MAIN44.ASM

.586

.model flat, stdcall

option casemap :none

include D:\masm32\include\kernel32.inc

include D:\masm32\include\user32.inc

include D:\masm32\include\windows.inc

include modulee.inc

include longopp.inc

includelib D:\masm32\lib\kernel32.lib

includelib D:\masm32\lib\user32.lib

.const

.data

.data

Valueb1 db 31

Caption1 db "A+B (1 variant)",0

Caption2 db "A+B (2 variant)",0

Caption3 db "A-B ",0

TextBuf1 db 52 dup(?)

TextBuf2 db 52 dup(?)

TextBuf3 db 88 dup(?)

ValueA1 dd 52 dup(?)

ValueB1 dd 52 dup(?)

ValueA2 dd 52 dup(?)

ValueB2 dd 52 dup(?)

ValueA3 dd 88 dup(?)

ValueB3 dd 88 dup(?)

Result1 dd 52 dup(0)

Result2 dd 52 dup(0)

Result3 dd 88 dup(0)

.code

main:

;А+B 1

mov eax , 80010001h

mov ecx, 52 ; ECX = потрібна кількість повторень

mov edx,0

cycleAB1:

mov dword ptr[ValueA1+4\*edx], eax

mov dword ptr[ValueB1+4\*edx], 80000001h

add eax , 10000h

inc edx

dec ecx ; лічильник зменшуємо на 1

jnz cycleAB1

push offset ValueA1

push offset ValueB1

push offset Result1

call Add\_416\_LONGOP

push offset TextBuf1

push offset Result1

push 416

call StrHex\_MY

invoke MessageBoxA, 0, ADDR TextBuf1, ADDR Caption1,0

;А+B 2

mov eax , 0Bh

mov ecx, 52 ; ECX = потрібна кількість повторень

mov edx,0

cycleAB2:

mov dword ptr[ValueA2+4\*edx], eax

mov dword ptr[ValueB2+4\*edx], 00000001h

add eax , 1h

inc edx

dec ecx ; лічильник зменшуємо на 1

jnz cycleAB2

push offset ValueA2

push offset ValueB2

push offset Result2

call Add\_416\_LONGOP

push offset TextBuf2

push offset Result2

push 416

call StrHex\_MY

invoke MessageBoxA, 0, ADDR TextBuf2, ADDR Caption2,0

;А-B

mov eax , 0Bh

mov ecx, 88 ; ECX = потрібна кількість повторень

mov edx,0

cycleAB3:

mov dword ptr[ValueA3+4\*edx], 0

mov dword ptr[ValueB3+4\*edx], eax

add eax , 1h

inc edx

dec ecx ; лічильник зменшуємо на 1

jnz cycleAB3

push offset ValueA3

push offset ValueB3

push offset Result3

call Sub\_704\_LONGOP

push offset TextBuf3

push offset Result3

push 704

call StrHex\_MY

invoke MessageBoxA, 0, ADDR TextBuf3, ADDR Caption3,0

invoke ExitProcess, 0

end main

LONGOP.ASM

.586

.model flat, c

.code

Add\_416\_LONGOP proc

push ebp

mov ebp,esp

mov esi, [ebp+16] ;ESI = адреса A

mov ebx, [ebp+12] ;EBX = адреса B

mov edi, [ebp+8] ;EDI = адреса результату

mov ecx, 52 ; ECX = потрібна кількість повторень

mov edx,0

clc ; обнулює біт CF регістру EFLAGS

cycle:

mov eax, dword ptr[esi+4\*edx]

adc eax, dword ptr[ebx+4\*edx] ; додавання групи з 32 бітів

mov dword ptr[edi+4\*edx], eax

inc edx

dec ecx ; лічильник зменшуємо на 1

jnz cycle

pop ebp

ret 12

Add\_416\_LONGOP endp

Sub\_704\_LONGOP proc

push ebp

mov ebp,esp

mov esi, [ebp+16] ;ESI = адреса A

mov ebx, [ebp+12] ;EBX = адреса B

mov edi, [ebp+8] ;EDI = адреса результату

mov ecx, 88 ; ECX = потрібна кількість повторень

mov edx,0

clc ; обнулює біт CF регістру EFLAGS

cycle:

mov eax, dword ptr[esi+4\*edx]

sbb eax, dword ptr[ebx+4\*edx] ; віднімання групи з 32 бітів

mov dword ptr[edi+4\*edx], eax

inc edx

dec ecx ; лічильник зменшуємо на 1

jnz cycle

pop ebp

ret 12

Sub\_704\_LONGOP endp

End

LONGOP.INC

EXTERN Add\_416\_LONGOP : proc

EXTERN Sub\_704\_LONGOP : proc

MODULEE.ASM

.586

.model flat, c

.code

;процедура StrHex\_MY записує текст шістнадцятькового коду

;перший параметр - адреса буфера результату (рядка символів)

;другий параметр - адреса числа

;третій параметр - розрядність числа у бітах (має бути кратна 8)

StrHex\_MY proc

push ebp

mov ebp,esp

mov ecx, [ebp+8] ;кількість бітів числа

cmp ecx, 0

jle @exitp

shr ecx, 3 ;кількість байтів числа

mov esi, [ebp+12] ;адреса числа

mov ebx, [ebp+16] ;адреса буфера результату

@cycle:

mov dl, byte ptr[esi+ecx-1] ;байт числа - це дві hex-цифри

mov al, dl

shr al, 4 ;старша цифра

call HexSymbol\_MY

mov byte ptr[ebx], al

mov al, dl ;молодша цифра

call HexSymbol\_MY

mov byte ptr[ebx+1], al

mov eax, ecx

cmp eax, 4

jle @next

dec eax

and eax, 3 ;проміжок розділює групи по вісім цифр

cmp al, 0

jne @next

mov byte ptr[ebx+2], 32 ;код символа проміжку

inc ebx

@next:

add ebx, 2

dec ecx

jnz @cycle

mov byte ptr[ebx], 0 ;рядок закінчується нулем

@exitp:

pop ebp

ret 12

StrHex\_MY endp

;ця процедура обчислює код hex-цифри

;параметр - значення AL

;результат -> AL

HexSymbol\_MY proc

and al, 0Fh

add al, 48 ;так можна тільки для цифр 0-9

cmp al, 58

jl @exitp

add al, 7 ;для цифр A,B,C,D,E,F

@exitp:

ret

HexSymbol\_MY endp

End

MODULEE.INC

EXTERN StrHex\_MY : proc