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»**

Группа: М8О-206Б-22

Студент: Ларин И.А.

Преподаватель: Миронов Е.С.

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата: 14.11.2023

Москва, 2023

**Постановка задачи**

**Вариант 14.**

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;

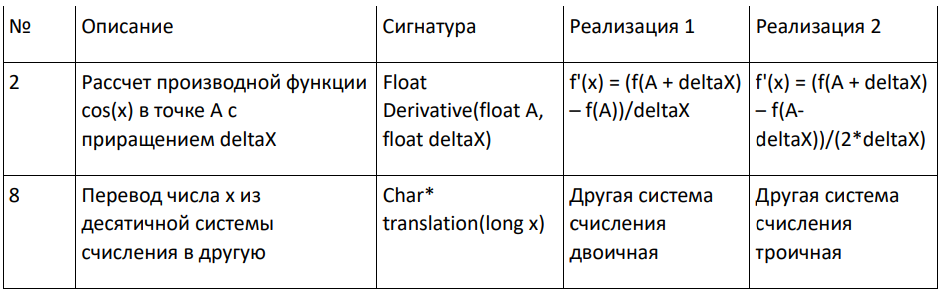
Тестовая программа (программа №1), которая используют одну из библиотек, используя знания полученные на этапе компиляции;

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:

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. ****

**Общий метод и алгоритм решения**

Используемые команды в программе:

1. **void \*dlopen(const char \*filename, int flag)** – загружает динамическую библиотеку, имя которой указано в строке filename, и возвращает прямой указатель на начало загруженной библиотеки;

2. **const char \*dlerror(void)** – возвращает строку символов с нулевым конечком, которая описывает последнюю ошибку, произошедшую во время обработки динамического связывания; если с момента последнего вызова dlerror() ошибок динамического связывания не было, dlerror() вернет NULL;

3. **void \*dlsym(void \*handle, char \*symbol)** – принимает указатель на начало загруженной библиотеки, возвращаемой dlopen(), и имя символа, и возвращает адрес, по которому этот символ загружается в память;

4. **int dlclose(void \*handle)** – уменьшает 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.

В программе с использованием статических библиотек мы вводим значение 1 или 2, если введём 1, то выполняется первая функция, а именно производная, а если ввести 2 то производится перевод в другую систему счисления. В реализации с динамическими библиотеками мы выбираем контракт, по этому контакту используется скомпилированная динамическая библиотека. Получаем нужные нам функции из этой библиотеки с помощью функций dlsym(). И в рантайме используем нужные нам функции. В отличии от статической реализации здесь ещё есть функция ноль, с помощь которой мы можем менять динамические библиотеки. Для этого сы закрываем старую и выполняем такие же действия с новой.

**Код программы**

First\_realization.c

#include "realization.h"

#include <stdlib.h>

double power(double base, int exponent) {

    double result = 1;

    for (int i = 0; i < abs(exponent); i++) {

        result \*= base;

    }

    if (exponent < 0) {

        return 1 / result;

    } else {

        return result;

    }

}

double factorial(int n) {

    if (n == 0) {

        return 1;

    } else {

        return n \* factorial(n - 1);

    }

}

double cosx(double x) {

    double result = 0;

    for (int i = 0; i < 10; i++) {

        result += power(-1, i) \* power(x, 2 \* i) / factorial(2 \* i);

    }

    return result;

}

float derivative(float point, float increment) {

  return (cosx(point + increment) - cosx(point)) / increment;

}

char\* translation(long num) {

  char\* result = (char\*)malloc(sizeof(char));

  int index = 0;

  while(num > 0) {

    result[index] = num % 2 + '0';

    index += 1;

    num /= 2;

    result = realloc(result, (index + 1) \* sizeof(char));

  }

  result[index] = '\0';

  char symbol;

  for(int permutation\_index = 0; permutation\_index < index / 2; ++permutation\_index) {

    symbol = result[permutation\_index];

    result[permutation\_index] = result[index - permutation\_index - 1];

    result[index - permutation\_index - 1] = symbol;

  }

  return result;

}

Second\_realization.c

#include "realization.h"

#include <stdlib.h>

double power(double base, int exponent) {

    double result = 1;

    for (int i = 0; i < abs(exponent); i++) {

        result \*= base;

    }

    if (exponent < 0) {

        return 1 / result;

    } else {

        return result;

    }

}

double factorial(int n) {

    if (n == 0) {

        return 1;

    } else {

        return n \* factorial(n - 1);

    }

}

double cosx(double x) {

    double result = 0;

    for (int i = 0; i < 10; i++) {

        result += power(-1, i) \* power(x, 2 \* i) / factorial(2 \* i);

    }

    return result;

}

float derivative(float point, float increment) {

  return (cosx(point + increment) - cosx(point - increment)) / (2 \* increment);

}

char\* translation(long num){

  char\* result = (char\*)malloc(sizeof(char));

  int index = 0;

  while(num > 0) {

    result[index] = num % 3 + '0';

    index += 1;

    num /= 3;

    result = realloc(result, (index + 1) \* sizeof(char));

  }

  result[index] = '\0';

  char symbol;

  for(int permutation\_index = 0; permutation\_index < index / 2; ++permutation\_index) {

    symbol = result[permutation\_index];

    result[permutation\_index] = result[index - permutation\_index - 1];

    result[index - permutation\_index - 1] = symbol;

  }

  return result;

}

Realization.h

#ifndef REALIZATION\_H

#define REALIZATION\_H

extern float derivative(float point, float increment);

extern char\* translation(long num);

#endif

Static\_main.c

#include "realization.h"

#include <stdio.h>

int main() {

  int c = 0;

  while(scanf("%d", &c) != EOF) {

      if(c == 1){

        float point, increment;

        if(scanf("%f %f", &point, &increment) == 2) {

          printf("%.6f\n", derivative(point, increment));

        }

      }

      else if(c ==2){

        long numeric;

        if(scanf("%ld", &numeric) == 1) {

          printf("%s\n", translation(numeric));

        }

      }

      else{

        printf("Invalid\n");

      }

    }

  return 0;

}

Dynamic\_main.c

#include <stdio.h>

#include <stdlib.h>

#include <dlfcn.h>

int contr = 1;

const char\* first\_library\_name = "libfirst.so";

const char\* second\_library\_name = "libsecond.so";

float (\*derivative)(float, float) = NULL;

char\* (\*translation)(long) = NULL;

void\* lib\_handle = NULL;

void load\_lib(int contr) {

  const char\* name;

    if(contr == 1){

      name = first\_library\_name;

    }

    else if(contr == 2){

      name = second\_library\_name;

    }

  lib\_handle = dlopen(name, RTLD\_LAZY);

  if (lib\_handle == NULL) {

    perror("dlopen");

    exit(EXIT\_FAILURE);

  }

}

void load\_contract() {

  load\_lib(contr);

  derivative = dlsym(lib\_handle, "derivative");

  translation = dlsym(lib\_handle, "translation");

}

void change\_contract() {

  dlclose(lib\_handle);

    if(contr == 1){

      contr = 2;

    }

    else if (contr == 2){

      contr = 1;

    }

  load\_contract();

}

int main() {

  load\_contract();

  int command = 0;

  while (scanf("%d", &command) != EOF) {

      if(command == 0){

        change\_contract();

        printf("Contract has been changed\n");

          if(contr == 1) {

            printf("Contract is first\n");

          }

          else if(contr == 2){

            printf("Contract is second\n");

          }

        }

      else if(command == 1){

        float point, increment;

        if(scanf("%f %f", &point, &increment) == 2) {

          printf("%.6f\n", derivative(point, increment));

        }

      }

      else if (command == 2){

        long numeric;

        if(scanf("%ld", &numeric) == 1) {

          printf("Translation from 10 to ");

            if(contr == 1){

              printf("2");

            }

            else if (contr == 2){

              printf("3");

            }

          }

          printf(" number system: %s\n", translation(numeric));

        }

      else

        printf("Invalid request\n");

  }

  return 0;

}

**Протокол работы программы**

**Тестирование:**

lya@ilya-ilyal:~/osi4laba$ ./program1

1 1 5

0.083695

2 8

1000

ilya@ilya-ilyal:~/osi4laba$ ./program2

1 1 5

0.161242

2 9

100

ilya@ilya-ilyal:~/osi4laba$ ./programd

1 1 5

0.083695

2 8

Translation from 10 to 2 number system: 1000

0

Contract has been changed

Contract is second

1 1 5

0.161242

2 9

Translation from 10 to 3 number system: 100

0

Contract has been changed

Contract is first

1 1 5

0.083695

2 8

Translation from 10 to 2 number system: 1000

1 1 9

-3.886546

**Strace:**

ilya@ilya-ilyal:~/osi4laba$ strace -f ./program1

execve("./program1", ["./program1"], 0x7ffcf29a51a8 /\* 59 vars \*/) = 0

brk(NULL) = 0x55c3777ad000

arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7fffcfdeb330) = -1 EINVAL (Недопустимый аргумент)

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f2c1f59d000

access("/etc/ld.so.preload", R\_OK) = -1 ENOENT (Нет такого файла или каталога)

**openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 3**

newfstatat(3, "", {st\_mode=S\_IFREG|0644, st\_size=53247, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 53247, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7f2c1f590000

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\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\0"..., 48, 848) = 48

pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\244;\374\204(\337f#\315I\214\234\f\256\271\32"..., 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) = 0x7f2c1f200000

mmap(0x7f2c1f228000, 1658880, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x28000) = 0x7f2c1f228000

mmap(0x7f2c1f3bd000, 360448, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1bd000) = 0x7f2c1f3bd000

mmap(0x7f2c1f415000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x214000) = 0x7f2c1f415000

mmap(0x7f2c1f41b000, 52816, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f2c1f41b000

close(3) = 0

mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f2c1f58d000

arch\_prctl(ARCH\_SET\_FS, 0x7f2c1f58d740) = 0

set\_tid\_address(0x7f2c1f58da10) = 27942

set\_robust\_list(0x7f2c1f58da20, 24) = 0

rseq(0x7f2c1f58e0e0, 0x20, 0, 0x53053053) = 0

mprotect(0x7f2c1f415000, 16384, PROT\_READ) = 0

mprotect(0x55c376b6d000, 4096, PROT\_READ) = 0

mprotect(0x7f2c1f5d7000, 8192, PROT\_READ) = 0

prlimit64(0, RLIMIT\_STACK, NULL, {rlim\_cur=8192\*1024, rlim\_max=RLIM64\_INFINITY}) = 0

munmap(0x7f2c1f590000, 53247) = 0

newfstatat(0, "", {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0x1), ...}, AT\_EMPTY\_PATH) = 0

getrandom("\x21\xd9\x11\x65\x8a\x62\xcf\x69", 8, GRND\_NONBLOCK) = 8

brk(NULL) = 0x55c3777ad000

brk(0x55c3777ce000) = 0x55c3777ce000

read(0, 1 1 5

"1 1 5\n", 1024) = 6

newfstatat(1, "", {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0x1), ...}, AT\_EMPTY\_PATH) = 0

write(1, "0.083695\n", 90.083695

) = 9

read(0, 2 8

"2 8\n", 1024) = 4

write(1, "1000\n", 51000

) = 5

read(0, "", 1024) = 0

exit\_group(0) = ?

+++ exited with 0 +++

ilya@ilya-ilyal:~/osi4laba$ strace -f ./program2

execve("./program2", ["./program2"], 0x7ffcea3ba748 /\* 59 vars \*/) = 0

brk(NULL) = 0x56517a474000

arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7ffc72a29920) = -1 EINVAL (Недопустимый аргумент)

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f82441f9000

access("/etc/ld.so.preload", R\_OK) = -1 ENOENT (Нет такого файла или каталога)

**openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 3**

newfstatat(3, "", {st\_mode=S\_IFREG|0644, st\_size=53247, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 53247, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7f82441ec000

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\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\0"..., 48, 848) = 48

pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\244;\374\204(\337f#\315I\214\234\f\256\271\32"..., 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) = 0x7f8243e00000

mmap(0x7f8243e28000, 1658880, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x28000) = 0x7f8243e28000

mmap(0x7f8243fbd000, 360448, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1bd000) = 0x7f8243fbd000

mmap(0x7f8244015000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x214000) = 0x7f8244015000

mmap(0x7f824401b000, 52816, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f824401b000

close(3) = 0

mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f82441e9000

arch\_prctl(ARCH\_SET\_FS, 0x7f82441e9740) = 0

set\_tid\_address(0x7f82441e9a10) = 28249

set\_robust\_list(0x7f82441e9a20, 24) = 0

rseq(0x7f82441ea0e0, 0x20, 0, 0x53053053) = 0

mprotect(0x7f8244015000, 16384, PROT\_READ) = 0

mprotect(0x56517a40c000, 4096, PROT\_READ) = 0

mprotect(0x7f8244233000, 8192, PROT\_READ) = 0

prlimit64(0, RLIMIT\_STACK, NULL, {rlim\_cur=8192\*1024, rlim\_max=RLIM64\_INFINITY}) = 0

munmap(0x7f82441ec000, 53247) = 0

newfstatat(0, "", {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0x1), ...}, AT\_EMPTY\_PATH) = 0

getrandom("\x71\x25\x35\xb0\xb5\xb1\x17\x52", 8, GRND\_NONBLOCK) = 8

brk(NULL) = 0x56517a474000

brk(0x56517a495000) = 0x56517a495000

read(0, 1 1 8

"1 1 8\n", 1024) = 6

newfstatat(1, "", {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0x1), ...}, AT\_EMPTY\_PATH) = 0

write(1, "-0.366912\n", 10-0.366912

) = 10

read(0, 1 1 5

"1 1 5\n", 1024) = 6

write(1, "0.161242\n", 90.161242

) = 9

read(0, 2 9

"2 9\n", 1024) = 4

write(1, "100\n", 4100

) = 4

read(0, 1 1 5

"1 1 5\n", 1024) = 6

write(1, "0.161242\n", 90.161242

) = 9

read(0, "", 1024) = 0

exit\_group(0) = ?

+++ exited with 0 +++

ilya@ilya-ilyal:~/osi4laba$ strace -f ./programd

execve("./programd", ["./programd"], 0x7ffc9e090798 /\* 59 vars \*/) = 0

brk(NULL) = 0x55563a1d3000

arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7fff30795a10) = -1 EINVAL (Недопустимый аргумент)

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f08e9109000

access("/etc/ld.so.preload", R\_OK) = -1 ENOENT (Нет такого файла или каталога)

**openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 3**

newfstatat(3, "", {st\_mode=S\_IFREG|0644, st\_size=53247, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 53247, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7f08e90fc000

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\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\0"..., 48, 848) = 48

pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\244;\374\204(\337f#\315I\214\234\f\256\271\32"..., 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) = 0x7f08e8e00000

mmap(0x7f08e8e28000, 1658880, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x28000) = 0x7f08e8e28000

mmap(0x7f08e8fbd000, 360448, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1bd000) = 0x7f08e8fbd000

mmap(0x7f08e9015000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x214000) = 0x7f08e9015000

mmap(0x7f08e901b000, 52816, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f08e901b000

close(3) = 0

mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f08e90f9000

arch\_prctl(ARCH\_SET\_FS, 0x7f08e90f9740) = 0

set\_tid\_address(0x7f08e90f9a10) = 28622

set\_robust\_list(0x7f08e90f9a20, 24) = 0

rseq(0x7f08e90fa0e0, 0x20, 0, 0x53053053) = 0

mprotect(0x7f08e9015000, 16384, PROT\_READ) = 0

mprotect(0x555639522000, 4096, PROT\_READ) = 0

mprotect(0x7f08e9143000, 8192, PROT\_READ) = 0

prlimit64(0, RLIMIT\_STACK, NULL, {rlim\_cur=8192\*1024, rlim\_max=RLIM64\_INFINITY}) = 0

munmap(0x7f08e90fc000, 53247) = 0

openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 3

newfstatat(3, "", {st\_mode=S\_IFREG|0644, st\_size=53247, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 53247, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7f08e90fc000

close(3) = 0

getrandom("\xe4\x42\x72\xa3\x4a\x8b\xd4\xc3", 8, GRND\_NONBLOCK) = 8

brk(NULL) = 0x55563a1d3000

brk(0x55563a1f4000) = 0x55563a1f4000

**openat(AT\_FDCWD, "/usr/local/lib/libfirst.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|0755, st\_size=15840, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 16464, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f08e90f4000

mmap(0x7f08e90f5000, 4096, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1000) = 0x7f08e90f5000

mmap(0x7f08e90f6000, 4096, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x2000) = 0x7f08e90f6000

mmap(0x7f08e90f7000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x2000) = 0x7f08e90f7000

close(3) = 0

mprotect(0x7f08e90f7000, 4096, PROT\_READ) = 0

munmap(0x7f08e90fc000, 53247) = 0

newfstatat(0, "", {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0x1), ...}, AT\_EMPTY\_PATH) = 0

read(0, 1 1 5

"1 1 5\n", 1024) = 6

newfstatat(1, "", {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0x1), ...}, AT\_EMPTY\_PATH) = 0

write(1, "0.083695\n", 90.083695

) = 9

read(0, 0

"0\n", 1024) = 2

munmap(0x7f08e90f4000, 16464) = 0

**openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 3**

newfstatat(3, "", {st\_mode=S\_IFREG|0644, st\_size=53247, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 53247, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7f08e90fc000

close(3) = 0

**openat(AT\_FDCWD, "/usr/local/lib/libsecond.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|0755, st\_size=15840, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 16464, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f08e90f4000

mmap(0x7f08e90f5000, 4096, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1000) = 0x7f08e90f5000

mmap(0x7f08e90f6000, 4096, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x2000) = 0x7f08e90f6000

mmap(0x7f08e90f7000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x2000) = 0x7f08e90f7000

close(3) = 0

mprotect(0x7f08e90f7000, 4096, PROT\_READ) = 0

munmap(0x7f08e90fc000, 53247) = 0

write(1, "Contract has been changed\n", 26Contract has been changed

) = 26

write(1, "Contract is second\n", 19Contract is second

) = 19

read(0, 1 5

"1 5\n", 1024) = 4

read(0, 5

"5\n", 1024) = 2

write(1, "-3.543861\n", 10-3.543861

) = 10

read(0, 0

"0\n", 1024) = 2

munmap(0x7f08e90f4000, 16464) = 0

**openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 3**

newfstatat(3, "", {st\_mode=S\_IFREG|0644, st\_size=53247, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 53247, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7f08e90fc000

close(3) = 0

openat(AT\_FDCWD, "/usr/local/lib/libfirst.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|0755, st\_size=15840, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 16464, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f08e90f4000

mmap(0x7f08e90f5000, 4096, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1000) = 0x7f08e90f5000

mmap(0x7f08e90f6000, 4096, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x2000) = 0x7f08e90f6000

mmap(0x7f08e90f7000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x2000) = 0x7f08e90f7000

close(3) = 0

mprotect(0x7f08e90f7000, 4096, PROT\_READ) = 0

munmap(0x7f08e90fc000, 53247) = 0

write(1, "Contract has been changed\n", 26Contract has been changed

) = 26

write(1, "Contract is first\n", 18Contract is first

) = 18

read(0, 1 1 5

"1 1 5\n", 1024) = 6

write(1, "0.083695\n", 90.083695

) = 9

read(0, 1 1 1

"1 1 1\n", 1024) = 6

write(1, "-0.956449\n", 10-0.956449

) = 10

read(0, "", 1024) = 0

exit\_group(0) = ?

+++ exited with 0 +++

**Вывод**

В результате данной лабораторной работы я научился работать с динамическими библиотеками. Динамические библиотеки можно загружать в ходе выполнения программы. Их загрузка во время выполнения программы упрощает компиляцию программы, а также уменьшает размер исполняемых файлов.