|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Министерство науки и высшего образования РФ  федеральное государственное образовательное учреждение  высшего образования  **«Балтийский государственный технический университет «ВОЕНМЕХ» им. Д.Ф. Устинова»**  **(БГТУ «ВОЕНМЕХ» им. Д.Ф. Устинова»)** | | | | |
| БГТУ.СМК-Ф-4.2-К5-01 | | | | |
| Факультет | |  | О |  | Естественнонаучный |
|  | |  | шифр |  | Наименование |
| Кафедра | |  | О7 |  | Информационные системы и программная инженерия |
|  | |  | шифр |  | Наименование |
| Дисциплина | |  | Компьютерный практикум | | |

Практическая работа №3

|  |
| --- |
| Тема работы |
| СОЗДАНИЕ ДИНАМИЧЕСКИХ БИБЛИОТЕК ПРИ ПОМОЩИ  НАБОРА КОМПИЛЯТОРОВ И УТИЛИТ GCC  И ИХ ПРИМЕНЕНИЕ |
| Вариант 7 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Выполнил студент группы | | | |  | О721Б |
| Желтовских Б.Э. | | | | | |
| Фамилия И.О. | | | | | |
| **ПРЕПОДАВАТЕЛЬ** | | | | | |
| Садырова А.К. | |  |  | | |
| Фамилия И.О. Подпись | | | | | |
| Оценка |  | | |  |  |
| «\_\_\_\_\_» |  | | |  | 2023 г. |

САНКТ-ПЕТЕРБУРГ

2023 г.Задания к лабораторной работе

1. Написать программу в соответствии с вариантом.

2. Массив и матрицу заполнять случайными числами от -50 до 50.

3. Функции для работы с массивами и матрицами поместить в две отдельные динамические библиотеки.

4. При запуске программы пользователь должен увидеть меню, в котором можно выбрать, с чем будет проходить работа: с матрицей или с массивом.

5. В зависимости от выбора пользователя загружается одна или другая динамическая библиотека.

6. Библиотеки должны быть скомпилированы с учетом возможного использования в ОС семейств Linux или Windows.

Варианты задания

Подсчитать число элементов матрицы Q (5х7) и массива R (57), кратных трем.

Текст программы:

prog3.cpp

#include <iostream>

#include <iomanip>

#include <cstdlib>

#include <ctime>

#include "p3ar.h"

#include "p3mat.h"

#include <windows.h>

#include <stdio.h>

const int ARRAY\_SIZE = 67;

const int MATRIX\_ROWS = 5;

const int MATRIX\_COLS = 7;

void LoadRun(const char\* const s, int\* arr, int\*\* matrix, int choice)

{

void\* lib;

lib = LoadLibrary(s);

if (!lib)

{

printf("Cannot open library '%s'\n", s);

return;

}

if (choice == 1) {

void (\*sortArr)(int\*, int) = (void (\*)(int\*, int))GetProcAddress((HINSTANCE)lib, "sortArrayDescending");

if (sortArr == NULL)

{

printf("Cannot load function sortArrayDescending from '%s'\n", s);

}

else

{

sortArr(arr, ARRAY\_SIZE); // Call the sortArrayDescending function

}

}

else if (choice == 2) {

void (\*sortMat)(int\*\*, int, int) = (void (\*)(int\*\*, int, int))GetProcAddress((HINSTANCE)lib, "sortMatrixDescending");

if (sortMat == NULL)

{

printf("Cannot load function sortMatrixDescending from '%s'\n", s);

}

else

{

sortMat(matrix, MATRIX\_ROWS, MATRIX\_COLS); // Call the sortMatrixDescending function

}

}

else {

printf("Invalid choice.\n");

}

FreeLibrary((HINSTANCE)lib); // Unload the library

}

void fillArray(int\* arr, int size)

{

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

{

arr[i] = rand() % 101 - 50;

}

}

void fillMatrix(int\*\* matrix, int rows, int cols)

{

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

{

for (int j = 0; j < cols; j++)

{

matrix[i][j] = rand() % 101 - 50;

}

}

}

void printArray(int\* arr, int size)

{

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

{

std::cout << arr[i] << " ";

}

std::cout << std::endl;

}

void printMatrix(int\*\* matrix, int rows, int cols)

{

int max\_width = 0;

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

{

for (int j = 0; j < cols; j++)

{

int element\_width = std::to\_string(matrix[i][j]).length();

if (element\_width > max\_width)

{

max\_width = element\_width;

}

}

}

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

{

for (int j = 0; j < cols; j++)

{

std::cout << std::setw(max\_width) << matrix[i][j] << " ";

}

std::cout << std::endl;

}

}

int main()

{

srand(time(0));

int\* arr = new int[ARRAY\_SIZE];

int\*\* matrix = new int\* [MATRIX\_ROWS];

for (int i = 0; i < MATRIX\_ROWS; i++)

{

matrix[i] = new int[MATRIX\_COLS];

}

fillArray(arr, ARRAY\_SIZE);

fillMatrix(matrix, MATRIX\_ROWS, MATRIX\_COLS);

int choice;

while (true)

{

std::cout << "Select what you want to work with:" << std::endl;

std::cout << "1. Array " << std::endl;

std::cout << "2. Matrix" << std::endl;

std::cout << "3. Exit" << std::endl;

std::cin >> choice;

if (choice < 1 || choice > 3)

{

std::cout << "Invalid choice. Please try again." << std::endl;

}

else

{

switch (choice)

{

case 1:

{

std::cout << "Original array:" << std::endl;

printArray(arr, ARRAY\_SIZE);

LoadRun("ar.dll", arr, matrix, choice);

}

break;

case 2:

{

std::cout << "Original matrix:" << std::endl;

printMatrix(matrix, MATRIX\_ROWS, MATRIX\_COLS);

LoadRun("mat.dll", arr, matrix, choice);

}

break;

case 3:

std::cout << "Exiting the program." << std::endl;

delete[] arr;

for (int i = 0; i < MATRIX\_ROWS; i++)

{

delete[] matrix[i];

}

delete[] matrix;

return 0;

}

}

}

delete[] arr;

for (int i = 0; i < MATRIX\_ROWS; i++)

{

delete[] matrix[i];

}

delete[] matrix;

return 0;

}

prog3ar.cpp

#include "p3ar.h"

#include <ostream>

#include <iostream>

extern "C" \_\_declspec(dllexport) void sortArrayDescending(int\* arr, int size)

{

int count = 0;

for (int i = 0; i < size; i++) {

if (arr[i] % 3 == 0) {

count++;

}

}

std::cout << "Multiples of three : " << count <<std::endl;

}P3ar.h

#ifdef \_\_cplusplus

extern "C" {

#endif

void sortArrayDescending(int\* arr, int size);

#ifdef \_\_cplusplus

}

#endif

p3ar.h

#ifdef \_\_cplusplus

extern "C" {

#endif

void sortArrayDescending(int\* arr, int size);

#ifdef \_\_cplusplus

}

#endif

prog3mat.cpp

#include "p3mat.h"

#include <iostream>

#include <ostream>

void sortMatrixDescending(int\*\* matrix, int rows, int cols)

{

int count = 0;

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

{

for (int j = 0; j < cols; j++)

{

if (matrix[i][j] % 3 == 0)

{

count++;

}

}

}

std::cout << "Multiples of three : " << count << std::endl;

}

p3mat.h

#ifdef \_\_cplusplus

extern "C" {

#endif

void sortMatrixDescending(int\*\* matrix, int rows, int cols);

#ifdef \_\_cplusplus

}

#endif

Результат:

Создаем о

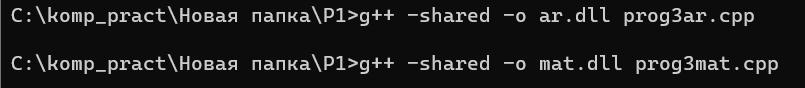
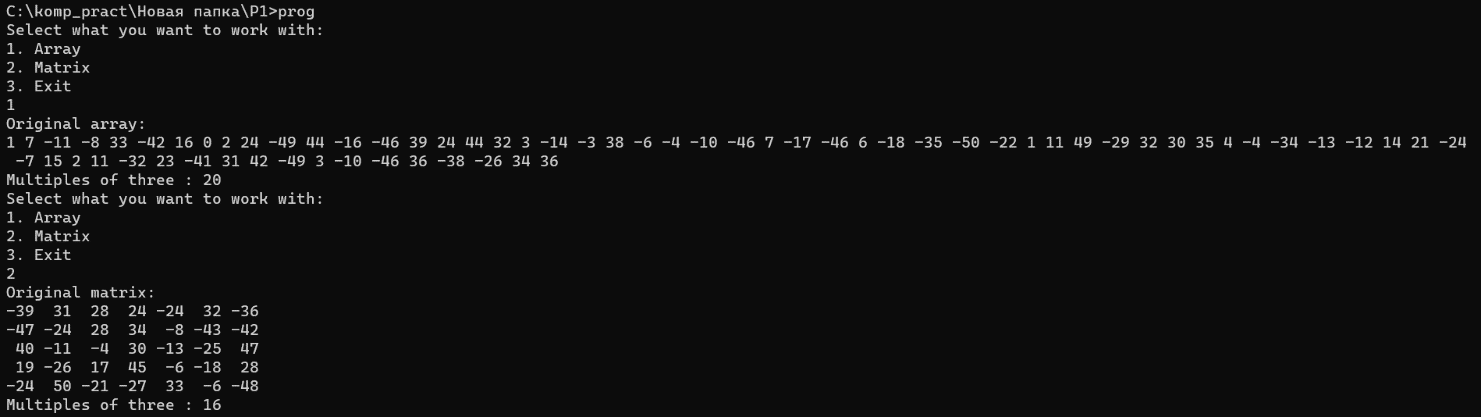
Создаем две динамические библиотеки с помощью команды g++ -shared -o …. .dll.

Рисунок 2 – Создание динамических библиотек

 Компиляция программы с добавлением двух динамических библиотек.

Рисунок 3 - Компиляция программы.

Запуск исполняемого файла prog.

Рисунок 4 – Работа программы

Вывод

В процессе выполнения задания я научился создавать две динамические библиотеки для работы с массивами и матрицами, разрабатывать пользовательские меню для выбора необходимой функциональности, динамически загружать соответствующую библиотеку в зависимости от выбора пользователя, и обеспечивать переносимость библиотек между операционными системами Linux и Windows.