Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Лабораторная работа №5 по курсу**

**«Операционные системы»**

**ДИНАМИЧЕСКИЕ БИБЛИОТЕКИ**

Студент: Лисин Р.С.

Группа: М8О–206Б–20

Вариант: 6

Преподаватель: Соколов Андрей Алексеевич

Оценка: 5

Дата: 20.11.21

Подпись: \_\_\_\_\_\_\_\_\_\_

Москва, 2021

**Постановка задачи**

**Цель работы**

Целью является приобретение практических навыков в:

* Создание динамических библиотек
* Создание программ, которые используют функции динамических библиотек

**Задание**

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 No1*), которая используют одну из библиотек, используя знания полученные на этапе компиляции;
* Тестовая программа (*программа No2*), 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 No2*). Можно реализовать лабораторную работу без данной функции, но максимальная оценка в этом случае будет «хорошо»;

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 6.

1. Рассчет интеграла функции sin(x) на отрезке [A, B] с шагом e Float SinIntegral(float A, float B, float e) двумя способами: подсчет интеграла методом прямоугольников, подсчет интеграла методом трапеций.

7. Подсчет площади плоской геометрической фигуры по двум сторонам Float Square(float A, float B): фигура – прямоугольник, фигура - прямоугольный треугольник.

В программе используются следующие системные вызовы:

1. **dlopen** – загружает динамическую библиотеку с указанным именем. В случае неуспеха возвращает NULL.
2. **dlclose –** уменьшает на единицу счетчик ссылок на указатель динамической библиотеки *handle*. Если нет других загруженных библиотек, использующих ее символы и если счетчик ссылок принимает нулевое значение, то динамическая библиотека выгружается.  В случае успеха возвращает 0, иначе ненулевой результат.
3. **dlsym** – использует указатель на динамическую библиотеку, возвращаемую dlopen, и оканчивающееся нулем символьное имя, а затем возвращает адрес, указывающий на нужный символ. В случае неуспеха dlsym возвращает NULL.
4. **dlerror** – возвращает сообщение об ошибке, если ошибки не произошло, то возвращает NULL.

**Основные файлы программы**

**Makefile:**

all: prog1 prog2

prog1: lib1.so

gcc main1.c -L /home/roma/os\_lab\_5 -l1 -Wall -o prog1 -lm

prog2: lib1.so lib2.so

gcc main2.c -ldl -o prog2 -lm

lib1.so: lib1.o

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

lib2.so: lib2.o

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

lib1.o:

gcc lib1.c -c -Wall -Werror -fpic -o lib1.o -lm

lib2.o:

gcc lib2.c -c -Wall -Werror -fpic -o lib2.o -lm

clean:

rm -rf \*.so \*.o prog1 prog2 –lm

**libs.h**

// Рассчет интеграла функции sin(x) на отрезке [A, B] с шагом e

float sinintegral(float a, float b, float e);

// Подсчет площади плоской геометрической фигуры по двум сторонам

float square(float a, float b);

**lib1.c**

#include <math.h>

#include "libs.h"

// Метод прямоугольников

float sinintegral(float a, float b, float e) {

int n = (b - a) / e;

float integral = 0.0;

float c = 0.0;

if (n < 0) {

n = -n;

}

if (a < b) {

c = a;

}

else {

c = b;

}

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

integral += sinf((2.0 \* c + e) / 2.0) \* e;

c += e;

}

return integral;

}

// Площадь прямоугольника

float square(float a, float b) {

return a \* b;

}

**lib2.c**

#include <math.h>

// Метод трапеций

float sinintegral(float a, float b, float e) {

int n = (b - a) / e;

float integral = 0.0;

float c = 0.0;

if (n < 0) {

n = -n;

}

if (a < b) {

c = a;

}

else {

c = b;

}

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

integral += (sinf(c) + sinf(c + e)) / 2.0 \* e;

c += e;

}

return integral;

}

// Площадь прямоугольного треугольника

float square(float a, float b) {

return 0.5 \* a \* b;

}

**main1.c**

#include <stdio.h>

#include "libs.h"

int main() {

int cmd = 0;

float A = 0.0, B = 0.0, step = 0.0;

float a = 0.0, b = 0.0;

printf("1 a b c --> count integral of sin(x) on [a, b] with step c\n");

printf("2 a b --> the square of a flat figure\n");

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

switch(cmd) {

case 0:

printf("This is not program 2\n");

break;

case 1:

scanf("%f%f%f", &A, &B, &step);

printf("Result: %f\n", sinintegral(A, B, step));

break;

case 2:

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

printf("Result: %f\n", square(a, b));

break;

}

}

return 0;

}

**main2.c**

#include <stdio.h>

#include <stdlib.h>

#include <dlfcn.h>

#define LIBNAME\_FIRST "lib1.so"

#define LIBNAME\_SECOND "lib2.so"

int main() {

float (\*sinintegral) (float, float, float) = NULL;

float (\*square) (float, float) = NULL;

void \*handle1 = dlopen(LIBNAME\_FIRST, RTLD\_LAZY);

if (!handle1) {

printf("%s\n", dlerror());

exit(1);

}

void \*handle2 = dlopen(LIBNAME\_SECOND, RTLD\_LAZY);

if (!handle2) {

printf("%s\n", dlerror());

exit(2);

}

int lib = 1;

int cmd = 0;

float A = 0.0, B = 0.0, step = 0.0;

float a = 0.0, b = 0.0;

char \*error;

printf("1 a b c --> count integral of sin(x) on [a, b] with step c\n");

printf("2 a b --> the square of a flat figure\n");

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

switch(cmd) {

case 0:

lib = (lib == 1) ? 2 : 1;

printf("Library was changed to %d \n", lib);

break;

case 1:

if (lib == 1) {

sinintegral = dlsym(handle1, "sinintegral");

}

else {

sinintegral = dlsym(handle2, "sinintegral");

}

if ((error = dlerror()) != NULL) {

printf("Dlsym error\n");

exit(3);

}

scanf("%f%f%f", &A, &B, &step);

printf("Result: %f\n", sinintegral(A, B, step));

break;

case 2:

if (lib == 1) {

square = dlsym(handle1, "square");

}

else {

square = dlsym(handle2, "square");

}

if ((error = dlerror()) != NULL) {

printf("Dlsym error\n");

exit(4);

}

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

printf("Result: %f\n", square(a, b));

break;

}

}

if (dlclose(handle1) != 0) {

perror("Dlclose error");

exit(5);

}

if (dlclose(handle2) != 0) {

perror("Dlclose error");

exit(6);

}

return 0;

}

**Пример работы**

roma@DESKTOP-JD58QU2:~/os\_lab\_5$ ./prog1

1 a b c --> count integral of sin(x) on [a, b] with step c

2 a b --> the square of a flat figure

1 1 2 3

Result: 0.000000

2 3 4

Result: 12.000000

2 6 7

Result: 42.000000

1 6 8 1

Result: 1.153120

^C

roma@DESKTOP-JD58QU2:~/os\_lab\_5$ ./prog2

1 a b c --> count integral of sin(x) on [a, b] with step c

2 a b --> the square of a flat figure

0

Library was changed to 2

1 2 4 1

Result: 0.217367

0

Library was changed to 1

1 2 4 1

Result: 0.247689

2 3 4

Result: 12.000000

0

Library was changed to 2

2 3 4

Result: 6.000000

**Вывод**

В ходе выполнения данной лабораторной работы я познакомился с тем, как создавать и использовать динамические библиотеки. Стоить отметить, что в простых программах с минимальной функциональностью использование статических библиотек может быть предпочтительней, однако в больших программах, использующих несколько библиотек, использование динамических библиотек позволяет снизить потребление оперативной и дисковой памяти, поскольку динамическую библиотеку достаточно один раз выгрузить в память, что ею могли пользоваться все нуждающиеся программы.