IDA output:

call	___main	
mov	dword ptr [esp+1A8h], 7	creates variable I will call a
mov	dword ptr [esp+1A4h], 64h	creates variable I will call b which is the max value of the iterator
mov	dword ptr [esp+1ACh], 0	creates variable I will call c which is an iterator
jmp	short loc_401619	jumps to loc_401619

loc_4015FC:

```
mov     eax, [esp+1ACh]                 moves c to eax
lea     edx, [eax+1]                    moved eax+1 or c+1 to edx
mov     eax, [esp+1ACh]                 moves c to eax
mov     [esp+eax*4+14h], edx            moves edx or c+1 to [esp+4c+14]
add     dword ptr [esp+1ACh], 1         increments b

Loc_401619:
mov     eax, [esp+1ACh]                 moves iterator c to eax
cmp     eax, [esp+1A4h]                 compares eax of c to b
jl      short loc_4015FC                jumps if eax is less than b so we have not reached
                                        end of loop
mov     eax, [esp+1A8h]                 moves a or 7 to eax
mov     [esp+8], eax    ; int           moves eax to [esp+8]
mov     eax, [esp+1A4h]                 moves b to eax
mov     [esp+4], eax    ; int           moves eax to [esp+4]
lea     eax, [esp+1B0h+var_19C]         moves [esp+432+-412] or [esp+14h] to eax
mov     [esp], eax      ; int *         moves eax to [esp]
call    __Z5proc1Piii   ; proc1(int *,int,int)  calls function
mov     [esp+4], eax
mov     dword ptr [esp], offset aD ; "%d\n"
call    _printf                         prints eax
mov     eax, 0
Leave
Retn
_main endp
```

****What the function does is loop through with an iterator that goes from 0 to 99 and adds the values of the index+1 to the array at index. Array[index] = index+1.**
So the array at the end stores values from 1 to 100.
It then calls a new function with the values of 1A8 which is 7, 1A4h which is 99 and the pointer to the beginning of the newly created array.
proc(array, 99, 7)....

```
 __Z5proc1Piii   proc near              ; CODE XREF: _main+7E↓p
.text:00401500
.text:00401500 var_10          = dword ptr -10h
.text:00401500 var_C           = dword ptr -0Ch
.text:00401500 var_8           = dword ptr -8
.text:00401500 var_4           = dword ptr -4
.text:00401500 arg_0           = dword ptr  8
.text:00401500 arg_4           = dword ptr  0Ch
.text:00401500 arg_8           = dword ptr  10h
.text:00401500
```

```
.text:00401500              push    ebp
.text:00401501              mov     ebp, esp                ebp = esp
.text:00401503              sub     esp, 10h                esp = esp-10h
.text:00401506              mov     [ebp+var_C], 0          moves 0 to [ebp+-0Ch]
.text:0040150D              mov     [ebp+var_10], 0         moves 0 to [epb+-10h]
.text:00401514              mov     [ebp+var_4], 0          moves 0 to [epb+-4]
.text:0040151B              jmp     loc_4015B7
.text:00401520 ; ---------------------------------------------------------------------------
.text:00401520
.text:00401520 loc_401520:                          ; CODE XREF: proc1(int *,int,int)+BD↓j
.text:00401520              mov     [ebp+var_8], 1
.text:00401527              jmp     short loc_40155E
.text:00401529 ; ---------------------------------------------------------------------------
.text:00401529
.text:00401529 loc_401529:                          ; CODE XREF: proc1(int *,int,int)+64↓j
.text:00401529              jmp     short loc_401538
.text:0040152B ; ---------------------------------------------------------------------------
.text:0040152B
.text:0040152B loc_40152B:                          ; CODE XREF: proc1(int *,int,int)+4B↓j
.text:0040152B              mov     eax, [ebp+var_C]
.text:0040152E              add     eax, 1
.text:00401531              cdq
.text:00401532              idiv    [ebp+arg_4]
.text:00401535              mov     [ebp+var_C], edx
.text:00401538
.text:00401538 loc_401538:                          ; CODE XREF: proc1(int *,int,int):loc_401529↑j
.text:00401538              mov     eax, [ebp+var_C]
.text:0040153B              lea     edx, ds:0[eax*4]
.text:00401542              mov     eax, [ebp+arg_0]
.text:00401545              add     eax, edx
.text:00401547              mov     eax, [eax]
.text:00401549              test    eax, eax
.text:0040154B              jz      short loc_40152B
.text:0040154D              add     [ebp+var_8], 1
.text:00401551              mov     eax, [ebp+var_C]
.text:00401554              add     eax, 1
.text:00401557              cdq
.text:00401558              idiv    [ebp+arg_4]
.text:0040155B              mov     [ebp+var_C], edx
.text:0040155E
.text:0040155E loc_40155E:                          ; CODE XREF: proc1(int *,int,int)+27↑j
.text:0040155E              mov     eax, [ebp+var_8]
.text:00401561              cmp     eax, [ebp+arg_8]
```

```
.text:00401564                jl      short loc_401529
.text:00401566                jmp     short loc_401575
.text:00401568 ; ---------------------------------------------------------------------------
.text:00401568
.text:00401568 loc_401568:                            ; CODE XREF: proc1(int *,int,int)+88↓j
.text:00401568                mov     eax, [ebp+var_C]
.text:0040156B                add     eax, 1
.text:0040156E                cdq
.text:0040156F                idiv    [ebp+arg_4]
.text:00401572                mov     [ebp+var_C], edx
.text:00401575
.text:00401575 loc_401575:                            ; CODE XREF: proc1(int *,int,int)+66↑j
.text:00401575                mov     eax, [ebp+var_C]
.text:00401578                lea     edx, ds:0[eax*4]
.text:0040157F                mov     eax, [ebp+arg_0]
.text:00401582                add     eax, edx
.text:00401584                mov     eax, [eax]
.text:00401586                test    eax, eax
.text:00401588                jz      short loc_401568
.text:0040158A                mov     eax, [ebp+var_C]
.text:0040158D                lea     edx, ds:0[eax*4]
.text:00401594                mov     eax, [ebp+arg_0]
.text:00401597                add     eax, edx
.text:00401599                mov     eax, [eax]
.text:0040159B                mov     [ebp+var_10], eax
.text:0040159E                mov     eax, [ebp+var_C]
.text:004015A1                lea     edx, ds:0[eax*4]
.text:004015A8                mov     eax, [ebp+arg_0]
.text:004015AB                add     eax, edx
.text:004015AD                mov     dword ptr [eax], 0
.text:004015B3                add     [ebp+var_4], 1
.text:004015B7
.text:004015B7 loc_4015B7:                            ; CODE XREF: proc1(int *,int,int)+1B↑j
.text:004015B7                mov     eax, [ebp+var_4]          moves to eax
.text:004015BA                cmp     eax, [ebp+arg_4]          compares eax to first argument
.text:004015BD                jl      loc_401520                jumps if is eax is less
.text:004015C3                mov     eax, [ebp+var_10]
.text:004015C6                leave
.text:004015C7                retn
.text:004015C7 __Z5proc1Piii  endp
.text:004015C7
.text:004015C8
```

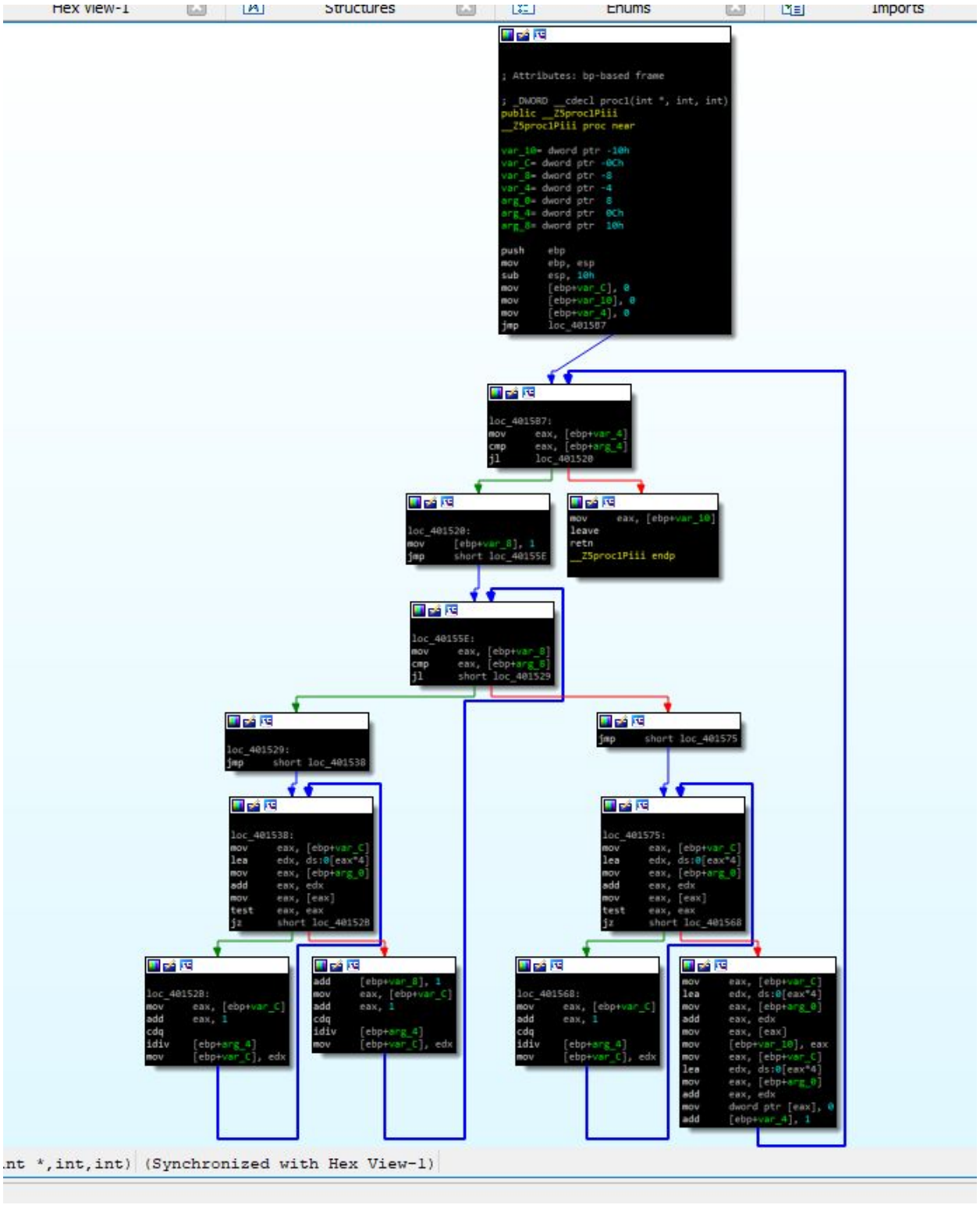
```
.text:004015C8 ; =============== S U B R O U T I N E
=======================================
.text:004015C8
.text:004015C8 ; Attributes: bp-based frame fuzzy-sp
.text:004015C8
.text:004015C8 ; int __cdecl main(int argc, const char **argv, const char **envp)
.text:004015C8                 public _main
.text:004015C8 _main           proc near               ; CODE XREF: ___tmainCRTStartup+25D↑p
.text:004015C8
.text:004015C8 var_19C         = dword ptr -19Ch
.text:004015C8 argc            = dword ptr  8
.text:004015C8 argv            = dword ptr  0Ch
.text:004015C8 envp            = dword ptr  10h
.text:004015C8
.text:004015C8                 push    ebp
.text:004015C9                 mov     ebp, esp
.text:004015CB                 and     esp, 0FFFFFFF0h
.text:004015CE                 sub     esp, 1B0h
.text:004015D4                 call    ___main
.text:004015D9                 mov     dword ptr [esp+1A8h], 7
.text:004015E4                 mov     dword ptr [esp+1A4h], 64h
.text:004015EF                 mov     dword ptr [esp+1ACh], 0
.text:004015FA                 jmp     short loc_401619
.text:004015FC ; ---------------------------------------------------------------------------
.text:004015FC
.text:004015FC loc_4015FC:                             ; CODE XREF: _main+5F↓j
.text:004015FC                 mov     eax, [esp+1ACh]
.text:00401603                 lea     edx, [eax+1]
.text:00401606                 mov     eax, [esp+1ACh]
.text:0040160D                 mov     [esp+eax*4+14h], edx
.text:00401611                 add     dword ptr [esp+1ACh], 1
.text:00401619
.text:00401619 loc_401619:                             ; CODE XREF: _main+32↑j
.text:00401619                 mov     eax, [esp+1ACh]
.text:00401620                 cmp     eax, [esp+1A4h]
.text:00401627                 jl      short loc_4015FC
.text:00401629                 mov     eax, [esp+1A8h]
.text:00401630                 mov     [esp+8], eax    ; int
.text:00401634                 mov     eax, [esp+1A4h]
.text:0040163B                 mov     [esp+4], eax    ; int
.text:0040163F                 lea     eax, [esp+1B0h+var_19C]
.text:00401643                 mov     [esp], eax      ; int *
.text:00401646                 call    __Z5proc1Piii   ; proc1(int *,int,int)
```

```
.text:0040164B                mov     [esp+4], eax
.text:0040164F                mov     dword ptr [esp], offset aD ; "%d\n"
.text:00401656                call    _printf
.text:0040165B                mov     eax, 0
.text:00401660                leave
.text:00401661                retn
.text:00401661 _main          endp
.text:00401661
.text:00401661 ; ---------------------------------------------------------------------------
.text:00401662                align 10h
.text:00401670
.text:00401670 ; =============== S U B R O U T I N E
=======================================
.text:00401670
.text:00401670 ; Attributes: static
.text:00401670
.text:00401670 ; BOOL __stdcall __dyn_tls_dtor(HANDLE hDllHandle, DWORD dwReason,
LPVOID lpreserved)
.text:00401670                public ___dyn_tls_dtor@12
.text:00401670 ___dyn_tls_dtor@12 proc near           ; DATA XREF: .CRT:___xl_d↓o
.text:00401670
.text:00401670 var_1C         = dword ptr -1Ch
.text:00401670 reason         = dword ptr -18h
.text:00401670 reserved       = dword ptr -14h
.text:00401670 hDllHandle     = dword ptr  4
.text:00401670 dwReason       = dword ptr  8
.text:00401670 lpreserved     = dword ptr  0Ch
.text:00401670
.text:00401670                sub     esp, 1Ch
.text:00401673                mov     eax, [esp+1Ch+dwReason]
.text:00401677                test    eax, eax
.text:00401679                jz      short loc_401690
.text:0040167B                cmp     eax, 3
.text:0040167E                jz      short loc_401690
.text:00401680                mov     eax, 1
.text:00401685                add     esp, 1Ch
.text:00401688                retn    0Ch
.text:00401688 ; ---------------------------------------------------------------------------
.text:0040168B                align 10h
.text:00401690
.text:00401690 loc_401690:                            ; CODE XREF: __dyn_tls_dtor(x,x,x)+9↑j
.text:00401690                                        ; __dyn_tls_dtor(x,x,x)+E↑j
.text:00401690                mov     edx, [esp+1Ch+lpreserved]
```

```
.text:00401694                mov     [esp+1Ch+reason], eax ; reason
.text:00401698                mov     eax, [esp+1Ch+hDllHandle]
.text:0040169C                mov     [esp+1Ch+reserved], edx ; reserved
.text:004016A0                mov     [esp+1Ch+var_1C], eax ; hDllHandle
.text:004016A3                call    ___mingw_TLScallback
.text:004016A8                mov     eax, 1
.text:004016AD                add     esp, 1Ch
.text:004016B0                retn    0Ch
.text:004016B0 ___dyn_tls_dtor@12 endp
```

****For this function it was too hard to follow in text form so I used IDA's graph form that is why the above is not annotated fully**

This proc1 function was very complex to someone who has not done a lot of assembly such as myself and I ran out of time trying to finish this problem. What I had figured out is in the Q4.c code, but I could not tell you the overall functionality other than the output should be 50.


Hex View-1
Structures
Enums
Imports
; Attributes: bp-based frame
; _DWORD __cdecl proc1(int *, int, int)
public __Z5proc1Piii
__Z5proc1Piii proc near
var_10= dword ptr -10h
var_C= dword ptr -0Ch
var_8= dword ptr -8
var_4= dword ptr -4
arg_0= dword ptr 8
arg_4= dword ptr 0Ch
arg_8= dword ptr 10h
push ebp
mov ebp, esp
sub esp, 10h
mov [ebp+var_C], 0
mov [ebp+var_10], 0
mov [ebp+var_4], 0
jmp loc_401587
loc_401587:
mov eax, [ebp+var_4]
cmp eax, [ebp+arg_4]
jl loc_401520
loc_401520:
mov [ebp+var_8], 1
jmp short loc_40155E
mov eax, [ebp+var_10]
leave
retn
__Z5proc1Piii endp
loc_40155E:
mov eax, [ebp+var_8]
cmp eax, [ebp+arg_8]
jl short loc_401529
loc_401529:
jmp short loc_401538
jmp short loc_401575
loc_401538:
mov eax, [ebp+var_C]
lea edx, ds:0[eax*4]
mov eax, [ebp+arg_0]
add eax, edx
mov eax, [eax]
test eax, eax
jz short loc_401528
loc_401575:
mov eax, [ebp+var_C]
lea edx, ds:0[eax*4]
mov eax, [ebp+arg_0]
add eax, edx
mov eax, [eax]
test eax, eax
jz short loc_401568
loc_401528:
mov eax, [ebp+var_C]
add eax, 1
cdq
idiv [ebp+arg_4]
mov [ebp+var_C], edx
add [ebp+var_8], 1
mov eax, [ebp+var_C]
add eax, 1
cdq
idiv [ebp+arg_4]
mov [ebp+var_C], edx
loc_401568:
mov eax, [ebp+var_C]
add eax, 1
cdq
idiv [ebp+arg_4]
mov [ebp+var_C], edx
mov eax, [ebp+var_C]
lea edx, ds:0[eax*4]
mov eax, [ebp+arg_0]
add eax, edx
mov eax, [eax]
mov [ebp+var_10], eax
mov eax, [ebp+var_C]
lea edx, ds:0[eax*4]
mov eax, [ebp+arg_0]
add eax, edx
mov dword ptr [eax], 0
add [ebp+var_4], 1
nt *,int,int) (Synchronized with Hex View-1)