Министерство образования и науки Российской Федерации Федеральное агентство по образованию

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Вятский государственный университет»

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

Изучение работы DLL

Отчет по лабораторной работе №5 дисциплины

«Технологии программирования»

Выполнил студент группы ИВТ-22 /Крючков И. С/ Проверил /Долженкова М. Л./

Киров 2022

1. Задание

Создать на основе лабораторной работы 3 библиотечные модули и разработать новое приложение . Интерфейсную часть приложения оформить в виде библиотеки неявной загрузки, а основной функционал - библиотеки явной загрузки. Проанализировать особенности вызовов и функционирование библиотек.

1. Теория

DLL (англ. dynamic-link library — «библиотека динамической компоновки», «динамически подключаемая библиотека») —динамическая библиотека, позволяющая многократное использование различными программными приложениями.

Формат файлов DLL придерживается тех же соглашений, что и формат исполняемых файлов, сочетая код, таблицы и ресурсы, отличаясь лишь интерпретацией некоторых полей.

Неявное связывание - это динамическое связывание во время загрузки программы, при этом код приложения ссылается на идентификаторы, содержащиеся в DLL, и тем самым заставляет загрузчик неявно подгрузить нужную библиотеку при запуске приложения. Будем считать, что исполняемый EXE модуль импортирует функции и переменные из DLL библиотеки. Тогда DLL экспортирует свои элементы в исполняемый модуль.

Явное связывание - определяет связь между требуемым DLL и клиентским приложением в процессе работы программы. Поток приложения загружает DLL в адресное пространство процесса, получает виртуальный адрес функций и вызывает эти функции по полученному адресу.

1. Результаты работы программы

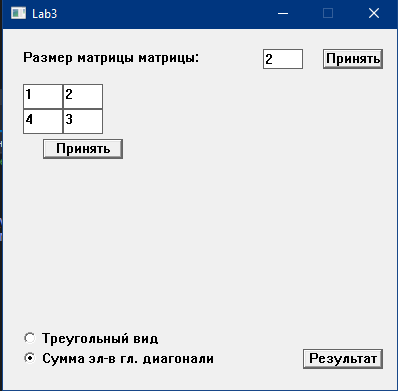


Рисунок 1 – Главное окно программы

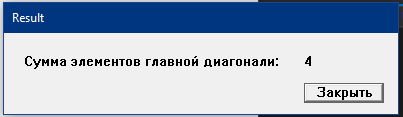


Рисунок 2 – Результат суммы элементов главной диагонали в дочернем окне

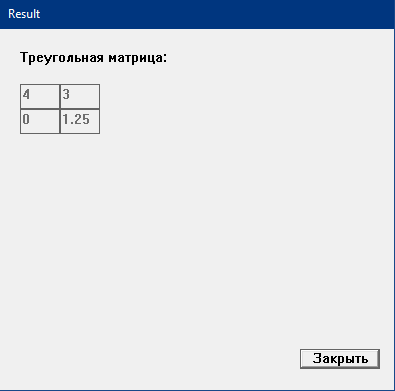


Рисунок 3 – Результат приведения матрицы к треугольному виду в дочернем окне

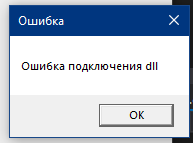


Рисунок 4 – Ошибка подключения dll

1. Листинг кода

**matrixDll.cpp**

#pragma once

#include "pch.h"

#include <utility>

#include <cmath>

#include "matrixDll.h"

using namespace std;

char get\_trangle\_matrix(double\*\*& a, int n) {

for (int i = 0; i < n; i++)

{

// поиск опорного элемента

int z = i;

char f = 0;

for (int h = z + 1; h < n; h++)

{

if (abs(a[z][i]) < abs(a[h][i]) || a[z][i] == 0){

if (a[z][i] == 0)

f = 1;

if (a[h][i] != 0)

swap(a[z], a[h]);

}

}

if (f == 1) {

return 1;

}

// прямой ход

for (int j = i + 1; j < n; j++){

double m = -a[j][i] / a[i][i];

for (int k = i; k < n; k++)

a[j][k] += a[i][k] \* m;

}

}

return 0;

}

**matrixDll.h**

#pragma once

extern "C" \_\_declspec(dllexport) char get\_trangle\_matrix(double\*\*& a, int n);

**matrixDesignDll.cpp**

#pragma once

#include "pch.h"

#include <windows.h>

#include <windowsx.h>

#include "matrixDesignDll.h"

HWND initMain(int w, int h, HINSTANCE hInstance) {

return CreateWindowEx(

NULL,

L"MainWndClass",

L"Lab3",

WS\_DLGFRAME | WS\_SYSMENU | WS\_MINIMIZEBOX,

CW\_USEDEFAULT, // x

0, // y

w, // width

h, // height

NULL, // id родительского окна

NULL,

hInstance,

NULL

);

}

HWND initChild(int w, int h, HINSTANCE hInstance, HWND parentWnd) {

return CreateWindowEx(

NULL,

L"ChildWndClass",

L"Result",

WS\_DLGFRAME,

CW\_USEDEFAULT, // x

0, // y

w, // width

h, // height

parentWnd, // id родительского окна

NULL,

hInstance,

NULL

);

}

**matrixDesignDll.h**

#pragma once

extern "C" \_\_declspec(dllexport) HWND initMain(int w, int h, HINSTANCE hInstance);

extern "C" \_\_declspec(dllexport) HWND initChild(int w, int h, HINSTANCE hInstance, HWND parentWnd);

**Неявное связывание**

#include "../Debug/matrixDesignDll.h"

mainWindow = initMain(widthMainWnd, heightMainWnd, hInstance);

childWindow = initChild(widthMainWnd, heightMainWnd, hInstance, mainWindow);

**Явное связывание**

HINSTANCE dll;

dll = LoadLibrary(L"matrixDll.dll");

get\_trangle\_matrix = (char(\*)(double\*\*&, int)) GetProcAddress(dll, "get\_trangle\_matrix");

if (get\_trangle\_matrix == NULL)

{

dl = true;

MessageBox(hwnd, L"Ошибка подключения dll", L"Ошибка", MB\_OK);

break;

}

else {

EnableWindow(mainWindow, false);

ShowWindow(childWindow, SW\_SHOW);

SetFocus(childWindow);

}

1. Вывод

В ходе выполнения лабораторной работы были созданы библиотечные модули для интерфейсной части приложения и основной логики. Подключение библиотеки с интерфейсной частью выполнено в виде неявной загрузки, библиотека с основным функционалом – в виде явной загрузки, с обработкой ошибки подключения.