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 II курсу ФІОТ доц. Павлов В. Г. групи ІМ-22 Басараб Станіслав Анатолійович  
номер у списку групи: 2

Київ 2024

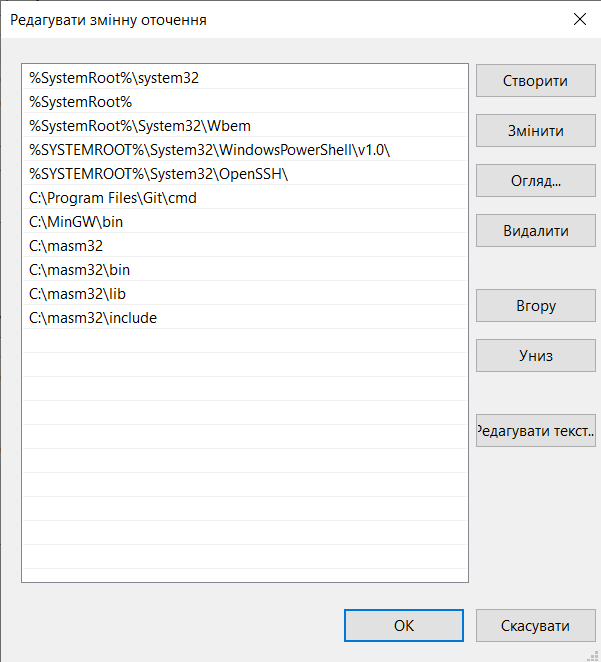
**Мета роботи**: Практичне створення та налаштування програмних середовищ 32- та 64-бітних версій MASM. Отримання навичок використання цих середовищ для розробки та відладки програм на мові Асемблер.

**Порядок виконання роботи**

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.
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.
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.
8. Скріншот вікна та лістинг програми на мові MASM32 привести у звіті з лабораторної роботи.
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-vc-redist?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/?.
11. Виконати компіляцію розробленого файлу та отримати виконуваний файл у форматі EXE.
12. Скріншот вікна та лістинг програми на мові MASM64 привести у звіті з лабораторної роботи.
13. Порівняти обидві версії програми та зробити висновки по лабораторній роботі.

**Виконання роботи**

Скріншот скорегованого змісту змінної оточення **Path** для MASM32:

****

Перелік опцій для команди **ML/?**:

Microsoft (R) Macro Assembler Version 6.14.8444 Copyright (C) Microsoft Corp 1981-1997. All rights reserved.

ML [ /options ] filelist [ /link linkoptions ]

/AT Enable tiny model (.COM file) /nologo Suppress copyright message /Bl<linker> Use alternate linker /Sa Maximize source listing /c Assemble without linking /Sc Generate timings in listing /Cp Preserve case of user identifiers /Sf Generate first pass listing /Cu Map all identifiers to upper case /Sl<width> Set line width /Cx Preserve case in publics, externs /Sn Suppress symbol-table listing /coff generate COFF format object file /Sp<length> Set page length /D<name>[=text] Define text macro /Ss<string> Set subtitle /EP Output preprocessed listing to stdout /St<string> Set title /F <hex> Set stack size (bytes) /Sx List false conditionals /Fe<file> Name executable /Ta<file> Assemble non-.ASM file /Fl[file] Generate listing /w Same as /W0 /WX /Fm[file] Generate map /WX Treat warnings as errors /Fo<file> Name object file /W<number> Set warning level /FPi Generate 80x87 emulator encoding /X Ignore INCLUDE environment path

/Fr[file] Generate limited browser info /Zd Add line number debug info /FR[file] Generate full browser info /Zf Make all symbols public /G<c|d|z> Use Pascal, C, or Stdcall calls /Zi Add symbolic debug info /H<number> Set max external name length /Zm Enable MASM 5.10 compatibility

/I<name> Add include path /Zp[n] Set structure alignment /link <linker options and libraries> /Zs Perform syntax check only

Перелік опцій для команди **LINK32/?**:

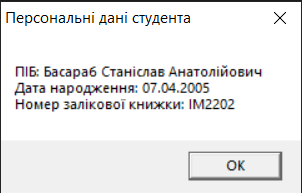
Microsoft (R) Incremental Linker Version 5.12.8078 Copyright (C) Microsoft Corp 1992-1998. All rights reserved.

usage: LINK [options] [files] [@commandfile]

options:

/ALIGN:# /BASE:{address|@filename,key} /COMMENT:comment /DEBUG /DEBUGTYPE:{CV|COFF} /DEF:filename /DEFAULTLIB:library /DLL /DRIVER[:{UPONLY|WDM}] /ENTRY:symbol /EXETYPE:DYNAMIC /EXPORT:symbol /FIXED[:NO] /FORCE[:{MULTIPLE|UNRESOLVED}] /GPSIZE:# /HEAP:reserve[,commit] /IMPLIB:filename /INCLUDE:symbol /INCREMENTAL:{YES|NO} /LARGEADDRESSAWARE[:NO] /LIBPATH:dir /MACHINE:{ALPHA|ARM|IX86|MIPS|MIPS16|MIPSR41XX|PPC|SH3|SH4} /MAP[:filename] /MAPINFO:{EXPORTS|FIXUPS|LINES} /MERGE:from=to /NODEFAULTLIB[:library] /NOENTRY /NOLOGO /OPT:{ICF[,iterations]|NOICF|NOREF|NOWIN98|REF|WIN98} /ORDER:@filename /OUT:filename /PDB:{filename|NONE} /PDBTYPE:{CON[SOLIDATE]|SEPT[YPES]} /PROFILE /RELEASE /SECTION:name,[E][R][W][S][D][K][L][P][X] /STACK:reserve[,commit] /STUB:filename /SUBSYSTEM:{NATIVE|WINDOWS|CONSOLE|WINDOWSCE|POSIX}[,#[.##]] /SWAPRUN:{CD|NET} /VERBOSE[:LIB] /VERSION:#[.#] /VXD /WARN[:warninglevel] /WINDOWSCE:{CONVERT|EMULATION} /WS:AGGRESSIVE

Скріншот вікна програми на мові **MASM32**:



Лістинг програми на мові **MASM32**:

.386

.model flat, stdcall

option casemap:none

include \masm32\include\windows.inc

include \masm32\include\user32.inc

includelib \masm32\lib\user32.lib

include \masm32\include\kernel32.inc

includelib \masm32\lib\kernel32.lib

.data

caption db "Персональні дані студента", 0

basarabData db "ПІБ: Басараб Станіслав Анатолійович", 13, 10,

"Дата народження: 07.04.2005", 13, 10,

"Номер залікової книжки: ІМ2202", 0

.code

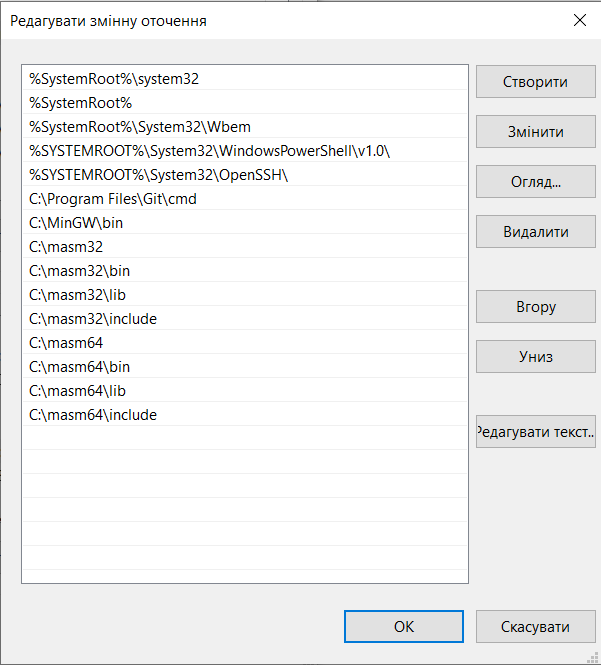
start:

invoke MessageBox, NULL, addr basarabData, addr caption, MB\_OK

invoke ExitProcess, 0

end start

Скріншот скорегованого змісту змінної оточення **Path** для MASM64:



Перелік опцій для команди **ML64/?**:

Microsoft (R) Macro Assembler (x64) Version 14.00.24210.0

Copyright (C) Microsoft Corporation. All rights reserved.

ML64 [ /options ] filelist [ /link linkoptions ]

/Bl<linker> Use alternate linker /Sf Generate first pass listing

/c Assemble without linking /Sl<width> Set line width

/Cp Preserve case of user identifiers /Sn Suppress symbol-table listing

/Cx Preserve case in publics, externs /Sp<length> Set page length

/D<name>[=text] Define text macro /Ss<string> Set subtitle

/EP Output preprocessed listing to stdout /St<string> Set title

/F <hex> Set stack size (bytes) /Sx List false conditionals

/Fe<file> Name executable /Ta<file> Assemble non-.ASM file

/Fl[file] Generate listing /w Same as /W0 /WX

/Fm[file] Generate map /WX Treat warnings as errors

/Fo<file> Name object file /W<number> Set warning level

/Fr[file] Generate limited browser info /X Ignore INCLUDE environment path

/FR[file] Generate full browser info /Zd Add line number debug info

/I<name> Add include path /Zf Make all symbols public

/link <linker options and libraries> /Zi Add symbolic debug info

/nologo Suppress copyright message /Zp[n] Set structure alignment

/Sa Maximize source listing /Zs Perform syntax check only

/ZH:SHA\_256 Use SHA256 for checksum

in debug info (experimental)

/Gy[-] separate functions for linker

/errorReport:<option> Report internal assembler errors to Microsoft

none - do not send report

prompt - prompt to immediately send report

queue - at next admin logon, prompt to send report

send - send report automatically

Перелік опцій для команди **LINK/?**:

Microsoft (R) Incremental Linker Version 14.00.24215.1

Copyright (C) Microsoft Corporation. All rights reserved.

usage: LINK [options] [files] [@commandfile]

options:

/ALIGN:#

/ALLOWBIND[:NO]

/ALLOWISOLATION[:NO]

/APPCONTAINER[:NO]

/ASSEMBLYDEBUG[:DISABLE]

/ASSEMBLYLINKRESOURCE:filename

/ASSEMBLYMODULE:filename

/ASSEMBLYRESOURCE:filename[,[name][,PRIVATE]]

/BASE:{address[,size]|@filename,key}

/CLRIMAGETYPE:{IJW|PURE|SAFE|SAFE32BITPREFERRED}

/CLRLOADEROPTIMIZATION:{MD|MDH|NONE|SD}

/CLRSUPPORTLASTERROR[:{NO|SYSTEMDLL}]

/CLRTHREADATTRIBUTE:{MTA|NONE|STA}

/CLRUNMANAGEDCODECHECK[:NO]

/DEBUG[:{FASTLINK|FULL|NONE}]

/DEF:filename

/DEFAULTLIB:library

/DELAY:{NOBIND|UNLOAD}

/DELAYLOAD:dll

/DELAYSIGN[:NO]

/DLL

/DRIVER[:{UPONLY|WDM}]

/DYNAMICBASE[:NO]

/ENTRY:symbol

/ERRORREPORT:{NONE|PROMPT|QUEUE|SEND}

/EXPORT:symbol

/EXPORTPADMIN[:size]

/FASTGENPROFILE[:{COUNTER32|COUNTER64|EXACT|MEMMAX=#|MEMMIN=#|NOEXACT|

NOPATH|NOTRACKEH|PATH|PGD=filename|TRACKEH}]

/FIXED[:NO]

/FORCE[:{MULTIPLE|UNRESOLVED}]

/FUNCTIONPADMIN[:size]

/GUARD:{CF|NO}

/GENPROFILE[:{COUNTER32|COUNTER64|EXACT|MEMMAX=#|MEMMIN=#|NOEXACT|

NOPATH|NOTRACKEH|PATH|PGD=filename|TRACKEH}]

/HEAP:reserve[,commit]

/HIGHENTROPYVA[:NO]

/IDLOUT:filename

/IGNORE:#

/IGNOREIDL

/IMPLIB:filename

/INCLUDE:symbol

/INCREMENTAL[:NO]

/INTEGRITYCHECK

/KERNEL

/KEYCONTAINER:name

/KEYFILE:filename

/LARGEADDRESSAWARE[:NO]

/LIBPATH:dir

/LTCG[:{INCREMENTAL|NOSTATUS|OFF|STATUS|}]

/MACHINE:{ARM|ARM64|EBC|X64|X86}

/MANIFEST[:{EMBED[,ID=#]|NO}]

/MANIFESTDEPENDENCY:manifest dependency

/MANIFESTFILE:filename

/MANIFESTINPUT:filename

/MANIFESTUAC[:{NO|UAC fragment}]

/MAP[:filename]

/MAPINFO:{EXPORTS}

/MERGE:from=to

/MIDL:@commandfile

/NOASSEMBLY

/NODEFAULTLIB[:library]

/NOENTRY

/NOIMPLIB

/NOLOGO

/NXCOMPAT[:NO]

/OPT:{ICF[=iterations]|LBR|NOICF|NOLBR|NOREF|REF}

/ORDER:@filename

/OUT:filename

/PDB:filename

/PDBSTRIPPED:filename

/PROFILE

/RELEASE

/SAFESEH[:NO]

/SECTION:name,[[!]{DEKPRSW}][,ALIGN=#]

/STACK:reserve[,commit]

/STUB:filename

/SUBSYSTEM:{BOOT\_APPLICATION|CONSOLE|EFI\_APPLICATION|

EFI\_BOOT\_SERVICE\_DRIVER|EFI\_ROM|EFI\_RUNTIME\_DRIVER|

NATIVE|POSIX|WINDOWS|WINDOWSCE}[,#[.##]]

/SWAPRUN:{CD|NET}

/TLBID:#

/TLBOUT:filename

/TIME

/TSAWARE[:NO]

/USEPROFILE[:{AGGRESSIVE|PGD=filename}]

/VERBOSE[:{CLR|ICF|INCR|LIB|REF|SAFESEH|UNUSEDLIBS}]

/VERSION:#[.#]

/WINMD[:{NO|ONLY}]

/WINMDDELAYSIGN[:NO]

/WINMDFILE:filename

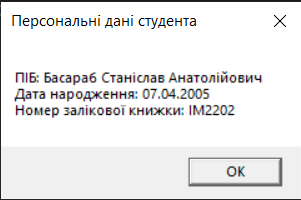
/WINMDKEYCONTAINER:name

/WINMDKEYFILE:filename

/WHOLEARCHIVE[:library]

/WX[:NO]

Скріншот вікна програми на мові **MASM64**:



Лістинг програми на мові **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

MsgCaption db 'Персональні дані студента', 0

MsgBoxText db 'ПІБ: Басараб Станіслав Анатолійович', 13, 10,

'Дата народження: 07.04.2005', 13, 10,

'Номер залікової книжки: ІМ2202', 0

.code

WinMain proc

sub rsp,28h

invoke MessageBox, NULL, &MsgBoxText, &MsgCaption, MB\_OK

invoke ExitProcess,NULL

WinMain endp

end

**Порівняння версій програм MASM32 та MASM64**

1. *Виведення MessageBox*:

Обидві версії програм призначені для виведення вікна MessageBox, що містить інформацію про студента.

1. *Модель адресації*:

MASM32 призначений для 32-бітних систем, тоді як MASM64 - для 64-бітних систем.

1. *Використання реєстрів*:

У MASM32 використовуються 32-бітні реєстри, тоді як у MASM64 використовуються 64-бітні реєстри.

1. *Директиви опцій*:

MASM64 використовує додаткові директиви опцій OPTION PROLOGUE та OPTION EPILOGUE, що вказують на використання rbp-фрейму та відсутність автоматичного епілогу.

1. *Використання адресної індексації*:

У MASM32 використовується addr для отримання адреси змінних, тоді як у MASM64 використовується &.

1. *Виклики функцій*:

В обидвох програмах функції MessageBox та ExitProcess викликаються через invoke. Це допомагає спростити виклик функцій та правильно передавати параметри у відповідному порядку.

**Висновок**

Під час виконання даної лабораторної роботи я здобув навички та знання зі створення та налаштування програмних середовищ 32- та 64-бітних версій MASM, вивчивши відмінності та особливості мов MASM32 та MASM64. Ознайомившись з архітектурою, системними викликами, реєстрами та стеком, я отримав важливі інструменти для розробки низькорівневих програм. Використання системних функцій через invoke та прямі виклики, форматування рядків та виведення інформації у вікно MessageBox допомогли мені засвоїти практичні нюанси роботи з мовами асемблера. Завдяки розробці простого віконного додатку, я отримав певний досвід роботи з Windows API на рівні асемблера. Загалом, вибір між MASM32 та MASM64 залежить від архітектури системи, на якій планується виконувати програму, і від специфіки завдання, яке потрібно вирішити.