Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Лабораторная работа № 5 по курсу**

**«Операционные системы»**

Студент: Румынина Екатерина Александровна

Группа: М8О-201Б-21

Вариант: 10

Преподаватель: Миронов Евгений Сергеевич

Оценка: \_\_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_\_

Москва, 2022

**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/KetRum0/mai_os_labs>

**Постановка задачи**

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.

Пользовательский ввод для обоих программ должен быть организован следующим образом:

1. Если пользователь вводит команду «0», то программа переключает одну реализацию контрактов на другую (необходимо только для программы №2).
2. «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;
3. «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.

## Контракты и реализации функций

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Описание | Сигнатура | Реализация 1 | Реализация 2 |
| 2 | Рассчет производной функции cos(x) в точке A с приращением deltaX | Float Derivative(float A, float deltaX) | f'(x) = (f(A + deltaX) – f(A))/deltaX | f'(x) = (f(A + deltaX) – f(A-deltaX))/(2\*deltaX) |
| 4 | Подсчёт наибольшего общего делителя для двух натуральных чисел | Int GCF(int A, int B) | Алгоритм Евклида | Наивный алгоритм. Пытаться разделить числа на все числа, что меньше A и B. |

**Общие сведения о программе**

main\_dynamic.c - программа, использующая динамическую библиотеку

main\_static.c – программа, использующая статическую библиотеку

lib1.c – первая реализация функций

lib2.c – вторая реализация функций

func.h – объявление функций

**Общий метод и алгоритм решения**

Для статической реализации необходимо скомпилировать исполняемый файл с библиотекой. Для динамической реализации, необходимо использовать системный вызов dlopen, чтобы подгрузить библиотеку во время runtime.

**Исходный код**

**main\_dynamic.c**

#include <stdlib.h>

#include <stdio.h>

#include <dlfcn.h>

#include <math.h>

#include <stdbool.h>

const char LIB1[] = "./libd1\_dynamic.so";

const char LIB2[] = "./libd2\_dynamic.so";

int main(int argc, char\* argv[]) {

void \*library;

bool flag = false;

int x, c, d;

double a, b;

library = dlopen(LIB2, RTLD\_LAZY);

if (!library) {

printf("Error dlopen(): %s\n", dlerror());

return 1;

}

double(\*Derivative)(double x, double y);

int(\*GCF)(int x, int y);

\*(void\*\*)(&Derivative) = dlsym(library, "Derivative");

\*(void\*\*)(&GCF) = dlsym(library, "GCF");

for (;;) {

scanf("%d", &x);

if (x == 0) {

dlclose(library);

if (flag) {

library = dlopen(LIB2, RTLD\_LAZY);

flag = false;

} else {

library = dlopen(LIB1, RTLD\_LAZY);

flag = true;

}

if (!library) {

printf("Error dlopen(): %s\n", dlerror());

return 1;

}

\*(void\*\*)(&Derivative) = dlsym(library, "Derivative");

\*(void\*\*)(&GCF) = dlsym(library, "GCF");

} else if (x == 1) {

scanf("%lf %lf", &a, &b);

printf("Result: ");

double res = Derivative(a, b);

printf("%lf\n", res);

} else if (x == 2) {

scanf("%d %d", &c, &d);

printf("Result: ");

int res = GCF(c, d);

printf("%d\n", res);

} else {

dlclose(library);

return 0;

}

}

return 0;

}

**main\_static.c**

#include "func.h"

#include <stdlib.h>

#include <stdio.h>

int main(int argc, char\* argv[]) {

int x;

float a, b;

int c, d;

for (;;) {

scanf("%d", &x);

if (x == 1) {

scanf("%f %f", &a, &b);

float res = Derivative(a, b);

printf("%f\n", res);

x = 0;

} else if (x == 2) {

scanf("%d %d", &c, &d);

int res = GCF(c, d);

printf("%d\n", res);

x = 0;

} else {

break;

}

}

return 0;

}

**func.h**

#ifndef FUNC\_H

#define FUNC\_H

#include <stdlib.h>

#include <stdio.h>

#include <math.h>

double Derivative(double a, double b);

int GCF(int a,int b);

#endif

**lib1.c**

double Derivative(double x, double delta){

double dif = (cos(x+delta) - cos(x))/delta;

return dif;

}

int GCF(int a, int b){

while(b != 0) {

a = a % b;

int tmp = a;

a = b;

b = tmp;

}

return a;

}

**lib2.c**

double Derivative(double x, double delta){

double dif = (cos(x+delta) - cos(x-delta))/(2\*delta);

return dif;

}

int GCF(int a, int b){

int m ;

int res=1;

if (a < b) m = a;

else m = b;

for (int i = m; i >= 1; --i) {

if ((a % i == 0) && (b % i == 0)) {

res = i;

break;

}

}

return res;

}

**Демонстрация работы программы**

ket@ket-laptop:~/Desktop/mai\_os\_labs/lab5$ ./main\_static\_1

1 1 0.0001

-0.841498

1 3 0.000001

-0.141120

2 3 9

3

2 27 51

3

^C

ket@ket-laptop:~/Desktop/mai\_os\_labs/lab5$ ./main\_static\_2

1 1 0.0001

-0.841471

1 3 0.000001

-0.141120

2 3 9

3

2 27 51

3

^C

ket@ket-laptop:~/Desktop/mai\_os\_labs/lab5$ ./main\_dynamic

1 1 0.0001

Result: -0.841471

1 3 0.000001

Result: -0.141120

2 3 9

Result: 3

2 27 51

Result: 3

0

1 1 0.0001

Result: -0.841498

1 3 0.000001

Result: -0.141120

2 3 9

Result: 3

2 27 51

Result: 3

^C

**Выводы**

В результате выполнения данной лабораторной работы я научилась создавать динамические библиотеки и программы, которые используют функции динамических библиотек.