

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: Матвеев Данил
Группа: М8О-207Б-21
Вариант: 22
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/MrDenli/OsLabs>

Постановка задачи

Целью является приобретение практических навыков в:

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

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:

1. Динамические библиотеки, реализующие контракты, которые заданы вариантом;
 - 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.

Провести анализ двух типов использования библиотек.

Пользовательский ввод для обеих 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.

Так же нужно сделать файл `stake` с особенностями согласно выданному варианту

Общие сведения о программе

У нас имеется два файла `lib1.cpp` и `lib2.cpp`, в каждом из которых представлена одна из реализаций выданной функции. `config.h` и `config.h.in` — файлы, служащие для отображения данных компиляции (вариант `stake`). `Main1.cpp` — файл, которому предписываются библиотеки на этапе компиляции. Из него получается два исполняемых файла `main1` и `main2`. `Main2.cpp` — файл, использующий динамические библиотеки. Из него получается исполняемый файл `main`. `Lib.h` нужен для подключения библиотек.

Общий метод и алгоритм решения

В одной реализации: перевод в двоичную систему и вывод простых чисел примитивным алгоритмом

В другой реализации: перевод в троичную систему и вывод простых чисел с помощью решета эратосфена

Реализуем три исполняемых файла. Первые два с библиотеками, подключенными на этапе линковки. Последний будет способен переключать реализации.

Исходный код

`lib1.cpp`

`#include<iostream>`

```
#include<cmath>
```

```
using namespace std;
```

```
// extern позволяет компилятору знать о типах и именах глобальных  
переменных без действительного создания этих переменных
```

```
extern "C" float SinIntegral(float a, float b, float e)
```

```
{  
    float square = 0;  
    for (float i = a; i <= b; i += e) {  
        square += e * sin(i);  
    }  
    return square;  
}
```

```
extern "C" char* translation(long x)
```

```
{  
    if (x == -1) {  
        cout<<"binary";  
    }  
    int cnt = 0;  
    int sizelong = 31;  
    char* binary = new char[sizelong];  
    for (int i = 0; i < sizelong; i++) {  
        binary[i] = '9';  
    }  
    while(x > 0) {  
        if (x%2 == 1) {  
            binary[sizelong - cnt - 1] = '1';  
        } else {
```

```

        binary[sizelong - cnt - 1] = '0';
    }
    x = x/2;
    cnt++;
}
return binary;
}

```

lib2.cpp

```
#include<iostream>
```

```
#include<cmath>
```

```
using namespace std;
```

```
extern "C" float SinIntegral(float a, float b, float e)
```

```

{
    float square = 0;
    for (float i = a; i < b; i += e) {
        square += e * ((sin(i) + sin(i + e)) / 2);
    }
    return square;
}

```

```
extern "C" char* translation(long x)
```

```

{
    if (x == -1) {
        cout<<"ternary";
    }
}

```

```

    int cnt = 0;
    int sizelong = 20;
    char* ternary = new char[sizelong];
    for (int i = 0; i < sizelong; i++) {
        ternary[i] = '9';
    }
    while(x > 0) {
        if (x%3 == 1) {
            ternary[sizelong - cnt - 1] = '1';
        } else if (x%3 == 2){
            ternary[sizelong - cnt - 1] = '2';
        } else {
            ternary[sizelong - cnt - 1] = '0';
        }
        x = x/3;
        cnt++;
    }
    return ternary;
}

```

main1.cpp

```

#include<iostream>
#include<stdio.h>
#include<cmath>
#include"lib.h"

```

using namespace std;

int main()

```

{
    cout << "Записывайте команды в виде: <command> <arg1> <arg2>
... <argn>" << endl;

    cout << "Если вы хотите посчитать интеграл функции sin(x) на
отрезке [a, b] с шагом e, введите: 1 <a> <b> <e> " << endl;

    cout << "Если вы хотите перевести число из десятичной системы
счисления, введите: 2 <x> " << endl;

    int command;
    while(cin >> command) {
        if (command == 2) {
            long x;
            cin >> x;
            char* rez;
            rez = translation(x);
            cout << "Число" << " " << x << " " << " в другой системе
- ";

            for (int i = 0; i < 32; i++) {
                if (rez[i] == '1' || rez[i] == '0' || rez[i] == '2') {
                    cout << rez[i];
                }
            }

            cout<<endl;
            delete rez;
        } else if (command == 1) {
            float a1, b1, e;
            cin >> a1 >> b1 >> e;
            float res1 = SinIntegral(a1, b1, e);
            cout << "Интеграл функции sin(x) на отрезке [" << a1 <<
", " << b1 << "]" с шагом " << e << " - " << res1 << endl;
        } else {

```



```

        cout << "Неверно введенная команда. Повторите ввод"
    << endl;
    }
}

}

main2.cpp
#include<iostream>
#include<stdio.h>
#include<cmath>
#include<dlfcn.h>
#include<string>

using namespace std;

int main()
{
    cout << "Сейчас вы находитесь в 1 реализации программы " <<
endl;
    cout << "Записывайте команды в виде: <command> <arg1> <arg2>
... <argn>" << endl;
    cout << "Если вы хотите посчитать интеграл функции sin(x) на
отрезке [a, b] с шагом e, введите 1 <a> <b> <e> " << endl;
    cout << "Если вы хотите перевести число из десятичной системы
счисления, введите: 2 <x> " << endl;
    cout << "Если вы хотите поменять реализацию программы,
введите 0 <a> <b> " << endl;

    int command;

    string lib1 = "./liblib1.dylib"; // хранятся динамические библиотеки

```

```
string lib2 = "./liblib2.dylib";
```

```
void* cur_lib = dlopen(lib1.c_str(), RTLD_LAZY); //загружает  
динамическую библиотеку
```

```
//RTLD_LAZY, подразумевающим разрешение неопределенных  
символов в виде кода, содержащегося в исполняемой динамической  
библиотеке
```

```
float (*SinIntegral)(float a, float b, float e);
```

```
char* (*translation)(long x);
```

```
SinIntegral = (float (*)(float, float, float))dlsym(cur_lib, "SinIntegral");
```

```
translation = (char (*)(long))dlsym(cur_lib, "translation");
```

```
int id = 1;
```

```
while(cin >> command) {
```

```
    if (command == 0) {
```

```
        dlclose(cur_lib);
```

```
        if (id == 1) {
```

```
            cur_lib = dlopen(lib2.c_str(), RTLD_LAZY);
```

```
            id = 2;
```

```
            cout << "Теперь вы находитесь во 2 реализации  
программы " << endl;
```

```
        } else {
```

```
            cur_lib = dlopen(lib1.c_str(), RTLD_LAZY);
```

```
            id = 1;
```

```
            cout << "Теперь вы находитесь в 1 реализации  
программы " << endl;
```

```
        }
```

```
        SinIntegral = (float (*)(float, float, float))dlsym(cur_lib,  
"SinIntegral");
```

```

translation = (char*)(*)(long))dlsym(cur_lib, "translation");

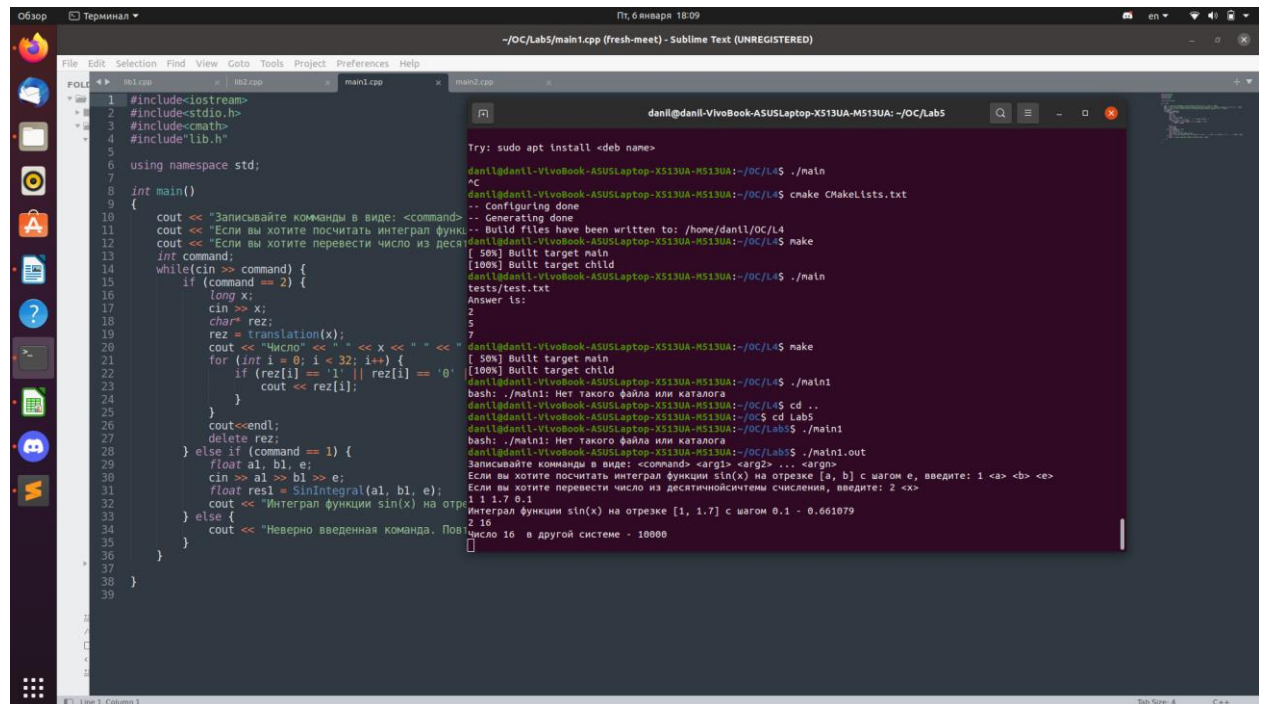
} else if (command == 2) {
    long x;
    cin >> x;
    char* rez;
    rez = translation(x);
    cout << "Число" << " " << x << " " << "в другой системе
- ";

    for (int i = 0; i < 32; i++) {
        if (rez[i] == '1' || rez[i] == '0' || rez[i] == '2') {
            cout << rez[i];
        }
    }
    cout<<endl;
    delete rez;
} else if (command == 1) {
    float a1, b1, e;
    cin >> a1 >> b1 >> e;
    cout << a1 << " " << b1 << " " << e << endl;
    float res1 = SinIntegral(a1, b1, e);
    cout << "Интеграл функции sin(x) на отрезке [" << a1 <<
", " << b1 << "] с шагом " << e << " - " << res1 << endl;
} else {
    cout << "Неверно введенная команда. Повторите ввод"
<< endl;
}
}
}

```

}

Демонстрация работы программы



```
1 #include<iostream>
2 #include<stdio.h>
3 #include<cmath>
4 #include<lib.h>
5
6 using namespace std;
7
8 int main()
9 {
10     cout << "Записывайте команды в виде: <command>"
11     cout << "Если вы хотите посчитать интеграл функт"
12     cout << "Если вы хотите перевести число из десяти"
13     int command;
14     while(cin >> command) {
15         if (command == 2) {
16             long x;
17             cin >> x;
18             char* rez;
19             rez = translation(x);
20             cout << "Число" << " " << x << " " << "
21             for (int i = 0; i < 32; i++) {
22                 if (rez[i] == '1' || rez[i] == '0')
23                     cout << rez[i];
24             }
25         }
26         cout<<endl;
27         delete rez;
28     } else if (command == 1) {
29         float a1, b1, e;
30         cin >> a1 >> b1 >> e;
31         float res1 = SinIntegral(a1, b1, e);
32         cout << "Интеграл функции sin(x) на отрезке [a1, b1] с шагом e: " << res1 << endl;
33     } else {
34         cout << "Неверно введенная команда. Повторите ввод." << endl;
35     }
36 }
37
38
39
```

```
Try: sudo apt install <deb name>
danil@danil-VivoBook-ASUSLaptop-X513UA-M513UA: ~/OC/L4$ ./main
^C
danil@danil-VivoBook-ASUSLaptop-X513UA-M513UA: ~/OC/L4$ cmake CMakeLists.txt
-- Configuring done
-- Generating done
-- Build files have been written to: /home/danil/OC/L4
danil@danil-VivoBook-ASUSLaptop-X513UA-M513UA: ~/OC/L4$ make
[ 50%] Built target main
[100%] Built target chld
danil@danil-VivoBook-ASUSLaptop-X513UA-M513UA: ~/OC/L4$ ./main
tests/test.txt
Answer is:
2
5
7
danil@danil-VivoBook-ASUSLaptop-X513UA-M513UA: ~/OC/L4$ make
[ 50%] Built target main
[100%] Built target chld
danil@danil-VivoBook-ASUSLaptop-X513UA-M513UA: ~/OC/L4$ ./main1
bash: ./main1: Нет такого файла или каталога
danil@danil-VivoBook-ASUSLaptop-X513UA-M513UA: ~/OC/L4$ cd ..
danil@danil-VivoBook-ASUSLaptop-X513UA-M513UA: ~/OC$ cd Lab5
danil@danil-VivoBook-ASUSLaptop-X513UA-M513UA: ~/OC/Lab5$ ./main1
bash: ./main1: Нет такого файла или каталога
danil@danil-VivoBook-ASUSLaptop-X513UA-M513UA: ~/OC/Lab5$ ./main1.out
Записывайте команды в виде: <command> <arg1> <arg2> ... <argn>
Если вы хотите посчитать интеграл функции sin(x) на отрезке [a, b] с шагом e, введите: 1 <a> <b> <e>
Если вы хотите перевести число из десятичной системы счисления, введите: 2 <x>
1 1.7 0.1
Интеграл функции sin(x) на отрезке [1, 1.7] с шагом 0.1 - 0.661079
2 16
Число 16 в другой системе - 10000
```

Выводы

Мне понравилась данная лабораторная работа, я научился работать с динамическими библиотеками и повторил MakeFile, что скорее всего будет полезно для меня в будущем.