Московский авиационный институт

(Национальный Исследовательский Университет)

Институт №8 “Компьютерные науки и прикладная математика”

Кафедра №806 “Вычислительная математика и программирование”

**Лабораторная работа №4 по курсу**

**«Операционные системы»**

Студент: Чапалда Мария Олеговна

Группа: М8О-201Б-22

Преподаватель: Миронов Евгений Сергеевич

Вариант: 28

Оценка: \_\_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_\_

Москва, 2023

Содержание

1. Постановка задачи

2. Вариант задания

3. Общие сведения о программе

5. Исходный код

6. Демонстрация работы программы

7. Вывод

**Цель работы**

Целью является приобретение практических навыков в:

Создание динамических библиотек

Создание программ, которые используют функции динамических библиотек

**Задание**

Требуется создать динамические библиотеки, которые реализуют определенный функционал.

Далее использовать данные библиотеки 2-мя способами:

Во время компиляции (на этапе «линковки»/linking)

Во время исполнения программы. Библиотеки загружаются в память с помощью

интерфейса ОС для работы с динамическими библиотеками

В конечном итоге, в лабораторной работе необходимо получить следующие части:

Динамические библиотеки, реализующие контракты, которые заданы вариантом;

Тестовая программа (программа №1), которая используют одну из библиотек, используя

знания полученные на этапе компиляции;

Тестовая программа (программа №2), которая загружает библиотеки, используя только их

местоположение и контракты.

Провести анализ двух типов использования библиотек.

Пользовательский ввод для обоих программ должен быть организован следующим образом:

Если пользователь вводит команду «0», то программа переключает одну реализацию контрактов на другую (необходимо только для программы №2). Можно реализовать лабораторную работу без данной функции, но максимальная оценка в этом случае будет «хорошо»;

«1 arg1 arg2 … argN», где после «1» идут аргументы для первой функции, предусмотренной контрактами. После ввода команды происходит вызов первой функции, и на экране появляется результат её выполнения;

«2 arg1 arg2 … argM», где после «2» идут аргументы для второй функции, предусмотренной контрактами. После ввода команды происходит вызов второй функции, и на экране появляется результат её выполнения.

**Общие сведения о программе**

Основная программа для динамической загрузки библиотек:

1. Загрузка библиотек: Программа начинается с попытки загрузить библиотеку, имя которой передается через командную строку. При успешной загрузке программа пытается получить адреса двух функций: PrimeCount для подсчета простых чисел и GCF для вычисления наибольшего общего делителя (НОД). Интерактивный ввод: Пользователь может вводить команды для переключения между библиотеками, вызывая разные реализации функций PrimeCount и GCF. Вызов функций: В зависимости от введенной команды, осуществляется вызов соответствующей функции с параметрами, введенными пользователем.

2. Пример реализации функций для статической компиляции: Представлен код для статически компилируемой программы, которая напрямую вызывает функции library1 и library2 без динамической загрузки. Это демонстрация альтернативного подхода, когда все функции доступны на этапе компиляции

**Исходный код:**

Library1.cpp

#include <cmath>

extern "C" float Pi(int K) {

    float pi = 0.0;

    for (int i = 0; i < K; ++i) {

        pi += pow(-1, i) / (2 \* i + 1);

    }

    return pi \* 4;

}

extern "C" float Square(float A, float B) {

    return A \* B; *// Площадь прямоугольника*

}

Library2.cpp

#include <cmath>

extern "C" float Pi(int K) {

    float pi = 1.0;

    for (int i = 1; i < K; ++i) {

        pi \*= (4.0 \* i \* i) / (4.0 \* i \* i - 1);

    }

    return pi \* 2;

}

extern "C" float Square(float A, float B) {

    return 0.5 \* A \* B; *// Площадь прямоугольного треугольника*

}

Static.cpp

#include <iostream>

extern "C" float Pi(int K);

extern "C" float Square(float A, float B);

int main() {

    std::cout << "Программа запущена. Ожидание ввода команд..." << std::endl;

    int command;

    while (std::cin >> command) {

        if (command == 1) {

            int K;

            std::cout << "Введите K для вычисления Pi: ";

            std::cin >> K;

            std::cout << "Вызов функции Pi с K = " << K << std::endl;

            std::cout << "Pi: " << Pi(K) << std::endl;

        } else if (command == 2) {

            float A, B;

            std::cout << "Введите стороны A и B для вычисления площади: ";

            std::cin >> A >> B;

            std::cout << "Вызов функции Square с A = " << A << " и B = " << B << std::endl;

            std::cout << "Площадь: " << Square(A, B) << std::endl;

        } else {

            std::cout << "Неизвестная команда. Пожалуйста, введите 1 (для Pi) или 2 (для Square)." << std::endl;

        }

    } g++

    std::cout << "Программа завершила работу." << std::endl;

    return 0;

}

Dynamic.cpp

#include <iostream>

#include <dlfcn.h>

typedef float (\*pi\_func)(int);

typedef float (\*square\_func)(float, float);

int main() {

    void\* lib\_handle = nullptr;

    pi\_func Pi = nullptr;

    square\_func Square = nullptr;

    auto load\_library = [&](const char\* library) {

        std::cout << "Загрузка библиотеки: " << library << std::endl;

        if (lib\_handle) {

            dlclose(lib\_handle); *// Закрыть старую библиотеку*

            std::cout << "Предыдущая библиотека закрыта." << std::endl;

        }

        lib\_handle = dlopen(library, RTLD\_LAZY);

        if (!lib\_handle) {

            std::cerr << "Ошибка загрузки библиотеки: " << dlerror() << std::endl;

            return false;

        }

        Pi = (pi\_func)dlsym(lib\_handle, "Pi");

        Square = (square\_func)dlsym(lib\_handle, "Square");

        if (!Pi || !Square) {

            std::cerr << "Ошибка загрузки функций: " << dlerror() << std::endl;

            return false;

        }

        std::cout << "Функции успешно загружены." << std::endl;

        return true;

    };

*// Начальная загрузка библиотеки*

    if (!load\_library("./library1.so")) return 1;

    int command;

    std::cout << "Введите команду (0 - сменить библиотеку, 1 - Pi, 2 - Square):" << std::endl;

    while (std::cin >> command) {

        if (command == 0) {

            std::cout << "Переключение библиотек..." << std::endl;

            if (!load\_library("./library2.so")) return 1;

        } else if (command == 1) {

            int K;

            std::cout << "Введите K для вычисления Pi: ";

            std::cin >> K;

            std::cout << "Pi: " << Pi(K) << std::endl;

        } else if (command == 2) {

            float A, B;

            std::cout << "Введите стороны A и B: ";

            std::cin >> A >> B;

            std::cout << "Square: " << Square(A, B) << std::endl;

        }

    }

    if (lib\_handle) {

        dlclose(lib\_handle);

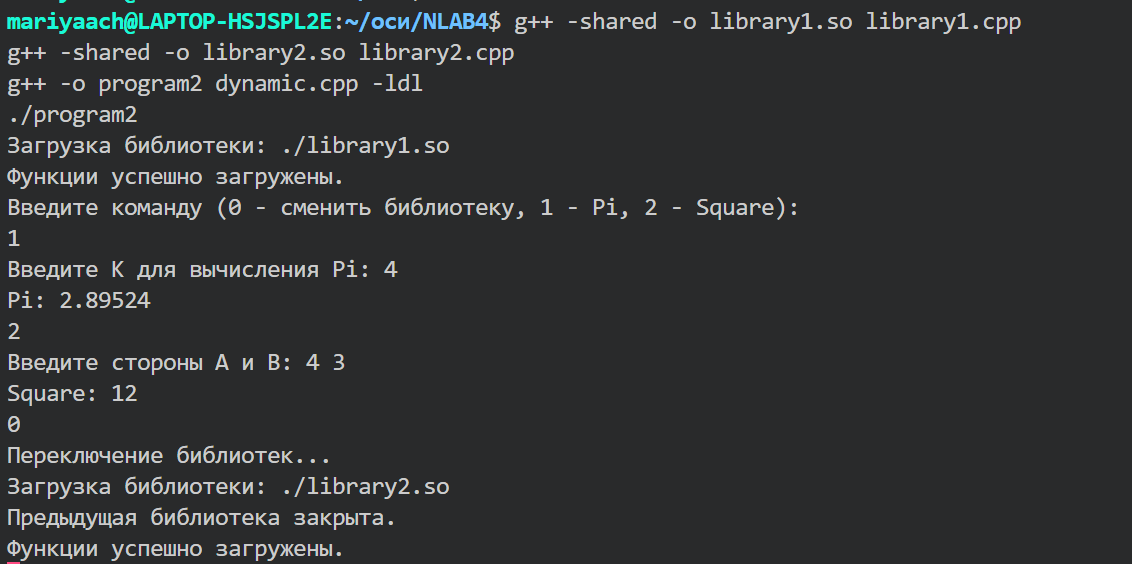
        std::cout << "Библиотека закрыта." << std::endl;

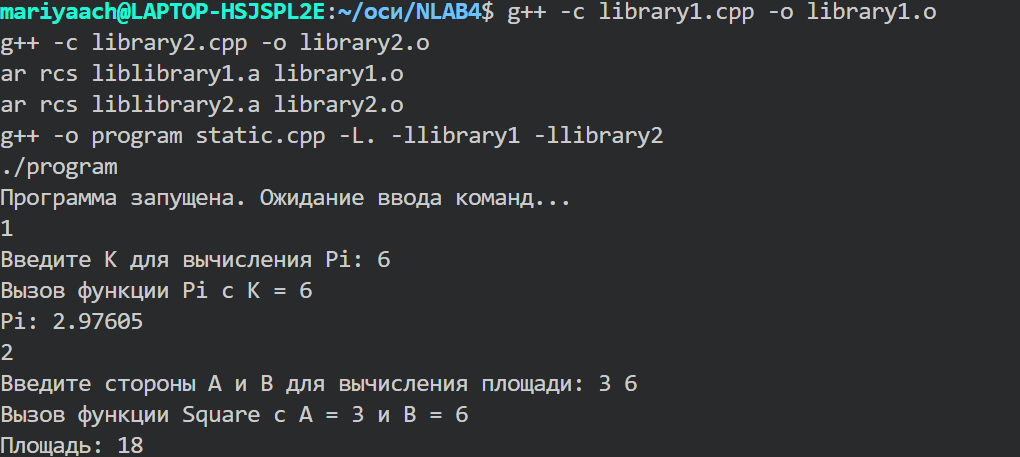
    }

    return 0;

}

**Демонстрация работы программы:**





**Выводы**

В ходе выполнения данной лабораторной работы я изучила отличия между динамической и статической линковкой библиотек, применила полученные знания на практике и обнаружила много нового и интересного. Это будет полезным для более глубокого изучения языка C++ в будущем.