Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**ЛАБОРАТОРНА РОБОТА № 9**

з дисципліни «Системне програмування-1» на тему

«Використання функцій API Win32 у програмах на асемблері»

ВИКОНАЛА:

студентка ІІ курсу ФІОТ

групи ІО-64

Бровченко Анастасія

Залікова - 6403

ПЕРЕВІРИВ:

ст.вик. Порєв В. М.

Київ – 2018

**Мета:** Навчитися використовувати у програмах на асемблері функції Windows динамічного виділення пам’яті та запису файлів

**Завдання:**

1. Створити у середовищі MS Visual Studio проект з ім’ям Lab9.
2. Написати вихідний текст програми згідно варіанту завдання. У проекті мають бути головний файл main9.asm та модулі module (за необхідності) та модуль longop попередніх робіт.
3. У цьому проекті кожний модуль може окремо компілюватися.
4. Скомпілювати вихідний текст і отримати виконуємий файл програми.
5. Перевірити роботу програми. Налагодити програму.
6. Отримати результати – файл числових значень згідно варіанту завдання.
7. Проаналізувати та прокоментувати результати, вихідний текст та дизасембльований машинний код програми.

**Варіант завдання:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № варіанту | Варіант циклу факторіалу | Код результату | Масиви для даних  підвищеної  розрядності | Масив-буфер для  імені файлу |
| 3 | 1 | шістнадцятковий | динамічні | динамічний |

**Роздруківка тексту програми:**

**module.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

**module.inc**

EXTERN StrHex\_MY : proc

**longop.asm**

.586

.model flat, c

.data

x dd 1

n dd 0

inner dd 14

outer dd 14

.code

Mul\_Nx32\_LONGOP proc

push ebp

mov ebp, esp

mov edi, [ebp + 12]

mov ebx, [ebp + 8]

mov x, ebx

mov n, 14

xor ebx, ebx

xor ecx, ecx

@mult32:

mov eax, dword ptr[edi + ecx]

mul x

mov dword ptr[edi + ecx], eax

clc

adc dword ptr[edi + ecx], ebx

mov ebx, edx

add ecx, 4

dec n

jnz @mult32

pop ebp

ret 8

Mul\_Nx32\_LONGOP endp

Mul\_NxN\_LONGOP proc

push ebp

mov ebp, esp

mov esi, dword ptr[ebp + 16]

mov edi, dword ptr[ebp + 12]

mov ebx, dword ptr[ebp + 8]

xor ecx, ecx

xor edx, edx

mov outer, 14

@cycle:

mov eax, dword ptr[esi + edx]

push edx

push ebx

mov ebx, ecx

sub ebx, edx

mul dword ptr[edi + ebx]

pop ebx

add dword ptr[ebx + ecx], eax

adc dword ptr[ebx + ecx + 4], edx

jnc @ncf

xor eax, eax

mov eax, ecx

@cf:

add eax, 4

add dword ptr[ebx + eax + 4], 1

jc @cf

@ncf:

pop edx

add ecx, 4

dec inner

jnz @cycle

add edx, 4

mov ecx, edx

mov inner, 14

dec outer

jnz @cycle

pop ebp

ret 12

Mul\_NxN\_LONGOP endp

end

**longop.inc**

XTERN Mul\_NxN\_LONGOP : proc

EXTERN Mul\_Nx32\_LONGOP : proc

**main.asm**

.586

.model flat, stdcall

option casemap: none

include \masm32\include\kernel32.inc

include \masm32\include\user32.inc

include \masm32\include\windows.inc

include \masm32\include\comdlg32.inc

include module.inc

include longop.inc

includelib \masm32\lib\comdlg32.lib

includelib \masm32\lib\kernel32.lib

includelib \masm32\lib\user32.lib

.data

counter dd 1

crlf db 13, 10

hFile dd 0

pVal dd ?

pRes dd ?

szFileName dd 64 dup(0)

pFileName db "lab9.txt", 0

TextBuf db 448 dup(0)

.code

MySaveFileName proc

LOCAL ofn : OPENFILENAME

invoke RtlZeroMemory, ADDR ofn, SIZEOF ofn

mov ofn.lStructSize, SIZEOF ofn

mov ofn.lpstrFile, offset szFileName

mov ofn.nMaxFile, SIZEOF szFileName

invoke GetSaveFileName, ADDR ofn

ret

MySaveFileName endp

main:

invoke GlobalAlloc, GPTR, 448

mov pRes, eax

mov dword ptr[eax], 1h

call MySaveFileName

cmp eax, 0

jz @exit

invoke CreateFile, ADDR szFileName,

GENERIC\_WRITE,

FILE\_SHARE\_WRITE,

0, CREATE\_ALWAYS,

FILE\_ATTRIBUTE\_NORMAL,

0

cmp eax, INVALID\_HANDLE\_VALUE

je @exit ; доступ до файлу неможливий

mov hFile, eax

; обчислення факторіала

@loop1:

push pRes

push counter

call Mul\_Nx32\_LONGOP

inc counter

push offset TextBuf

push pRes

push 160

call StrHex\_MY

invoke WriteFile, hFile, ADDR TextBuf, 112, NULL, 0

invoke WriteFile, hFile, ADDR crlf, 2, NULL, 0

cmp counter,36

jle @loop1

invoke CloseHandle, hFile

@exit:

invoke GlobalFree, pRes

invoke ExitProcess, 0

end main

**Результат роботи програми:**

00000000 00000000 00000000 00000000 00000001

00000000 00000000 00000000 00000000 00000002

00000000 00000000 00000000 00000000 00000006

00000000 00000000 00000000 00000000 00000018

00000000 00000000 00000000 00000000 00000078

00000000 00000000 00000000 00000000 000002D0

00000000 00000000 00000000 00000000 000013B0

00000000 00000000 00000000 00000000 00009D80

00000000 00000000 00000000 00000000 00058980

00000000 00000000 00000000 00000000 00375F00

00000000 00000000 00000000 00000000 02611500

00000000 00000000 00000000 00000000 1C8CFC00

00000000 00000000 00000000 00000001 7328CC00

00000000 00000000 00000000 00000014 4C3B2800

00000000 00000000 00000000 00000130 77775800

00000000 00000000 00000000 00001307 77758000

00000000 00000000 00000000 0001437E EECD8000

00000000 00000000 00000000 0016BEEC CA730000

00000000 00000000 00000000 01B02B93 06890000

00000000 00000000 00000000 21C3677C 82B40000

00000000 00000000 00000002 C5077D36 B8C40000

00000000 00000000 0000003C EEA4C2B3 E0D80000

00000000 00000000 00000579 70CD7E29 33680000

00000000 00000000 00008362 9343D3DC D1C00000

00000000 00000000 000CD4A0 619FB090 7BC00000

00000000 00000000 014D9849 EA37EEAC 91800000

00000000 00000000 232F0FCB B3E62C33 58800000

00000000 00000003 D925BA47 AD2CD59D AE000000

00000000 0000006F 99461A1E 9E1432DC B6000000

00000000 00000D13 F6370F96 865DF5DD 54000000

00000000 0001956A D0AAE33A 4560C5CD 2C000000

00000000 0032AD5A 155C6748 AC18B9A5 80000000

00000000 0688589C C0E9505E 2F2FEE55 80000000

00000000 DE1BC4D1 9EFCAC82 445DA75B 00000000

0000001E 5DCBE8A8 BC8B95CF 58CDE171 00000000

00000445 30ACB7BA 83A11128 7CF3B3E4 00000000

**Висновок:**

У ході виконання лабораторної роботи було створено програму для покрокового запису у файл значення факторіалу числа, а також було закріплено основні навички у використанні функцій API Win32 у програмах на асемблері.