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 № 7**

з дисципліни «Системне програмування-1» на тему

«Програмування операцій ділення чисел»

ВИКОНАЛА:

студентка ІІ курсу ФІОТ

групи ІО-64

Бровченко Анастасія

Залікова - 6403

ПЕРЕВІРИВ:

ст.вик. Порєв В. М.

Київ – 2018

**Мета:** Навчитися програмувати на асемблері ділення чисел, вивчити перетворення з двойкової у десяткову систему числення.

**Завдання:**

1. Створити у середовищі MS Visual Studio проект з ім’ям **Lab7**.

2. Написати вихідний текст програми згідно варіанту завдання. У проекті

мають бути три модуля на асемблері:

- головний модуль: файл **main7.asm**. Цей модуль створити та написати заново, частково використавши текст модуля main5.asm попередньої роботи №5;

- другий модуль: модуль **module** попередньої роботи №6;

- третій модуль: модуль **longop** попередньої роботи №6.

3. Додати у модулі процедури, які потрібні для виконання завдання. Обгрунтувати розподіл процедур по модулям.

4. У цьому проекті кожний модуль може окремо компілюватися.

5. Скомпілювати вихідний текст і отримати виконуємий файл програми.

6. Перевірити роботу програми. Налагодити програму.

7. Отримати результати – кодовані значення чисел згідно варіанту завдання.

8. Проаналізувати та прокоментувати результати, вихідний текст та дизасембльований машинний код програми.

**Варіант завдання:**

1. Потрібно запрограмувати на асемблері вивід значення факторіалу *n*! У **десятковому коді**. Використати програмний код обчислення факторіалу попередньої лабораторної роботи №5. Для кожного студента своє значення *n*

*n* = 30 + 2×*H*, *n* = 36, де *H* – це номер студента у журналі.

2. Перетворення у десятковий код запрограмувати діленням числа підвищеної розрядності на 10. Ділення на 10 виконати двома способами – діленням "у стовпчик" та діленням груп бітів. Ділення групами бітів запрограмувати або як 8-бітове, або як 16-бітове, або як 32-бітове відповідно до варіанту. Варіант ділення групами бітів обирається відповідно залишку ділення номеру студента у журналі (*H)* на 3:

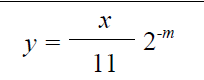
- залишок = 0 – ділення групами по 8 бітів;

- залишок = 1 – ділення групами по 16 бітів;

- залишок = 2 – ділення групами по 32 бітів.

3. Програмний код ділення "у стовпчик" та ділення групами бітів оформити у вигляді процедур. Процедури повинні містити такі параметри: адреса ділимого, розрядність ділимого, значення дільника (B) та інші (за необхідності).

4. Запрограмувати на асемблері також обчислення формули, яку вибрати з таблиці, наведеної нижче.



**Роздруківка тексту програми:**

**module.asm**

.586

.model flat, c

.data

num11 dd 11

.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)

Func proc

push ebp

mov ebp,esp

mov ecx, [ebp+8] ;m

mov eax, [ebp+12] ;X

xor edx, edx

mov ebx, eax

shl ebx, 1

jnc @plus

sub edx, 1

@plus:

idiv num11

sar eax, cl

pop ebp

ret 8

Func endp

End

**module.inc**

EXTERN Func : proc

**longop.asm**

.586

.model flat, c

.data

x dd 1

n dd 0

num10 db 10

inner dd 0

num7 db 7

minn db 0

spacee db 3

.code

Mul\_Nx32\_LONGOP proc

push ebp

mov ebp, esp

mov edi, [ebp + 12]

mov ebx, [ebp + 8]

mov x, ebx

mov n, 13

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

Div\_Column\_LONGOP proc

xor ebx, ebx

xor ecx, ecx

dec edx

cmp byte ptr[esi + edx], 0

jnz @cycleout

inc bl

@cycleout:

mov ch, byte ptr[esi + edx]

@cycleinner:

shl cl, 1

shl bh, 1

shl ch, 1

jnc @zero

inc bh

@zero:

cmp bh, num10

jc @less

inc cl

sub bh, num10

@less:

inc inner

cmp inner, 8

jnz @cycleinner

mov byte ptr[esi + edx], cl

mov inner, 0

sub edx, 1

jnc @cycleout

ret

Div\_Column\_LONGOP endp

Str\_Dec proc

;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)

push ebp

mov ebp,esp

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

shr edx, 3 ;кількість байтів числа

mov esi, [ebp+12] ;адреса числа

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

mov eax, edx

shl eax, 2

mov cl, byte ptr[esi + edx - 1]

and cl, 128

cmp cl, 128

jnz @plus

mov minn, 1

push edx

@minus:

not byte ptr[esi + edx - 1]

sub edx, 1

jnz @minus

inc byte ptr[esi + edx]

pop edx

@plus:

@cycle:

push edx

call Div\_Column\_LONGOP

pop edx

add bh, 48

mov byte ptr[edi + eax], bh

dec eax

cmp bl, 0

jz @cycle

dec edx

jnz @cycle

cmp minn, 1

jc @nomin

mov byte ptr[edi + eax + 1], 45

dec eax

@nomin:

inc eax

@space:

mov byte ptr[edi + eax], 32

sub eax, 1

jnc @space

pop ebp

ret 12

Str\_Dec endp

end

**longop.inc**

EXTERN Mul\_Nx32\_LONGOP : proc

EXTERN Str\_Dec : proc

EXTERN Div\_Column\_LONGOP : proc

**main.asm**

.586

.model flat, stdcall

include \masm32\include\kernel32.inc

include \masm32\include\user32.inc

include module.inc

include longop.inc

includelib \masm32\lib\kernel32.lib

includelib \masm32\lib\user32.lib

option casemap :none

.data

Caption db "n!" ,0

Caption1 db "Function" ,0

textBuf dd 80 dup(?)

textBuf1 dd 30 dup(?)

var dd 13 dup(0)

x dd 36

y dd 0

test1 dd 13 dup(4294967295)

test1res dd 26 dup(0)

test2 dd 13 dup(4294967295)

test31 dd 13 dup(4294967295)

test32 dd 13 dup(0)

test3res dd 26 dup(0)

.code

main:

mov [var], 1

@fact:

push offset var

push x

call Mul\_Nx32\_LONGOP

dec x

jne @fact

push offset textBuf

push offset var

push 160

call Str\_Dec

invoke MessageBoxA, 0, ADDR textBuf, ADDR Caption, 0

push -1475574672 ; X, +- 2^31 2640

push 4 ; m, 0-31

call Func

mov y, eax

push offset textBuf1

push offset y

push 32

call Str\_Dec

invoke MessageBoxA, 0, ADDR textBuf1, ADDR Caption1, 0

invoke ExitProcess,0

end main

**Висновок:**

Під час виконання лабораторної роботи були покращені навички написання власних модулів, роботи з циклами. Наввчились програмувати побітові операції на асемблері, вивчили основні команди обробки бітів.