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: Полонский Кирилл Андреевич

Группа: М8О-208Б-20

Вариант: 11

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/kirillpolonskii/OS/tree/master/os_lab5>

**Постановка задачи**

Требуется создать динамические библиотеки, которые реализуют рассчет производной функции cos(x) в точке A с приращением deltaX и рассчет значения числа Пи при заданной длине ряда (K). 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

**Общие сведения о программе**

Файлы implementation1.c и implementation2.c содержат реализации функций из задания, файлы prog1.c и prog2.c содержат использование статической и динамической библиотеки, сборка осуществляется с помощью утилиты make.

В Makefile описаны флаги -fsanitize=address и -g для отслеживания ошибок и строк, в которых они возникают, а также флаги динамической и статической компиляции библиотек.

**Общий метод и алгоритм решения**

Пользовательский ввод для обоих программ организован следующим образом:

1. Если пользователь вводит команду «0», то программа переключает одну реализацию контрактов на другую (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.

**Исходный код**

implementation1.h

#include "math.h"

float Derivative(float A, float deltaX);

float Pi(int k);

implementation1.c

#include "implementation1.h"

#include "stdio.h"

float Derivative(float A, float deltaX){

printf("in 1st impl**\n**");

return (cos(A + deltaX) - cos(A)) / deltaX;

}

float Pi(int k){

float sum = 0.0;

for(float i = 0.0; i <= k; ++i){

sum += pow(-1, i) / (2.0 \* i + 1);

}

return sum \* 4.0;

}

implementation2.h

#include "math.h"

float Derivative(float A, float deltaX);

float Pi(int k);

implementation2.c

#include "implementation2.h"

#include "stdio.h"

float Derivative(float A, float deltaX){

printf("in 2st impl**\n**");

return (cos(A + deltaX) - cos(A - deltaX)) / (2.0 \* deltaX);

}

float Pi(int k){

float product = 1.0;

for(float i = 1.0; i <= k; ++i){

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

}

return product \* 2.0;

}

prog1.c

#include "implementation2.h"

#include "implementation1.h"

#include "stdio.h"

int main(){

int command;

while(scanf("%d ", &command) != EOF){

if (command == 1){

float A, deltaX;

scanf("%f %f", &A, &deltaX);

printf("cos'(%f) = %f**\n**", A, Derivative(A, deltaX));

}

else {

int k;

scanf("%d", &k);

printf("pi = %f**\n**", Pi(k));

}

}

return 0;

}

prog2.c

#include "implementation1.h"

#include "implementation2.h"

#include "stdio.h"

#include "dlfcn.h"

int main(){

int command, curLib = 0;

char\* libs[] = {"lib1.so", "lib2.so"};

void\* lib\_handler = dlopen(libs[curLib], RTLD\_LAZY); // initial library is first (with implementation1)

if (lib\_handler == NULL){

return 1;

}

float (\*Derivative)(float A, float deltaX);

float (\*Pi)(int k);

Derivative = dlsym(lib\_handler, "Derivative");

Pi = dlsym(lib\_handler, "Pi");

while(scanf("%d", &command) != EOF){

if (command == 0){

dlclose(lib\_handler);

curLib = 1 - curLib;

lib\_handler = dlopen(libs[curLib], RTLD\_LAZY);

if (lib\_handler == NULL){

return 1;

}

Derivative = dlsym(lib\_handler, "Derivative");

Pi = dlsym(lib\_handler, "Pi");

}

else if (command == 1){

float A, deltaX;

scanf("%f %f", &A, &deltaX);

printf("cos'(%f) = %f**\n**", A, Derivative(A, deltaX));

}

else{

int k;

scanf("%d", &k);

printf("pi = %f**\n**", Pi(k));

}

}

return 0;

**Демонстрация работы программы**

kirill@kirill-acpire:~/labsMAI/sem3/OS/os\_lab5/src$ ./prog1\_1

1 3.14 0.001

in 1st impl

cos'(3.140000) = -0.001093

2 400

pi = 3.144087

kirill@kirill-acpire:~/labsMAI/sem3/OS/os\_lab5/src$ ./prog1\_2

1 3.14 0.001

in 2st impl

cos'(3.140000) = -0.001592

2 400

pi = 3.139632

kirill@kirill-acpire:~/labsMAI/sem3/OS/os\_lab5/src$ ./prog2

1 3.14 0.001

in 1st impl

cos'(3.140000) = -0.001093

2 400

pi = 3.144087

0

1 3.14 0.001

in 2st impl

cos'(3.140000) = -0.001592

2 400

pi = 3.139632

**Выводы**

В ходе выполнения лабораторной работы я научился работать со статическими и динамическими библиотеками.