**Міністерство Освіти І НАУКИ України**

**Національний університет "Львівська політехніка"**

Інститут **ІКНІ**

Кафедра **ПЗ**

**ЗВІТ**

До лабораторної роботи № 7

**На тему:** *“Статичні та динамічні бібліотеки. WINDOWS та LINUX”*

**З дисципліни:** *“Операційні системи”*

**Лектор:**

Старший викладач ПЗ

Грицай О.Д.

**Виконав:**

ст. гр. ПЗ-22

Солтисюк Д.А.

**Прийняв:**

Старший викладач ПЗ

Грицай О.Д.

« \_\_\_\_ » \_\_\_\_\_\_\_\_ 2022 р.

∑= \_\_\_\_

Львів – 2022

**Тема роботи:** Статичні та динамічні бібліотеки. WINDOWS та LINUX

**Мета роботи:** Ознайомитися з статичними та динамічними бібліотеками в операційних

системах WINDOWS та LINUX. Навчитися реалізовувати статичні та динамічні

бібліотеки.

**Індивідуальне завдання**

1. Реалізувати лабораторну роботу №5 (згідно варіанту) у вигляді статичної та динамічної бібліотеки в ОС WINDOWS.

2. Запустити створену динамічну бібліотеку з командної стрічки (cmd.exe) за допомогою rundll32.exe.

3. Створити окрему програму і реалізувати статичний зв'язок між програмою та бібліотекою із п. 1.

4. Створити окрему програму і реалізувати динамічний зв'язок між програмою та бібліотекою із п. 1.

5. Експортувати головну функцію бібліотеки під іншим іменем із п. 1.

6. Реалізувати лабораторну роботу №6 у вигляді статичної та динамічної (поділюваної) бібліотеки в ОС LINUX.

7. Створити окрему програму і реалізувати статичний зв'язок між програмою та бібліотекою із п. 2

8. Створити окрему програму і реалізувати динамічний зв'язок між програмою та бібліотекою із п. 2.

9. Порівняти результати виконання програми та роботу бібліотек під ОС Windows та Linux.

10. Результати виконання роботи відобразити у звіті.

**Хід роботи**

**Windows**

**main.cpp**

#include <Windows.h>

#include <iostream>

#include "lib1.h"

*//#define RUNTIME\_LOAD*

#ifndef RUNTIME\_LOAD

#include "dll1.h"

#include "main.h"

#else

typedef int(\_\_cdecl\* DLLSUMUP)(int\*, int);

#endif

int main() {

int len;

std::cout << "Type array size: ";

std::cin >> len;

int\* arr = **new** int[len];

std::srand(**static\_cast**<unsigned int>(std::time(nullptr)));

**for** (int i = 0; i < len; i++) arr[i] = rand() % 10;

int sum = libSumUp(arr, len);

std::cout << "Sum from static lib: " << sum << std::endl;

#ifndef RUNTIME\_LOAD

int dllSum = dllSumUp(arr, len);

std::cout << "Sum from dynamic lib (using load-time linking): " << dllSum

<< std::endl;

#else

HINSTANCE hinstLib;

DLLSUMUP sumFn;

hinstLib = LoadLibrary(TEXT("dll1.dll"));

**if** (!hinstLib) {

std::cout << "Failed to load ddl" << std::endl;

**return** -1;

}

sumFn = (DLLSUMUP)GetProcAddress(hinstLib, "dllSumUp");

**if** (!sumFn) {

std::cout << "Failed to link dll on runtime" << std::endl;

} **else** {

int sum = sumFn(arr, len);

std::cout << "Sum from dynamic lib (using run-time linking): " << sum

<< std::endl;

}

FreeLibrary(hinstLib);

#endif

}

**lib1.cpp**

#include "pch.h"

#include "framework.h"

#include "lib1.h"

int libSumUp(int\* arr, int len) {

int sum = 0;

**for** (int i = 0; i < len; i++) sum += arr[i];

**return** sum;

}

**lib1.h**

#pragma once

int libSumUp(int\* arr, int len);

**dll1.cpp**

#include "pch.h"

#include "dll1.h"

int dllSumUp(int\* arr, int len) {

int sum = 0;

**for** (int i = 0; i < len; i++) sum += arr[i];

**return** sum;

}

**dll1.h**

#pragma once

#ifndef DLLSUM\_EXPORTS

#define DLLSUM\_API \_\_declspec(dllexport)

#else

#define DLLSUM\_API \_\_declspec(dllimport)

#endif

extern "C" DLLSUM\_API int dllSumUp(int\* arr, int len);

pasted-image.tiff

Рис. 1. Виконання програми (динамічний зв'язок з динамічною бібліотекою та статичний зв'язок з динамічною бібліотекою)

**Linux**

**main.cpp**

#include <iostream>

#include "aSum.h"

#define RUNTIME\_LOAD

#ifndef RUNTIME\_LOAD

#include "soSum.h"

#else

#include <dlfcn.h>

int (\*soSumUp)(int\*, int);

#endif

int main() {

int len;

std::cout << "Enter array len: ";

std::cin >> len;

int\* arr = **new** int[len];

srand(**static\_cast**<unsigned int>(time(nullptr)));

**for** (int i = 0; i < len; i++) arr[i] = rand() % 10;

int aSum = sumUp(arr, len);

std::cout << "Sum from static lib: " << aSum << std::endl;

#ifndef RUNTIME\_LOAD

int soSum = soSumUp(arr, len);

std::cout << "Sum from dynamic lib (using load-time linking): " << soSum

<< std::endl;

#else

void\* lib;

lib = dlopen("./libSum.so", RTLD\_LAZY);

**if** (!lib) std::cout << "Failed to link on runtime" << std::endl;

soSumUp = (int (\*)(int\*, int))dlsym(lib, "soSumUp");

int soSum = (\*soSumUp)(arr, len);

std::cout << "Sum from dynamic lib (using run-time linking): " << soSum

<< std::endl;

dlclose(lib);

#endif

**aSum.cpp**

#include "aSum.h"

int sumUp(int\* arr, int len) {

int sum = 0;

**for** (int i = 0; i < len; i++) sum += arr[i];

**return** sum;

}

aSum.h

int sumUp(int\* arr, int len);

soSum.cpp

#include "soSum.h"

int soSumUp(int\* arr, int len) {

int sum = 0;

**for** (int i = 0; i < len; i++) sum += arr[i];

**return** sum;

}

soSum.h

extern "C" int soSumUp(int\* arr, int len);

run.sh

*#!/bin/sh*

export LD\_LIBRARY\_PATH=.

g++ -c main.cpp *# main.o creating object files*

g++ -c aSum.cpp *# aSum.o*

g++ -fPIC -c soSum.cpp *# soSum.o with position independent asm code*

ar rc libSum.a aSum.o *# libSum.a creating static library (archive)*

g++ -shared -o libSum.so soSum.o *# libSum.so creating dynamic library (shared object)*

g++ main.o libSum.a libSum.so -ldl *# a.out*

./a.out

rm \*.a \*.o \*.so \*.out

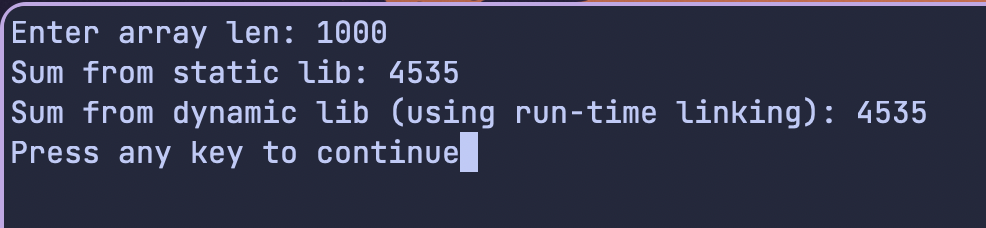


Рис. 2. Виконання програми (динамічний зв'язок з динамічною бібліотекою та статичний зв'язок з динамічною бібліотекою)

**Висновок**

Під час виконання лабораторної роботи я ознайомився з статичними та динамічними бібліотеками в операційних системах WINDOWS та LINUX. Навчився реалізовувати статичні та динамічні бібліотеки