Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

ЛАБОРАТОРНА РОБОТА № 10

З дисципліни «Системне програмування»

ВИКОНАВ:

студент 2 курсу ФІОТ

групи ІО-41

Логвинчук А. І.

ПЕРЕВІРИВ:

ст. вик. Порєв В. М.

Київ – 2016

**Тема:** Використання у проекті C++ модулів на асемблері

**Мета:** Навчитися створювати програми на С++ з використанням модулів на

асемблері.

**Варіант завдання:**

Вар. 20.

F = A2-B+C2

Розрядність операндів – 192 біти.

**Програмний код**

**Lab10.cpp:**

void MyWork1(HWND hWnd)

{

long oA[6] = {0x80000001, 0x80000001, 0x80000001, 0x80000001,

0x80000001, 0x80000001};

long oB[12] = {0x80010001, 0x80020001, 0x80030001, 0x80040001,

0x80050001, 0x80060001, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000};

long oC[6] = {0x80010001, 0x80020001, 0x80030001, 0x80040001,

0x80050001, 0x80060001};

long aSqr[12] = { 0 };

long cSqr[12] = { 0 };

long aPlusC[12] = { 0 };

long resultFinal[12] = { 0 };

char TextBuf[256];

Mul\_LONGOP(24, aSqr, oA, oA);

Mul\_LONGOP(24, aSqr, oC, oC);

Add\_LONGOP(48, aPlusC, aSqr, cSqr);

Sub\_LONGOP(48, resultFinal, oB, aPlusC);

StrHex\_MY(384, resultFinal, TextBuf);

MessageBox(hWnd, TextBuf, "A^2-B+C^2", MB\_OK);

}

**longop.h:**

extern "C"

{

void Mul\_LONGOP(long bytes, long \*dest, long \*pB, long \*pA);

void Add\_LONGOP(long bytes, long \*dest, long \*pB, long \*pA);

void Sub\_LONGOP(long bytes, long \*dest, long \*pB, long \*pA);

}

**longop.asm:**

.586

.model flat, c

.data

cote dd 0

temp dd 12 dup(0)

buf dd 0

buffer dd 0

Nbit dd 0

.code

Mul\_LONGOP proc bytes:DWORD, dest:DWORD, pB:DWORD, pA:DWORD

mov esi, pA ; ESI = адреса множеного

mov ebx, pB ; EBX = адреса множника

mov edi, dest ; EDI = адреса результату

mov ecx, bytes ; кількість байтів

shr ecx, 2

mov cote, ecx

xor ecx, ecx

@loop:

push ebx

push ecx

push edi

shl ecx, 2

add ebx, ecx

add edi, ecx

shr ecx, 2

xor ecx, ecx

@loop\_p: ; для парних доданків

mov eax, [esi+4\*ecx]

mul dword ptr [ebx]

add [edi+4\*ecx], eax

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

adc byte ptr [edi+4\*ecx+8], 0

inc ecx

inc ecx

cmp ecx, cote

jl @loop\_p

xor ecx, ecx

inc ecx

@loop\_np: ; для непарних доданків

mov eax, [esi+4\*ecx]

mul dword ptr [ebx]

add [edi+4\*ecx], eax

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

adc byte ptr [edi+4\*ecx+8], 0

inc ecx

inc ecx

cmp ecx, cote

jl @loop\_np

pop edi

pop ecx

pop ebx

inc ecx

cmp ecx, cote

jl @loop

ret

Mul\_LONGOP endp

Div10\_LONGOP proc

mov esi, [ebp+24] ; адреса діленого

mov eax, [ebp+20] ; дільник

mov ebx, [ebp+16] ; розрядність діленого

mov edi, [ebp+12] ; адреса частки

mov ecx, [ebp+8] ; адреса залишку

mov buf, eax

shr ebx, 5 ; 4

dec ebx

xor edx, edx

@loop:

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

div dword ptr[buf]

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

dec ebx

cmp ebx, 0

jge @loop

mov dword ptr[ecx], edx

ret

Div10\_LONGOP endp

Add\_LONGOP proc bytes:DWORD, dest:DWORD, pB:DWORD, pA:DWORD

mov esi, pA ; ESI = адреса A

mov ebx, pB ; EBX = адреса B

mov edi, dest ; EDI = адреса результату

mov ecx, bytes ; ECX = потрібна кількість повторень

shr ecx, 2

xor edx, edx

clc ; обнулює біт CF регістру EFLAGS

@loop:

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

cmp edx, ecx

jl @loop

ret

Add\_LONGOP endp

Sub\_LONGOP proc bytes:DWORD, dest:DWORD, pB:DWORD, pA:DWORD

mov esi, pA ; ESI = адреса A

mov ebx, pB ; EBX = адреса B

mov edi, dest ; EDI = адреса результату

mov ecx, bytes ; ECX = потрібна кількість повторень

shr ecx, 2

xor edx, edx

clc ; обнулює біт CF регістру EFLAGS

@loop:

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

cmp edx, ecx

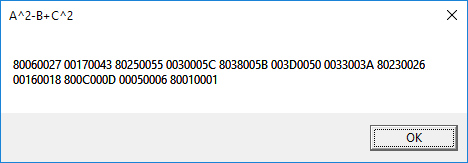
jl @loop

ret

Sub\_LONGOP endp

end

**Результати роботи програми:**



**Висновок**

Під час виконання роботи я навчився підключати модулі асемблерні модулі до проектів С++, дізнався про різні конвенції виклику та написав процедуру обчислення функції згідно варіанту.