**Міністерство освіти і науки України Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»**

**Факультет інформатики та обчислювальної техніки**

**Кафедра обчислювальної техніки**

Лабораторна робота №1

з дисципліни «Системне програмування» на тему

“Налаштування програмних середовищ MASM32 та

MASM64”

Виконав: Перевірив:

Студент ІІ курсу ФІОТ доц. Павлов В. Г.

групи ІМ-22

Сачко Максим Євгенійович

номер у списку групи (варіант): 20

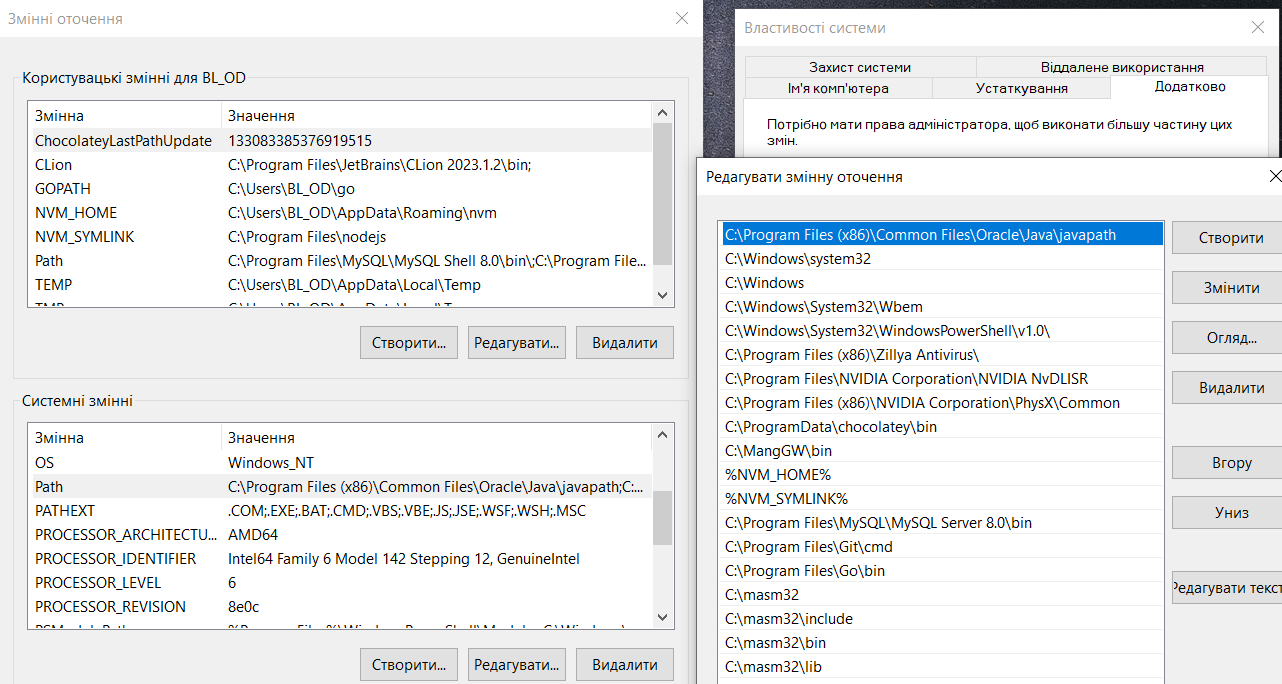
Київ 2024

**Мета роботи:** Практичне створення та налаштування програмних середовищ 32- та 64-бітних версій MASM. Отримання навичок використання цих середовищ для розробки та відладки програм на мові Асемблер.

**1. Розгорнути програмний пакет MASM32 у кореневій теці визначеного логічного диску**

**2. Виконати корегування змінних оточення, а саме додати до змісту змінної Path шляхи до головної теки пакету та деяких внутрішніх тек: bin, lib та include. Зробити скріншот скорегованого змісту змінної оточення Path.**

Скріншот зміненого вмісту змінної **Path для MASM32**:

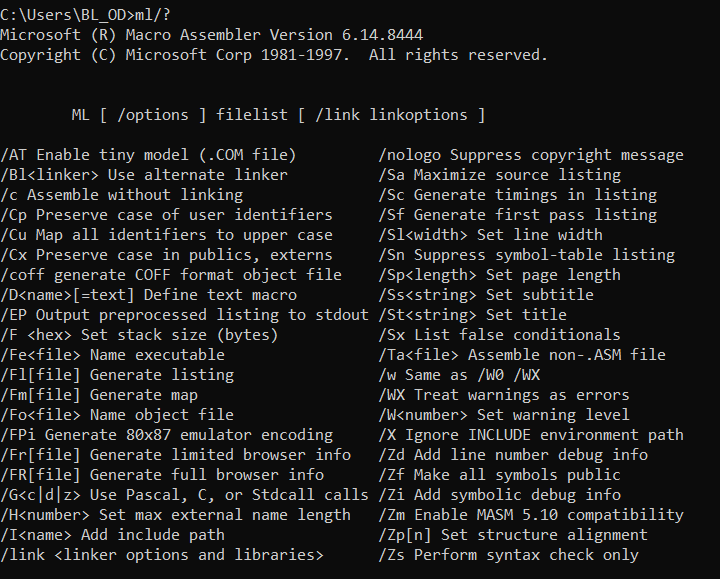


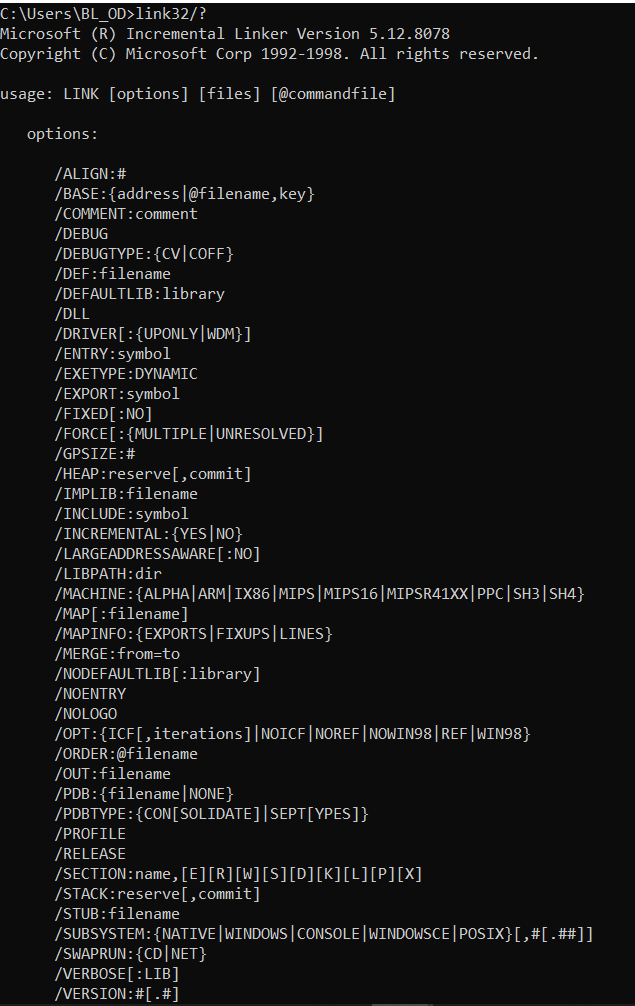
**3. Перезавантажити операційну систему, щоб зміни вступили у дію.**

**4. Запустити вбудований редактор qeditor.exe, а у ньому відкрити консольне вікно.**

**5. Перевірити пошук операційною системою команд MASM32, для чого спочатку ввести команду ML/?, а потім LINK32/?. При правильному корегуванні змінної Path внаслідок кожної з команд буде отримання переліку опцій для кожної з команд. Обидва отримані переліки опцій скопіювати у звіт з лабораторної роботи. При негативному результаті повторити пункти 2-5.**

Переліки опцій команд **ML/?** та **LINK32/?**:

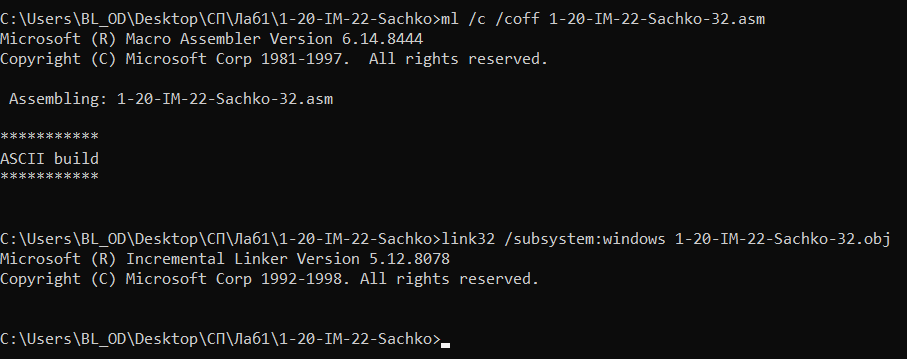




**6. У середовищі MASM32 розробити програму на мові Assembler, за допомогою якої у віконному інтерфейсі повідомлення MessageBox виводяться наступні персональні дані студента – ПІБ, дата народження, номер залікової книжки тощо, причому кожне з нового рядка.**

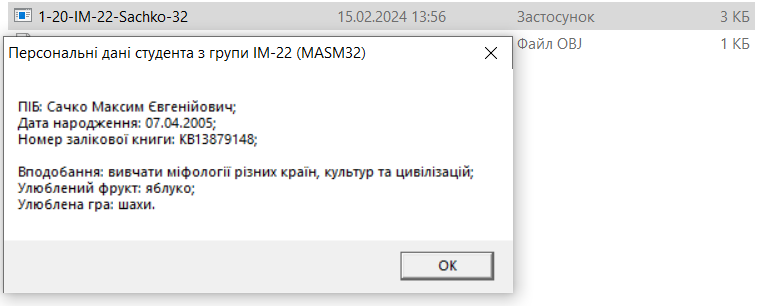
**7. Виконати компіляцію** **розробленого файлу та отримати виконуваний файл у форматі EXE**

Скріншот команд та успішної компіляції розробленого файлу у виконуваний файл у форматі **EXE**:



**8. Скріншот вікна та лістинг програми на мові MASM32 привести у звіті з лабораторної роботи.**

Скріншот вікна для програми розробленої в середовищі **MASM32**:



**Лістинг** **програми розробленої в середовищі MASM32:**

.386

.model flat, stdcall

option casemap:none

include \masm32\include\windows.inc

include \masm32\include\user32.inc

include \masm32\include\kernel32.inc

includelib \masm32\lib\user32.lib

includelib \masm32\lib\kernel32.lib

.data

infoOfStudentSachko db "ПІБ: Сачко Максим Євгенійович;",13,10

db "Дата народження: 07.04.2005;",13,10

db "Номер залікової книги: КВ13879148;",13,10

db 13,10

db "Вподобання: вивчати міфології різних країн, культур та цивілізацій;",13,10

db "Улюблений фрукт: яблуко;",13,10

db "Улюблена гра: шахи.",0

captionOlympus db "Персональні дані студента з групи IM-22 (MASM32)",0

.code

SachkoFistLab:

mov eax, offset infoOfStudentSachko

mov ebx, offset captionOlympus

invoke MessageBox, NULL, eax, ebx, MB\_OK

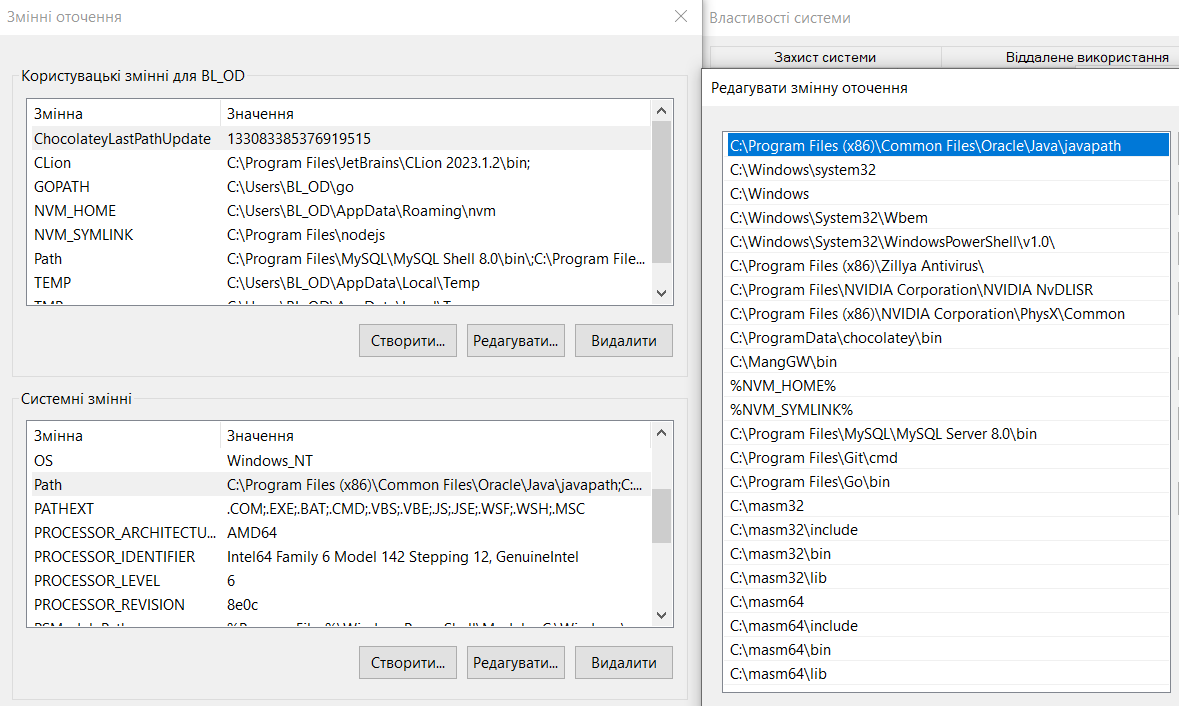
invoke ExitProcess, 0

end SachkoFistLab

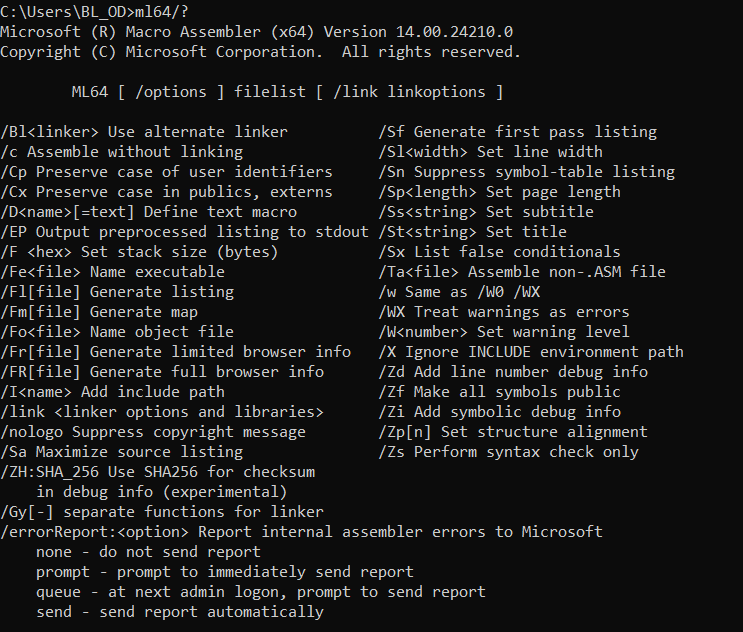
**9. Перевірити наявність в операційній системі 32-бітної та 64-бітної версій Visual C++ (2015-2022). При їх відсутності виконати встановлення зі власних або наданих ресурсів. (**[**https://learn.microsoft.com/ru-RU/cpp/windows/latest-supported-vcredist?view=msvc-160#visual-studio-2015-2017-2019-and-2022**](https://learn.microsoft.com/ru-RU/cpp/windows/latest-supported-vcredist?view=msvc-160#visual-studio-2015-2017-2019-and-2022)**)**

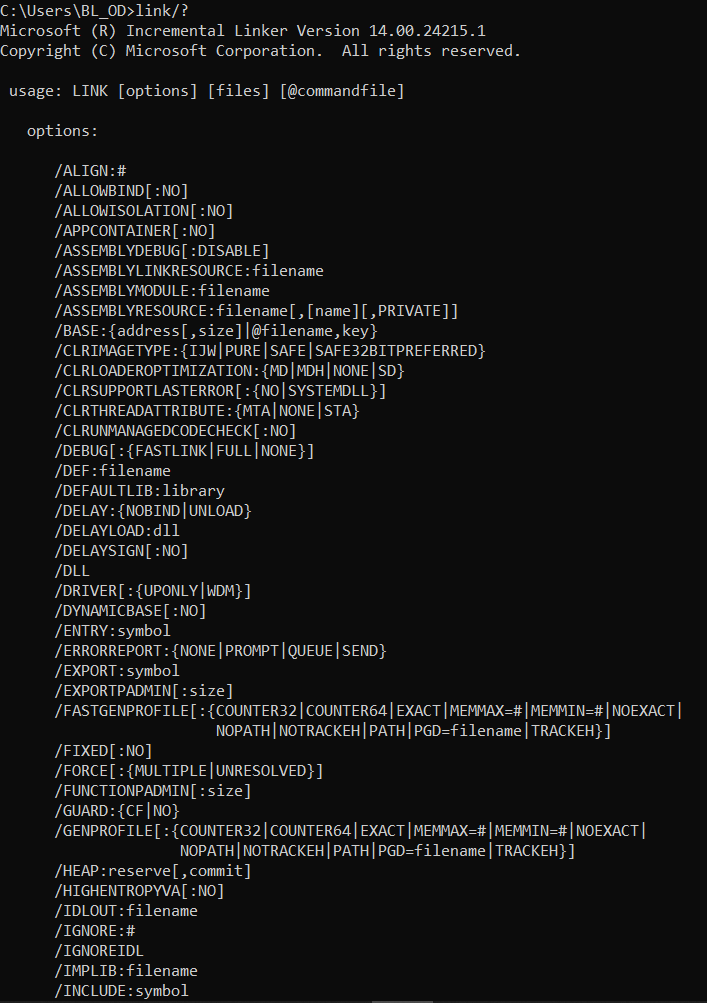
**10. Повторити пункти 1-7 для розгортання та налаштування у інший теці кореневого каталогу визначеного логічного диску програмного пакету MASM64. У пункті 5 перевірити виконання команд ML64/?, а потім LINK/? [1].**

Скріншот зміненого вмісту змінної **Path для MASM64**:



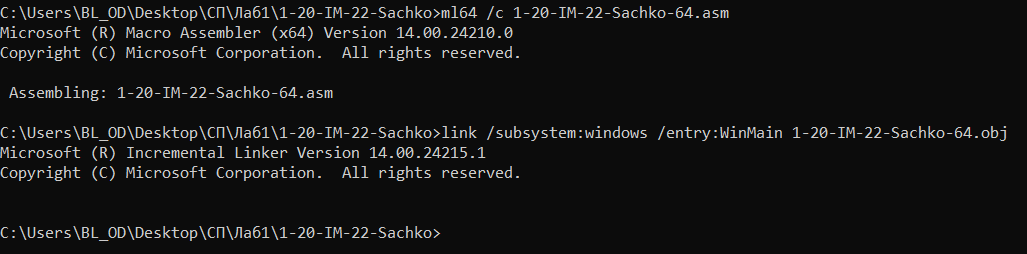
Переліки опцій команд **ML64/?** та **LINK/?**:





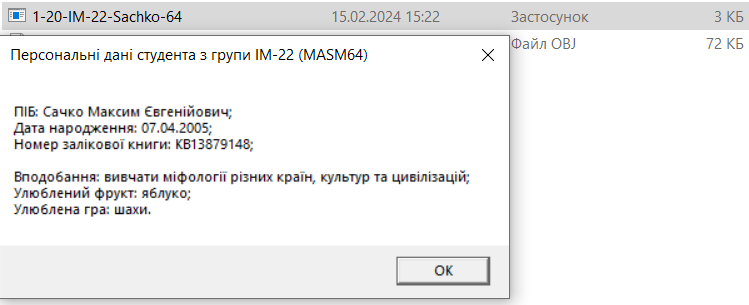
**11. Виконати компіляцію розробленого файлу та отримати виконуваний файл у форматі EXE**

Скріншот команд та успішної компіляції розробленого файлу у виконуваний файл у форматі **EXE**:



**12. Скріншот вікна та лістинг програми на мові MASM64 привести у звіті з лабораторної роботи.**

Скріншот вікна для програми розробленої в середовищі **MASM64**:



**Лістинг програми розробленої в середовищі MASM64:**

OPTION DOTNAME

option casemap:none

include \masm64\include\temphls.inc

include \masm64\include\win64.inc

include \masm64\include\kernel32.inc

includelib \masm64\lib\kernel32.lib

include \masm64\include\user32.inc

includelib \masm64\lib\user32.lib

OPTION PROLOGUE:rbpFramePrologue

OPTION EPILOGUE:none

.data

infoOfStudentSachko db "ПІБ: Сачко Максим Євгенійович;",13,10

db "Дата народження: 07.04.2005;",13,10

db "Номер залікової книги: КВ13879148;",13,10

db 13,10

db "Вподобання: вивчати міфології різних країн, культур та цивілізацій;",13,10

db "Улюблений фрукт: яблуко;",13,10

db "Улюблена гра: шахи.",0

captionOlympus db "Персональні дані студента з групи IM-22 (MASM64)",0

.code

WinMain proc

sub rsp,28h

mov rdx, offset infoOfStudentSachko

mov rbx, offset captionOlympus

invoke MessageBox, NULL, rdx, rbx, MB\_OK

invoke ExitProcess,NULL

WinMain endp

end

**13. Порівняти обидві версії програми та зробити висновки по лабораторній роботі.**

**Порівняння програм:**

**1. Середовище:**

Програма 1 розроблена в середовищі MASM32, а Програма 2 – в середовищі MASM64.

**2. Архітектура:**

Програма 1 використовує 32-бітну архітектуру (х86), а Програма 2 – 64-бітну (х86-64).

**3. Модель адресації:**

Програма 1 використовує ‘.386’ та .’model flat, stdcall’, що вказує на 32-бітну адресацію та використання стекового конвенції виклику функцій. Програма 2 використовує ‘OPTION DOTNAME’ та ‘OPTION PROLOGUE:rbpFramePrologue’, що вказує на 64-бітну адресацію та використання RBP-заснованої стекової конвенції.

**4. Регістри:**

У Програмі 2 використовують регістри 64-бітної архітектури (наприклад, RDX, RBX), що дозволяє оптимізувати обробку даних та забезпечує кращу продуктивність., тоді як Програма 1 використовує 32-бітні регістри (наприклад, EAX, EBX).

**5. Стек:**

Програма 2 використовує ‘sub rsp, 28h’ для резервування місця на стеці, тоді як Програма 1 використовує вказівку стекового фрейму для цього А так MASM32 використовує стек на базі ESP, а MASM64 – стек на базі RSP.

**6. Використання реєстрів та команд:**

Програма 1 використовує команди типу ‘MOV’, ‘INVOKE’ для роботи з реєстрами та виклику функцій. Аналогічно використовує ці команди й Програма 2, але з 64-бітними реєстрами.

Обидві версії програми виконують аналогічні функції та виводять MessageBox і завершують виконання програми через ExitProcess. Проте, основна відмінність полягає в тому, що Програма 1 призначена для 32-бітних систем та використовує 32-бітні регістри та адреси, а Програма 2 – для 64-бітних систем та використовує 64-бітні регістри та адреси.

**Висновок:** Виконуючи цю лабораторну роботу я отримав практичні навички створення та налаштування програмних середовищ 32-бітної та 64-бітної версій MASM. Навчився використовувати ці середовища для розробки та відладки програм на мові Assembler.

Версія програми, написана для MASM32, буде виконуватися на 32-розрядних системах Windows, тоді як версія для MASM64 – на 64-розрядних системах. Тому вибір між MASM32 та MASM64 залежить від того, для якої архітектури ви пишете програму. У сучасних системах, особливо на 64-бітних, рекомендується використовувати MASM64 для оптимізації та сумісності з сучасними системами.