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: Куценко Б.Д.

Группа: М80-207Б-20

Преподаватель: Миронов Е.С.

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата:

**Содержание**

1. Постановка задачи.
2. Общие сведения о программе.
3. Код программы.
4. Демонстрация работы программы.
5. Вывод.

## **Постановка задачи**

Требуется создать динамические библиотеки, которые реализуют определенный 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 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:

1. «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. «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.

**Вариант 29:**

Составить и отладить программу на языке С с использованием дин. библиотек, со следующими котрактами и функциями.

## 

## **Общие сведения о программе**

Проект состоит из пяти исходных файлов main.c, din\_main.c, lib.h, lib1.c, lib2.c. Первые два это программы, в который тестируются функции из библиотек. Третий файл – интерфейс библиотек. И, наконец, 4 и 5 это файлы, в которых хранятся реализации функции библиотек. В lib1.c хранятся функции с Реализацией 1, а в lib2.c с Реализацией 2.

Программы использует следующие системные вызовы:

1. **dlopen** - загружает динамическую библиотеку, имя которой мы передаем, и возвращает прямой указатель на начало динамической библиотеки.
2. **dlclose** уменьшает на единицу счетчик ссылок на указатель динамической библиотеки. Если нет других загруженных библиотек, использующих ее символы и если счетчик ссылок принимает нулевое значение, то динамическая библиотека выгружается.
3. **dlsym** использует указатель на динамическую библиотеку, возвращаемую dlopen, и оканчивающееся нулем символьное имя, а затем возвращает адрес, указывающий, откуда загружается этот символ.
4. **dlerror** возвращает удобочитаемую строку, описывающую самую последнюю ошибку, возникшую в dlopen, dlsym или dlclose с момента последнего вызова dlerror.

## **Код программы**

**Makefile**

compile: Prog1 Prog2

Prog1: lib.h main.o lib1.o

gcc main.o lib1.o -o Prog1

Prog2: lib1.so lib2.so din\_main.c

gcc din\_main.c -ldl -o Prog2 -Wl,-rpath -Wl,.

main.o:

gcc -c main.c

lib1.o: lib.h lib1.c

gcc -c lib1.c -lm

lib1.so: lib1.c

gcc -lm -shared -fPIC lib1.c -o lib1.so -lm

lib2.so: lib2.c

gcc -lm -shared -fPIC lib2.c -o lib2.so -lm

clean:

rm \*.so \*.o Prog1 Prog2

**lib.h:**

#ifndef LIB\_H

#define LIB\_H

extern double ComputePi(int k);

extern double ComputeE(int x);

#endif // LIB\_H

**lib1.c:**

#include "lib.h"

#include <math.h>

#include <stdio.h>

long long facto(long long f)

{

return (f == 0) ? 1: (f \* facto(f - 1));

}

// coumpute Pi with Leibniz Series

double ComputePi(int k)

{

long double res = 0;

long double flag = 0;

for (long long i = 0; i <= k; i++) {

flag = (i & 1) ? -1: 1;

res += flag / (long long)(2 \* i + 1);

}

res \*= 4;

return res;

}

// compute e with taylor series

double ComputeE(int x) {

x = (x > 20) ? 20: x;

double res = 0;

for (int i = 0; i <= x; i++) {

res += (double)1/facto(i);

}

return res;

}

**lib2.c**

#include "lib.h"

#include <stdio.h>

#include <math.h>

#define M\_PI 3.14159265358979323846

//compute Pi with Wallis formula

double ComputePi(int k)

{

k = (k > 44749) ? 44749: k;

long double res = 1;

for (int i = 1; i <= k; i++) {

if( i > 23170) res \*= 1.0000000005;

else res \*= (long double)(4 \* i \* i) / (4 \* i \* i - 1);

}

res \*= 2;

return (double)res;

}

// compute e with (1 + 1 / x) ^ x

double ComputeE(int x)

{

x = (x > 0) ? x: x \* -1;

return pow((1 + (double)1 / x), x);

}

**main.c**

#include <stdio.h>

#include "lib.h"

int main()

{

char mode;

printf("To calculate the number pi with the long series k enter -- 1\n");

printf("To calculate the number e enter -- 2 \n");

while ((mode = getchar()) != EOF) {

if (mode == '\n') continue;

if (mode == '1') {

int k;

printf("Enter the series lenght k: ");

scanf("%d", &k);

printf("The value of the counted number pi: %.9lf\n", ComputePi(k));

} else if (mode == '2') {

int x;

printf("Enter the number: ");

scanf("%d", &x);

printf("The value of the counted number e: %lf\n", ComputeE(x));

} else {

printf("This command is not supported, enter 1 or 0\n");

// long long n;

// scanf("%lld", &n);

// printf("factorial %lld = %lld\n", n, facto(n));

}

}

// int i = 20000000;

// ComputePi(i);

return 0;

**din\_main.c**

#include <stdio.h>

#include <stdlib.h>

#include <dlfcn.h>

typedef enum {

FIRST,

SECOND,

} MODE;

MODE r = FIRST;

const char\* libName1 = "lib1.so";

const char\* libName2 = "lib2.so";

double (\*ComputePi)(int k) = NULL;

double (\*ComputeE)(int x) = NULL;

char \*err;

void \*libHandle = NULL;

void loadDLibs(MODE context){

const char \*name;

name = (context == FIRST) ? libName1 : libName2;

libHandle = dlopen(name, RTLD\_LAZY); //RTLD\_LAZY подразумевает разрешение неопределенных символов в виде кода, содержащегося в исполняемой динамической библиотеке

if(!libHandle){

fprintf(stderr, "%s\n", dlerror());

exit(EXIT\_FAILURE);

}

}

void unloadDLibs(){

dlclose(libHandle);

}

void loadContext(){

loadDLibs(r);

ComputePi = dlsym(libHandle, "ComputePi");

ComputeE = dlsym(libHandle, "ComputeE");

if((err = dlerror())) {

fprintf(stderr, "%s\n", err);

exit(EXIT\_FAILURE);

}

}

void changeContext(){

unloadDLibs();

if(r == FIRST){

r = SECOND;

} else {

r = FIRST;

}

loadContext();

}

void print(MODE context, int counter) {

(counter > 0) ? printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n") : counter;

(counter > 0) ? printf("Press Enter to countinue...\n") : counter;

(counter > 0) ? getchar() : counter;

(counter > 0) ? getchar() : counter;

system("clear");

printf("+-------------------------------------Mode: %d-------------------------------------+\n", context + 1);

switch (context)

{

case FIRST :

printf("|Enter -- 1. To calculate the number pi with Leibniz Series with the long series k |\n");

printf("|Enter -- 2. To calculate the number e with Teylor series with the long series k |\n");

break;

case SECOND :

printf("|Enter -- 1. To calculate the number pi with Wallis formula with the long series k|\n");

printf("|Enter -- 2. To calculate the number e with limit (1 + 1 / x) ^ x. |\n");

break;

}

printf("+----------------------------------------------------------------------------------+\n");

printf("|Enter -- 0. If you want to change methods(Mode) of calculation. |\n");

printf("|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\n");

}

int main(){

r = FIRST;

loadContext();

int task = 0;

print(r, 0);

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

if(task == 0){

changeContext();

print(r, 0);

puts("Ok. Mode was changed");

if(r == FIRST){

puts("The first mode is now enabled");

} else{

puts("The second mode is now enabled");

}

print(r, 1);

continue;

} else if(task == 1) {

int k;

printf("\nEnter the series lenght k: ");

scanf("%d", &k);

printf("The value of the counted number pi = %.9lf\n", ComputePi(k));

} else if(task == 2) {

//Сделать перевод строк(перепутал вариант)

int x;

printf("\nEnter the number: ");

scanf("%d", &x);

printf("The value of the counted number e: %lf\n", ComputeE(x));

} else printf("This command is not supported, enter 1, 2 or 0\n");

print(r, 1);

}

system("clear");

unloadDLibs();

## **Демонстрация работы программы**

**Main.c**

## sokol@DESKTOP-4ULOTQV:~/os/lab5/test$ ./Prog1

## To calculate the number pi with the long series k enter -- 1

## To calculate the number e enter -- 2

## 1

## Enter the series lenght k: 1000000

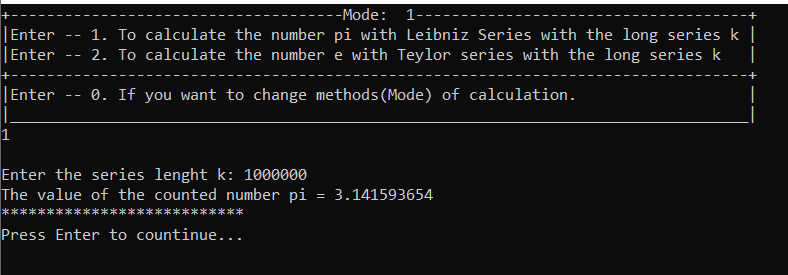
## The value of the counted number pi: 3.141593654

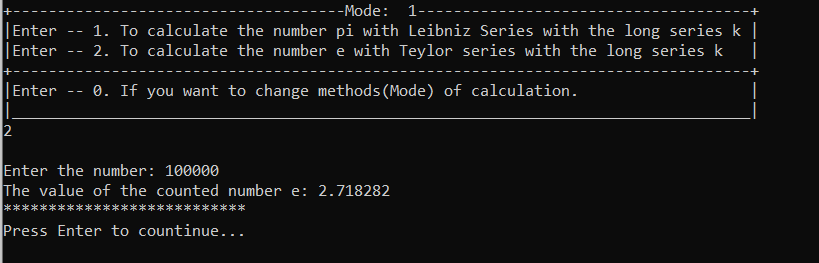
## 2

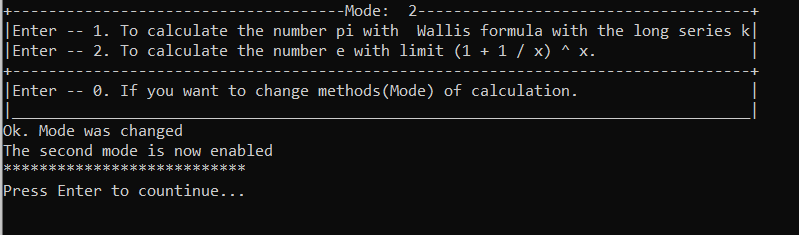
## Enter the number: 1000000

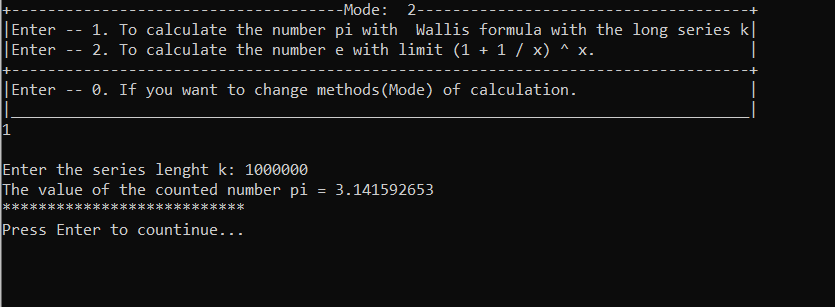
## The value of the counted number e: 2.718282

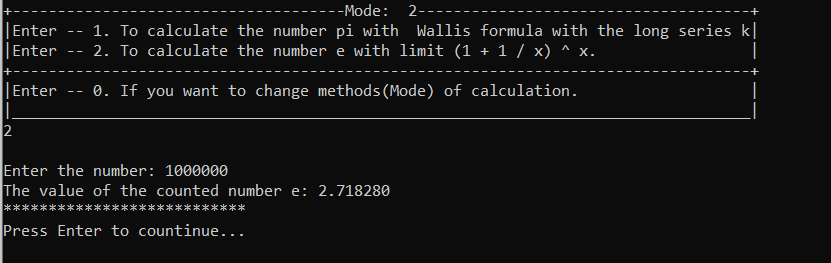
**Din\_main.c**

****

****







## **Вывод**

В данной ЛР мне пришлось познакомиться с использованием динамических библиотек. Этот метод полезен, когда надо добавить какое-либо обновление в программу, не перекомпилируя весь код. Также, можно менять реализации функций прямо во время программы. Динамические библиотеки часто используются для загрузки плагинов: не нужно пересобирать заново весь проект.