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-214Б-24

Студент: Ельцова Д.А.

Преподаватель: Бахарев В.Д.

Оценка: _____

Дата: 16.12.25

Москва, 2025

Постановка задачи

Вариант 34.

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

Контракты и реализации:

1. **Подсчет площади плоской геометрической фигуры по двум сторонам**:
   Сигнатура функции: float area(float a, float b);
   - Реализация №1: Фигура прямоугольник
   - Реализация №2: Фигура прямоугольный

2. **Перевод числа x из десятичной системы счисления в другую**:
   Сигнатура функции: char *convert(int x);
   - Реализация №1: Перевод в двоичную
   - Реализация №2: Перевод в троичную

Требуется разработать две независимые программы:

1. Программа №1 (Статическое связывание):
   Принцип работы: Использование динамической библиотеки с фиксированной привязкой на этапе компиляции
   Механизм связывания: Линковка (linking) осуществляется до запуска программы в процессе сборки
   Особенность: Формируется жесткая зависимость между программой и конкретной версией библиотеки
   Результат: Создается исполняемый файл, требующий обязательного наличия связанной библиотеки в системе во время выполнения
2. Программа №2 (Динамическая загрузка):
   Принцип работы: Отложенная загрузка библиотек непосредственно в процессе выполнения программы
   Механизм связывания: Использование интерфейса операционной системы для динамического подключения библиотек.
   Ключевая функция: Поддержка интерактивного переключения между альтернативными реализациями без необходимости перезапуска приложения
   Требование к путям: Библиотеки загружаются по относительным путям, что обеспечивает переносимость решения

Взаимодействие с пользователем организовано через консольный интерфейс, поддерживающий два режима работы: интерактивный ввод команд с клавиатуры и пакетную обработку команд из текстового файла, при этом в обоих случаях операции ввода-вывода выполняются через низкоуровневые системные вызовы.

Общий метод и алгоритм решения

Основные использованные системные вызовы:

Для работы с динамическими библиотеками:

- 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*.

Для низкоуровневого ввода-вывода:

- ssize_t read(int fd, void* buf, size_t count); – чтение данных из файлового дескриптора.
- ssize_t write(int fd, const void* buf, size_t count);– запись данных в файловый дескриптор.
- int open(const char* pathname, int flags); – открытие файла для чтения команд.
- int close(int fd); – закрытие файлового дескриптора.

Описание программы

В рамках лабораторной работы созданы два файла исходного кода библиотек (lib1.c, lib2.c), которые компилируются с флагами -fPIC (позиционно-независимый код) и -shared для создания динамических библиотек lib1.so и lib2.so.

lib1.so реализует:

1. Функцию area(float a, float b) – вычисление площади прямоугольника по формуле S = a × b.
2. Функцию convert(int x) – перевод числа в двоичную систему счисления с использованием побитовых операций.

lib2.so реализует:

1. Функцию area(float a, float b) – вычисление площади прямоугольного треугольника по формуле S = (a × b) / 2.
2. Функцию convert(int x) – перевод числа в троичную систему счисления с использованием деления на 3.

Программа №1 (prog1):

Реализует статическую компоновку. При компиляции указывается путь к библиотеке (-L.) и её имя (-l1). Для корректного запуска требуется установка переменной окружения LD_LIBRARY_PATH=. или использование флага линковщика -Wl,-rpath,., указывающего загрузчику искать библиотеку в текущей директории.

Алгоритм работы:

- Программа вызывает функции библиотеки напрямую через заголовочный файл libs.h
- Поддерживает два режима работы: интерактивный и обработку команд из файла
- Ввод данных парсится с помощью strtok() и atoi()/atof()
- Вывод осуществляется через буфер с использованием snprintf() и системного вызова write()

Программа №2 (prog2):
Реализует динамическую загрузку. Программа не слинкована с библиотеками при компиляции, а загружает их во время выполнения.
Алгоритм работы:

- При запуске или по команде «0» программа вызывает dlopen() для загрузки соответствующего .so файла
- С помощью dlsym() программа получает указатели на функции area и convert
- Вызовы функций происходят через полученные указатели
- При переключении библиотек старая библиотека выгружается через dlclose(), и загружается новая
- Поддерживаются оба режима работы: интерактивный и из файла
- Реализована полноценная обработка ошибок загрузки библиотек через dlerror()

Код программы

libs.h

```
#ifndef LIBS_H

#define LIBS_H


typedef float area_func(float a, float b);

typedef char* convert_func(int x);


#endif // LIBS_H
```

lib1.c

```
#include <stdlib.h>

#include <string.h>


#ifdef _MSC_VER

#define EXPORT __declspec(dllexport)

#else

#define EXPORT

#endif


EXPORT float area(float a, float b) {

    return a * b;
```

```
}

EXPORT char* convert(int x) {
    if (x == 0) {
        char* result = (char*)malloc(2 * sizeof(char));
        if (result) {
            result[0] = '0';
            result[1] = '\0';
        }
        return result;
    }

    int temp = x;
    int length = 0;
    while (temp > 0) {
        temp >>= 1;
        length++;
    }

    char* result = (char*)malloc((length + 1) * sizeof(char));
    if (!result) return NULL;

    result[length] = '\0';
    temp = x;
    for (int i = length - 1; i >= 0; i--) {
        result[i] = (temp & 1) ? '1' : '0';
        temp >>= 1;
    }

    return result;
}
```

lib2.c

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

#ifdef _MSC_VER
#define EXPORT __declspec(dllexport)
#else
#define EXPORT
#endif

EXPORT float area(float a, float b) {
    return (a * b) / 2.0f;
}

EXPORT char* convert(int x) {
    if (x == 0) {
        char* result = (char*)malloc(2 * sizeof(char));
        if (result) {
            result[0] = '0';
            result[1] = '\0';
        }
        return result;
    }

    int temp = x;
    int length = 0;
    while (temp > 0) {
        temp /= 3;
        length++;
    }
```

```
    char* result = (char*)malloc((length + 1) * sizeof(char));
    if (!result) return NULL;

    result[length] = '\0';
    temp = x;
    for (int i = length - 1; i >= 0; i--) {
        int remainder = temp % 3;
        result[i] = '0' + remainder;
        temp /= 3;
    }

    return result;
}
```

prog1.c

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

#include "libs.h"

extern float area(float a, float b);
extern char* convert(int x);

#define BUFFER_SIZE 1024

static float area_stub(float a, float b) {
    (void)a; (void)b;
```

```
    const char msg[] = "error: area function not available\n";
    write(STDERR_FILENO, msg, sizeof(msg) - 1);
    return 0.0f;
}

static char* convert_stub(int x) {
    (void)x;
    const char msg[] = "error: convert function not available\n";
    write(STDERR_FILENO, msg, sizeof(msg) - 1);
    char* result = (char*)malloc(2 * sizeof(char));
    if (result) {
        result[0] = '0';
        result[1] = '\0';
    }
    return result;
}

void process_command_1(char* args, int output_fd) {
    float a, b;
    if (sscanf(args, "%f %f", &a, &b) == 2) {
        float result = area(a, b);
        char buf[BUFFER_SIZE];
        int length = snprintf(buf, sizeof(buf) - 1,
                              "Area with sides %.2f and %.2f = %.2f\n",
                              a, b, result);
        buf[length] = '\0';
        write(output_fd, buf, length);
    } else {
        const char msg[] = "error: wrong arguments for command 1\n"
                           "usage: 1 A B\n";
        write(STDERR_FILENO, msg, sizeof(msg) - 1);
    }
}
```

```
void process_command_2(char* args, int output_fd) {
    int x;
    if (sscanf(args, "%d", &x) == 1) {
        char* result = convert(x);
        char buf[BUFFER_SIZE];
        int length = snprintf(buf, sizeof(buf) - 1,
                              "Number %d in binary: %s\n", x, result);
        buf[length] = '\0';
        write(output_fd, buf, length);
        free(result);
    } else {
        const char msg[] = "error: wrong argument for command 2\n"
                           "usage: 2 X\n";
        write(STDERR_FILENO, msg, sizeof(msg) - 1);
    }
}

int process_input_file(const char* filename) {
    int fd = open(filename, O_RDONLY);
    if (fd < 0) {
        char buf[BUFFER_SIZE];
        int len = snprintf(buf, sizeof(buf), "error: cannot open file %s\n", filename);
        write(STDERR_FILENO, buf, len);
        return 0;
    }

    char buffer[BUFFER_SIZE];
    int bytes_read;

    while ((bytes_read = read(fd, buffer, BUFFER_SIZE - 1)) > 0) {
        buffer[bytes_read] = '\0';
```

```
char* line = buffer;
char* line_end;

while ((line_end = strchr(line, '\n')) != NULL) {
    *line_end = '\0';

    if (strlen(line) > 0) {
        char* token = strtok(line, " \t");
        if (!token) continue;

        if (strcmp(token, "exit") == 0) {
            close(fd);
            return 1;
        }

        if (strcmp(token, "0") == 0) {
            const char info[] = "Static linking: using library 1\n"
                                "  - Area: rectangle (a * b)\n"
                                "  - Convert: to binary system\n";
            write(STDOUT_FILENO, info, sizeof(info) - 1);
        } else {
            char* args = token + strlen(token) + 1;
            if (*args == '\0') args = NULL;

            int cmd = atoi(token);
            switch (cmd) {
                case 1:
                    process_command_1(args, STDOUT_FILENO);
                    break;
                case 2:
                    process_command_2(args, STDOUT_FILENO);
                    break;
                default:
```

```
                            const char err[] = "error: unknown command\n";
                            write(STDERR_FILENO, err, sizeof(err) - 1);
                            break;
                        }
                    }
                }

                line = line_end + 1;
            }
        }

        close(fd);
        return 1;
    }

    int main(int argc, char** argv) {
        if (argc > 1) {
            process_input_file(argv[1]);
        } else {
            const char msg[] = "Program 1 (Static Linking)\n"
                               "Commands: 0 | 1 A B | 2 X | exit\n"
                               "Usage: ./prog1 [commands_file.txt]\n"
                               "> ";
            write(STDOUT_FILENO, msg, sizeof(msg) - 1);

            int bytes_read = 0;
            char buffer[BUFFER_SIZE];

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

                char* newline = strchr(buffer, '\n');
                if (newline) *newline = '\0';
```

```
char* token = strtok(buffer, " \t\n");
if (!token) {
    write(STDOUT_FILENO, "> ", 2);
    continue;
}

if (strcmp(token, "exit") == 0) {
    break;
}

if (strcmp(token, "0") == 0) {
    const char info[] = "Current implementation: Library 1\n"
                        "  - Area: rectangle (a * b)\n"
                        "  - Convert: to binary system\n"
                        "Cannot switch implementations in static linking\n"
                        "> ";
    write(STDOUT_FILENO, info, sizeof(info) - 1);
    continue;
}

char* args = token + strlen(token) + 1;
if (*args == '\0') args = NULL;

int cmd = atoi(token);
switch (cmd) {
    case 1:
        process_command_1(args, STDOUT_FILENO);
        break;
    case 2:
        process_command_2(args, STDOUT_FILENO);
        break;
    default:
```

```
                    const char err[] = "error: unknown command\n";
                    write(STDERR_FILENO, err, sizeof(err) - 1);
                    break;
            }

            write(STDOUT_FILENO, "> ", 2);
        }
    }

    return 0;
}
```

prog2.c

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

#include "libs.h"

#define BUFFER_SIZE 1024

static area_func* area_impl = NULL;
static convert_func* convert_impl = NULL;
static void* library_handle = NULL;
static const char* LIB_NAMES[] = {"./lib1.so", "./lib2.so"};
static int current_lib = 0;

static float area_stub(float a, float b) {
```

```
    (void)a; (void)b;
    const char msg[] = "warning: area function not available\n";
    write(STDERR_FILENO, msg, sizeof(msg) - 1);
    return 0.0f;
}

static char* convert_stub(int x) {
    (void)x;
    const char msg[] = "warning: convert function not available\n";
    write(STDERR_FILENO, msg, sizeof(msg) - 1);
    char* result = (char*)malloc(2 * sizeof(char));
    if (result) {
        result[0] = '0';
        result[1] = '\0';
    }
    return result;
}

int load_library(int lib_index) {
    if (library_handle) {
        dlclose(library_handle);
        library_handle = NULL;
    }

    library_handle = dlopen(LIB_NAMES[lib_index], RTLD_LAZY);

    if (library_handle == NULL) {
        const char msg[] = "warning: library failed to load\n";
        write(STDERR_FILENO, msg, sizeof(msg) - 1);

        area_impl = area_stub;
        convert_impl = convert_stub;
        return 0;
```

```
    }

    area_impl = (area_func*)dlsym(library_handle, "area");
    if (area_impl == NULL) {
        const char msg[] = "warning: failed to find area function implementation\n";
        write(STDERR_FILENO, msg, sizeof(msg) - 1);
        area_impl = area_stub;
    }

    convert_impl = (convert_func*)dlsym(library_handle, "convert");
    if (convert_impl == NULL) {
        const char msg[] = "warning: failed to find convert function implementation\n";
        write(STDERR_FILENO, msg, sizeof(msg) - 1);
        convert_impl = convert_stub;
    }

    current_lib = lib_index;
    return 1;
}

void switch_library() {
    int new_lib = (current_lib == 0) ? 1 : 0;

    char buf[BUFFER_SIZE];
    int length = snprintf(buf, sizeof(buf) - 1,
                          "Switching from library %d to library %d\n",
                          current_lib + 1, new_lib + 1);
    buf[length] = '\0';
    write(STDOUT_FILENO, buf, length);

    if (load_library(new_lib)) {
        const char* msg = (new_lib == 0) ?
            "Loaded library 1: rectangle area, binary conversion\n" :
```

```
            "Loaded library 2: triangle area, ternary conversion\n";
        write(STDOUT_FILENO, msg, strlen(msg));
    }
}

void process_command_1(char* args, int output_fd) {
    float a, b;
    if (sscanf(args, "%f %f", &a, &b) == 2) {
        float result = area_impl(a, b);
        char buf[BUFFER_SIZE];
        int length = snprintf(buf, sizeof(buf) - 1,
                              "Area with sides %.2f and %.2f = %.2f\n",
                              a, b, result);
        buf[length] = '\0';
        write(output_fd, buf, length);
    } else {
        const char msg[] = "error: wrong arguments for command 1\n"
                           "usage: 1 A B\n";
        write(STDERR_FILENO, msg, sizeof(msg) - 1);
    }
}

void process_command_2(char* args, int output_fd) {
    int x;
    if (sscanf(args, "%d", &x) == 1) {
        char* result = convert_impl(x);
        char buf[BUFFER_SIZE];
        const char* system_name = (current_lib == 0) ? "binary" : "ternary";
        int length = snprintf(buf, sizeof(buf) - 1,
                              "Number %d in %s: %s\n", x, system_name, result);
        buf[length] = '\0';
        write(output_fd, buf, length);
        free(result);
```

```
    } else {
        const char msg[] = "error: wrong argument for command 2\n"
                           "usage: 2 X\n";
        write(STDERR_FILENO, msg, sizeof(msg) - 1);
    }
}

int process_input_file(const char* filename) {
    int fd = open(filename, O_RDONLY);
    if (fd < 0) {
        char buf[BUFFER_SIZE];
        int len = snprintf(buf, sizeof(buf), "error: cannot open file %s\n", filename);
        write(STDERR_FILENO, buf, len);
        return 0;
    }

    char buffer[BUFFER_SIZE];
    int bytes_read;

    while ((bytes_read = read(fd, buffer, BUFFER_SIZE - 1)) > 0) {
        buffer[bytes_read] = '\0';

        char* line = buffer;
        char* line_end;

        while ((line_end = strchr(line, '\n')) != NULL) {
            *line_end = '\0';

            if (strlen(line) > 0) {
                char* token = strtok(line, " \t");
                if (!token) continue;

                if (strcmp(token, "exit") == 0) {
```

```
                    close(fd);
                    return 1;
                }

                char* args = token + strlen(token) + 1;
                if (*args == '\0') args = NULL;

                int cmd = atoi(token);
                switch (cmd) {
                    case 0:
                        switch_library();
                        break;
                    case 1:
                        process_command_1(args, STDOUT_FILENO);
                        break;
                    case 2:
                        process_command_2(args, STDOUT_FILENO);
                        break;
                    default:
                        const char err[] = "error: unknown command\n";
                        write(STDERR_FILENO, err, sizeof(err) - 1);
                        break;
                }
            }

            line = line_end + 1;
        }
    }

    close(fd);
    return 1;
}
```

```
int main(int argc, char** argv) {
    area_impl = area_stub;
    convert_impl = convert_stub;

    if (!load_library(0)) {
        const char msg[] = "warning: using stub implementations\n";
        write(STDERR_FILENO, msg, sizeof(msg) - 1);
    }

    if (argc > 1) {
        process_input_file(argv[1]);
    } else {
        const char msg[] = "Program 2 (Dynamic Loading)\n"
                           "Commands: 0 | 1 A B | 2 X | exit\n"
                           "Usage: ./prog2 [commands_file.txt]\n"
                           "> ";
        write(STDOUT_FILENO, msg, sizeof(msg) - 1);

        int bytes_read = 0;
        char buffer[BUFFER_SIZE];

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

            char* newline = strchr(buffer, '\n');
            if (newline) *newline = '\0';

            char* token = strtok(buffer, " \t\n");
            if (!token) {
                write(STDOUT_FILENO, "> ", 2);
                continue;
            }
```

```
        if (strcmp(token, "exit") == 0) {
            break;
        }

        char* args = token + strlen(token) + 1;
        if (*args == '\0') args = NULL;

        int cmd = atoi(token);
        switch (cmd) {
            case 0:
                switch_library();
                break;
            case 1:
                process_command_1(args, STDOUT_FILENO);
                break;
            case 2:
                process_command_2(args, STDOUT_FILENO);
                break;
            default:
                const char err[] = "error: unknown command\n";
                write(STDERR_FILENO, err, sizeof(err) - 1);
                break;
        }

        write(STDOUT_FILENO, "> ", 2);
    }
}

if (library_handle) {
    dlclose(library_handle);
}

return 0;
```

```
}
```

Протокол работы программы

Тестирование:

commands.txt

```
0

1 5 3

2 10

0

1 5 3

2 10

exit
```

```
suslik@WIN-L3ULFBUQJMS:/mnt/c/Users/daria/Desktop/уч и зад/3 сем/оси/lr4$ ./prog1
commands.txt

Static linking: using library 1

  - Area: rectangle (a * b)

  - Convert: to binary system

Area with sides 5.00 and 3.00 = 15.00

Number 10 in binary: 1010

Static linking: using library 1

  - Area: rectangle (a * b)

  - Convert: to binary system

Area with sides 5.00 and 3.00 = 15.00

Number 10 in binary: 1010

suslik@WIN-L3ULFBUQJMS:/mnt/c/Users/daria/Desktop/уч и зад/3 сем/оси/lr4$ ./prog2
commands.txt

Switching from library 1 to library 2

Loaded library 2: triangle area, ternary conversion

Area with sides 5.00 and 3.00 = 7.50

Number 10 in ternary: 101

Switching from library 2 to library 1

Loaded library 1: rectangle area, binary conversion

Area with sides 5.00 and 3.00 = 15.00

Number 10 in binary: 1010

suslik@WIN-L3ULFBUQJMS:/mnt/c/Users/daria/Desktop/уч и зад/3 сем/оси/lr4$ ./prog2
```

```
Program 2 (Dynamic Loading)
Commands: 0 | 1 A B | 2 X | exit
Usage: ./prog2 [commands_file.txt]
> 0
Switching from library 1 to library 2
Loaded library 2: triangle area, ternary conversion
> 1 10 6
Area with sides 10.00 and 6.00 = 30.00
> 2 42
Number 42 in ternary: 1120
> 0
Switching from library 2 to library 1
Loaded library 1: rectangle area, binary conversion
> 1 10 6
Area with sides 10.00 and 6.00 = 60.00
> 2 42
Number 42 in binary: 101010
> exit
suslik@WIN-L3ULFBUQJMS:/mnt/c/Users/daria/Desktop/уч и зад/3 сем/оси/lr4$ ./prog1
Program 1 (Static Linking)
Commands: 0 | 1 A B | 2 X | exit
Usage: ./prog1 [commands_file.txt]
> 1 0 0
Area with sides 0.00 and 0.00 = 0.00
> 1 0 5
Area with sides 0.00 and 5.00 = 0.00
> 1 5 0
Area with sides 5.00 and 0.00 = 0.00
> 2 0
Number 0 in binary: 0
> 2 1
Number 1 in binary: 1
> 2 2
Number 2 in binary: 10
> exit
```

```
suslik@WIN-L3ULFBUQJMS:/mnt/c/Users/daria/Desktop/уч и зад/3 сем/оси/lr4$ ./prog2
Program 2 (Dynamic Loading)
Commands: 0 | 1 A B | 2 X | exit
Usage: ./prog2 [commands_file.txt]
> 0
Switching from library 1 to library 2
Loaded library 2: triangle area, ternary conversion
> 1 4 7
Area with sides 4.00 and 7.00 = 14.00
> 2 16
Number 16 in ternary: 121
> 0
Switching from library 2 to library 1
Loaded library 1: rectangle area, binary conversion
> 1 4 7
Area with sides 4.00 and 7.00 = 28.00
> 2 16
Number 16 in binary: 10000
> 0
Switching from library 1 to library 2
Loaded library 2: triangle area, ternary conversion
> 1 4 7
Area with sides 4.00 and 7.00 = 14.00
> 2 16
Number 16 in ternary: 121
> exit
suslik@WIN-L3ULFBUQJMS:/mnt/c/Users/daria/Desktop/уч и зад/3 сем/оси/lr4$ ./prog1
Program 1 (Static Linking)
Commands: 0 | 1 A B | 2 X | exit
Usage: ./prog1 [commands_file.txt]
> 1 1000 500
Area with sides 1000.00 and 500.00 = 500000.00
> 2 1024
Number 1024 in binary: 10000000000
> 1 3.14 2.71
```

```
Area with sides 3.14 and 2.71 = 8.51

> 2 1000

Number 1000 in binary: 1111101000

> exit

suslik@WIN-L3ULFBUQJMS:/mnt/c/Users/daria/Desktop/уч и зад/3 сем/оси/lr4$ ./prog2

Program 2 (Dynamic Loading)

Commands: 0 | 1 A B | 2 X | exit

Usage: ./prog2 [commands_file.txt]

> 0

Switching from library 1 to library 2

Loaded library 2: triangle area, ternary conversion

> 1 7.5 2.0

Area with sides 7.50 and 2.00 = 7.50

> 2 7

Number 7 in ternary: 21

> 0

Switching from library 2 to library 1

Loaded library 1: rectangle area, binary conversion

> 1 7.5 2.0

Area with sides 7.50 and 2.00 = 15.00

> 2 7

Number 7 in binary: 111

> exit

suslik@WIN-L3ULFBUQJMS:/mnt/c/Users/daria/Desktop/уч и зад/3 сем/оси/lr4$
```

Strace:

1.

suslik@WIN-L3ULFBUQJMS:/mnt/c/Users/daria/Desktop/уч и зад/3 сем/оси/lr4$ **strace -f ./prog1 commands.txt**

```
execve("./prog1", ["./prog1", "commands.txt"], 0x7fff43ec1e40 /* 27 vars */) = 0

brk(NULL)                               = 0x55c451827000

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7f5f266a1000

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

openat(AT_FDCWD, "./glibc-hwcaps/x86-64-v4/lib1.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT
(No such file or directory)
```

```
      openat(AT_FDCWD, "./glibc-hwcaps/x86-64-v3/lib1.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT
(No such file or directory)

      openat(AT_FDCWD, "./glibc-hwcaps/x86-64-v2/lib1.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT
(No such file or directory)

      openat(AT_FDCWD, "./lib1.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|0777, st_size=15504, ...}) = 0

      getcwd("/mnt/c/Users/daria/Desktop/\321\203\321\207 \320\270 \320\267\320\260\320\264/3
\321\201\320\265\320\274/\320\276\321\201\320\270/lr4", 128) = 62

      mmap(NULL, 16408, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f5f2669c000

      mmap(0x7f5f2669d000, 4096, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x7f5f2669d000

      mmap(0x7f5f2669e000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x2000) = 0x7f5f2669e000

      mmap(0x7f5f2669f000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7f5f2669f000

      close(3)                                = 0

      openat(AT_FDCWD, "./glibc-hwcaps/x86-64-v4/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT
(No such file or directory)

      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=19963, ...}) = 0

      mmap(NULL, 19963, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f5f26697000

      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) = 0x7f5f26485000

      mmap(0x7f5f264ad000, 1605632, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x28000) = 0x7f5f264ad000
```

```
      mmap(0x7f5f26635000, 323584, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x1b0000) = 0x7f5f26635000

      mmap(0x7f5f26684000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x1fe000) = 0x7f5f26684000

      mmap(0x7f5f2668a000, 52624, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS,
-1, 0) = 0x7f5f2668a000

      close(3)                                  = 0

      mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7f5f26482000

      arch_prctl(ARCH_SET_FS, 0x7f5f26482740) = 0

      set_tid_address(0x7f5f26482a10)         = 1306

      set_robust_list(0x7f5f26482a20, 24)     = 0

      rseq(0x7f5f26483060, 0x20, 0, 0x53053053) = 0

      mprotect(0x7f5f26684000, 16384, PROT_READ) = 0

      mprotect(0x7f5f2669f000, 4096, PROT_READ) = 0

      mprotect(0x55c430ff4000, 4096, PROT_READ) = 0

      mprotect(0x7f5f266d9000, 8192, PROT_READ) = 0

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

      munmap(0x7f5f26697000, 19963)           = 0

      openat(AT_FDCWD, "commands.txt", O_RDONLY) = 3

      read(3, "0\n1 5 3\n2 10\n0\n1 5 3\n2 10\nexit\n", 1023) = 31

      write(1, "Static linking: using library 1\n"..., 90Static linking: using library 1

        - Area: rectangle (a * b)

        - Convert: to binary system

      ) = 90

      write(1, "Area with sides 5.00 and 3.00 = "..., 38Area with sides 5.00 and 3.00 = 15.00

      ) = 38

      getrandom("\x34\xa9\x0a\xce\xad\x80\x68\x92", 8, GRND_NONBLOCK) = 8

      brk(NULL)                                 = 0x55c451827000

      brk(0x55c451848000)                       = 0x55c451848000

      write(1, "Number 10 in binary: 1010\n", 26Number 10 in binary: 1010

      ) = 26

      write(1, "Static linking: using library 1\n"..., 90Static linking: using library 1

        - Area: rectangle (a * b)

        - Convert: to binary system

      ) = 90

      write(1, "Area with sides 5.00 and 3.00 = "..., 38Area with sides 5.00 and 3.00 = 15.00
```

```
) = 38

write(1, "Number 10 in binary: 1010\n", 26Number 10 in binary: 1010

) = 26

close(3)                                = 0

exit_group(0)                           = ?

+++ exited with 0 +++

suslik@WIN-L3ULFBUQJMS:/mnt/c/Users/daria/Desktop/уч и зад/3 сем/оси/lr4$
```

2.

```
suslik@WIN-L3ULFBUQJMS:/mnt/c/Users/daria/Desktop/уч и зад/3 сем/оси/lr4$ strace -f
./prog2 commands.txt

execve("./prog2", ["./prog2", "commands.txt"], 0x7ffe6ad3efc0 /* 27 vars */) = 0

brk(NULL)                               = 0x55df20f78000

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7fba5a13d000

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=19963, ...}) = 0

mmap(NULL, 19963, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7fba5a138000

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) = 0x7fba59f26000

mmap(0x7fba59f4e000, 1605632, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x28000) = 0x7fba59f4e000

mmap(0x7fba5a0d6000, 323584, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x1b0000) = 0x7fba5a0d6000

mmap(0x7fba5a125000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x1fe000) = 0x7fba5a125000

mmap(0x7fba5a12b000, 52624, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS,
-1, 0) = 0x7fba5a12b000

close(3)                                = 0

mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7fba59f23000

arch_prctl(ARCH_SET_FS, 0x7fba59f23740) = 0
```

```
set_tid_address(0x7fba59f23a10)         = 1320

set_robust_list(0x7fba59f23a20, 24)     = 0

rseq(0x7fba59f24060, 0x20, 0, 0x53053053) = 0

mprotect(0x7fba5a125000, 16384, PROT_READ) = 0

mprotect(0x55dee513c000, 4096, PROT_READ) = 0

mprotect(0x7fba5a175000, 8192, PROT_READ) = 0

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

munmap(0x7fba5a138000, 19963)           = 0

getrandom("\xe2\x9e\xf9\x3e\x18\xa7\x51\xb2", 8, GRND_NONBLOCK) = 8

brk(NULL)                               = 0x55df20f78000

brk(0x55df20f99000)                     = 0x55df20f99000

openat(AT_FDCWD, "./lib1.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

fstat(3, {st_mode=S_IFREG|0777, st_size=15504, ...}) = 0

getcwd("/mnt/c/Users/daria/Desktop/\321\203\321\207 \320\270 \320\267\320\260\320\264/3
\321\201\320\265\320\274/\320\276\321\201\320\270/lr4", 128) = 62

mmap(NULL, 16408, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7fba5a138000

mmap(0x7fba5a139000, 4096, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x7fba5a139000

mmap(0x7fba5a13a000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x2000) = 0x7fba5a13a000

mmap(0x7fba5a13b000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7fba5a13b000

close(3)                                = 0

mprotect(0x7fba5a13b000, 4096, PROT_READ) = 0

openat(AT_FDCWD, "commands.txt", O_RDONLY) = 3

read(3, "0\n1 5 3\n2 10\n0\n1 5 3\n2 10\nexit\n", 1023) = 31

write(1, "Switching from library 1 to libr"..., 38Switching from library 1 to library 2

) = 38

munmap(0x7fba5a138000, 16408)           = 0

openat(AT_FDCWD, "./lib2.so", O_RDONLY|O_CLOEXEC) = 4

read(4, "\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

fstat(4, {st_mode=S_IFREG|0777, st_size=15576, ...}) = 0

getcwd("/mnt/c/Users/daria/Desktop/\321\203\321\207 \320\270 \320\267\320\260\320\264/3
\321\201\320\265\320\274/\320\276\321\201\320\270/lr4", 128) = 62

mmap(NULL, 16408, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 4, 0) = 0x7fba5a138000
```

mmap(0x7fba5a139000, 4096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 4, 0x1000) = 0x7fba5a139000

mmap(0x7fba5a13a000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 4, 0x2000) = 0x7fba5a13a000

mmap(0x7fba5a13b000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 4, 0x2000) = 0x7fba5a13b000

close(4) = 0

mprotect(0x7fba5a13b000, 4096, PROT_READ) = 0

write(1, "Loaded library 2: triangle area,"..., 52Loaded library 2: triangle area, ternary conversion

) = 52

write(1, "Area with sides 5.00 and 3.00 = "..., 37Area with sides 5.00 and 3.00 = 7.50

) = 37

write(1, "Number 10 in ternary: 101\n", 26Number 10 in ternary: 101

) = 26

write(1, "Switching from library 2 to libr"..., 38Switching from library 2 to library 1

) = 38

munmap(0x7fba5a138000, 16408) = 0

openat(AT_FDCWD, "./lib1.so", O_RDONLY|O_CLOEXEC) = 4

read(4, "\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\0"..., 832) = 832

fstat(4, {st_mode=S_IFREG|0777, st_size=15504, ...}) = 0

getcwd("/mnt/c/Users/daria/Desktop/\321\203\321\207 \320\270 \320\267\320\260\320\264/3\321\201\320\265\320\274/\320\276\321\201\320\270/lr4", 128) = 62

mmap(NULL, 16408, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 4, 0) = 0x7fba5a138000

mmap(0x7fba5a139000, 4096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 4, 0x1000) = 0x7fba5a139000

mmap(0x7fba5a13a000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 4, 0x2000) = 0x7fba5a13a000

mmap(0x7fba5a13b000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 4, 0x2000) = 0x7fba5a13b000

close(4) = 0

mprotect(0x7fba5a13b000, 4096, PROT_READ) = 0

write(1, "Loaded library 1: rectangle area"..., 52Loaded library 1: rectangle area, binary conversion

) = 52

write(1, "Area with sides 5.00 and 3.00 = "..., 38Area with sides 5.00 and 3.00 = 15.00

) = 38

write(1, "Number 10 in binary: 1010\n", 26Number 10 in binary: 1010

```
) = 26
close(3)                                = 0
munmap(0x7fba5a138000, 16408)           = 0
exit_group(0)                           = ?
+++ exited with 0 +++
suslik@WIN-L3ULFBUQJMS:/mnt/c/Users/daria/Desktop/уч и зад/3 сем/оси/lr4$
```

Вывод

В ходе выполнения лабораторной работы были успешно созданы динамические библиотеки для Linux и реализованы две программы, демонстрирующие различные подходы к их использованию: статическое связывание на этапе компиляции и динамическую загрузку во время выполнения с использованием интерфейса dlfcn.h.