Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

ЛАБОРАТОРНА РОБОТА № 5

З дисципліни «Системне програмування»

ВИКОНАВ:

студент 2 курсу ФІОТ

групи ІО-41

Логвинчук А. І.

ПЕРЕВІРИВ:

ст. вик. Порєв В. М.

Київ – 2016

**Тема:** Програмування множення чисел операцій підвищеної розрядності

**Мета:** Навчитися програмувати на асемблері множення чисел підвищеної розрядності, а також отримати перші навички програмування

власних процедур у модульному проекті.

**Варіант**: 20

n = 30 + 2\*20 = 70

Розрядність n!\*n! – 333 біти.

**Програмний код**

**longop.asm**

.586

.model flat, c

.data

operand1 dd 0h

operand2 dd 0h

res dd 0h

buffer dd 1h

numBit dd 0h

.code

Multip\_Nx32 proc

push ebp

mov ebp, esp

mov ebx, [ebp+16]

mov ecx, [ebp+12]

mov esi, [ebp+8]

clc

xor edi, edi

@loop:

mov edx, [ebx+4\*edi]

mov eax, [ecx]

mul edx

add eax, [esi+4\*edi]

mov [esi+4\*edi], eax

adc edx, [esi+4\*edi+4]

mov [esi+4\*edi+4], edx

inc edi

cmp edi, 11

jl @loop

pop ebp

ret 16

Multip\_Nx32 endp

Multip\_NxN proc

push ebp

mov ebp, esp

mov esi, [ebp+12]

mov ebx, [ebp+8]

mov ecx, 12

@outer\_loop:

push ecx

mov ecx, 12

@inner\_loop:

push ecx

mov ecx, operand1

mov eax, [esi+4\*ecx]

mov ecx, operand2

mul dword ptr[esi+4\*ecx]

mov ecx, res

clc

adc eax, [ebx+4\*ecx]

mov [ebx+4\*ecx], eax

adc edx, [ebx+ecx\*4+4]

mov [ebx+4\*ecx+4], edx

mov eax, [ebx+4\*ecx+4]

inc operand1

inc res

pop ecx

dec ecx

jnz @inner\_loop

inc operand2

xor eax, eax

mov operand1, eax

mov eax, operand2

mov res, eax

pop ecx

dec ecx

jnz @outer\_loop

xor eax, eax

mov operand1, eax

mov operand2, eax

mov res, eax

pop ebp

ret 8

Multip\_NxN endp

**main5.asm**

.586

.model flat, stdcall

include D:\DEV\masm32\include\kernel32.inc

include D:\DEV\masm32\include\user32.inc

include module.inc

include longop.inc

includelib D:\DEV\masm32\lib\kernel32.lib

includelib D:\DEV\masm32\lib\user32.lib

; Варіант n=30+20\*2=70

.data

Caption1 db "Обчислення факторіалу n!", 0

Caption2 db "Обчислення квадрату факторіалу n! x n!", 0

Caption3 db "Перший тест множення N x 32", 0

Caption4 db "Другий тест множення N x N", 0

TextBuff1 db 100 dup(?)

TextBuff2 db 100 dup(?)

TextBuff3 db 100 dup(?)

TextBuff4 db 100 dup(?)

value1 dd 12 dup(?)

value2 dd 1h, 11 dup(0h)

value3 dd 11 dup(0FFFFFFFFh)

value4 dd 12 dup(0FFFFFFFFh)

value5 dd 0FFFFFFFFh

counter dd 1h

n dd 70

result dd 22 dup(?)

resultTest1 dd 12 dup(0)

resultTest2 dd 22 dup(0)

.code

main:

; обчислення факторіала

@loop1:

push offset value1

push offset value2

push counter

call Mu1tip\_Nx32

inc counter

xor ecx, ecx

@loop2:

mov eax, [value1 + 4\*ecx]

mov [value2 + 4\*ecx], eax

inc ecx

cmp ecx, 12

jl @loop2

dec n

jnz @loop1

push offset TextBuff1

push offset value1

push 336

call StrHex\_MY

invoke MessageBoxA, 0, ADDR TextBuff1, ADDR Caption1, 0

; квадрат факторіала

push offset value1

push offset result

call Multip\_NxN

push offset TextBuff2

push offset result

push 672

call StrHex\_MY

invoke MessageBoxA, 0, ADDR TextBuff2, ADDR Caption2, 0

; тест Nx32

push offset value3

push offset value5

push offset resultTest1

call Multip\_Nx32

push offset TextBuff3

push offset resultTest1

push 384

call StrHex\_MY

invoke MessageBoxA, 0, ADDR TextBuff3, ADDR Caption3, 0

; тест NxN

push offset value4

push offset resultTest2

call Mu1tip\_NxN

push offset TextBuff4

push offset resultTest2

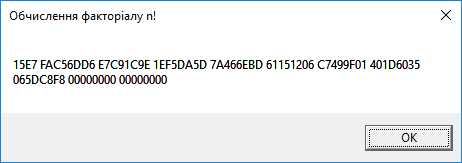
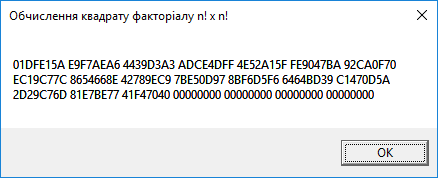
push 768

call StrHex\_MY

invoke MessageBoxA, 0, ADDR TextBuff4, ADDR Caption4, 0

invoke ExitProcess, 0

end main

**Результати роботи програми:**

**Висновок**

В ході виконання роботи я навчився програмувати множення чисел підвищеної розрядності та покращив навички створення власних процедур на мові ассемблера.