МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ (НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Институт №8 «Компьютерные науки и прикладная математика»
Кафедра 806 «Вычислительная математика и программирование»

Лабораторная работа №4
по курсу «Операционные системы»

Выполнила: Власова Е.Р.
Группа: М8О-208БВ-24
Преподаватель: Миронов Е.С.

Москва, 2025

Условие:

Требуется создать динамические библиотеки, которые реализуют заданный вариантом 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 информацию полученные на этапе компиляции; тестовая программа (программа №2), которая загружает библиотеки, используя только их относительные пути и контракты. Провести анализ двух типов использования библиотек. Пользовательский ввод для обоих программ должен быть организован следующим образом: 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»; «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; «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.

Цель работы:

Целью является приобретение практических навыков в создании динамических библиотек, создании программ, которые используют функции динамических библиотек.

Вариант:

9

№	Описание	Сигнатура	Реализация 1	Реализация 2
§	§ Рассчет производной функции cos(x) в точке A с приращением deltaX	§ Float Derivative(float A, float deltaX)	f'(x) = (f(A + deltaX) – f(A))/deltaX	f'(x) = (f(A + deltaX) – f(A-deltaX))/(2*deltaX)
§	§ 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

Задание:

Метод решения:

Программа представляет собой демострацию различных подходов реализации динамических библиотек. Она демонстрирует два подхода к вычислениям: первая программа (fist) статически включает реализации функций для вычисления производной функции cos(x) и подсчета простых чисел, вторая программа (second) динамически загружает разные реализации этих же функций (формула вперед/центральная разность для производной, наивный алгоритм/решето для простых чисел) через библиотеки .so, позволяя выбирать алгоритм во время выполнения.

Описание программы:

Программа состоит из следующих файлов: derivative-center.c, derivative-forward.c, primecount-naive.c, primecount-sieve.c, first.c, second.c

Я использую следующие системные вызовы: execve, brk, mmap, openat, read, write, close, mprotect, exit-group

Выводы:

В этой лабораторной работе я разобралась с двумя разными способами сборки программ. Первый способ — статический, когда весь код компилируется в один файл. Второй — динамический, когда основные части программы загружаются отдельно, как дополнительные модули.

Мне понравилось, как одна и та же задача решается по-разному. В первом случае всё просто и собрано в кучу, во втором — гибко, можно менять части программы на ходу.

Я научилась создавать .so-библиотеки, работать с dlopen, dlsym — это функции для загрузки кода во время работы программы. Динамиче-

ская загрузка оказалась удобной штукой. Так можно делать программы с плагинами или менять алгоритмы без пересборки. В общем, работа показала, что есть не только один способ собрать программу, и иногда гибкость важнее простоты.

Исходная программа:

derivative_center.c

```
#include <math.h>

float Derivative(float A, float deltaX) {
    float f_A_plus = cosf(A + deltaX);
    float f_A_minus = cosf(A - deltaX);
    return (f_A_plus - f_A_minus) / (2 * deltaX);
}
```

derivative_forward.c

```
#include <math.h>

float Derivative(float A, float deltaX) {
    float f_A_plus = cosf(A + deltaX);
    float f_A = cosf(A);
    return (f_A_plus - f_A) / deltaX;
}
```

primecount_naive.c

```
#include <math.h>

int isPrime(int n) {
    if (n < 2) return 0;
    if (n == 2) return 1;
    if (n % 2 == 0) return 0;

    int limit = (int)sqrt(n);
    for (int i = 3; i <= limit; i += 2) {
        if (n % i == 0) return 0;
    }
    return 1;
}

int PrimeCount(int A, int B) {
    int count = 0;
    for (int i = A; i <= B; i++) {
        if (isPrime(i)) {
            count++;
        }
    }
```

```
    return count;
}
```

primecount_sieve.c

```
#include <stdlib.h>
#include <string.h>

int PrimeCount(int A, int B) {
    if (B < 2) return 0;
    if (A < 2) A = 2;

    int size = B + 1;
    char* sieve = (char*)malloc(size * sizeof(char));
    if (!sieve) return -1;

    memset(sieve, 1, size);
    sieve[0] = sieve[1] = 0;

    for (int i = 2; i * i <= B; i++) {
        if (sieve[i]) {
            for (int j = i * i; j <= B; j += i) {
                sieve[j] = 0;
            }
        }
    }

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

    free(sieve);
    return count;
}
```

first.c

```
#include <stdio.h>

#include "include/derivative_forward.c"
```

```
#include "include/primecount_naive.c"

void solve() {
    int number_of_function;
    scanf("%d", &number_of_function);

    if (number_of_function == 1) {
        float A, deltaX;
        scanf("%f %f", &A, &deltaX);

        float result = Derivative(A, deltaX);
        printf("Derivative of cos(x) at x = %f is approximately %f\n", A, result);
    } else {
        int A, B;
        scanf("%d %d", &A, &B);

        int result = PrimeCount(A, B);
        printf("Prime numbers count in [%d, %d] = %d\n", A, B, result);
    }
}

int main() {
    solve();
    return 0;
}
```

second.c

```
#include <dlfcn.h>
#include <stdio.h>
#include <stdlib.h>

float (*Derivative)(float, float);
int (*PrimeCount)(int, int);

void* handle_derivative;
void* handle_primecount;

void solve() {
    int number_of_function;
    scanf("%d", &number_of_function);
```

```
    if (number_of_function == 1) {
        float A, deltaX;
        scanf("%f %f", &A, &deltaX);

        float result = (*Derivative)(A, deltaX);
        printf("Derivative of cos(x) = %f\n", result);
    } else {
        int A, B;
        scanf("%d %d", &A, &B);

        int result = (*PrimeCount)(A, B);
        printf("Prime numbers count = %d\n", result);
    }
}

void load_libraries() {
    int realization;
    scanf("%d", &realization);

    if (realization == 0) {
        handle_derivative = dlopen("./libraries/libderivative_forward.so", RTLD_LAZY)
        handle_primecount = dlopen("./libraries/libprimecount_naive.so", RTLD_LAZY);
    } else {
        handle_derivative = dlopen("./libraries/libderivative_central.so",
        RTLD_LAZY);
        handle_primecount = dlopen("./libraries/libprimecount_sieve.so", RTLD_LAZY);
    }

    if (!handle_derivative) {
        fprintf(stderr, "Error loading derivative library: %s\n", dlerror());
        exit(1);
    }
    if (!handle_primecount) {
        fprintf(stderr, "Error loading primecount library: %s\n", dlerror());
        exit(1);
    }
    dlerror();

    Derivative = dlsym(handle_derivative, "Derivative");
    PrimeCount = dlsym(handle_primecount, "PrimeCount");
```

```
    char* error = dlerror();
    if (error != NULL) {
        fprintf(stderr, "Error loading symbols: %s\n", error);
        exit(1);
    }
}

void close_libraries() {
    dlclose(handle_derivative);
    dlclose(handle_primecount);
}

int main() {
    load_libraries();
    solve();
    close_libraries();
    return 0;
} return 0;
}
```

Strace

```
strace ./fist

execve("./fist", ["./fist"], 0x7ffc7b1a6d80 /* 58 vars */) = 0
brk(NULL)                                       = 0x55a1a2c2e000
arch_prctl(0x3001 /* ARCH_??? */, 0x7fff8808f4e0) = -1 EINVAL
(Invalid argument)
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE
|MAP_ANONYMOUS,
-1, 0) = 0x7f3e2b5a7000
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=118366, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 118366, PROT_READ,
MAP_PRIVATE, 3, 0) = 0x7f3e2b5890
00
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"..., 832) = 832
```

```
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=1769288, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 1783040, PROT_READ, MAP_PRIVATE|
MAP_DENYWRITE, 3, 0)
= 0x7f3e2b3e0000
mmap(0x7f3e2b404000, 1359872, PROT_READ|
PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x24000) = 0x7f3e2b404000
mmap(0x7f3e2b545000, 299008, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x165000) = 0x7f3e2b545000
mmap(0x7f3e2b590000, 24576, PROT_READ|
PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x1af000) = 0x7f3e2b590000
mmap(0x7f3e2b596000, 13440, PROT_READ
|PROT_WRITE, MAP_PRIVATE|MAP_FIXED
|MAP_ANONYMOUS, -1, 0) =
0x7f3e2b596000
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\0\0\
0\0\0\0\0\0\0"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=2140920, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|
MAP_ANONYMOUS, -1, 0) = 0x7f3e2b587000
mmap(NULL, 2160800, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0)
= 0x7f3e2b1d4000
mmap(0x7f3e2b1f6000, 1654784, PROT_READ|
PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x22000) = 0x7f3e2b1f6000
mmap(0x7f3e2b38c000, 360448, PROT_READ,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1b8000) = 0x7f3e2b38c000
mmap(0x7f3e2b3e5000, 24576,
PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x210000) = 0x7f3e2b3e5000
mmap(0x7f3e2b3eb000, 52832, PROT_READ|
PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f3e2b3eb000
close(3)                                = 0
mprotect(0x7f3e2b3e5000, 16384, PROT_READ) = 0
mprotect(0x7f3e2b590000, 16384, PROT_READ) = 0
mprotect(0x7f3e2b5a9000, 8192, PROT_READ) = 0
munmap(0x7f3e2b589000, 118366)          = 0
```

```
read(0, "1\n", 1024)                    = 2
read(0, "1.5 0.001\n", 1024)           = 10
write(1, "Derivative of cos(x) at x = 1.5"..., 71Derivative o
f cos(x) at x = 1.500000 is approximately -0.997495
) = 71
exit_group(0)                           = ?
+++ exited with 0 +++
```

```
strace ./second
execve("./second", ["./second"], 0x7fff96e82ca0 /* 58 vars */)
= 0
brk(NULL)                               = 0x55963d3c1000
arch_prctl(0x3001 /* ARCH_??? */, 0x7fffd96a93d0) = -1 EINVAL
(Invalid argument)
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS,
-1, 0) = 0x7f1d5c84b000
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=118366, ...},
AT_EMPTY_PATH) = 0
mmap(NULL, 118366, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f1d5c82d000
close(3)                                = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libdl.so.2", 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=18856, ...},
AT_EMPTY_PATH) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|
MAP_ANONYMOUS, -1, 0) = 0x7f1d5c849000
mmap(NULL, 20752, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE,
3, 0) = 0x7f1d5c824000
mmap(0x7f1d5c825000, 8192, PROT_READ|
PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x1000) = 0x7f1d5c825000
mmap(0x7f1d5c827000, 8192, PROT_READ, MAP_PRIVATE|MAP_FIXED|
MAP_DENYWRITE, 3, 0x3000)
= 0x7f1d5c827000
mmap(0x7f1d5c828000, 4096,
PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED
```

```
|MAP_DENYWRITE, 3, 0x3000) =
0x7f1d5c828000
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\3\0>\0
\1\0\0\0\0\0\0\0\0\0\0"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0755,
st_size=2140920, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 2160800, PROT_READ, MAP_PRIVATE|
MAP_DENYWRITE, 3, 0)
= 0x7f1d5c622000
mmap(0x7f1d5c644000, 1654784, PROT_READ
|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x22000) = 0x7f1d5c644000
mmap(0x7f1d5c7da000, 360448, PROT_READ,
MAP_PRIVATE|MAP_FIXED
|MAP_DENYWRITE, 3, 0x1b8000) = 0x7f1d5c7da000
mmap(0x7f1d5c833000, 24576, PROT_READ|
PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x210000) = 0x7f1d5c833000
mmap(0x7f1d5c839000, 52832, PROT_READ|
PROT_WRITE, MAP_PRIVATE|MAP_FIXED|
MAP_ANONYMOUS, -1, 0) = 0x7f1d5c839000
close(3)                                = 0
mprotect(0x7f1d5c833000, 16384, PROT_READ) = 0
mprotect(0x7f1d5c828000, 4096, PROT_READ) = 0
mprotect(0x7f1d5c84d000, 8192, PROT_READ) = 0
munmap(0x7f1d5c82d000, 118366)          = 0
read(0, "0\n", 1024)                    = 2
read(0, "1\n", 1024)                    = 2
read(0, "1.5 0.001\n", 1024)           = 10
openat(AT_FDCWD, "./libraries/libderivative_forward.so", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\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=16232, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|
MAP_ANONYMOUS, -1, 0) = 0x7f1d5c847000
mmap(NULL, 18448, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE,
3, 0) = 0x7f1d5c81e000
mmap(0x7f1d5c81f000, 4096, PROT_READ|
PROT_EXEC, MAP_PRIVATE|MAP_FIXED|
```

```
MAP_DENYWRITE,
3, 0x1000) = 0x7f1d5c81f000
mmap(0x7f1d5c820000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED
|MAP_DENYWRITE, 3, 0x2000)
= 0x7f1d5c820000
mmap(0x7f1d5c821000, 4096, PROT_READ|
PROT_WRITE, MAP_PRIVATE
|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000)
= 0x7f1d5c821000
close(3)                                = 0
openat(AT_FDCWD, "./libraries/libprimecount_naive.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"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0775, st_size=16272, ...},
AT_EMPTY_PATH) = 0
mmap(NULL, 8192, PROT_READ|
PROT_WRITE,
MAP_PRIVATE|
MAP_ANONYMOUS, -1, 0) = 0x7f1d5c845000
mmap(NULL, 18456, PROT_READ, MAP_PRIVATE|
MAP_DENYWRITE, 3, 0)
= 0x7f1d5c819000
mmap(0x7f1d5c81a000, 4096, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|
MAP_DENYWRITE, 3,
0x1000) = 0x7f1d5c81a000
mmap(0x7f1d5c81b000, 4096,
PROT_READ, MAP_PRIVATE|MAP_FIXED|
MAP_DENYWRITE, 3,
0x2000) = 0x7f1d5c81b000
mmap(0x7f1d5c81c000, 4096, PROT_READ|
PROT_WRITE, MAP_PRIVATE
|MAP_FIXED|MAP_DENYWRITE, 3,
0x2000) = 0x7f1d5c81c000
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"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=1769288, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 1783040, PROT_READ, MAP_PRIVATE|
MAP_DENYWRITE, 3, 0) = 0x7f1d5c478000
```

```
mmap(0x7f1d5c49c000, 1359872, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x24000) = 0x7f1d5c49c000
mmap(0x7f1d5c5dd000, 299008, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x165000) = 0x7f1d5c5dd000
mmap(0x7f1d5c628000, 24576, PROT_READ|
PROT_WRITE, MAP_PRIVATE|MAP_FIXED|
MAP_DENYWRITE, 3, 0x1af000)
= 0x7f1d5c628000
mmap(0x7f1d5c62e000, 13440, PROT_READ|
PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0)
= 0x7f1d5c62e000
close(3)                                = 0
write(1, "Derivative of cos(x) = -0.997495"...,
39Derivative of cos(x) = -0.997495
) = 39
exit_group(0)                           = ?
+++ exited with 0 +++
```