     ebp, esp
                  push    0Ch
                  push    6
                  mov     eax, [ebp+8]
                  push    eax
                  call    B6Sublayer2_2
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    0FFFFFFF7h
                  push    2
                  push    0Eh
                  mov     ecx, [ebp+8]
                  push    ecx
                  call    B6_Sublayer2_3
                  add     esp, 10h
                  mov     [ebp+8], eax
                  push    0Ah
                  push    4
                  mov     edx, [ebp+8]
                  push    edx
                  call    B6Sublayer2_2
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  mov     eax, [ebp+8]
                  xor     eax, 2ACA701Ah
                  mov     [ebp+8], eax
                  push    19h
                  push    2
                  mov     ecx, [ebp+8]
                  push    ecx
                  call    B6sublayer2_1
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    0Eh
                  push    6
                  mov     edx, [ebp+8]
                  push    edx
                  call    B6Sublayer2_2
                  add     esp, 0Ch
                  mov     [ebp+8], eax
                  push    0FFFFFFFEh
                  push    3
                  push    7
                  mov     eax, [ebp+8]
```

```
                push    eax
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                push    0Bh
                push    3
                mov     ecx, [ebp+8]
                push    ecx
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     edx, [ebp+8]
                xor     edx, 8C6C2052h
                mov     [ebp+8], edx
                push    0FFFFFFFEh
                push    0Ah
                push    8
                mov     eax, [ebp+8]
                push    eax
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                push    0FFFFFFFCh
                mov     ecx, [ebp+8]
                push    ecx
                call    B6_Sublayer2_4
                add     esp, 8
                mov     [ebp+8], eax
                push    0FFFFFFF8h
                push    6
                push    0Bh
                mov     edx, [ebp+8]
                push    edx
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                xor     eax, 10AFC3E4h
                mov     [ebp+8], eax
                mov     eax, [ebp+8]
                pop     ebp
                retn
     }
}



__declspec(naked) void sub_47ECFE(void)
{
     __asm
     {
                push    ebp
```

```
mov     ebp, esp
push    0FFFFFFE5h
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    7
push    0Eh
mov     ecx, [ebp+8]
push    ecx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFE8h
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    0FFFFFFECh
push    3
push    1Bh
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    12h
push    0Bh
mov     ecx, [ebp+8]
push    ecx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFEAh
push    0
push    1Dh
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
mov     eax, [ebp+8]
xor     eax, 414B8E93h
mov     [ebp+8], eax
push    0Dh
push    7
mov     ecx, [ebp+8]
push    ecx
call    B6Sublayer2_2
add     esp, 0Ch
```

```
mov     [ebp+8], eax
push    0FFFFFFFBh
push    8
push    10h
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    6
push    19h
mov     eax, [ebp+8]
push    eax
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFE1h
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    0Ch
push    3
mov     edx, [ebp+8]
push    edx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
mov     eax, [ebp+8]
xor     eax, 0B4324752h
mov     [ebp+8], eax
push    0FFFFFFFAh
push    1
push    14h
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    17h
push    4
mov     edx, [ebp+8]
push    edx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    0
push    0Fh
mov     eax, [ebp+8]
push    eax
call    B6sublayer2_1
```

```
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0FFFFFFECh
                mov     ecx, [ebp+8]
                push    ecx
                call    B6_Sublayer2_4
                add     esp, 8
                mov     [ebp+8], eax
                push    0Ch
                push    0Dh
                mov     edx, [ebp+8]
                push    edx
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0Ch
                push    7
                mov     eax, [ebp+8]
                push    eax
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     ecx, [ebp+8]
                xor     ecx, 96174FB5h
                mov     [ebp+8], ecx
                mov     eax, [ebp+8]
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_47EF85(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                push    15h
                push    7
                mov     eax, [ebp+8]
                push    eax
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    0FFFFFFFDh
                push    4
                push    15h
                mov     ecx, [ebp+8]
                push    ecx
                call    B6_Sublayer2_3
```

```
add     esp, 10h
mov     [ebp+8], eax
push    0FFFFFFF5h
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    2
push    0Bh
mov     eax, [ebp+8]
push    eax
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFECh
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    0FFFFFFFEh
push    1
push    7
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    16h
push    8
mov     eax, [ebp+8]
push    eax
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    0Eh
push    10h
mov     ecx, [ebp+8]
push    ecx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFFBh
push    9
push    12h
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
mov     eax, [ebp+8]
```

```
                xor     eax, 0DDF185A2h
                mov     [ebp+8], eax
                push    0FFFFFFFCh
                push    13h
                push    0Bh
                mov     ecx, [ebp+8]
                push    ecx
                call    B6_Sublayer2_3
                add     esp, 10h
                mov     [ebp+8], eax
                push    14h
                push    5
                mov     edx, [ebp+8]
                push    edx
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    11h
                push    0Bh
                mov     eax, [ebp+8]
                push    eax
                call    B6sublayer2_1
                add     esp, 0Ch
                mov     [ebp+8], eax
                push    16h
                push    7
                mov     ecx, [ebp+8]
                push    ecx
                call    B6Sublayer2_2
                add     esp, 0Ch
                mov     [ebp+8], eax
                mov     edx, [ebp+8]
                xor     edx, 8B8BAF82h
                mov     [ebp+8], edx
                mov     eax, [ebp+8]
                pop     ebp
                retn
    }
}



__declspec(naked) void sub_47F22B(void)
{
    __asm
    {
                push    ebp
                mov     ebp, esp
                push    2
                push    1Bh
                mov     eax, [ebp+8]
                push    eax
```

```
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFF6h
push    0
push    12h
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, offset unk_552CA9
mov     [ebp+8], edx
push    3
push    18h
mov     eax, [ebp+8]
push    eax
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    0Eh
push    6
mov     ecx, [ebp+8]
push    ecx
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
mov     edx, [ebp+8]
xor     edx, 1AD7F4D0h
mov     [ebp+8], edx
push    3
push    15h
mov     eax, [ebp+8]
push    eax
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFF5h
push    0
push    1Fh
mov     ecx, [ebp+8]
push    ecx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    9
push    2
mov     edx, [ebp+8]
push    edx
call    B6Sublayer2_2
add     esp, 0Ch
```

```
mov     [ebp+8], eax
push    1
push    1Ah
mov     eax, [ebp+8]
push    eax
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
mov     ecx, [ebp+8]
xor     ecx, 87E29F3Ch
mov     [ebp+8], ecx
push    0FFFFFFEEh
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_4
add     esp, 8
mov     [ebp+8], eax
push    0FFFFFFFCh
push    5
push    17h
mov     eax, [ebp+8]
push    eax
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    1
push    1Dh
mov     ecx, [ebp+8]
push    ecx
call    B6sublayer2_1
add     esp, 0Ch
mov     [ebp+8], eax
push    0FFFFFFF9h
push    8
push    17h
mov     edx, [ebp+8]
push    edx
call    B6_Sublayer2_3
add     esp, 10h
mov     [ebp+8], eax
push    16h
push    6
mov     eax, [ebp+8]
push    eax
call    B6Sublayer2_2
add     esp, 0Ch
mov     [ebp+8], eax
push    1Ah
push    4
mov     ecx, [ebp+8]
push    ecx
call    B6sublayer2_1
```

```
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 push    0FFFFFFFBh
                 mov     edx, [ebp+8]
                 push    edx
                 call    B6_Sublayer2_4
                 add     esp, 8
                 mov     [ebp+8], eax
                 push    0FFFFFFFCh
                 push    2
                 push    0Eh
                 mov     eax, [ebp+8]
                 push    eax
                 call    B6_Sublayer2_3
                 add     esp, 10h
                 mov     [ebp+8], eax
                 mov     ecx, [ebp+8]
                 xor     ecx, 17019638h
                 mov     [ebp+8], ecx
                 push    13h
                 push    7
                 mov     edx, [ebp+8]
                 push    edx
                 call    B6Sublayer2_2
                 add     esp, 0Ch
                 mov     [ebp+8], eax
                 mov     eax, [ebp+8]
                 xor     eax, 6760502h
                 mov     [ebp+8], eax
                 mov     eax, [ebp+8]
                 pop     ebp
                 retn
     }
}
```
------------------------------------------------------------------

**THE END**

Thanx to Arilou and Neoxquick who helped me with Armadillo. Thanx to my personal beta-reader, Devine9.
Greetings to all my RET friends and all UIC guys.
A super greet to the super The_Enigma aka Giulia :)
GoodBye!

```
[AndreaGeddon]    andreageddon@gmail.com       my mail
[RET]             www.reteam.org               RET's great site
[UIC]             www.quequero.org             Italian University
of Cracking
```