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-214БВ-24

Студент: Александров М.С.

Преподаватель: Бахарев В.Д.

Оценка: _____

Дата: 17.12.25

Москва, 2025

Постановка задачи

Вариант 26.

Требуется создать две динамические библиотеки, реализующие один и тот же интерфейс (контракт), но с различными алгоритмами обработки данных.

Контракты и реализации:

1. **Подсчёт наибольшего общего делителя для двух натуральных чисел:**
   - Реализация №1: Алгоритм Евклида
   - Реализация №2: Наивный алгоритм: пытаться разделить числа на все числа, что меньше a и b
2. **Отсортировать целочисленный массив:**
   - Реализация №1: Пузырьковая сортировка
   - Реализация №2: Сортировка Хоара

Необходимо разработать две программы:

1. **Программа №1:** Использует одну из библиотек, связывание происходит на этапе компиляции. Программа жёстко зависит от наличия библиотеки при запуске.
2. **Программа №2:** Загружает библиотеки в RUNTIME по их относительным путям. Программа должна поддерживать переключение между реализациями по команде пользователя без перезапуска.

Взаимодействие с пользователем осуществляется через консоль. Ввод команд и аргументов обрабатывается вручную, ввод-вывод осуществляется через системные вызовы.

Общий метод и алгоритм решения

Использованные системные вызовы:

- *void* dlopen(const char* filename, int flags); – открывает динамическую библиотеку и возвращает дескриптор (handle).*
- *void* dlsym(void* handle, const char* symbol); – возвращает адрес символа (функции) в памяти загруженной библиотеки.*
- *int dlclose(void* handle); – уменьшает счетчик ссылок на библиотеку и выгружает её, если счетчик равен 0.*
- *char* dlerror(void); – возвращает текстовое описание последней ошибки, возникшей в функциях dl*.*

В рамках лабораторной работы были созданы два файла исходного кода библиотек (*library1.c, library2.c*), которые компилируются с флагами *-fPIC* (позиционно-независимый код) и *-shared* для создания динамических библиотек *libd1.so* и *libd2.so*.

Библиотеки:

- ***libd1.so*** реализует функцию *gcd* с помощью алгоритма Евклида и функцию *sort* сортировкой пузырьком.
- ***libd2.so*** реализует функцию gcd с помощью наивного алгоритма и функцию *sort* сортировкой Хоара.

Программа №1 (*prog1*):

Реализует статическую компоновку. При компиляции указывается путь к библиотеке (*-L.*) и её имя (*-ld1*). Для корректного запуска используется флаг линковщика *-Wl,-rpath,.*, указывающий загрузчику искать библиотеку в текущей директории. Программа вызывает функции библиотеки напрямую через заголовочный файл *libraries.h*. Ввод данных парсится с помощью *strtok* и *atoi*, вывод осуществляется через буфер с использованием *snprintf* и системного вызова *write*.

Программа №2 (*prog2*):

Реализует динамическую загрузку. Программа не слинкована с библиотеками при компиляции.

1. При запуске или по команде «0» программа вызывает *dlopen*() для загрузки соответствующего *.so* файла.
2. С помощью *dlsym*() программа получает указатели на функции *gcd* и *sort*.
3. Вызовы функций происходят через полученные указатели.
4. При переключении библиотек старая библиотека выгружается через *dlclose*(), и загружается новая.
5. Обработка ввода-вывода аналогична первой программе, но добавлена обработка ошибок загрузки библиотек (через *dlerror*).

Таким образом, продемонстрирована разница между жёсткой привязкой библиотек (требует компиляции для смены реализации) и гибкой загрузкой плагинов во время выполнения.

Код программы

libraries.h

```
#ifndef LIBRARIES_H
#define LIBRARIES_H

#include <stddef.h>

int gcd(int a, int b);
int *sort(int *array, size_t n);

#endif
```

library1.c

```
#include "libraries.h"


// Евклид
int gcd(int a, int b) {
    int temp;
    while (b != 0) {
        temp = b;
        b = a % b;
        a = temp;
    }
    return a;
}

// Пузырьковая
int *sort(int *array, size_t n) {
    int tmp, noSwap;

    for (int i = n - 1; i >= 0; --i) {
        noSwap = 1;
        for (int j = 0; j < i; ++j) {
```

```
            if (array[j] > array[j + 1]) {
                tmp = array[j];
                array[j] = array[j + 1];
                array[j + 1] = tmp;
                noSwap = 0;
            }
        }
        if (noSwap) break;
    }

    return array;
}
```

library2.c

```
#include "libraries.h"


// Наивная
int gcd(int a, int b) {
    int min = (a < b) ? a : b;
    int res = 1;
    for (int i = 1; i <= min; ++i) {
        if (a % i == 0 && b % i == 0) {
            res = i;
        }
    }

    return res;
}

// Хоар
void quick_sort(int *array, int left, int right) {
    if (left >= right) return;

    int pivot = array[(left + right) / 2];
    int i = left;
    int j = right;
    int tmp;

    while (i <= j) {
        while (array[i] < pivot) ++i;

        while (array[j] > pivot) --j;

        if (i <= j) {
            tmp = array[i];
            array[i] = array[j];
            array[j] = tmp;
            ++i;
            --j;
        }
    }

    if (left < j) quick_sort(array, left, j);
    if (i < right) quick_sort(array, i, right);
}

int *sort(int *array, size_t n) {
    if (n > 0) quick_sort(array, 0, (int)n - 1);
    return array;
}
```

program1.c

```
#include <stdio.h>
#include <string.h>
#include <stddef.h>
#include <stdlib.h>
#include <unistd.h>
#include "libraries.h"


void func_1()
{
    char *arg1 = strtok(NULL, " \t\n");
    char *arg2 = strtok(NULL, " \t\n");

    size_t length = 0;
    char buffer[1024];

    if (arg1 && arg2) {
        int result = gcd(atoi(arg1), atoi(arg2));
        length = snprintf(buffer, 1024, "gcd(%d, %d) = %d\n", atoi(arg1), atoi(arg2), result);
        write(STDOUT_FILENO, buffer, length);
    }
}

void func_2() {
    char* size_str = strtok(NULL, " \t\n");
    if (!size_str) return;

    int size = atoi(size_str);
    int* array = (int*)malloc(size * sizeof(int));
    if (!array) {
        const char* message = "Malloc error\n";
        write(STDERR_FILENO, message, strlen(message));
        return;
    }


    for (int i = 0; i < size; ++i) {
        char* value = strtok(NULL, " \t\n");
        if (value) array[i] = atoi(value);
        else array[i] = 0;
    }

    sort(array, size);

    {
        const char* message = "Sorted array: ";
        write(STDOUT_FILENO, message, strlen(message));
    }

    char buffer[1024];
    for (int i = 0; i < size; ++i) {
        int len = snprintf(buffer, 1024, "%d ", array[i]);
        write(STDOUT_FILENO, buffer, len);
    }

    write(STDOUT_FILENO, "\n", 1);
    free(array);
}

int main(void) {
```

```
    {
        const char* message = "Program 1.\nCommands: 1 a b | 2 Size arg1...\n";
        write(STDOUT_FILENO, message, strlen(message));
    }

    int bytes = 0;
    char buffer[1024];

    while ((bytes = read(STDIN_FILENO, buffer, 1024 - 1)) > 0) {
        buffer[bytes] = '\0';


        char *token = strtok(buffer, " \t\n");
        if (!token) {
            continue;
        }

        int func = atoi(token);
        switch (func) {
        case 1: {
            func_1();
            break;
        }

        case 2: {
            func_2();
            break;
        }

        default:
            break;
        }
    }

    return 0;
}
```

program2.c

```
#include <stdio.h>
#include <string.h>
#include <stddef.h>
#include <stdlib.h>
#include <unistd.h>
#include <dlfcn.h>
#include "libraries.h"


typedef int (*gcd_func)(int, int);
typedef int* (*sort_func)(int*, size_t);

enum CurrentLibrary {
    FIRST = 0,
    SECOND = 1
};


int command_0(const char **lib_names, void **library, int *current_lib,
              gcd_func *gcd, sort_func *sort) {

    dlclose(*library);

    switch (*current_lib) {
```

```
    case FIRST: {
        *current_lib = SECOND;
        break;
    }
    case SECOND: {
        *current_lib = FIRST;
        break;
    }
    default:
        return 1;
    }

    char buffer[1024];

    *library = dlopen(lib_names[*current_lib], RTLD_LAZY);
    if (!*library) {
        int length = snprintf(buffer, 1024, "Error: invalid switch libs; %s\n", dlerror());
        write(STDERR_FILENO, buffer, length);
        return 1;
    }

    *gcd = dlsym(*library, "gcd");
    if (!gcd)
    {
        const char msg[] = "Error: unable to get gcd function\n";
        write(STDOUT_FILENO, msg, sizeof(msg));
    }

    *sort = dlsym(*library, "sort");
    if (!sort) {
        const char msg[] = "Error: unable to get sort fucntion\n";
        write(STDOUT_FILENO, msg, sizeof(msg));
    }

    int length = snprintf(buffer, 1024, "Switched to %s library\n", lib_names[*current_lib]);
    write(STDOUT_FILENO, buffer, length);

    return 0;
}

void command_1(gcd_func gcd)
{
    char *arg1 = strtok(NULL, " \t\n");
    char *arg2 = strtok(NULL, " \t\n");

    size_t length = 0;
    char buffer[1024];

    if (arg1 && arg2) {
        int result = gcd(atoi(arg1), atoi(arg2));
        length = snprintf(buffer, 1024, "gcd(%d, %d) = %d\n", atoi(arg1), atoi(arg2), result);
        write(STDOUT_FILENO, buffer, length);
    }
}

void command_2(sort_func sort)
{
    char* size_str = strtok(NULL, " \t\n");
    if (!size_str) return;

    int size = atoi(size_str);
```

```
    int* array = (int*)malloc(size * sizeof(int));
    if (!array) {
        const char* message = "Malloc error\n";
        write(STDERR_FILENO, message, strlen(message));
        return;
    }

    for (int i = 0; i < size; ++i) {
        char* value = strtok(NULL, " \t\n");
        if (value) array[i] = atoi(value);
        else array[i] = 0;
    }

    sort(array, size);

    {
        const char* message = "Sorted array: ";
        write(STDOUT_FILENO, message, strlen(message));
    }

    char buffer[1024];
    for (int i = 0; i < size; ++i) {
        int len = snprintf(buffer, 1024, "%d ", array[i]);
        write(STDOUT_FILENO, buffer, len);
    }

    write(STDOUT_FILENO, "\n", 1);
    free(array);
}


int main() {
    const char *lib_names[] = {"./libd1.so", "./libd2.so"};
    int current_lib = FIRST;

    gcd_func gcd = NULL;
    sort_func sort = NULL;

    void *library = dlopen(lib_names[current_lib], RTLD_LAZY);
    char buffer[1024];
    if (!library) {
        int length = snprintf(buffer, 1024, "Error: unable to load lib; %s\n", dlerror());
        write(STDERR_FILENO, buffer, length);
        return 1;
    }

    gcd = dlsym(library, "gcd");
    if (!gcd) {
        const char msg[] = "Error: unable to get gcd function\n";
        write(STDOUT_FILENO, msg, sizeof(msg));
    }

    sort = dlsym(library, "sort");
    if (!sort) {
        const char msg[] = "Error: unable to get sort function\n";
        write(STDERR_FILENO, msg, sizeof(msg));
    }

    {
        const char *msg = "Program 2.\nCommands: 0 (Switch) | 1 a b | 2 Size arg1...\n";
        write(STDOUT_FILENO, msg, strlen(msg));
```

```
    }

    int bytes_read;
    while ((bytes_read = read(STDIN_FILENO, buffer, sizeof(buffer) - 1)) > 0) {
        buffer[bytes_read] = '\0';
        char *cmd_str = strtok(buffer, " \t\n");

        if (cmd_str == NULL) {
            continue;
        }

        int cmd = atoi(cmd_str);
        switch (cmd) {
        case 0: {
            int result = command_0(lib_names, &library, &current_lib, &gcd, &sort);
            if (result) {
                return result;
            }
            break;
        }
        case 1: {
            command_1(gcd);
            break;
        }
        case 2: {
            command_2(sort);
            break;
        }
        default:
            break;
        }
    }

    if (library) {
        dlclose(library);
    }

    return 0;
}
```

Протокол работы программы

Тестирование:

```
maks-alex@DESKTOP-QFPFVP1:~/MAI_OS/lab4/src$ ./program1
Program 1.
Commands: 1 a b | 2 Size arg1...
1 27 15
gcd(27, 15) = 3
2 7 -4 6 9 123 -4000 3 0
Sorted array: -4000 -4 0 3 6 9 123
```

```
maks-alex@DESKTOP-QFPFVP1:~/MAI_OS/lab4/src$ ./program2
Program 2.
Commands: 0 (Switch) | 1 a b | 2 Size arg1...
1 27 15
gcd(27, 15) = 3
2 7 -123 23 0 15 6 100 2
Sorted array: -123 0 2 6 15 23 100
0
Switched to ./libd2.so library
1 27 15
gcd(27, 15) = 3
2 7 -123 23 0 15 6 100 2
Sorted array: -123 0 2 6 15 23 100
0
Switched to ./libd1.so library
```

Strace:

strace -f ./program1

execve("./program1", ["./program1"], 0x7fffef6b19d8 /* 29 vars */) = 0

brk(NULL) = 0x55ad65b33000

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7fe0c1386000

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

openat(AT_FDCWD, "./glibc-hwcaps/x86-64-v3/libd1.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD, "./glibc-hwcaps/x86-64-v2/libd1.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD, "./libd1.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

fstat(3, {st_mode=S_IFREG|0755, st_size=15136, ...}) = 0

getcwd("/home/maks-alex/MAI_OS/lab4/src", 128) = 32

mmap(NULL, 16400, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7fe0c1381000

mmap(0x7fe0c1382000, 4096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x7fe0c1382000

mmap(0x7fe0c1383000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7fe0c1383000

mmap(0x7fe0c1384000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7fe0c1384000

close(3) = 0

openat(AT_FDCWD, "./glibc-hwcaps/x86-64-v3/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD, "./glibc-hwcaps/x86-64-v2/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD, "./libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

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

fstat(3, {st_mode=S_IFREG|0644, st_size=20003, ...}) = 0

mmap(NULL, 20003, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7fe0c137c000

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\220\243\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

fstat(3, {st_mode=S_IFREG|0755, st_size=2125328, ...}) = 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, 2170256, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7fe0c116a000

mmap(0x7fe0c1192000, 1605632, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x7fe0c1192000

mmap(0x7fe0c131a000, 323584, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x1b0000) = 0x7fe0c131a000

mmap(0x7fe0c1369000, 24576, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1fe000) = 0x7fe0c1369000

mmap(0x7fe0c136f000, 52624, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7fe0c136f000

close(3)   = 0

mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7fe0c1167000

arch_prctl(ARCH_SET_FS, 0x7fe0c1167740) = 0

set_tid_address(0x7fe0c1167a10)   = 17141

set_robust_list(0x7fe0c1167a20, 24)   = 0

rseq(0x7fe0c1168060, 0x20, 0, 0x53053053) = 0

mprotect(0x7fe0c1369000, 16384, PROT_READ) = 0

mprotect(0x7fe0c1384000, 4096, PROT_READ) = 0

mprotect(0x55ad2bce1000, 4096, PROT_READ) = 0

mprotect(0x7fe0c13be000, 8192, PROT_READ) = 0

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

munmap(0x7fe0c137c000, 20003)   = 0

write(1, "Program 1.\nCommands: 1 a b | 2 S"..., 44Program 1.

Commands: 1 a b | 2 Size arg1...

) = 44

read(0, 1 27 15

"1 27 15\n", 1023)   = 8

write(1, "gcd(27, 15) = 3\n", 16gcd(27, 15) = 3

)   = 16

read(0, 2 7 -4 6 9 123 -4000 3 0

"2 7 -4 6 9 123 -4000 3 0\n", 1023) = 25

getrandom("\x5f\x7d\x4d\x61\x6c\xdf\x20\x3c", 8, GRND_NONBLOCK) = 8

brk(NULL)   = 0x55ad65b33000

brk(0x55ad65b54000)   = 0x55ad65b54000

write(1, "Sorted array: ", 14Sorted array:)   = 14

write(1, "-4000 ", 6-4000)   = 6

write(1, "-4 ", 3-4) = 3

write(1, "0 ", 20) = 2

write(1, "3 ", 23) = 2

write(1, "6 ", 26) = 2

write(1, "9 ", 29) = 2

write(1, "123 ", 4123) = 4

write(1, "\n", 1) = 1

read(0, "", 1023) = 0

exit_group(0) = ?
+++ exited with 0 +++

strace -f ./program2

execve("./program2", ["./program2"], 0x7ffcc9ae8f08 /* 29 vars */) = 0

brk(NULL) = 0x55e32cbf3000

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7fec70134000

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

fstat(3, {st_mode=S_IFREG|0644, st_size=20003, ...}) = 0

mmap(NULL, 20003, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7fec7012f000

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\220\243\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

fstat(3, {st_mode=S_IFREG|0755, st_size=2125328, ...}) = 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, 2170256, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7fec6ff1d000

mmap(0x7fec6ff45000, 1605632, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x7fec6ff45000

mmap(0x7fec700cd000, 323584, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1b0000) = 0x7fec700cd000

mmap(0x7fec7011c000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1fe000) = 0x7fec7011c000

mmap(0x7fec70122000, 52624, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7fec70122000

close(3) = 0

mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =

0x7fec6ff1a000

arch_prctl(ARCH_SET_FS, 0x7fec6ff1a740) = 0

set_tid_address(0x7fec6ff1aa10)   = 18258

set_robust_list(0x7fec6ff1aa20, 24)   = 0

rseq(0x7fec6ff1b060, 0x20, 0, 0x53053053) = 0

mprotect(0x7fec7011c000, 16384, PROT_READ) = 0

mprotect(0x55e2f9e8d000, 4096, PROT_READ) = 0

mprotect(0x7fec7016c000, 8192, PROT_READ) = 0

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

munmap(0x7fec7012f000, 20003)   = 0

getrandom("\x77\xdc\xc3\xad\x66\x61\xa0\x46", 8, GRND_NONBLOCK) = 8

brk(NULL)   = 0x55e32cbf3000

brk(0x55e32cc14000)   = 0x55e32cc14000

openat(AT_FDCWD, "./libd1.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

fstat(3, {st_mode=S_IFREG|0755, st_size=15136, ...}) = 0

getcwd("/home/maks-alex/MAI_OS/lab4/src", 128) = 32

mmap(NULL, 16400, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7fec7012f000

mmap(0x7fec70130000, 4096, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x7fec70130000

mmap(0x7fec70131000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7fec70131000

mmap(0x7fec70132000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7fec70132000

close(3)   = 0

mprotect(0x7fec70132000, 4096, PROT_READ) = 0

write(1, "Program 2.\nCommands: 0 (Switch) "..., 57Program 2.

Commands: 0 (Switch) | 1 a b | 2 Size arg1...

) = 57

read(0, 1 27 15

"1 27 15\n", 1023)   = 8

write(1, "gcd(27, 15) = 3\n", 16gcd(27, 15) = 3)   = 16

read(0, 2 7 -123 23 0 15 6 100 2

"2 7 -123 23 0 15 6 100 2\n", 1023) = 25

write(1, "Sorted array: ", 14Sorted array:)   = 14

write(1, "-123 ", 5-123) = 5

write(1, "0 ", 20) = 2

write(1, "2 ", 22) = 2

write(1, "6 ", 26) = 2

write(1, "15 ", 315) = 3

write(1, "23 ", 323) = 3

write(1, "100 ", 4100) = 4

write(1, "\n", 1) = 1

read(0, 0

"0\n", 1023) = 2

munmap(0x7fec7012f000, 16400) = 0

openat(AT_FDCWD, "./libd2.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

fstat(3, {st_mode=S_IFREG|0755, st_size=15368, ...}) = 0

getcwd("/home/maks-alex/MAI_OS/lab4/src", 128) = 32

mmap(NULL, 16408, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7fec7012f000

**mmap(0x7fec70130000, 4096, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x7fec70130000**

mmap(0x7fec70131000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7fec70131000

**mmap(0x7fec70132000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7fec70132000**

close(3) = 0

mprotect(0x7fec70132000, 4096, PROT_READ) = 0

write(1, "Switched to ./libd2.so library\n", 31Switched to ./libd2.so library

) = 31

read(0, 1 27 15

"1 27 15\n", 1023) = 8

write(1, "gcd(27, 15) = 3\n", 16gcd(27, 15) = 3

) = 16

read(0, 2 7 -123 23 0 15 6 100 2"2 7 -123 23 0 15 6 100 2\n", 1023) = 25

write(1, "Sorted array: ", 14Sorted array:) = 14

write(1, "-123 ", 5-123) = 5

write(1, "0 ", 20) = 2

write(1, "2 ", 22) = 2

write(1, "6 ", 26) = 2

write(1, "15 ", 315) = 3

write(1, "23 ", 323) = 3

write(1, "100 ", 4100) = 4

write(1, "\n", 1) = 1

read(0, 0

"0\n", 1023) = 2

munmap(0x7fec7012f000, 16408) = 0

openat(AT_FDCWD, "./libd1.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

fstat(3, {st_mode=S_IFREG|0755, st_size=15136, ...}) = 0

getcwd("/home/maks-alex/MAI_OS/lab4/src", 128) = 32

mmap(NULL, 16400, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7fec7012f000

mmap(0x7fec70130000, 4096, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x7fec70130000

mmap(0x7fec70131000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7fec70131000

mmap(0x7fec70132000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7fec70132000

close(3) = 0

mprotect(0x7fec70132000, 4096, PROT_READ) = 0

write(1, "Switched to ./libd1.so library\n", 31Switched to ./libd1.so library) = 31

read(0, "", 1023) = 0

munmap(0x7fec7012f000, 16400) = 0

exit_group(0) = ?

+++ exited with 0 +++

Вывод

В ходе выполнения лабораторной работы были успешно созданы динамические библиотеки в *Linux* и реализованы программы для их использования двумя способами: через статическую компоновку и через динамическую подгрузку с использованием *dlfcn.h.*