Міністерство освіти та науки України

Національний університет «Львівська політехніка»

Кафедра СКС



**ЗВІТ**

До лабораторної роботи №1

з дисципліни

СИСТЕМНЕ ПРОГРАМУВАННЯ, ЧАСТИНА 2

**ЛАБОРАТОРНА РОБОТА № 1**

Варіант-25

Виконав:

ст. гр. КІ-307

Равчак В.А.

Прийняв:

Козак Н.Б.

Львів 2024

ЛАБОРАТОРНА РОБОТА №1. ЗМІШАНЕ ПРОГРАМУВАННЯ МОВАМИ С/C++ ТА АСЕМБЛЕРА

**Мета:** оволодіти навиками створення програм, частини яких написані різними мовами програмування. Засвоїти правила взаємодії між програмними модулями різних мов програмування.

# ЗАВДАННЯ

1. Створити перший програмний проект **для 32-х розрядної платформи**, який реалізує схему викликів **С-ASM-C** та здійснює обчислення заданого виразу, згідно варіанту (таблиця 1.4). Проект повинний складатися з двох модулів, передача параметрів між якими здійснюється через стек. Константа передається через спільну пам’ять, як глобальні дані. Множення/ділення рекомендовано робити через команди зсувів.

**Основний модуль – створюється мовою С/С++**. Він повинен забезпечувати:

* + ввід даних з клавіатури;
  + обчислення виразу засобами мови С/С++;
  + вивід на екран результатів обчислення виразу.
  + виклик асемблерної процедури обчислення виразу.

**Модуль безпосередніх обчислень (асемблерний модуль)** – здійснює обчислення виразу засобами асемблера і реалізує вивід на екран результату обчислення шляхом виклику стандартної функції мови С/C++ printf(), або її аналога.

1. Створити другий програмний проект **для 64-х розрядної платформи**, який також реалізує схему викликів **С-ASM-C** та здійснює обчислення заданого виразу, згідно варіанту. Проект повинна складатися з двох модулів, передача параметрів між якими повинна відбуватися, згідно угод для платформи х64. Константа передається через спільну пам’ять, як глобальна змінна. Множення/ділення рекомендовано робити через команди зсувів.

Призначення програмних модулів таке ж, як в першому програмному проекті.

1. Відлагодити та протестувати обидва програмні проекти. Результати роботи програм продемонструвати викладачу.
2. Скласти звіт про виконану роботу з приведенням текстів програм та коментарів до неї, а також результатів її роботи.
3. Дати відповідь на контрольні запитання викладача.

*Таблиця 1.4.*

# Варіанти завдань

|  |  |  |
| --- | --- | --- |
| **№** | **Вираз** | **K** |
| 25. | X=(K-B2-C1)\*2+E4/4 | 2B05025 |

де: А, В, С, D, E, F - знакові цілі числа, довжиною в байтах, згідно з індексом, значення константи К подано у 16-ій системі числення.

**Виконання роботи:**

1. Код для 32-х розрядної платформи:

#include <stdio.h>

extern "C" unsigned int calc(short, char, int);

extern "C" unsigned int K = 0x2B05025;

int main()

{

short B;

char C;

int E;

int X;

printf("Enter numbers:\n");

printf("B = ");

scanf\_s("%d", &B);

printf("C = ");

scanf\_s("%d", &C);

printf("E = ");

scanf\_s("%d", &E);

printf("\n(K-B-C)\*2+E/4 = %d\n", ((K - B - C) \* 2) + (E / 4));

X = calc(B, C, E);

printf("\nResult of procedure calc is: %d\n", X);

return 0;

}

.686

.model flat, c

printf proto c : vararg

EXTERN K : DWORD

.data

msg db 'Output from asm module is: %d', 0

.code

calc PROC

; Збереження базового указівника

push ebp

mov ebp, esp

; Завантаження аргументів зі стеку

movsx eax, word ptr [ebp+8] ; Завантажуємо B2 (short) і розширюємо до 32-біт

movsx ecx, byte ptr [ebp+12] ; Завантажуємо C1 (char) і розширюємо до 32-біт

mov edx, dword ptr [ebp+16] ; Завантажуємо E4 (int)

; Обчислюємо K - B2 - C1

mov ebx, K ; Завантажуємо значення K в регістр ebx

sub ebx, eax ; Віднімаємо B2

sub ebx, ecx ; Віднімаємо C1

; Помножуємо результат на 2

shl ebx, 1 ; ebx = (K - B2 - C1) \* 2

; Ділимо E4 на 4 і додаємо до результату

sar edx, 2 ; edx = E4 / 4

add ebx, edx ; Додаємо результат поділу до ebx

; Виводимо результат

push ebx

push offset msg

call printf

add esp,8

; Повертаємо значення результату в eax

mov eax, ebx

; Відновлюємо базовий указівник і повертаємося

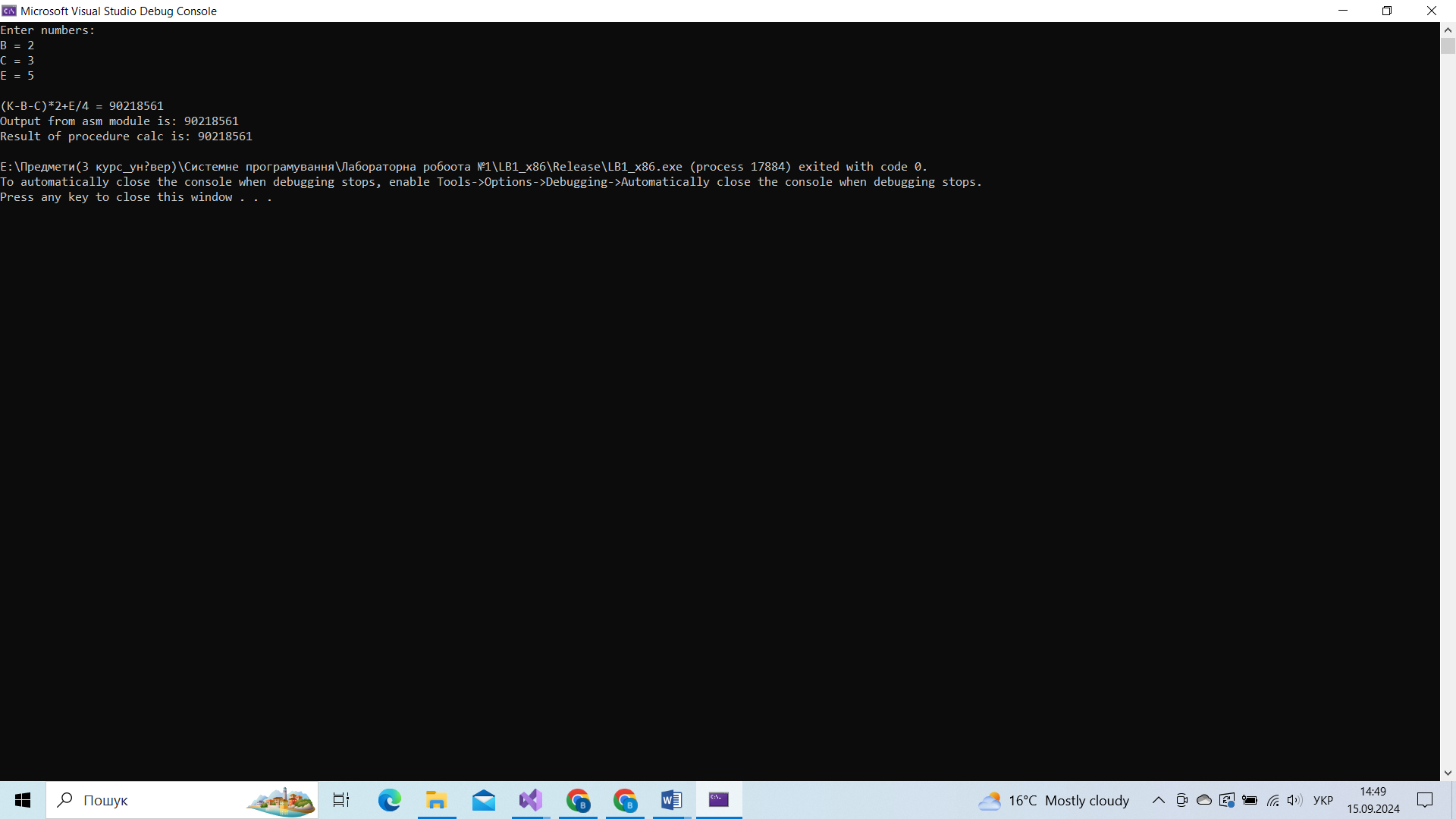
pop ebp

ret

calc ENDP

END

**Результат виконання програми:**



1. Код для 64-х розрядної платформи:

#include <iostream>

// Declare the assembly function

extern "C" int a\_calc(short, char, int);

extern "C" int K = 0x2B05025;

int main() {

short B;

char C;

int E;

int X;

printf("Enter numbers:\n");

printf("B = ");

scanf\_s("%d", &B);

printf("C = ");

scanf\_s("%d", &C);

printf("E = ");

scanf\_s("%d", &E);

printf("\n(K-B-C)\*2+E/4 = %d\n", ((K - B - C) \* 2) + (E / 4));

X = a\_calc(B, C, E);

printf("\nResult of procedure calc is: %d\n", X);

return 0;

}

IFDEF RAX

includelib kernel32.lib

includelib user32.lib

includelib libcmt.lib

includelib legacy\_stdio\_definitions.lib

externdef printf:proc

EXTERN K: DWORD

.data

format db '64 ASM res is %d', 0 ; Format string with %d for integer

.code

a\_calc proc

sub rsp, 28h; NEED TO BE ALLIGNED TO 16

xor rax, rax

xor rbx,rbx

mov ebx, K

sub rbx, rcx

sub rbx, rdx

shl rbx, 1

mov rcx,rbx

shr r8,2

add rcx,r8

mov r12,rcx ;preserver for ret

mov rdx,rcx

lea rcx, offset format

call printf

add rsp, 28h

mov rax,r12

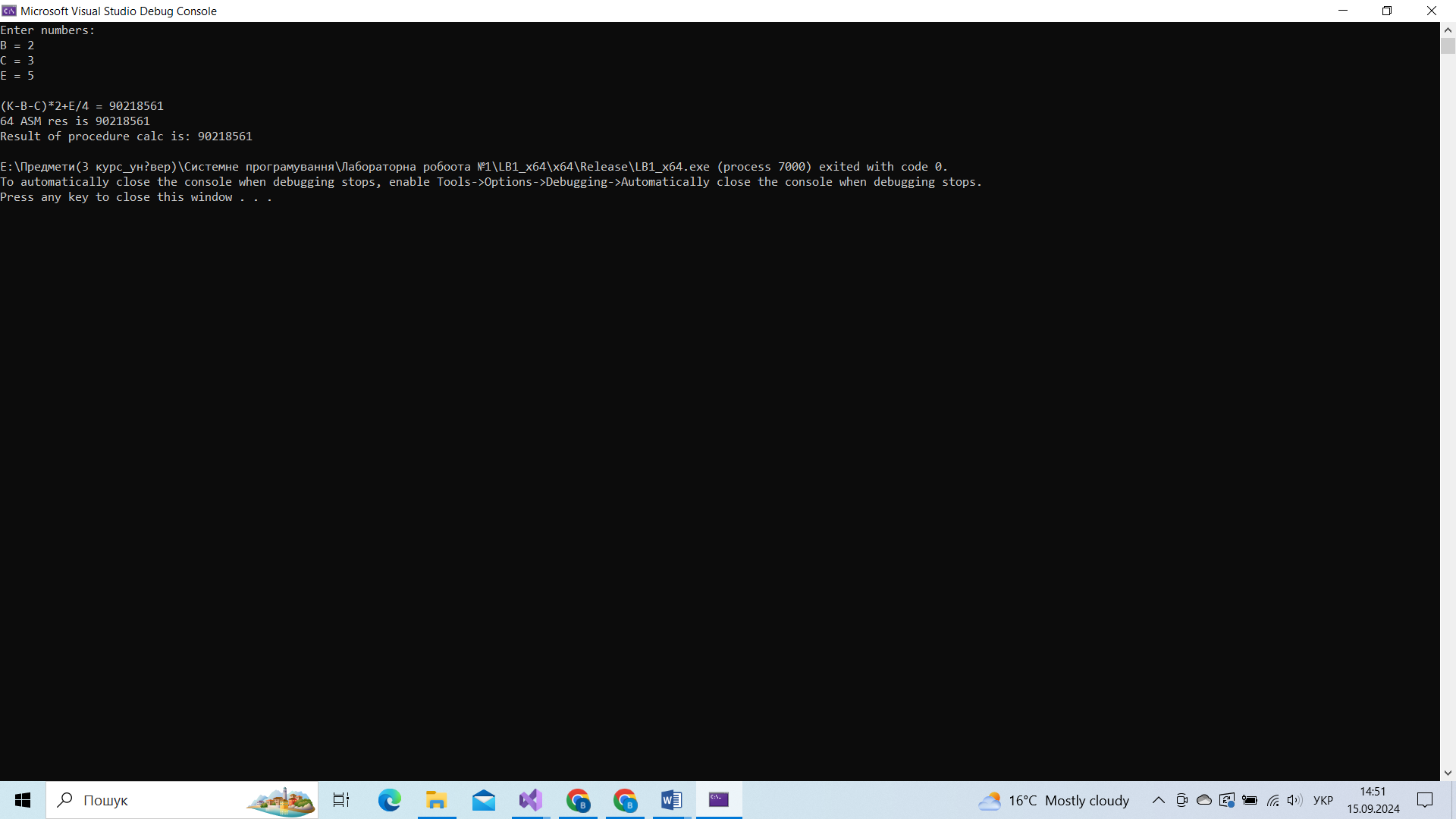
ret

a\_calc endp

ENDIF; RAX

end

**Результат виконання програми:**



**Висновок:** на даній лабораторній роботі я навчився створювати програми, які написані різними мовами програмування, засвоїв взаємодію високорівневої мови C++ та assembler.