МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

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: Степанов Н.Е.
Группа: М8О-208Б-23
Вариант: 31
Преподаватель: Миронов Е. С.

Дата: _____

Оценка: _____

Подпись: _____

Москва, 2024

Содержание

Репозиторий

https://github.com/n0w3e/os_labs/tree/lab4

Постановка задачи

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 заданный вариантом функционал. Далее использовать данные библиотеки 2-мя способами:

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

2. 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 информацию полученные на этапе компиляции;

- Тестовая программа (программа №2), которая загружает библиотеки, используя только их относительные пути и контракты.

Провести анализ двух типов использования библиотек.

Контракты и реализации функций(**мой вариант**):

6	Рассчет значения числа e(основание натурального логарифма)	Float E(int x)	(1 + 1/x) ^ x	Сумма ряда по n от 0 до x, где элементы ряда равны: (1/(n!))
7	Подсчет площади плоской геометрической фигуры подвум сторонам	Float Square(float A, float B)	Фигура прямоугольник	Фигура прямоугольный треугольник

Общие сведения о программе

Программа представляет собой набор файлов, которые реализуют математические функции и позволяют пользователю выбирать между различными реализациями этих функций во время выполнения. Основная идея заключается в использовании динамической загрузки библиотек для выбора реализации функций.

Используются библиотеки **<cmath>** и **<stdexcept>** для математических функций и для обработки исключений соответственно.

Основной подход заключается в динамической загрузке библиотек с помощью функций **dlopen**, **dlsym** и **dlclose**.

Общий метод и алгоритм решения

Функции **E** и **Square** реализованы в отдельных файлах (**impl1.cpp** и **impl2.cpp**), что позволяет легко заменять их реализации. Эти файлы компилируются в динамические библиотеки (**libmy_math.so** и **libmy_math_alternative.so**), которые загружаются во время выполнения.

Функция **E(int x)**:

В impl1.cpp:

- Проверяется, что **x** — положительное целое число.

- Вычисляется приближение числа **e** по формуле **(1 + 1/x)^x**

В impl2.cpp:

- Проверяется, что **x** — неотрицательное целое число.

- Вычисляется приближение числа **e** с использованием ряда Тейлора (сумма ряда **1/n!**).

Функция **Square(float A, float B)**:

В impl1.cpp:

- Проверяется, что стороны **A** и **B** — положительные числа.

- Вычисляется площадь прямоугольника как произведение сторон **A * B**.

В impl2.cpp:

- Проверяется, что стороны **A** и **B** — положительные числа.

- Вычисляется площадь прямоугольника как половина произведения сторон **(A * B) / 2**.

Вызывается **dlopen** для загрузки соответствующей библиотеки (**libmy_math.so** или **libmy_math_alternative.so**. Вызывается **dlsym** для получения указателей на функции **E** и **Square**. Пользователь выбирает операцию (1 — вычисление числа e, 2 — вычисление площади прямоугольника). Пользователь вводит необходимые данные (значение **x** или стороны **A** и **B**). Вызывается соответствующая функция (**E** или **Square**) через указатель. Результат выводится на экран. Вызывается **dlclose** для закрытия библиотеки.

Исходный код

my_math.h:

```
#ifndef MY_MATH_H
#define MY_MATH_H

#ifdef _WIN32
   #ifdef BUILD_MY_MATH
      #define MY_MATH_API __declspec(dllexport)
   #else
      #define MY_MATH_API __declspec(dllimport)
   #endif
#else
   #define MY_MATH_API
#endif

extern "C" {
   MY_MATH_API float E(int x);
   MY_MATH_API float Square(float A, float B);
}

#endif
```

impl1.cpp:

```
#include "my_math.h"
#include <cmath>
```

```
#include <stdexcept>

float E(int x) {
    if (x <= 0) {
        throw std::invalid_argument("x must be a positive integer.");
    }
    return pow(1.0 + 1.0 / x, x);
}

float Square(float A, float B) {
    if (A <= 0 || B <= 0) {
        throw std::invalid_argument("Sides must be positive.");
    }
    return A * B;
}
```

impl2.cpp:

```
#include "my_math.h"
#include <cmath>
#include <stdexcept>

float E(int x) {
    if (x < 0) {
        throw std::invalid_argument("x must be a non-negative integer.");
    }
    float result = 0.0;
    float factorial = 1.0;
    for (int n = 0; n <= x; ++n) {
        if (n > 0) factorial *= n;
```

```
        result += 1.0 / factorial;
    }
    return result;
}

float Square(float A, float B) {
    if (A <= 0 || B <= 0) {
        throw std::invalid_argument("Sides must be positive.");
    }
    return (A * B) / 2.0;
}
```

main_dynamic.cpp:

```
#include <iostream>
#include <dlfcn.h>
#include <stdexcept>

int main() {
    try {
        void* handle = dlopen("./libmy_math_alternative.so", RTLD_LAZY);
        if (!handle) {
            throw std::runtime_error(dlerror());
        }

        auto E = (float (*)(int))dlsym(handle, "E");
        auto Square = (float (*)(float, float))dlsym(handle, "Square");
        if (!E || !Square) {
            dlclose(handle);
            throw std::runtime_error(dlerror());
```

```
    }

    int choice;
    std::cout << "Choose an operation:\n"
              << "1. Calculate e\n"
              << "2. Calculate area\n";
    std::cin >> choice;

    if (choice == 1) {
        int x;
        std::cout << "Enter x: ";
        std::cin >> x;

        std::cout << "e: " << E(x) << "\n";
    } else if (choice == 2) {
        float A, B;
        std::cout << "Enter A and B: ";
        std::cin >> A >> B;

        std::cout << "Area: " << Square(A, B) << "\n";
    } else {
        std::cout << "Invalid choice.\n";
    }

    dlclose(handle);
} catch (const std::exception& e) {
    std::cerr << "Error: " << e.what() << "\n";
}
```

```
    return 0;
}
```

main.cpp:

```
#include <iostream>
#include <dlfcn.h>
#include <stdexcept>

int main() {
    try {
        void* handle = dlopen("./libmy_math.so", RTLD_LAZY);
        if (!handle) {
            throw std::runtime_error(dlerror());
        }

        auto E = (float (*)(int))dlsym(handle, "E");
        auto Square = (float (*)(float, float))dlsym(handle, "Square");
        if (!E || !Square) {
            dlclose(handle);
            throw std::runtime_error(dlerror());
        }

        int choice;
        std::cout << "Choose an operation:\n"
                  << "1. Calculate e\n"
                  << "2. Calculate area\n";
        std::cin >> choice;

        if (choice == 1) {
```

```
            int x;
            std::cout << "Enter x: ";
            std::cin >> x;

            std::cout << "e: " << E(x) << "\n";
        } else if (choice == 2) {
            float A, B;
            std::cout << "Enter A and B: ";
            std::cin >> A >> B;

            std::cout << "Area: " << Square(A, B) << "\n";
        } else {
            std::cout << "Invalid choice.\n";
        }

        dlclose(handle);
    } catch (const std::exception& e) {
        std::cerr << "Error: " << e.what() << "\n";
    }

    return 0;
}
```

Демонстрация работы программы

```
n0wee@DESKTOP-8QSPN1P:~/Coding/os_labs/build/lab4$ ./lab4_static
Choose an operation:
1. Calculate e
2. Calculate area
1

10
```

Enter x: 10

e: 2.59374

n0wee@DESKTOP-8QSPN1P:~/Coding/os_labs/build/lab4$ ./lab4_dynamic

Choose an operation:

1. Calculate e

2. Calculate area

1

Enter x: 10

e: 2.71828

n0wee@DESKTOP-8QSPN1P:~/Coding/os_labs/build/lab4$ ./lab4_static

Choose an operation:

1. Calculate e

2. Calculate area

2

Enter A and B: 3 4

Area: 12

n0wee@DESKTOP-8QSPN1P:~/Coding/os_labs/build/lab4$ ./lab4_dynamic

Choose an operation:

1. Calculate e

2. Calculate area

2

Enter A and B: 3 4

Area: 6

Выводы

В процессе работы с программой были изучены ключевые концепции программирования на языке C++ и работы с динамическими библиотеками. Я научился использовать функции **dlopen**, **dlsym** и **dlclose** для динамической загрузки и работы с библиотеками, что позволяет гибко выбирать реализации функций во время выполнения. Этот опыт помог мне лучше понять принципы работы с динамическими библиотеками.