|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Кафедра |  | О7 |  | Информационные системы и программная инженерия |
|  |  | шифр |  | наименование кафедры, по которой выполняется работа |
| Дисциплина |  | Компьютерный практикум | | |
|  |  | наименование дисциплины | | |

|  |  |  |
| --- | --- | --- |
| ПРАКТИЧЕСКАЯ РАБОТА | №3 |  |
|  |  |  |
|  | | |
| СОЗДАНИЕ ДИНАМИЧЕСКИХ БИБЛИОТЕК | | |
| ПРИ ПОМОЩИ НАБОРА КОМПИЛЯТОРОВ И | | |
| УТИЛИТ GCC И ИХ ПРИМЕНЕНИЕ | | |
| Вариант №24 | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ОБУЧАЮЩИЙСЯ** | | | | | | |
| группы | | | |  | | О738Б |
|  |  | Рогачёв В. В. | | | | |
| подпись |  | фамилия и инициалы | | | | |
|  | | | | |
| дата сдачи | | | | |
| **ПРОВЕРИЛ** | | | | | | |
|  | | | | | | | |
| ученая степень, ученое звание, должность | | | | | | | |
|  |  | Гладевич А. А. | | | | | |
| подпись |  | фамилия и инициалы | | | | | |
| Оценка / балльная оценка | | |  | | | |
|  | | | | |
| дата проверки | | | | |

**СОДЕРЖАНИЕ**

[1 Практическая работа №3 3](#_Toc177940507)

[1.1 Цель и постановка задачи 3](#_Toc177940508)

[1.2 Формулировка вариативной части работы 3](#_Toc177940509)

[1.3 Выполнение практической работы 3](#_Toc177940510)

[1.3.1 Текст программы 3](#_Toc177940511)

[1.3.2 Создание динамических библиотек и демонстрация работы 8](#_Toc177940512)

[1.3.3 Демонстрация работы программы 9](#_Toc177940513)

# **1 Практическая работа №3**

## **1.1 Цель и постановка задачи**

Цель работы: создать динамическую библиотеку при помощи набора компиляторов и утилит GCC.

Практическая работа включает в себя шесть заданий:

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 учётом возможного использования в ОС семейств Linux или Windows.

## **1.2 Формулировка вариативной части работы**

При выполнении учебной практики был выбран вариант №6.

Подсчитать число элементов матрицы Q (9х11) и массива R (38), остаток от деления которых на пять равен единице.

## **1.3 Выполнение практической работы**

### 1.3.1 Текст программы

1.3.1.1 Содержание файла array\_lib.cpp

#include <cstdlib>

#include <ctime>

extern "C" int count\_mod5\_equals\_1\_array(int\* array, int size) {

int count = 0;

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

if (array[i] % 5 == 1) {

++count;

}

}

return count;

}

extern "C" void fill\_array(int\* array, int size) {

srand(static\_cast<unsigned>(time(0)));

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

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

}

}

1.3.1.2 Содержание файла matrix\_lib.cpp

#include <cstdlib>

#include <ctime>

extern "C" int count\_mod5\_equals\_1\_matrix(int matrix[9][11]) {

int count = 0;

for (int i = 0; i < 9; ++i) {

for (int j = 0; j < 11; ++j) {

if (matrix[i][j] % 5 == 1) {

++count;

}

}

}

return count;

}

extern "C" void fill\_matrix(int matrix[9][11]) {

srand(static\_cast<unsigned>(time(0)));

for (int i = 0; i < 9; ++i) {

for (int j = 0; j < 11; ++j) {

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

}

}

}

1.3.1.3 Содержание файла main.cpp

#include <iostream>

#ifdef \_WIN32

#include <windows.h>

#else

#include <dlfcn.h>

#endif

int main() {

system("chcp 65001>nul");

int choice;

std::cout << "Выберите:\n1. Работа с массивом (38 элементов)\n2. Работа с матрицей (9x11)\n";

std::cin >> choice;

#ifdef \_WIN32

HINSTANCE hLib;

if (choice == 1) {

hLib = LoadLibrary("array\_lib.dll");

} else {

hLib = LoadLibrary("matrix\_lib.dll");

}

#else

void\* hLib;

if (choice == 1) {

hLib = dlopen("./array\_lib.so", RTLD\_LAZY);

} else {

hLib = dlopen("./matrix\_lib.so", RTLD\_LAZY);

}

#endif

if (!hLib) {

std::cerr << "Ошибка загрузки библиотеки!" << std::endl;

return 1;

}

if (choice == 1) {

int array[38];

#ifdef \_WIN32

void (\*fill\_array)(int\*, int) = (void (\*)(int\*, int)) GetProcAddress(hLib, "fill\_array");

int (\*count\_mod5\_equals\_1\_array)(int\*, int) = (int (\*)(int\*, int)) GetProcAddress(hLib, "count\_mod5\_equals\_1\_array");

#else

void (\*fill\_array)(int\*, int) = (void (\*)(int\*, int)) dlsym(hLib, "fill\_array");

int (\*count\_mod5\_equals\_1\_array)(int\*, int) = (int (\*)(int\*, int)) dlsym(hLib, "count\_mod5\_equals\_1\_array");

#endif

fill\_array(array, 38);

int count = count\_mod5\_equals\_1\_array(array, 38);

std::cout << "Число элементов массива, остаток от деления на 5 которых равен 1: " << count << std::endl;

} else if (choice == 2) {

int matrix[9][11];

#ifdef \_WIN32

void (\*fill\_matrix)(int[9][11]) = (void (\*)(int[9][11])) GetProcAddress(hLib, "fill\_matrix");

int (\*count\_mod5\_equals\_1\_matrix)(int[9][11]) = (int (\*)(int[9][11])) GetProcAddress(hLib, "count\_mod5\_equals\_1\_matrix");

#else

void (\*fill\_matrix)(int[9][11]) = (void (\*)(int[9][11])) dlsym(hLib, "fill\_matrix");

int (\*count\_mod5\_equals\_1\_matrix)(int[9][11]) = (int (\*)(int[9][11])) dlsym(hLib, "count\_mod5\_equals\_1\_matrix");

#endif

fill\_matrix(matrix);

int count = count\_mod5\_equals\_1\_matrix(matrix);

std::cout << "Число элементов матрицы, остаток от деления на 5 которых равен 1: " << count << std::endl;

}

#ifdef \_WIN32

FreeLibrary(hLib);

#else

dlclose(hLib);

#endif

std::cout << "Нажимите Enter для завершения работы программы...";

std::cin.ignore();

std::cin.get();

return 0;

}

### 1.3.2 Создание динамических библиотек и демонстрация работы

Для выполнения задания было создано две библиотеки: одна для работы с массивом, другая — для работы с матрицей. Эти библиотеки содержат функции для заполнения данных случайными числами и подсчета элементов, которые при делении на 5 дают остаток 1. Далее наступил этап компиляции. Для Linux и Windows используется разная команда для компиляции динамических библиотек.

Для системы Linux динамическая библиотека компилируется с помощью команды >g++ -fPIC -shared -o «файл».so «файл».cpp, где g++ — компилятор C++, -fPIC — этот флаг указывает компилятору создавать позиционно-независимый код (Position Independent Code), который необходим для создания динамических библиотек, -shared — указывает компилятору создать динамическую библиотеку, -o «файл».so — имя выходного файла динамической библиотеки, а «файл».cpp — исходный файл с кодом библиотеки.

Для системы Windows динамическая библиотека компилируется с помощью команды >g++ -shared -o «файл».dll «файл».cpp, где g++ — компилятор C++, -shared — флаг для создания динамической библиотеки,  
-o «файл».dll — имя выходного файла динамической библиотеки,  
«файл».cpp — исходный файл с кодом библиотеки.

Далее требуется скомпилировать основной файл. Для   
компиляции основной программы в Linux используется команда   
>g++ main.cpp -o main -ldl, где g++ — компилятор C++, main.cpp — исходный файл основной программы, -o main — имя исполняемого файла (в данном случае это main), -ldl — флаг, который указывает компилятору слинковать программу с библиотекой dl, которая необходима для динамической загрузки библиотек в Linux.

Для компиляции основной программы Windows используется команда >g++ main.cpp -o main.exe, где g++ — компилятор C++,   
main.cpp — исходный файл основной программы, -o main.exe — имя выходного файла (в данном случае это main.exe), исполняемый файл для Windows.

После настал этап запуска программы. Для того чтобы запустить программу в системе Linux нужно воспользоваться командой >./main. Для того чтобы запустить программу в Windows через командную строку с помощью команды >main.exe.

### 1.3.3 Демонстрация работы программы

При запуске программы пользователю предлагается выбор между работой с массивом или с матрицей, что представлено на рисунке 1.

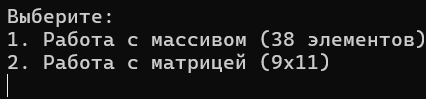


Рисунок 1 – Вывод меню

После выбора пользователем варианта программа загружает соответствующую библиотеку и вызывает функцию для заполнения данных и подсчёта элементов. Работа программы представлена на рисунках 2-3.

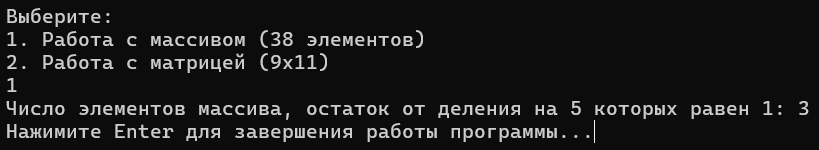


Рисунок 2 – Работа программы при выборе 1

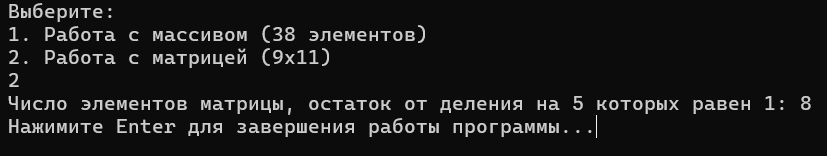


Рисунок 3 – Работа программы при выборе 2

Поле выполнения функций и подсчёта элементов программа завершает свою работу, что представлено на рисунке 4.



Рисунок 4 – Завершение работы программы