Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Лабораторная работа №4 по курсу**

**«Операционные системы»**

**ДИНАМИЧЕСКИЕ БИБЛИОТЕКИ**

Студент: Головенко Анатолий Валерьевич

Группа: М8О–212Б–22

Вариант: 18

Преподаватель: Соколов Андрей Алексеевич

Оценка: \_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_

Москва, 2023.

**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.

Вариант 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**

Программа компилируется из файлов compile.cpp и runtime.cpp, также используется заголовочный файл functions.hpp. В программе используются следующие системные вызовы:

1. **dlopen** – загружает динамический общий объект (общую библиотеку) из файла, имя которого указано в строке filename (завершается null) и возвращает непрозрачный описатель на загруженный объект.
2. **dlsym** – функция возвращает адрес, по которому символ расположен в памяти(указывается одним из аргументов).
3. **dlclose** – уменьшает счётчик ссылок на динамически загружаемый общий объект, на который ссылается handle. Если счётчик ссылок достигает нуля, то объект выгружается. Все общие объекты, которые были автоматически загружены при вызове dlopen() для объекта, на который ссылается handle, рекурсивно закрываются таким же способом.

**Общий метод и алгоритм решения**.

Первый вариант исполнения реализуется в программе complile.cpp. При помощи cmake создаются 2 библиотеки с разными версиями имплементации функций. Далее одна из библиотек линкуется к исполняемому файлу.

Второй вариант исполнения реализуется в программе runtime.cpp. В нем поключение библиотек происходит при помощи системных вызовов. Также реализована возможность переключения между разными вариантами имплементации.

Различные имплементации представлены в файлах impl1.cpp и impl2.cpp.

**Основные файлы программы**

**compile.cpp:**

#include <iostream>

#include "functions.hpp"

auto main() -> int

{

std::cout << "This program is linking libraries during compilation. Instructions: " << std::endl

<< "\"1\" - count prime nums" << std::endl

<< "\"2\" - calculating exponent" << std::endl

<< "\"-1\" - exit" << std::endl

<< "Input instruction: ";

int option;

std::cin >> option;

while (option != -1)

{

switch (option)

{

case 1:

int A, B;

std::cin >> A >> B;

std::cout << "The count is: " << PrimeCount(A, B) << std::endl

<< "Input instruction: ";

std::cin >> option;

break;

case 2:

int num;

std::cin >> num;

std::cout << "The E is: " << E(num) << std::endl

<< "Input instruction: ";

std::cin >> option;

break;

default:

option = -1;

}

}

return 0;

}

**runtime.cpp:**

#include <dlfcn.h>

#include <iostream>

auto main() -> int

{

std::cout << "This program is linking libraries during runtime. Instructions: " << std::endl

<< "\"0\" - change implementation" << std::endl

<< "\"1\" - count prime nums" << std::endl

<< "\"2\" - calculating exponent" << std::endl

<< "\"-1\" - exit" << std::endl

<< "Input instruction: ";

int option;

std::cin >> option;

int impl = 1;

auto handle = dlopen("libimpl1.dylib", RTLD\_LAZY);

char \*error;

error = dlerror();

if (error)

{

std::cerr << error << std::endl;

return 1;

}

int (\*prime)(int, int);

prime = reinterpret\_cast<decltype(prime)>(dlsym(handle, "PrimeCount"));

error = dlerror();

if (error)

{

std::cerr << error << std::endl;

return 1;

}

float (\*exp)(int);

exp = reinterpret\_cast<decltype(exp)>(dlsym(handle, "E"));

error = dlerror();

if (error)

{

std::cerr << error << std::endl;

return 1;

}

while (option != -1)

{

switch (option)

{

case 0:

dlclose(handle);

error = dlerror();

if (error)

{

std::cerr << error << std::endl;

return 1;

}

switch (impl)

{

case 1:

impl = 2;

handle = dlopen("libimpl2.dylib", RTLD\_LAZY);

break;

case 2:

int impl = 1;

handle = dlopen("libimpl1.dylib", RTLD\_LAZY);

break;

}

char \*error;

error = dlerror();

if (error)

{

std::cerr << error << std::endl;

return 1;

}

prime = reinterpret\_cast<decltype(prime)>(dlsym(handle, "PrimeCount"));

error = dlerror();

if (error)

{

std::cerr << error << std::endl;

return 1;

}

exp = reinterpret\_cast<decltype(exp)>(dlsym(handle, "E"));

error = dlerror();

if (error)

{

std::cerr << error << std::endl;

return 1;

}

std::cout << "Implementation switched" << std::endl

<< "Input instruction: ";

std::cin >> option;

break;

case 1:

int A, B;

std::cin >> A >> B;

std::cout << "The count is: " << prime(A, B) << std::endl

<< "Input instruction: ";

std::cin >> option;

break;

case 2:

int num;

std::cin >> num;

std::cout << "The E is: " << exp(num) << std::endl

<< "Input instruction: ";

std::cin >> option;

break;

default:

option = -1;

}

}

dlclose(handle);

error = dlerror();

if (error)

{

std::cerr << error << std::endl;

return 1;

}

return 0;

}

**functions.hpp:**

#pragma once

extern "C"

{

int PrimeCount(int A, int B);

}

extern "C"

{

float E(int x);

}

**impl1.cpp:**

#include <iostream>

#include "functions.hpp"

extern "C"

{

bool bruteforce(int n)

{

int i = 2;

while (i \* i <= n)

{

if (n % i == 0)

{

return true;

}

++i;

}

return false;

}

int PrimeCount(int A, int B)

{

int count = 0;

for (int i = A; i <= B; ++i)

{

if (not bruteforce(i))

{

count++;

}

}

return count;

}

}

extern "C"

{

float E(int x)

{

float ans = 1;

for (int i = 1; i < x; ++i)

{

ans \*= (1 + 1.0 / x);

}

return ans;

}

}

**impl2.cpp:**

#include <iostream>

#include "functions.hpp"

extern "C"

{

int PrimeCount(int A, int B)

{

bool sieve[B];

int count = 0;

for (int i = 1; i <= B; i++)

{ // Изначально все числа не вычеркнуты.

sieve[i] = true;

}

for (int i = 2; i <= B; i++)

{

if (sieve[i])

{ // если i не вычеркнуто

if (i >= A)

{

count++;

}

for (int j = i \* i; j <= B; j += i)

{ // вычеркиваем все кратные числа начиная с i^2

sieve[j] = false;

}

}

}

return count;

}

}

extern "C"

{

long long factorial(int x)

{

long long ans = x;

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

{

ans \*= i;

}

return ans;

}

float E(int x)

{

float ans = 1.0;

for (int i = 1; i <= x; ++i)

{

ans += 1.0 / (factorial(i));

}

return ans;

}

}

**CMakeLists.txt:**

cmake\_minimum\_required(VERSION 3.10)

project(Dynamic\_Libraries)

set(CMAKE\_CXX\_STANDARD\_REQUIRED ON)

set(CMAKE\_CXX\_STANDARD 17)

add\_library(impl1 SHARED impl1.cpp)

add\_library(impl2 SHARED impl2.cpp)

add\_executable(comp compile.cpp)

target\_link\_libraries(comp impl1)

add\_executable(runtime runtime.cpp)

**Пример работы**

**Test 1:**

**anatolii@MacBook-Pro-Anatolii ~/D/O/h/b/src (main)> /Users/anatolii/Desktop/Oper\_Syst/hometask\_4/build/src/comp**

This program is linking libraries during compilation. Instructions:

"1" - count prime nums

"2" - calculating exponent

"-1" - exit

Input instruction: 1

10 100

The count is: 21

Input instruction: 1

2 5

The count is: 3

Input instruction: 2

10

The E is: 2.35795

Input instruction: 2

100

The E is: 2.67803

Input instruction: -1

anatolii@MacBook-Pro-Anatolii ~/D/O/h/b/src (main)>

**Test 2:**

**anatolii@MacBook-Pro-Anatolii ~/D/O/h/b/src (main)> /Users/anatolii/Desktop/Oper\_Syst/hometask\_4/build/src/runtime**

This program is linking libraries during runtime. Instructions:

"0" - change implementation

"1" - count prime nums

"2" - calculating exponent

"-1" - exit

Input instruction: 1

2 5

The count is: 3

Input instruction: 2 10

The E is: 2.35795

Input instruction: 2 1000

The E is: 2.71421

Input instruction: 1

2 10

The count is: 4

Input instruction: 0

Implementation switched

Input instruction: 1

2 5

The count is: 3

Input instruction:

1

2 10

The count is: 4

Input instruction: 2

10

The E is: 2.71828

Input instruction: 2

3

The E is: 2.66667

Input instruction: 0

Implementation switched

Input instruction: 0

Implementation switched

Input instruction: 1

50 77

The count is: 6

Input instruction: 1

50 100

The count is: 10

Input instruction: 2

10000

The E is: 2.71787

Input instruction: -1

anatolii@MacBook-Pro-Anatolii ~/D/O/h/b/src (main)>

**Вывод**

Во время выполнения данной работы я познакомился с динамическими библиотеками. Я изучил создание библиотек при помощи CMake, а также мною было изучено подключение библиотек 2 способами: на этапе компиляции и на этапе исполнения программы. Подключение библиотек на этапе компиляции позволяет динамически изменять функции в зависимости от подключенной библиотеки, но замедляет время выполнения, так как требуется время на подключение библиотек.