Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Лабораторная работа №5 по курсу**

**«Операционные системы»**

**Тема работы**

**“Динамические библиотеки”**

Студент: Зубко Дмитрий Валерьевич

Группа: М8О-208Б-20

Вариант: 18

Преподаватель: Миронов Евгений Сергеевич

Оценка: \_\_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_\_

Москва, 2021

**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/usernameMAI/OS/tree/main/os\_lab5

**Постановка задачи**

**Цель работы**  
Целью является приобретение практических навыков в:

1. Создание динамических библиотек

2. 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 №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.

Задания по варианту 18:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 3 | Подсчёт количества простых чисел на отрезке [A, B] (A, B - натуральные) | Int PrimeCount(int A, int B) | Наивный алгоритм. Проверить делимость текущего числа на все предыдущие числа. | Решето Эратосфена |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 6 | Рассчет значения числа е(основание натурального логарифма) | Float E(int x) | (1 + 1/x) ^ x | Сумма ряда по n от 0 до x, где элементы ряда равны: (1/(n!)) |

**Общие сведения о программе**

1) void \*dlopen(const char \**filename*, int *flag*); dlopen загружает динамическую библиотеку, имя которой указано в строке *filename*, и возвращает прямой указатель на начало динамической библиотеки. Если *filename* не является полным именем файла (т.е. не начинается с "/"), то файл ищется в следующих местах:

2) void \*dlsym(void \**handle*, char \**symbol*); dlopenзагружает динамическую библиотеку, имя которой указано в строке *filename*, и возвращает прямой указатель на начало динамической библиотеки. Если *filename* не является полным именем файла (т.е. не начинается с "/"), то файл ищется в следующих местах:

3) int dlclose(void \*handle*);* dlcloseуменьшает 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. Если динамическая библиотека экпортировала функцию, названную \_fini, то эта функция вызывается перед выгрузкой библиотеки. 

**Общий метод и алгоритм решения**

1) Изучить процесс компиляции программ.

2) Изучить динамические библиотеки.

3) Реализовать две библиотеки согласно заданию.

4) Реализовать две программы для работы с динамическими библиотеками.

**Исходный код**

**first.cpp**

#include <iostream>  
  
extern "C" int PrimeCount(int A, int B);  
extern "C" float E(int x);  
  
int main(){  
 int command;  
 while((std::cout << "Enter command: ") && (std::cin >> command)){  
 if(command == 1){  
 std::cout << "Enter A and B: ";  
 int A, B;  
 std::cin >> A >> B;  
 std::cout << "PrimeCount in [A; B] " << PrimeCount(A, B) << std::endl;  
 }  
 else if(command == 2){  
 int x;  
 std::cout << "Enter precision: ";  
 std::cin >> x;  
 std::cout << "Approximate exponent value " << E(x) << std::endl;  
 }  
 }  
}

**second.cpp**

#include <cstdlib>  
#include <iostream>  
#include <dlfcn.h>  
  
int main(){  
 std::cout << "Enter num library: ";  
 int lib\_num;  
 std::cin >> lib\_num;  
 if(lib\_num < 1 || lib\_num > 2){  
 std::cout << "error lib\n";  
 exit(1);  
 }  
 --lib\_num;  
 int command;  
 const char\* libs[] = {"libd1.so", "libd2.so"};  
 void\* library\_handle;  
 library\_handle = dlopen (libs[lib\_num], RTLD\_LAZY);  
 if(!library\_handle){  
 std::cout << "Error in dlopen\n";  
 exit(1);  
 }  
  
 int (\*PrimeCount)(int A, int B);  
 float (\*E)(int x);  
  
 PrimeCount = (int(\*)(int, int))dlsym(library\_handle, "PrimeCount");  
 E = (float(\*)(int))dlsym(library\_handle, "E");  
  
 std::cout << "Enter command 0, 1 or 2\n";  
 while(std::cin >> command) {  
 switch (command) {  
 case 0:  
 dlclose(library\_handle);  
 lib\_num = (lib\_num + 1) % 2;  
 library\_handle = dlopen(libs[lib\_num], RTLD\_LAZY);  
 if(!library\_handle){  
 std::cout << "Error in dlopen\n";  
 exit(1);  
 }  
 PrimeCount = (int(\*)(int, int))dlsym(library\_handle, "PrimeCount");  
 E = (float(\*)(int))dlsym(library\_handle, "E");  
 std::cout << "Change contract\n";  
 break;  
 case 1:  
 std::cout << "Enter A and B: ";  
 int A, B;  
 std::cin >> A >> B;  
 std::cout << "PrimeCount in [A; B] " << PrimeCount(A, B) << std::endl;  
 break;  
 case 2:  
 int x;  
 std::cout << "Enter precision: ";  
 std::cin >> x;  
 std::cout << "Approximate exponent value " << E(x) << std::endl;  
 break;  
 default:  
 std::cout << "Enter 0, 1 or 2!\n";  
 break;  
 }  
 }  
 dlclose(library\_handle);  
}

**lib1.cpp**

#include <cmath>  
  
extern "C" int PrimeCount(int A, int B);  
extern "C" float E(int x);  
  
int PrimeCount(int A, int B) {  
 int count = 0;  
 if (B < 2)  
 return 0;  
 if (A < 3) {  
 A = 3;  
 ++count;  
 }  
 for (int number = A; number <= B; ++number) {  
 for (int divider = 2; divider < number; ++divider) {  
 if (number % divider == 0)  
 break;  
 if (divider == number - 1)  
 ++count;  
 }  
 }  
 return count;  
}  
  
float E(int x){  
 if(x < 0)  
 return -1;  
 return pow(1.0 + 1.0 / x, x);  
}  
**lib2.cpp**

#include <vector>  
  
extern "C" int PrimeCount(int A, int B);  
extern "C" float E(int x);  
  
int PrimeCount(int A, int B){  
 if (B < 2)  
 return 0;  
 if(A < 2)  
 A = 2;  
 int n = B;  
 std::vector<char> prime(n + 1, true);  
 prime[0] = prime[1] = false;  
 for(int i = 2; i <= n; ++i){  
 if(prime[i]){  
 if(i \* i <= n){  
 for(int j = i \* i; j <= n; j += i){  
 prime[j] = false;  
 }  
 }  
 }  
 }  
 int count = 0;  
 for(int i = A; i <= B; ++i)  
 count += prime[i];  
 return count;  
}  
  
float E(int x) {  
 float ans = 1;  
 if (x < 0)  
 return -1;  
 for(int i = 1; i <= x; ++i){  
 float fact = 1;  
 for(int j = 1; j <= i; ++j)  
 fact \*= j;  
 ans += 1.0 / fact;  
 }  
 return ans;  
}

**Демонстрация работы программы**

Запуск программы:

g++ -fPIC -c lib1.cpp -o d1.o

g++ -shared d1.o -o libd1.so

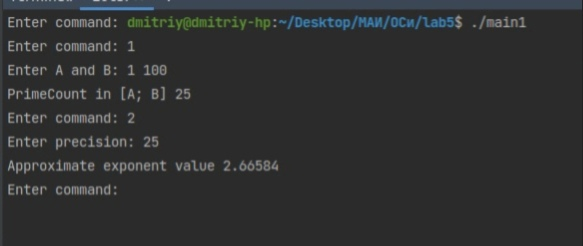
g++ -fPIC -c lib2.cpp -o d2.o

g++ -shared d2.o -o libd2.so

g++ first.cpp -L. -ld1 -o main1 -Wl,-rpath -Wl,.

g++ second.cpp -o main2 -ldl

./main1



./main2



**Выводы**

В данной лабораторной работе я узнал о динамических и статических библиотеках. Узнал их отличия. Динамические библиотеки помогают уменьшить размер исполняемых файлов. Их загрузка упрощает компиляцию. Их можно подключать к программе на этапе линкови или в самой программе.