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: Е. А. Айрапетова

Группа: М8О–206Б–20

Вариант: 24

Преподаватель: Соколов Андрей Алексеевич

Оценка: \_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_

Москва, 2020.

**Постановка задачи**

## Цель работы

Целью является приобретение практических навыков в:

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

## Задание

Требуется создать динамическую библиотеку, которая реализует определенный функционал. Далее использовать данную библиотеку 2-мя способами:

1. Во время компиляции (на этапе «линковки»/linking)
2. Во время исполнения программы, подгрузив библиотеку в память с помощью системных вызовов

В конечном итоге, программа должна состоять из следующих частей:

* Динамическая библиотека, реализующая заданных вариантом интерфейс;
* Тестовая программа, которая используют библиотеку, используя знания полученные на этапе компиляции;
* Тестовая программа, которая использует библиотеку, используя только местоположение динамической библиотеки и ее интерфейс.

Провести анализ между обоими типами использования библиотеки.

* Пользовательский ввод должен быть организован следующим образом:
* Команда «0»: переключить одну реализацию контрактов на другую
* Команда «1 args»: вызов первой функции контрактов
* Команда «2 args»: вызов второй функции контрактов

Контракты:

* Подсчёт наибольшего общего делителя для двух натуральных чисел (основание натурального логарифма) – при помощи Алгоритма Евклида и при помощи наивного алгоритма (попытаться разделить числа на все числа, что меньше A и B).
* Подсчет площади плоской геометрической фигуры по двум сторонам – площадь прямоугольника и прямоугольного треугольника.

**Общие сведения о программе**

Программа компилируется с помощью makefile. Для подключения библиотеки на этапе компиляции выполняются следующие действия:

Получение объектного файла основной программы.

Компиляция файла библиотеки с ключом -shared. Получаем .so файл.

Линковка библиотеки и объектного файла основной программы.

Для динамической загрузки библиотек используются средства библиотеки dlfcn.h.

Для указания пути к библиотеке в Makefile используем флаг -Wl,-rpath=./

Контракты реализованы в functions1.c и functions2.c.

**Общий метод и алгоритм решения**.

Программа принимает на вход неограниченное количество команд следующего вида:

0: переключение библиотеки. Выполняется при помощи функций dlopen, dlclose, dlsym, dlerror. Все системные ошибки обрабатываются.

1.1: расчет НОД с помощью Алгоритма Евклида: сравниваем числа, большему числу присваиваем значение разности входных чисел, до тех пор, пока они не станут равными. Во втором случае применяем наивный алгоритм – делим числа на все что меньше меньшего из них и сравниваем остатки.

2: Формулу прямоугольника вычисляем как a\*b, прямоугольного треугольника – как a\*b/2.

При вводе неверной команды программа выводит соответствующее сообщения.

**Основные файлы программы**

**functions.h**

extern float Square(float a, float b);

extern int GCF(int x, int y);

**functions1.c**

#include "functions.h"

float Square(float a, float b){

return a\*b;

}

int GCF(int x, int y){

while (x != y){

if (x>=y)

x = x - y;

else

y = y - x;

}

return x;

}

**functions2.c**

#include "functions.h"

float Square(float a, float b) {

return (a\*b)/2;

}

void Swap(int\* x, int\* y) {

int tmp = \*x;

\*x = \*y;

\*y = tmp;

}

int GCF(int x, int y) {

if (x>y)

Swap(&x,&y);

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

if (x % i == 0 && y % i == 0)

return i;

}

return 1;

}

**static.c**

#include "functions.h"

#include <stdio.h>

#include <stdlib.h>

int main(){

int c, x, y;

float a, b;

while (scanf("%d", &c) > 0){

if (c == 1){

if(scanf("%d %d", &x, &y) == 2)

printf("GCF(%d %d) = %d\n", x, y, GCF(x,y));

}

else if (c == 2){

if(scanf("%f %f", &a, &b) == 2)

printf("Square = %f\n", Square(a, b));

}

}

}

**dynamic.c**

//#include "functions.h"

#include <stdio.h>

#include <stdlib.h>

#include <dlfcn.h>

#include <math.h>

#define check(VALUE,BADVALUE,MSG) if (VALUE == BADVALUE){perror(MSG); exit(1); }

int main() {

char \*libnames[] = {"./libfunctions1.so", "./libfunctions2.so"};

int lib = 0;

int (\*GCF)(int, int) = NULL;

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

void \*handle;

handle = dlopen(libnames[lib], RTLD\_NOW);

check(handle,NULL,dlerror());

GCF = dlsym(handle,"GCF");

check(GCF,NULL,dlerror());

Square = dlsym(handle,"Square");

check(Square,NULL,dlerror());

int f,x,y;

float a, b;

while (scanf("%d", &f) > 0){

if (f == 0){

if (dlclose(handle) != 0)

perror(dlerror());

lib = (lib+1)%2;

handle = dlopen(libnames[lib],RTLD\_NOW);

check(handle,NULL, dlerror());

GCF = dlsym(handle,"GCF");

check(GCF,NULL,dlerror());

Square = dlsym(handle,"Square");

check(Square,NULL,dlerror());

}

else if (f == 1){

if (scanf("%d %d", &x, &y) == 2)

printf("GCF(%d,%d) = %d\n", x, y, GCF(x,y));

}

else if (f == 2){

if(scanf("%f %f", &a, &b) == 2)

printf("Square = %f\n", Square(a, b));

}

else

printf("Incorrect comand!\n");

}

if (dlclose(handle) != 0)

perror(dlerror());

}

**makefile**

all: static dynamic libfunctions

static: libfunctions

gcc -o static static.c -L./ -lfunctions2 -Wl,-rpath=./

dynamic: libfunctions

gcc -rdynamic -o dynamic dynamic.c -ldl

libfunctions: functions

gcc -shared -o libfunctions1.so functions1.o -lm

gcc -shared -o libfunctions2.so functions2.o

functions: functions.h functions1.c functions2.c

gcc -fPIC -c functions1.c

gcc -fPIC -c functions2.c

clean:

rm -f \*.so \*.o static dynamic

**Пример работы**

jane@Evgenia:/mnt/c/Files/ОСи/ЛР5$ ./dynamic

1

45 15

GCF(45,15) = 15

1

35 125

GCF(35,125) = 5

2

3 4

Square = 12.000000

jane@Evgenia:/mnt/c/Files/ОСи/ЛР5$ ./static

2

4 4

Square = 8.000000

1

35 125

GCF(35 125) = 5

**Вывод**

В процессе выполнения лабораторной работы я изучила динамические библиотеки. Динамические библиотеки содержат функции, которые будут загружены в оперативную память только тогда, когда понадобятся, поэтому использования таких библиотек существенно экономит память и ускоряет процесс сборки программы.