

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: Кажкин Денис
Группа: М8О-207Б-21
Вариант: 7
Преподаватель: Черемисинов Максим
Оценка: _____
Дата: _____
Подпись: _____

Москва, 2022

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/DKazhekin/OS>

Постановка задачи

Цель работы

Целью является приобретение практических навыков в:

1. Создание динамических библиотек
2. Создание программ, которые используют функции динамических библиотек
3. Работа со сборочной системой

Задание

Разработать:

- Динамические библиотеки, реализующие функции исчисления определенного интеграла (методом прямоугольников и трапеций) и перевода десятичного числа в другие системы счисления (двоичная и троичная).
- Тестовая программа (*программа №1*), которая использует одну из библиотек, используя знания полученные на этапе компиляции;
- Тестовая программа (*программа №2*), которая загружает библиотеки, используя только их местоположение и контракты.
- Составить Makefile (для сборки программы №1 и №2).

Общие сведения о программе

Программа 1 компилируется из файла `program1.c`, `libmath1.c`.

Программа 2 компилируется из файла `program2.c`, `libmath1.so`, `libmath2.so` (создаваемые в Makefile).

Также используется заголовочные файлы: `math1.h`, `math2.h`

Общий метод и алгоритм решения

Для начала я реализовал файлы `libmath1.c` и `libmath2.c`, в которых определены функции исчисления интеграла и перевода числа в другую систему счисления, и определил для них соответствующие заголовочные файлы `math1.h` и `math2.h`

Затем приступил к написанию первой программы, где все тривиально. Прописаны проверки на

аргументы , подающиеся вместе с исполняемым файлом, и соответственно сам цикл вывода ответа при помощи исполнения функции интеграла, полученной из статичной библиотеки на этапе компиляции (я выбрал именно функцию интеграла для первой программы)

Далее реализовал вторую программу, где используются функции из обеих библиотек, причем динамических. Также прописаны проверки на входные аргументы с исполняемым файлом. Дополнительно реализована возможность «переключать режим» функции при печати символа «0» с помощью указателя, который меняет функции, что дополнительно сопровождается пользовательской печатью для более простого взаимодействия с программой.

Так как в проекте много Си файлов и библиотек, то необходимо было составить Makefile для упрощения сборки. Его я прописал так, что при вызове «make» компилируется программа 1 и программа 2 под исполняемыми файлами program1 и program2. Вторая программа отлична тем, что дополнительно создаются «.so» файлы, которые линкуются с исполняемым файлом и подгружаются в момент запуска программы.

Исходный код

Program1.c:

```
#include "stdio.h"
#include "stdlib.h"
#include "math1.h"

int main(int argc, char *argv[]){
    // Declaring a pointer to realisation of an integral function
    float (*ptr_to_fun)(float a, float b, float n);
    ptr_to_fun = &trapez_integral;

    // Argument error check
    if (argc == 1){
        perror("INCORRECT INPUT KEYS (at least 3 arguments needed)\n");
        exit(0);
    }
    else {
        // Argument error check
        if ((*argv[1] == '1') & (((argc - 2) % 3) != 0)) {
```

```

        perror("INCORRECT INPUT KEYS (for every computation of integral 3 args needed)\n");
        exit(0);
    }

    // Answer printing
    else{
        int n = 2;
        for (int i = 0; i < ((argc - 2) / 3); i++){
            printf("%f\n", ptr_to_fun(atoi(argv[n]), atoi(argv[n+1]), atoi(argv[n+2])));
            n += 3;
        }
    }
    return 0;
}
}

```

Program2.c:

```

#include "stdio.h"
#include "stdlib.h"
#include "math1.h"
#include "math2.h"

int main(int argc, char *argv[]){
    // Argument error check
    if (argc == 1){
        perror("INCORRECT INPUT KEYS (need more arguments)\n");
        exit(0);
    }
    else {
        // Argument error check
        if (*argv[1] == '1') {
            if ((*argv[1] == '1') & (((argc - 2) % 3) != 0)) {
                perror("INCORRECT INPUT KEYS (for every computation of integral 3 args needed)\n");
                exit(0);
            }
            else {

```

```

int check = 1;

int tmp = 1;

// Declaring a pointer to realisation of an integral function
float (*ptr_to_fun)(float a, float b, float n);

printf("Current method - trapez\n");

int n = 2;
for (int i = 0; i < ((argc - 2) / 3); i++) {

    printf("%s", "Enter '0' if you want to switch, '1' otherwise\n");
    scanf("%d", &tmp);

    if (tmp == 0) {
        if (check == 1) {
            check = 0;
        }
        else {
            check = 1;
        }
    }

    if (check == 1) {
        ptr_to_fun = &trapez_integral;
        printf("%f\n", ptr_to_fun(atof(argv[n]), atof(argv[n + 1]), atof(argv[n + 2])));
        n += 3;
        printf("Current method - trapez\n");
    }
    else {
        ptr_to_fun = &rectangle_integral;
        printf("%f\n", ptr_to_fun(atof(argv[n]), atof(argv[n + 1]), atof(argv[n + 2])));
        n += 3;
        printf("Current method - rectangle\n");
    }
}

```

```

    }

    }

}

}

else if (*argv[1] == '2') {

    int check = 1;
    int tmp = 1;

    int (*ptr_to_fun)(long a);

    printf("Current translation - to binary\n");

    int n = 2;
    for (int i = 0; i < (argc - 2); i++) {

        printf("%s", "Enter '0' if you want to switch, '1' otherwise\n");
        scanf("%d", &tmp);

        if (tmp == 0) {
            if (check == 1) {
                check = 0;
            }
            else {
                check = 1;
            }
        }

        if (check == 1) {
            ptr_to_fun = &dec2bin;
            int k = atoi(argv[n]);
            printf("%s ", argv[n]);

```

```

        printf("%s", "----> ");
        printf("%d\n", ptr_to_fun(k));
        n += 1;
        printf("Current method - to binary\n");
    }
    else {
        ptr_to_fun = &dec2third;
        int k = atoi(argv[n]);
        printf("%s ", argv[n]);
        printf("%s", "----> ");
        printf("%d\n", ptr_to_fun(k));
        n += 1;
        printf("Current method - to ternary\n");
    }

}

}

return 0;
}
}

```

Libmath1.c:

```

#include "stdio.h"
#include "math.h"

float trapez_integral(float a, float b, float n){
    int i;
    double h,x,sum=0,integral;
    /*Begin Trapezoidal Method: */
    h=fabs(b-a)/n;
    for(i=1;i<n;i++){
        x=a+i*h;
        sum=sum+sin(x);
    }
}

```



```

    }
    integral=(h/2)*(sin(a)+sin(b)+2*sum);
    return integral;
}

float rectangle_integral(float a, float b, float n){
    double h,S=0,x;
    int i;
    /*Begin Rectangle Method: */
    h=fabs(b-a)/n;
    for(i=0;i<n-1;i++)
    {
        x=a+i*h;
        S=S+sin(x);
    }
    S=h*S;
    return S;
}

```

Libmath2.c:

```

#include "stdio.h"
#include "math.h"
#include "stdlib.h"
#include "string.h"

int dec2bin(long num){
    int bin = 0, k = 1;
    while (num != 0)
    {
        bin += (num % 2) * k;
        k *= 10;
        num /= 2;
    }
    return bin;
}

```

```

int dec2third(long num) {
    int third = 0, k = 1;
    while (num)
    {
        third += (num % 3) * k;
        k *= 10;
        num /= 3;
    }
    return third;
}

```

Math1.h:

```

#ifndef MATH1_H
#define MATH1_H

```

```

float trapez_integral(float a, float b, float n);
float rectangle_integral(float a, float b, float n);

```

```

#endif

```

Math2.h:

```

#ifndef MATH2_H
#define MATH2_H

```

```

int dec2bin(long num);
int dec2third(long num);

```

```

#endif

```

Makefile:

```

CC = clang
CFLAGS = -Wall -g

```

```

all:

```

```

    LD_LIBRARY_PATH=.

```

```
$(CC) $(CFLAGS) program1.c libmath1.c -o program1
```

```
$(CC) $(CFLAGS) program2.c -L. -lmath1 -lmath2 -o program2
```

libmath1.o: libmath1.c math1.h

```
$(CC) $(CFLAGS) -c libmath1.c
```

libmath2.o: libmath2.c math2.h

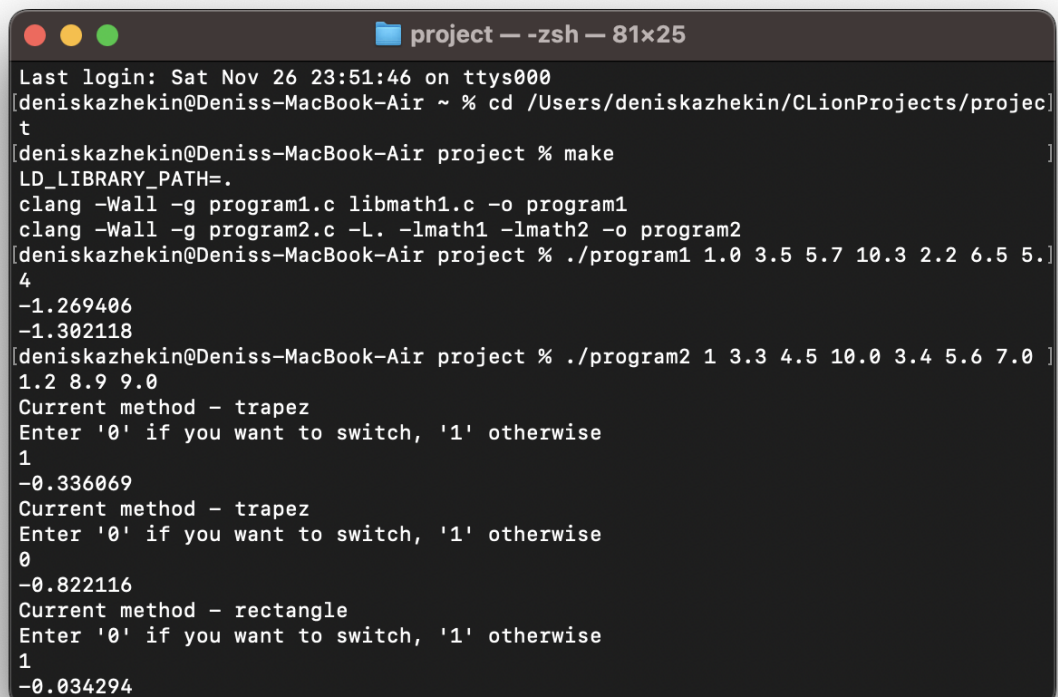
```
$(CC) $(CFLAGS) -c libmath2.c
```

libmath1.so: libmath1.c math1.h

```
$(CC) $(CFLAGS) -fPIC -shared -o $@ libmath1.c -lc
```

libmath2.so: libmath2.c math2.h

```
$(CC) $(CFLAGS) -fPIC -shared -o $@ libmath2.c -lc
```



```
project — zsh — 81x25
Last login: Sat Nov 26 23:51:46 on ttys000
[deniskazhekin@Deniss-MacBook-Air ~ % cd /Users/deniskazhekin/CLionProjects/project
[deniskazhekin@Deniss-MacBook-Air project % make
LD_LIBRARY_PATH=.
clang -Wall -g program1.c libmath1.c -o program1
clang -Wall -g program2.c -L. -lmath1 -lmath2 -o program2
[deniskazhekin@Deniss-MacBook-Air project % ./program1 1.0 3.5 5.7 10.3 2.2 6.5 5.]
4
-1.269406
-1.302118
[deniskazhekin@Deniss-MacBook-Air project % ./program2 1 3.3 4.5 10.0 3.4 5.6 7.0 ]
1.2 8.9 9.0
Current method - trapez
Enter '0' if you want to switch, '1' otherwise
1
-0.336069
Current method - trapez
Enter '0' if you want to switch, '1' otherwise
0
-0.822116
Current method - rectangle
Enter '0' if you want to switch, '1' otherwise
1
-0.034294
```

Демонстрация работы программы

```
project — -zsh — 81x25
Current method - rectangle
[deniskazhekin@Deniss-MacBook-Air project % ./program2 2 1 2 3 4 5 6 7 8 ]
Current translation - to binary
Enter '0' if you want to switch, '1' otherwise
0
1 ----> 1
Current method - to ternary
Enter '0' if you want to switch, '1' otherwise
1
2 ----> 2
Current method - to ternary
Enter '0' if you want to switch, '1' otherwise
1
3 ----> 10
Current method - to ternary
Enter '0' if you want to switch, '1' otherwise
1
4 ----> 11
Current method - to ternary
Enter '0' if you want to switch, '1' otherwise
0
5 ----> 101
Current method - to binary
Enter '0' if you want to switch, '1' otherwise
0
```

```
project — -zsh — 81x25
Enter '0' if you want to switch, '1' otherwise
1
3 ----> 10
Current method - to ternary
Enter '0' if you want to switch, '1' otherwise
1
4 ----> 11
Current method - to ternary
Enter '0' if you want to switch, '1' otherwise
0
5 ----> 101
Current method - to binary
Enter '0' if you want to switch, '1' otherwise
0
6 ----> 20
Current method - to ternary
Enter '0' if you want to switch, '1' otherwise
0
7 ----> 111
Current method - to binary
Enter '0' if you want to switch, '1' otherwise
1
8 ----> 1000
Current method - to binary
deniskazhekin@Deniss-MacBook-Air project %
```

Выводы

Выполнив лабораторную работу цели были достигнуты: я научился создавать программы с использованием динамических и статических библиотек, а также укрепил свои знания в составлении Makefile'ов.