Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Лабораторная работа №5 по курсу**

**«Операционные системы»**

**Тема работы**

**“Динамические библиотеки”**

Студент: Молчанов Владислав Дмитриевич

Группа: М8О-208Б-20

Вариант: 17

Преподаватель: Миронов Евгений Сергеевич

Оценка: \_\_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_\_

Москва, 2021

**Содержание**

1. Репозиторий
2. Постановка задачи
3. Общие сведения о программе
4. Общий метод и алгоритм решения
5. Исходный код
6. Демонстрация работы программы
7. Выводы

**Репозиторий**

https://github.com/molch4nov/OS

**Постановка задачи**

Задача: реализовать 2 динамические библиотеки и 2 программы для работы с ними. Первая программа будет загружать библиотеку (одну) на этапе компиляции при помощи ключа -lmylib, а вторая программа будет подключать две динамические библиотеки при помощи dl-функций в самом коде.

**Общие сведения о программе**

Для выполнения данной лабораторной работы я предварительно создал 5 файлов: первые два - first.cpp и second.cpp являются исходным кодом для наших динамических библиотек. Файлы comp.cpp и launch.cpp являются двумя программами, которые нужно было реализовать по заданию. compilation.cpp является программой, к которой библиотека подгружается на этапе компиляции, а launch.cpp является программой, к которой библиотека подключается непосредственно в самом коде.

Помимо этого, для удобства компиляции всех программ я создал MakeFile со следующим набором команд:  
1) g++ -fPIC -c first.cpp -o d1.o

g++ -fPIC -c second.cpp -o d2.o

При помощи этих команд наши cpp-библиотеки превращаются в объектные файлы. Это, так называемый, “промежуточный этап” создания динамических библиотек.

2) g++ -shared d1.o -o libd1.so

g++ -shared d2.o -o libd2.so

При помощи флага -shared мы создаем наши нужные по заданию динамические библиотеки.

3) g++ comp.cpp -L. -ld1 -o main1 -Wl,-rpath -Wl,.

Этой строчкой мы делаем исполняемый файл из нашей программы compilation.cpp, при этом компилируем мы только с одной библиотекой (то есть компиляция может проходить либо с ключом -ld1, либо с ключом -ld2).

4) g++ launch.cpp -L. -ldl -o main2 -Wl,-rpath -Wl,.

Этой строчкой мы делаем исполняемый файл из нашей программы launch.cpp, только теперь с флагом -ldl. Далее в нашей программе main2 будут доступны 2 динамические библиотеки, действия над которыми будут обрабатываться при помощи следующих функций:  
void\* dlopen(...) - вгружает нашу библиотеку;

void\* dlsym(...) - присваивает указателю на функцию ее адрес в библиотеке

int dlclose(...) - освобождает указатель на библиотеку

5) rm -r \*.so \*.o main1 main2

При помощи команды make clean происходит удаление всех созданных файлов, вследствие чего в папке остаются исходные 5 объектов.

**Общий метод и алгоритм решения**

В самом начале выполнения лабораторной работы я реализовал две библиотеки: first.cpp и second.cpp. В библиотеке first.cpp реализованы подсчет количества простых чисел на отрезке и расчёт числа Пи через ряд Лейбница. В библиотеке second.cpp реализовано подсчет простых чисел на отрезке с помощью решета Эратосфена и реализован подсчет числа Пи формулой Валлиса. Далее в файле comp.cpp я реализовал обычное считывание команды при помощи проверки равенства функции scanf на -1 (вводится EOF - Ctrl+D на Ubuntu) и конструкции switch-case. Если вводится команда, отличная от 1 или 2, вылезает сообщение о том, что ввод был осуществлен неправильно. Если вводится 1, то считается количество простых чисел. Если вводится 2, то считается Пи. Что же касается launch.cpp, то там суть почти та же.

**Исходный код**

**first.cpp**

extern "C" int PrimeCount(int A, int B);

extern "C" float Pi(int K);

#include <cmath>

int PrimeCount(int A, int B){

    int ans = 0;

    bool flag = true;

    for(int i = A; i <= B; i++){

        for(int j = 2; j < B; j++){

            if(i % j == 0 && i != j){

                flag = false;

            }

        }

        if(flag == true){

            ans += 1;

        }

        flag = true;

    }

    if(A == 1){

        return ans - 1;

    }

    else{

        return ans;

    }

}

float Pi(int K){

    if(K < 0){

        return -1;

    }

    float pi = 1.0;

    for(int i = 1; i <= K; i++){

        pi += pow((-1), i)/(2\*i+1);

    }

    return pi \* 4;

}

**second.cpp**

extern "C" int PrimeCount(int A, int B);

extern "C" float Pi(int K);

#include <cmath>

#include <vector>

using namespace std;

int PrimeCount(int A, int B){

    int ans = 0;

    const long long N = 15485863;

    vector<bool>simple(N, true);

    vector<long long> v;

    for(int i = 2; i <= N; ++i) {

        if(simple[i] == true) {

            for(int j = i \* 2; j <= N; j += i) {

                simple[j] = false;

            }

            v.push\_back(i);

        }

    }

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

        if(v[i] >= A && v[i] <= B){

            ans += 1;

        }

    }

    return ans;

}

float Pi(int K){

    if(K < 0){

        return -1;

    }

    float pi = 1;

    for(int i = 1; i <= K; i++){

        pi \*= (4\*pow(i,2))/(4\*pow(i,2) - 1);

    }

    pi \*= 2;

    return pi;

}

**comp.cpp**#include <iostream>

using namespace std;

extern "C" int PrimeCount(int A, int B);

extern "C" float Pi(int K);

int main(){

    int checker;

    cout << "Choose a function" << endl;

    bool flag = true;

    while(flag){

        scanf("%d", &checker);

        switch(checker){

            case 1: {

                int A,B;

                cout << "Enter A and B numbers" << endl;

                cin >> A >> B;

                int ans = PrimeCount(A, B);

                cout << "Your answer: " << ans << endl;

                break;

            }

            case 2: {

                float K;

                cout << "Enter positive K" << endl;

                cin >> K;

                float ans = Pi(K);

                cout << "Pi  =  " << ans << endl;

                break;

            }

            case 3: {

                flag = false;

                break;

            }

            default:

            cout << "Only one or one + one or 3" << endl;

                break;

        }

    }

    return 0;

}

**launch.cpp**#include <stdio.h>

#include <iostream>

#include <stdlib.h>

#include <dlfcn.h>

using namespace std;

int main () {

    const char\* lib\_array[] = {"libd1.so", "libd2.so"};

    int cur, StartLib;

    cout << "Enter start library: " << endl;

    cout << "1 for using first library" << endl;

    cout << "2 for using second library" << endl;

    cin >> StartLib;

    bool flag = true;

    while (flag) {

        if (StartLib == 1) {

            cur = 0;

            flag = false;

        }

        else if (StartLib == 2) {

            cur = 1;

            flag = false;

        }

        else {

            cout << "Only one or one + one" << endl;

            cin >> StartLib;

        }

    }

    void\* handle = NULL;

    handle = dlopen(lib\_array[cur], RTLD\_LAZY);

    if (!handle) {

        cout << "An error while opening library has been detected" << endl;

        exit(EXIT\_FAILURE);

    }

    int (\*PrimeCount)(int A, int B);

    float (\*Pi)(int K);

    PrimeCount = (int(\*)(int, int))dlsym(handle, "PrimeCount");

    Pi = (float(\*)(int))dlsym(handle, "Pi");

    int command;

    cout << "Choose: " << endl;

    cout << "1 for changing the contract;" << endl;

    cout << "2 for calculating the count of simple numbers; " << endl;

    cout << "3 for calculating the Pi; " << endl;

    while (printf("Please enter your command: ") && (scanf("%d", &command)) != EOF) {

        if (command == 1) {

            dlclose(handle); //освобождает указатель на библиотеку и программа перестает ей пользоваться

            if (cur == 0) {

                cur = 1;

                handle = dlopen(lib\_array[cur], RTLD\_LAZY);

                if (!handle) {

                    cout << "An error while opening library has been detected" << endl;

                    exit(EXIT\_FAILURE);

                }

                PrimeCount = (int(\*)(int, int))dlsym(handle, "PrimeCount");

                Pi = (float(\*)(int))dlsym(handle, "Pi");

            }

            else if (cur == 1) {

                cur = 0;

                handle = dlopen(lib\_array[cur], RTLD\_LAZY);

                if (!handle) {

                    cout << "An error while opening library has been detected" << endl;

                    exit(EXIT\_FAILURE);

                }

                PrimeCount = (int(\*)(int, int))dlsym(handle, "PrimeCount");

                Pi = (float(\*)(int))dlsym(handle, "Pi");

            }

        cout << "You have changed contracts!" << endl;

        }

        else if (command == 2) {

            int A, B;

            cout << "Enter A and B "<< endl;

            cin >> A >> B;

            int ans = PrimeCount(A, B);

            cout << "Your answer: " << ans << endl;

        }

        else if (command == 3) {

            float pi; int K;

            cout << "Enter K:" << endl;

            cin >> K;

            pi = Pi(K);

            cout << "Your answer: " << pi << endl;

        }

        else {

            cout << "You had to enter only 1, 2 or 3!" << endl;

        }

    }

    dlclose(handle);

    return 0;

}

**Демонстрация работы программы  
  
  
  
  
  
vladislav@DESKTOP-OL36FK8:/mnt/c/Users/vlad-/Desktop/os\_lab5/src$ make**

**g++ -fPIC -c second.cpp -o d2.o**

**g++ -shared d2.o -o libd2.so**

**g++ comp.cpp -L. -ld2 -o main1 -Wl,-rpath -Wl,.**

**g++ -fPIC -c first.cpp -o d1.o**

**g++ -shared d1.o -o libd1.so**

**g++ launch.cpp -L. -ldl -o main2 -Wl,-rpath -Wl,.**

**vladislav@DESKTOP-OL36FK8:/mnt/c/Users/vlad-/Desktop/os\_lab5/src$ ./main1**

**Choose a function**

**1**

**Enter A and B numbers**

**10 5**

**Your answer: 0**

**2**

**Enter positive K**

**14**

**Pi = 3.0879**

**1**

**Enter A and B numbers**

**1 7**

**Your answer: 4**

**3**

**Выводы**Данная лабораторная работа научила меня пользоваться dl-функциями, благодаря реализации исполняемых файлов по заданию, я закрепил навык работы с динамическими библиотеками и полностью осознал их отличие от статических библиотек.