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»

Динамические библиотеки. Создание динамических библиотек. Создание программ, которые используют функции динамических библиотек.

Группа: М80-210Б-22

Студент: Бонокин Д.С.

Вариант:4

Преподаватель: Соколов А.А.

Оценка: _____

Дата: _____

Подпись:_____

Москва, 2023

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. Если пользователь вводит команду «0», то программа переключает одну реализацию контрактов на другую (необходимо только для программы №2). Можно реализовать лабораторную работу без данной функции, но максимальная оценка в этом случае будет «хорошо»;

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.

Вариант №4.

Контракты и реализации функций

1) Рассчет интеграла функции sin(x) на отрезке [A, B] с шагом e Float SinIntegral(float A, float B, float e)

Подсчет интеграла методом прямоугольников.

Подсчет интеграла методом трапеций.

2) Рассчет значения числа Пи при заданной длине ряда (K)

Рассчет значения числа Пи через ряд Лейбница

Рассчет значения числа Пи через формулу Валлиса

Листинг программы

Pi-sin.h

```
#ifndef _PI_SIN_
#define _PI_SIN_
#include <math.h>
#include <stdio.h>

float Pi(int k);
float sin_integral(float A, float B, float e);

#endif
```

Pi-sin-1.c

```
#include "../include/pi-sin.h"
#include <math.h>

float Pi(int k){
   float res = -1;
   for (int n = 0; n <= k; n++) {
      res = res + (-pow(-1, n) / (2 * n - 1));
   }
```

```
    return 4 * res;
}

float sin_integral(float A, float B, float e){
    float dx = (B - A) / e;
    int steps = (B - A) / dx;
    float current = A;
    float result = 0;
    for (int i = 0; i < steps; i++){
        result += dx * sin(current + dx / 2);
        current += dx;
    }
    result += (B - current) * sin((B + current) / 2);
    return result;
}
```

Pi-sin-2.c

```
#include "../include/pi-sin.h"
#include <math.h>

float Pi(int k){
    float res = 1;
    for (int n = 1; n <= k; n++) {
        res *= ((4 * pow(n, 2)) / (4 * pow(n, 2) - 1));
    }
    return 2 * res;
}

float sin_integral(float A, float B, float e){
    float dx = (B - A) / e;
    int steps = (B - A) / dx;
    float current = A;
```

```
    float result = 0;
    for (int i = 0; i < steps; ++i){
        result += dx * (sin(current) + sin(current + dx)) / 2;
        current += dx;
    }
    result += (B - current) * (sin(B) + sin(current)) / 2;
    return result;
}
```

Static.c

```
#include "include/pi-sin.h"



void information(){
    printf("\n1 - Pi\n2 - sin_integral\n -1 - EXIT\n");
}

int main(){
    int command;
    information();
    while(scanf("%d", &command) != EOF){
        switch (command){
        case -1:
            return 0;
        case 1:
            printf("\nenter lenght of series\n");
            int k;
            if (scanf("%d", &k) == EOF){
                printf("\ninvalid argument\n");
                break;
            };
            printf("\nPi = %f\n", Pi(k));
```

```
            break;
        case 2:
            printf("\nenter arguments: A B e\n");
            float a, b, e;
            if (scanf("%f %f %f", &a, &b, &e) == EOF){
                printf("\ninvalid arguments\n");
                break;
            };
            printf("\nintegral of sin(x) on [%f, %f] is %f\n", a, b, sin_integral(a, b, e));
            break;
        default:
            printf("\ninvalid command\n");
            break;
        }
        information();
    }
}
```

Dynamic.c

```
#include <stdio.h>
#include <dlfcn.h>

void* Handle = NULL;

float (*Pi)(int) = NULL;
float (*sin_integral)(float, float, float) = NULL;


int mode = 1;


void load_lib(){
```

```
    switch (mode){
        case 1:
                Handle = dlopen("../build/liblib1.so", RTLD_LAZY);
                break;
        case 2:
                Handle = dlopen("../build/liblib2.so", RTLD_LAZY);
                break;
        default:
            printf("invalid mode");
            return;
    }
    if (Handle == NULL){
            perror("dlopen");
            exit(-1);
    }
}

void load_contract(){
    load_lib();
    Pi = dlsym(Handle, "Pi");
    sin_integral = dlsym(Handle, "sin_integral");
}

void swap_contract(){
    dlclose(Handle);
    if(mode == 1){
        mode = 2;
    }
    else {
        mode = 1;
    }
```

```
    load_contract();
}

void information(){
    printf("\n1 - Pi\n2 - sin_integral\n3 - swap contract\n -1 - EXIT\n");
}


int main(){
    int command;
    information();
    load_contract();
    while(scanf("%d", &command) != EOF){
        switch (command){
        case -1:
            return 0;
        case 1:
            printf("\nenter lenght of series\n");
            int k;
            if (scanf("%d", &k) == EOF){
                printf("\ninvalid argument\n");
                break;
            };
            printf("\nPi = %f\n", (*Pi)(k));
            break;
        case 2:
            printf("\nenter arguments: A B e\n");
            float a, b, e;
            if (scanf("%f %f %f", &a, &b, &e) == EOF){
                printf("\ninvalid arguments\n");
                break;
```

```
            };
            printf("\nintegral of sin(x) on [%f, %f] is %f\n", a, b, (*sin_integral)(a, b, e));
            break;
        case 3:
            swap_contract();
            printf("\nswap\n");
            break;
        default:
            printf("\ninvalid command\n");
            break;
        }
        information();
    }
    dlclose(Handle);
}
```

CMakeLists.txt

```
cmake_minimum_required(VERSION 3.17)
project(os_lab_4)

set(CMAKE_CXX_STANDARD 17)
set(CMAKE_CXX_STANDARD_REQUIRED ON)
set(INCLUDE_DIR ${CMAKE_CURRENT_SOURCE_DIR}/include)
set(LIB_DIR ${CMAKE_CURRENT_SOURCE_DIR}/lib)

add_library(lib1 SHARED ${LIB_DIR}/pi-sin-1.c)
add_library(lib2 SHARED ${LIB_DIR}/pi-sin-2.c)

target_link_libraries(lib1 m)
target_link_libraries(lib2 m)
target_include_directories(lib1 PUBLIC all)
target_include_directories(lib2 PUBLIC all)
```

```
add_executable(static static.c)
add_executable(dinamic dinamic.c)

target_link_libraries(static PRIVATE lib1)
target_include_directories(dinamic PRIVATE all)
```

Примеры работы

```
danil@danil-HYM-WXX:~/Desktop/lab_os/lab4$ cd build
danil@danil-HYM-WXX:~/Desktop/lab_os/lab4/build$ ./dinamic

1 - Pi
2 - sin_integral
3 - swap contract
 -1 - EXIT
1

enter lenght of series
2

Pi = 2.666667

1 - Pi
2 - sin_integral
3 - swap contract
 -1 - EXIT
3

swap

1 - Pi
2 - sin_integral
```

3 - swap contract

-1 - EXIT

1

enter lenght of series

2

Pi = 2.844445

1 - Pi

2 - sin_integral

3 - swap contract

-1 - EXIT

3

swap

1 - Pi

2 - sin_integral

3 - swap contract

-1 - EXIT

1

enter lenght of series

1000

Pi = 3.140593

1 - Pi

2 - sin_integral

3 - swap contract

-1 - EXIT

2

enter arguments: A B e

1 2 10

|rectangle sin|

integral of sin(x) on [1.000000, 2.000000] is 0.956847

1 - Pi

2 - sin_integral

3 - swap contract

-1 - EXIT

3

swap

1 - Pi

2 - sin_integral

3 - swap contract

-1 - EXIT

1

enter lenght of series

1000

Pi = 3.140807

1 - Pi

2 - sin_integral

3 - swap contract

-1 - EXIT

2

enter arguments: A B e

1 2 10

|trapezoid sin|

integral of sin(x) on [1.000000, 2.000000] is 0.955652

1 - Pi

2 - sin_integral

3 - swap contract

-1 - EXIT

-1

danil@danil-HYM-WXX:~/Desktop/lab_os/lab4/build$ ./static

1 - Pi

2 - sin_integral

-1 - EXIT

1

enter lenght of series

100

Pi = 3.131593

1 - Pi

2 - sin_integral

-1 - EXIT

2

enter arguments: A B e

1 2 3

|rectangle sin|

integral of sin(x) on [1.000000, 2.000000] is 0.960892

1 - Pi

2 - sin_integral

-1 - EXIT

-1

Вывод

В ходе выполнения данной лабораторной работы я познакомился с тем, как создавать и использовать динамические библиотеки.

Динамические библиотеки используются во всех крупных проектах, чтобы при внесении изменений надо было перекомпилировать только одну библиотеку, а не весь проект. Также они удобны тем, что достаточно один раз выгрузить динамическую библиотеку в память и ей смогут пользоваться все нуждающиеся программы.