Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа №4 по курсу

«Операционные системы»

Группа: М8О-209Б-22

Студент: Концебалов О.С.

Преподаватель: Пономарев Н.В.

Оценка: _____

Дата: 09.12.2023

Москва, 2023.

Содержание

Постановка задачи

Требуется создать динамические библиотеки, которые реализуют определенный функционал. Далее использовать данные библиотеки 2-мя способами:

- Во время компиляции (на этапе «линковки»/linking)
- 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.

Общие сведения о программе

Программа состоит из папки с библиотеками third_party, в которой находятся две папки eulers_num и prime_numbers. В каждой из которых есть папки include с заголовочными файлами библиотеки и src с реализацией функций из заголовочных файлов. В них находятся файлы calculate_eulers_num.h , prime_count.h, calculate_eulers_num.cpp и prime_count.cpp соответственно. В корневой папке программы находятся два файла run.cpp – для запуска с использованием библиотек во время компиляции и run_runtime.cpp – для запуска с использованием библиотек во время рантайма. Также есть Makefile для удобной работы с программой.

Общий метод и алгоритм решения

Пользователь запускает одну из версий – во время компиляции или во время рантайма. После этого происходит выбор необходимой версии библиотеки и функции, реализованный с помощью switch-case. Если выбрали во время компиляции, то код просто компилируется и запускается, если выбрали во время рантайма, то создаются указатели на функции, загружаются библиотеки и аналогично считается результат. Для динамического использования библиотек используется <dlfcn.h>.

Код программы

./third_party/eulers_number/include/calculate_eulers_num.h

```
#ifndef CALCULATE_EULERS_NUM_H
#define CALCULATE_EULERS_NUM_H

#include <cmath>
#include <iostream>

namespace numbers::eulers {

float E(int);

float sum_of_series(int x);

}; // namespace number::eulers

#endif // #ifndef CALCULATE_EULERS_NUM_H
```

./third_party/eulers_number/src/calculate_eulers_num.cpp

```
#include "../include/calculate_eulers_num.h"

float numbers::eulers::E(int x)
{
    if (x <= 0) {
        throw std::invalid_argument("E ERROR: given number "
                                    "must be more than zero");
    }

    return pow(1 + 1.0 / x, x);
}

float numbers::eulers::sum_of_series(int x)
{
    if (x <= 0) {
        throw std::invalid_argument("E ERROR: given number "
                                    "must be more than zero");
    }
```

```
    float sum { 1.0 };
    float factorial { 1.0 };

    for (int n = 1; n <= x; ++n) {
        factorial *= n;
        sum += 1.0 / factorial;
    }

    return sum;
}
```

./third_party/prime_numbers/include/prime_count.h

```
#ifndef PRIME_COUNT_H
#define PRIME_COUNT_H

#include <cmath>
#include <iostream>
#include <vector>

namespace numbers::prime {

int naive_prime_count(int, int);

int eratosphene_prime_count(int, int);

}; // namespace numbers::prime

#endif // #ifndef PRIME_COUNT_H
```

./third_party/prime_numbers/src/prime_count.cpp

```
#include "../include/prime_count.h"

int numbers::prime::naive_prime_count(int A, int B)
{
    if (B < A) {
        throw std::invalid_argument("naive_prime_count ERROR:
first num "
                                    "must be less or equal than
second num");
```

```
    }

    int counter { 0 };

    for (int i = A; i <= B; ++i) {
        if (i == 0 || i == 1) {
            continue;
        }

        int count_divider { 0 };
        for (int j = 2; j <= i; ++j) {
            if (i % j == 0) {
                ++count_divider;
            }
        }

        if (count_divider <= 1) {
            ++counter;
        }
    }

    return counter;
}

int numbers::prime::eratosphene_prime_count(int A, int B)
{
    if (B < A) {
        throw std::invalid_argument("naive_prime_count ERROR:
first num "
                                    "must be less or equal than
second num");
    }

    std::vector<bool> is_prime(B + 1, true);

    is_prime[0] = is_prime[1] = false;

    for (int i = 2; i * i <= B; ++i) {
        if (is_prime[i]) {
            for (int j = i * i; j <= B; j += i) {
```

```
                is_prime[j] = false;
            }
        }
    }

    int counter = 0;
    for (int i = A; i <= B; ++i) {
        if (is_prime[i]) {
            ++counter;
        }
    }

    return counter;
}
```

./run.cpp

```
#include <iostream>

#include
"third_party/eulers_number/include/calculate_eulers_num.h"
#include "third_party/prime_numbers/include/prime_count.h"

int main(int argc, char** argv) {
    if (argc < 3 || argc > 4) {
        std::cerr << "Invalid input data!\nExample:\nmake
mode=1 a=1 b=1\nor\nmake run mode=2 a=1" << std::endl;
        return -1;
    }

    int mode = std::atoi(argv[1]);

    if (mode != 1 && mode != 2) {
        std::cerr << "Invalid data!\nValid value of variable
\"mode\" is 1 or 2" << std::endl;
        return -1;
    }

    int a = std::atoi(argv[2]);
    int b = std::atoi(argv[3]);
```

```
    switch (mode) {
        case 1:
            std::cout << "Your choice: primes count" <<
std::endl;
            std::cout << "Naive implement: " <<
numbers::prime::naive_prime_count(a, b) << std::endl;
            std::cout << "Eratosthenes method: " <<
numbers::prime::eratosphene_prime_count(a, b) << std::endl;
            return 0;

        case 2:
            std::cout << "Your choice: calculate Euler's
number" << std::endl;
            std::cout << "(1 + 1 / x) ^ x method: " <<
numbers::eulers::E(a) << std::endl;
            std::cout << "Sum of series method: " <<
numbers::eulers::sum_of_series(a) << std::endl;
            return 0;
  }

}
```

./run_runtime.cpp

```
#include <iostream>
#include <dlfcn.h>

int (*naive_prime_count)(int, int);
int (*eratosphene_prime_count)(int, int);
float (*E)(int);
float (*sum_of_series)(int);

bool init_lib() {
    void* hdl_prime_lib =
dlopen("/home/baronpipistron/MAI_OS/4_Lab/build/libprime_number
s.so", RTLD_LAZY);
    void* hdl_eulers_lib =
dlopen("/home/baronpipistron/MAI_OS/4_Lab/build/libeulers.so",
RTLD_LAZY);

    if (hdl_prime_lib == nullptr) {
```

```
        std::cerr << "init_lib ERROR: hdl_prime_lib is nullptr"
<< std::endl;
        return false;
    }

    if (hdl_eulers_lib == nullptr) {
        std::cerr << "init_lib ERROR: hdl_eulers_lib is
nullptr" << std::endl;
        return false;
    }

    naive_prime_count = (int (*)(int, int))
dlsym(hdl_prime_lib,
"_ZN7numbers5prime17naive_prime_countEii");
    eratosphene_prime_count = (int (*)(int, int))
dlsym(hdl_prime_lib,
"_ZN7numbers5prime23eratosphene_prime_countEii");
    E = (float (*)(int)) dlsym(hdl_eulers_lib,
"_ZN7numbers6eulers1EEi");
    sum_of_series = (float (*)(int)) dlsym(hdl_eulers_lib,
"_ZN7numbers6eulers13sum_of_seriesEi");

    if (naive_prime_count == nullptr) {
        std::cerr << "init_lib ERROR: naive_prime_count is
nullptr" << std::endl;
        return false;
    }

    if (eratosphene_prime_count == nullptr) {
        std::cerr << "init_lib ERROR: eratosphene_prime_count
is nullptr" << std::endl;
        return false;
    }

    if (E == nullptr) {
        std::cerr << "init_lib ERROR: E is nullptr" <<
std::endl;
        return false;
    }
```

```
    if (sum_of_series == nullptr) {
        std::cerr << "init_lib ERROR: sum_of_series if nullptr"
<< std::endl;
        return false;
    }

    return true;
}

int main(int argc, char** argv) {
    if (argc < 3 || argc > 4) {
        std::cerr << "invalid arguments!" << std::endl;
        return -1;
    }

    bool check_flag = init_lib();
        if (check_flag == false) {
        std::cerr << "Error with open libs" << std::endl;
        return -1;
    }

    int mode = std::atoi(argv[1]);
    int a = std::atoi(argv[2]);
    int b = std::atoi(argv[3]);
    int impl_flag = 0;

    switch (mode) {
        case 2:
            std::cout << "Input implementation flag(0 - (1 + 1
/ x) ^ x, oth - sum of series): ";
            std::cin >> impl_flag;

            if (impl_flag == 0) {
                std::cout << "(1 + 1 / x) ^ x method
implementation: ";
                std::cout << E(a) << std::endl;
            } else {
                std::cout << "sum of series method: ";
                std::cout << sum_of_series(a) << std::endl;
             }
```

```
            break;

        case 1:
            std::cout << "Input implementation flag(0 - naive,
oth - eratosphene): ";
            std::cin >> impl_flag;

            if (impl_flag == 0) {
                std::cout << "naive primes count
implementation: ";
                std::cout << naive_prime_count(a, b) <<
std::endl;
            } else {
                std::cout << "eratosthenes primes count
implementation: ";
                std::cout << eratosphene_prime_count(a, b) <<
std::endl;
            }
            break;

        default:
            std::cerr << "Invalid flag of mode!" << std::endl;
            break;
  }

  return 0;
}
```

Использование утилиты strace

cd build; strace -f ./run_runtime 1 0 10=0

execve("./run_runtime", ["./run_runtime", "1", "0", "10=0"], 0x7ffe1b0f6810 /* 64 vars */) = 0

brk(NULL) = 0x564eb9396000

arch_prctl(0x3001 /* ARCH_??? */, 0x7ffe69009ca0) = -1 EINVAL (Invalid argument)

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7fb77b714000

access("/etc/ld.so.preload", R_OK) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3

newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=66003, ...}, AT_EMPTY_PATH) = 0

mmap(NULL, 66003, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7fb77b703000

close(3) = 0

openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libstdc++.so.6", O_RDONLY|O_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0\0"..., 832) = 832

newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=2260296, ...}, AT_EMPTY_PATH) = 0

mmap(NULL, 2275520, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7fb77b400000

mprotect(0x7fb77b49a000, 1576960, PROT_NONE) = 0

mmap(0x7fb77b49a000, 1118208, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x9a000) = 0x7fb77b49a000

mmap(0x7fb77b5ab000, 454656, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1ab000) = 0x7fb77b5ab000

mmap(0x7fb77b61b000, 57344, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x21a000) = 0x7fb77b61b000

mmap(0x7fb77b629000, 10432, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7fb77b629000

close(3) = 0

openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P\237\2\0\0\0\0"..., 832) = 832

pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784

pread64(3, "\4\0\0\0 \0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0"..., 48, 848) = 48

pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0 =\340\2563\265?\356\25x\261\27\313A#\350"..., 68, 896) = 68

newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=2216304, ...}, AT_EMPTY_PATH) = 0

pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784

mmap(NULL, 2260560, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7fb77b000000

mmap(0x7fb77b028000, 1658880, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x7fb77b028000

mmap(0x7fb77b1bd000, 360448, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1bd000) = 0x7fb77b1bd000

mmap(0x7fb77b215000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x214000) = 0x7fb77b215000

mmap(0x7fb77b21b000, 52816, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7fb77b21b000

close(3) = 0

openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libm.so.6", O_RDONLY|O_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0\0"..., 832) = 832

newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=940560, ...}, AT_EMPTY_PATH) = 0

mmap(NULL, 942344, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7fb77b319000

mmap(0x7fb77b327000, 507904, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0xe000) = 0x7fb77b327000

mmap(0x7fb77b3a3000, 372736, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x8a000) = 0x7fb77b3a3000

mmap(0x7fb77b3fe000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0xe4000) = 0x7fb77b3fe000

close(3) = 0

openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libgcc_s.so.1", O_RDONLY|O_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"..., 832) = 832

newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=125488, ...}, AT_EMPTY_PATH) = 0

mmap(NULL, 127720, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7fb77b6e3000

mmap(0x7fb77b6e6000, 94208, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x3000) = 0x7fb77b6e6000

mmap(0x7fb77b6fd000, 16384, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1a000) = 0x7fb77b6fd000

mmap(0x7fb77b701000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1d000) = 0x7fb77b701000

close(3) = 0

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7fb77b6e1000

arch_prctl(ARCH_SET_FS, 0x7fb77b6e23c0) = 0

set_tid_address(0x7fb77b6e2690) = 28718

set_robust_list(0x7fb77b6e26a0, 24) = 0

rseq(0x7fb77b6e2d60, 0x20, 0, 0x53053053) = 0

mprotect(0x7fb77b215000, 16384, PROT_READ) = 0

mprotect(0x7fb77b701000, 4096, PROT_READ) = 0

mprotect(0x7fb77b3fe000, 4096, PROT_READ) = 0

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7fb77b6df000

mprotect(0x7fb77b61b000, 45056, PROT_READ) = 0

mprotect(0x564eb8638000, 4096, PROT_READ) = 0

mprotect(0x7fb77b74e000, 8192, PROT_READ) = 0

prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0

munmap(0x7fb77b703000, 66003) = 0

getrandom("\x2d\xf1\x96\x61\xb2\xa4\xb7\xaf", 8, GRND_NONBLOCK) = 8

brk(NULL) = 0x564eb9396000

brk(0x564eb93b7000) = 0x564eb93b7000

futex(0x7fb77b62977c, FUTEX_WAKE_PRIVATE, 2147483647) = 0

openat(AT_FDCWD, "/home/baronpipistron/MAI_OS/4_Lab/build/libprime_numbers.so", O_RDONLY|O_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"..., 832) = 832

newfstatat(3, "", {st_mode=S_IFREG|0775, st_size=33368, ...}, AT_EMPTY_PATH) = 0

mmap(NULL, 29192, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7fb77b70c000

mmap(0x7fb77b70f000, 8192, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x3000) = 0x7fb77b70f000

mmap(0x7fb77b711000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x5000) = 0x7fb77b711000

mmap(0x7fb77b712000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x5000) = 0x7fb77b712000

close(3) = 0

mprotect(0x7fb77b712000, 4096, PROT_READ) = 0

openat(AT_FDCWD, "/home/baronpipistron/MAI_OS/4_Lab/build/libeulers.so", O_RDONLY|O_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"..., 832) = 832

newfstatat(3, "", {st_mode=S_IFREG|0775, st_size=16640, ...}, AT_EMPTY_PATH) = 0

mmap(NULL, 16496, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7fb77b707000

mmap(0x7fb77b708000, 4096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x7fb77b708000

mmap(0x7fb77b709000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7fb77b709000

mmap(0x7fb77b70a000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7fb77b70a000

close(3) = 0

mprotect(0x7fb77b70a000, 4096, PROT_READ) = 0

newfstatat(1, "", {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...}, AT_EMPTY_PATH) = 0

write(1, "Input implementation flag(0 - na"..., 57Input implementation flag(0 - naive, oth - eratosphene):) = 57

newfstatat(0, "", {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...}, AT_EMPTY_PATH) = 0

read(0, 0

"0\n", 1024) = 2

write(1, "naive primes count implementatio"..., 37naive primes count implementation: 4

) = 37

lseek(0, -1, SEEK_CUR) = -1 ESPIPE (Illegal seek)

exit_group(0) = ?

+++ exited with 0 +++

Демонстрация работы программы

```
baronpipistron@BaronPIpistron:~/MAI_OS/4_Lab$ make run-comptime mode=1 a=0 b=10
cd build; ./run 1 0 10=0
Your choice: primes count
Naive implement: 4
Eratosthenes method: 4
baronpipistron@BaronPIpistron:~/MAI_OS/4_Lab$ make run-comptime mode=2 a=100
cd build; ./run 2 100 =0
Your choice: calculate Euler's number
(1 + 1 / x) ^ x method: 2.70481
Sum of series method: 2.71828
baronpipistron@BaronPIpistron:~/MAI_OS/4_Lab$ make run-runtime mode=1 a=0 b=10
cd build; ./run_runtime 1 0 10=0
Input implementation flag(0 - naive, oth - eratosphene): 2
eratosthenes primes count implementation: 4
baronpipistron@BaronPIpistron:~/MAI_OS/4_Lab$ make run-runtime mode=2 a=100
cd build; ./run_runtime 2 100 =0
Input implementation flag(0 - (1 + 1 / x) ^ x, oth - sum of series): 0
(1 + 1 / x) ^ x method implementation: 2.70481
baronpipistron@BaronPIpistron:~/MAI_OS/4_Lab$
```

Вывод

Довольно интересная лабораторная, которая учит работе с динамическими и статическими библиотеками. Очень хорошо показывает самую основу этих процессов, смотрим на все изнутри, а не из вершины абстракции. Выполнять было не очень сложно, есть довольно хорошие статьи на эту тему. Самым сложным оказалось, что необходимо добавить свои библиотеки в переменные окружения. Казалось бы, очевидно, но долго не мог понять почему не работает. Также думал, что в функции dlsym вторым аргументом необходимо указывать свои кастомные имена, а оказалось, что необходимо имя, которое дала функции сама машина. Тоже долго с этим разбирался. Считаю, что лаба довольно полезная и может пригодится в дальнейшей работе.