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-213БВ-24

Студент: Месропян А.Э.

Преподаватель: Бахарев В.Д.

Оценка: _____

Дата: 21.11.25

Москва, 2025

Постановка задачи

Вариант 36.

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 ее 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 ее выполнения.

Функция 1:

8. 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

Функция 2:

9. Отсортировать целочисленный массив:

Сигнатура функции: int *sort(int *array, size_t n);

Реализация №1: Пузырьковая сортировка

Реализация №2: Сортировка Хоара

Общий метод и алгоритм решения

Программа 1: Статическая Линковка

Общий метод: Создание исполняемого файла, в который функции convert (перевод в двоичную систему) и sort (пузырьковая сортировка) встраиваются на этапе компиляции, образуя единый бинарный файл. Взаимодействие с пользователем происходит через консольный цикл ввода/вывода.

Использованные функции:

1. read(int fd, void *buf, size_t count): Чтение пользовательского ввода из STDIN_FD.
2. write(int fd, const void *buf, size_t count): Вывод результатов и сообщений в STDOUT_FD.
3. strlen(const char *s): Определение длины строки для вывода.
4. snprintf(char *str, size_t size, const char *format, ...): Форматированный вывод в буфер.
5. malloc() и free(): Управление динамической памятью для результата функции convert.
6. parse_int(const char *str, int *val): Вспомогательная функция для парсинга целых чисел из строки.

Алгоритм работы программы (prog1_static.c):

Инициализация

- Программа запускается и выводит приветственное сообщение и список доступных функций (1: convert – двоичный перевод, 2: sort – пузырьковая сортировка).

Цикл обработки команд

- Основной цикл: Считывание команды и аргументов из STDIN с помощью read(). Цикл продолжается, пока не введена команда "q".
- Парсинг команды: Первый символ строки определяет команду (1 или 2). Остальная часть строки передается как аргументы.

Обработка команды 1 (handle_function_1)

- Входные данные: Целое число x.
- Парсинг: Аргумент парсится в целое число с помощью parse_int().
- Вызов функции: Вызывается статически связанная функция char* convert(int x).

- Логика convert: Преобразует число x в строковое представление в **двоичной системе счисления** (v1).
- Вывод: Результат (двоичная строка) выводится в STDOUT, и выделенная память освобождается (free).

Обработка команды 2 (handle_function_2)

- Входные данные: Список целых чисел в строке.
- Парсинг: Строка аргументов парсится, извлекая до 100 целых чисел.
- Копирование: Числа копируются в динамически выделенный массив.
- Вызов функции: Вызывается статически связанная функция int* sort(int* array, size_t n).
- Логика sort: Сортирует массив с использованием алгоритма **Пузырьковой сортировки** (v1).
- Вывод: Исходный и отсортированный массивы выводятся в STDOUT. Выделенная память освобождается.

Завершение работы

- При вводе "q" цикл завершается, и программа выходит с кодом 0.

Программа 2: Динамическая Линковка

Общий метод: Реализация интерфейса, который динамически загружает и выгружает реализации функций (convert и sort) из внешних разделяемых библиотек (.so) во время выполнения, позволяя переключаться между разными версиями функций без перекомпиляции основной программы.

Использованные системные вызовы (Dynamic Linking):

1. void *dlopen(const char *filename, int flag): Загрузка разделяемой библиотеки (библиотеки v1 и v2 для каждой функции).
2. void *dlsym(void *handle, const char *symbol): Получение адреса функции (convert или sort) из загруженной библиотеки.
3. int dlclose(void *handle): Выгрузка разделяемой библиотеки.
4. write() и read(): Используются для пользовательского ввода/вывода.

Алгоритм работы программы (prog2_dynamic.c):

Инициализация (load_libraries)

- **Загрузка библиотек:** Вызывается dlopen() для всех четырех библиотек:

    - Ф1 V1: libconvert_bin.so (двоичный перевод)
    - Ф1 V2: libconvert_tri.so (троичный перевод)
    - Ф2 V1: libsort_bubble.so (пузырьковая сортировка)
    - Ф2 V2: libsort_hoare.so (сортировка Хоара/быстрая)
- **Получение символов:** С помощью dlsym() устанавливаются начальные указатели:
    - current_convert -> convert из V1 (двоичный).
    - current_sort -> sort из V1 (пузырьковая).
- Программа начинает работу, выводя информацию о текущих версиях.

Цикл обработки команд

- Основной цикл: Чтение команд из STDIN с помощью custom_read_line().

Обработка команды 0 (switch_implementations)

- **Переключение Ф1:** Текущая версия для convert меняется (1 -> 2 или 2 -> 1). Новый указатель на функцию получается через dlsym() из соответствующего хэндла (lib_handle_1_v1 или lib_handle_1_v2).
- **Переключение Ф2:** Текущая версия для sort меняется (1 -> 2 или 2 -> 1). Новый указатель на функцию получается через dlsym() из соответствующего хэндла (lib_handle_2_v1 или lib_handle_2_v2).
- Вывод: Сообщение о том, какие реализации теперь активны.

Обработка команды 1 (handle_function_1)

- Вызов: Вызывается функция через указатель current_convert(), который может указывать как на двоичную (V1), так и на **троичную** (V2) реализацию.
- Вывод: Результат и текущая активная версия функции выводятся в STDOUT.

Обработка команды 2 (handle_function_2)

- Вызов: Вызывается функция через указатель current_sort(), который может указывать как на **пузырьковую** (V1), так и на **Хоара (быструю)** (V2) реализацию.
- Вывод: Исходный и отсортированный массивы, а также текущая активная версия функции выводятся в STDOUT.

Завершение работы

- При вводе "q" цикл завершается.

- **Выгрузка библиотек (unload_libraries):** Вызывается dlclose() для всех четырех загруженных хэндлов, освобождая системные ресурсы.
- Программа выходит с кодом 0.

Код программы

contract.h

```h
#ifndef CONTRACT_H
#define CONTRACT_H

#include <stddef.h>

typedef char*(*ConvertFunc)(int);

typedef int*(*SortFunc)(int* array, size_t n);

#endif
```

lib1_v1.c

```c
#include "../include/contract.h"
#include <stdlib.h>
#include <string.h>

#define BUF_SIZE 66

char* convert(int x) {
    if (x == 0) {
        char* res = (char *)malloc(2);
        if (!res) {
            return NULL;
        }
        strcpy(res, "0");
        return res;
    }
    char* buf = (char *)malloc(BUF_SIZE);
    if (!buf) {
        return NULL;
    }
    int temp = x;
    int is_negative = (x < 0);
    if (is_negative) {
        temp = -temp;
    }
    int i = BUF_SIZE - 2;
    buf[BUF_SIZE - 1] = '\0';
    while (temp > 0) {
        buf[i--] = (temp % 2) + '0';
        temp /= 2;
    }
    if (is_negative) {
        buf[i--] = '-';
    }
    char* res = strdup(&buf[i + 1]);
    free(buf);
    return res;
}
```

lib1_v2.c

```c
#include "../include/contract.h"
#include <stdlib.h>
#include <string.h>

#define BUF_SIZE 66

char* convert(int x) {
    if (x == 0) {
        char* res = (char *)malloc(2);
        if (res == NULL) {
            return NULL;
        }
        strcpy(res, "0");
        return res;
    }
    int temp = x;
    int is_negative = (temp < 0);
    if (is_negative) {
        temp = -temp;
    }
    char* buf = (char *)malloc(BUF_SIZE);
    if (buf == NULL) {
        return NULL;
    }
    int i = BUF_SIZE - 2;
    buf[BUF_SIZE - 1] = '\0';
    while (temp > 0) {
        buf[i--] = (temp % 3) + '0';
        temp /= 3;
    }

    if (is_negative) {
        buf[i--] = '-';
    }
    char* res = strdup(&buf[i + 1]);
    return res;
}
```

lib2_v1.c

```c
#include "../include/contract.h"

int* sort(int* array, size_t n) {
    if (array == NULL || n == 0) {
        return array;
    }

    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (array[j] > array[j + 1]) {
                int temp = array[j];
                array[j] = array[j + 1];
                array[j + 1] = temp;
            }
        }
    }
    return array;

}
```

lib2_v2.c

```c
#include "../include/contract.h"

static void swap(int *a, int *b) {
    int t = *a;
    *a = *b;
    *b = t;
}

static size_t partition(int *array, size_t low, size_t high) {
    int pivot = array[high];
    size_t i = low;

    for (size_t j = low; j < high; j++) {
        if (array[j] <= pivot) {
            swap(&array[i], &array[j]);
            i++;
        }
    }
    swap(&array[i], &array[high]);
    return i;
}

static void quick_sort_recursive(int *array, size_t low, size_t high) {
    if (low < high) {
        size_t pi = partition(array, low, high);

        if (pi > 0) quick_sort_recursive(array, low, pi - 1);
        quick_sort_recursive(array, pi + 1, high);
    }
}

int *sort(int *array, size_t n) {
    if (array == NULL || n < 2) return array;
    quick_sort_recursive(array, 0, n - 1);
    return array;
}
```

prog1_static.c

```c
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <unistd.h>
#include "../include/contract.h"

int snprintf(char *str, size_t size, const char *format, ...);

char* convert(int x);
int* sort(int* array, size_t n);

#define STDOUT_FD 1
#define STDIN_FD 0

static void print_str(const char* s) {
    write(STDOUT_FD, s, strlen(s));
}

static void print_format(const char* format, int num) {
    char buf[128];
    int len = snprintf(buf, sizeof(buf), format, num);
    if (len > 0) {
        write(STDOUT_FD, buf, (size_t)len);
    }
}

static int parse_int(const char *str, int *val) {
    if (!str || !*str) return 0;
    long result = 0;
    int sign = 1;
    const char *p = str;

    while (*p && isspace(*p)) p++;

    if (*p == '-') {
        sign = -1;
        p++;
    } else if (*p == '+') {
        p++;
    }

    if (!isdigit(*p)) return 0;

    while (isdigit(*p)) {
        result = result * 10 + (*p - '0');
```

```
        p++;
    }

    *val = (int)(result * sign);
    return 1;
}


static void handle_function_1(const char* arg_str) {
    int x;
    if (!parse_int(arg_str, &x)) {
        print_str("Ошибка ввода. Для функции 1 требуется целое число\n");
        return;
    }

    char* res = convert(x);
    if (res) {
        print_str("Результат(двоичный) : ");
        print_str(res);
        print_str("\n");
        free(res);
    } else {
        print_str("Ошибка выделения памяти\n");
    }
}

static void handle_function_2(const char* arg_str) {
    int temp_arr[100];
    size_t count = 0;
    const char *p = arg_str;

    while (*p) {
        int val;
        while (*p && isspace(*p)) p++;

        const char *start = p;
        if (*p == '-' || *p == '+') p++;
        while (*p && isdigit(*p)) p++;
        const char *end = p;

        if (end > start) {
            char num_str[30];
            size_t len = end - start;
            if (len >= sizeof(num_str)) len = sizeof(num_str) - 1;
            memcpy(num_str, start, len);
```

```
            num_str[len] = '\0';

            if (parse_int(num_str, &val)) {
                temp_arr[count++] = val;
                if (count >= 100) break;
            }
        }

        if (*p == '\0' || end == start) break;
    }

    if (count == 0) {
        print_str("Ошибка ввода. Требуется список целых чисел\n");
        return;
    }

    int* arr_to_sort = (int *)malloc(count * sizeof(int));
    if (arr_to_sort == NULL) {
        print_str("Ошибка выделения памяти\n");
        return;
    }

    memcpy(arr_to_sort, temp_arr, count * sizeof(int));

    print_str("Исходный массив: ");
    for (size_t i = 0; i < count; i++) {
        print_format("%d ", temp_arr[i]);
    }
    print_str("\n\n");

    int* sorted_arr = sort(arr_to_sort, count);

    print_str("Результат (Пузырьковая): ");
    for(size_t i = 0; i < count; i++) {
        print_format("%d ", sorted_arr[i]);
    }
    print_str("\n\n");

    free(arr_to_sort);
}


int main() {
    char line[512];
    ssize_t bytes_read;
```

```
    print_str("Программа 1: Статическая линковка\n");
    print_str("Функция 1: Перевод в двоичную систему счисления\n");
    print_str("Функция 2: Пузырьковая сортировка\n");
    print_str("Введите команду: (1 [число] или 2 [массив чисел] или 'q' для выхода)\n");

    while ((bytes_read = read(STDIN_FD, line, sizeof(line) - 1)) > 0) {
        line[bytes_read] = '\0';

        if (line[bytes_read - 1] == '\n') {
            line[bytes_read - 1] = '\0';
        }

        if (line[0] == 'q') {
            break;
        }

        if (line[0] == '\0') continue;

        int cmd = line[0] - '0';
        char* args = line + 1;
        while (*args == ' ' || *args == '\t') {
            args++;
        }

        switch(cmd) {
            case 1:
                handle_function_1(args);
                break;
            case 2:
                handle_function_2(args);
                break;
            default:
                print_str("Недоступная команда, введите 1 или 2\n");
        }

        print_str("Введите команду: (1 [число] или 2 [массив чисел] или 'q' для выхода)\n");
    }
    return 0;
}
```

prog2_dynamic.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include "../include/contract.h"
#include <dlfcn.h>
#include <unistd.h>
#include <stdarg.h>

static ConvertFunc current_convert = NULL;
static SortFunc current_sort = NULL;

static void *lib_handle_1_v1 = NULL;
static void *lib_handle_1_v2 = NULL;
static void *lib_handle_2_v1 = NULL;
static void *lib_handle_2_v2 = NULL;

static int current_impl_1 = 1;
static int current_impl_2 = 1;

#define LIB1_V1_PATH "bin/libconvert_bin.so"
#define LIB1_V2_PATH "bin/libconvert_tri.so"
#define LIB2_V1_PATH "bin/libsort_bubble.so"
#define LIB2_V2_PATH "bin/libsort_hoare.so"

#define MAX_OUTPUT_BUFFER 512

static void custom_write(int fd, const char *format, ...) {
    char buffer[MAX_OUTPUT_BUFFER];
    va_list args;
    va_start(args, format);
    int len = snprintf(buffer, sizeof(buffer), "%s", format);
    va_end(args);

    if (len > 0) {
        write(fd, buffer, (size_t)len);
    }
}

static int custom_read_line(char *buffer, size_t max_len) {
    ssize_t bytes_read = read(STDIN_FILENO, buffer, max_len - 1);
    if (bytes_read <= 0) {
        return 0;
    }
    buffer[bytes_read] = '\0';
```

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

    return (int)bytes_read;
}


static int load_libraries() {
    lib_handle_1_v1 = dlopen(LIB1_V1_PATH, RTLD_LAZY);
    lib_handle_1_v2 = dlopen(LIB1_V2_PATH, RTLD_LAZY);
    lib_handle_2_v1 = dlopen(LIB2_V1_PATH, RTLD_LAZY);
    lib_handle_2_v2 = dlopen(LIB2_V2_PATH, RTLD_LAZY);

    if (!lib_handle_1_v1 || !lib_handle_1_v2 || !lib_handle_2_v1 || !lib_handle_2_v2) {
        char err_msg[MAX_OUTPUT_BUFFER];
        snprintf(err_msg, sizeof(err_msg), "Ошибка загрузки одной из библиотек: %s\n", dlerror());
        write(STDERR_FILENO, err_msg, strlen(err_msg));
        return 0;
    }

    current_convert = (ConvertFunc)dlsym(lib_handle_1_v1, "convert");
    current_sort = (SortFunc)dlsym(lib_handle_2_v1, "sort");

    if (!current_convert || !current_sort) {
        char err_msg[MAX_OUTPUT_BUFFER];
        snprintf(err_msg, sizeof(err_msg), "Ошибка поиска символа 'convert' или 'sort': %s\n", dlerror());
        write(STDERR_FILENO, err_msg, strlen(err_msg));
        return 0;
    }

    return 1;
}

static void unload_libraries() {
    if (lib_handle_1_v1) dlclose(lib_handle_1_v1);
    if (lib_handle_1_v2) dlclose(lib_handle_1_v2);
    if (lib_handle_2_v1) dlclose(lib_handle_2_v1);
    if (lib_handle_2_v2) dlclose(lib_handle_2_v2);
}
```

```
static void switch_implementations() {

    current_impl_1 = (current_impl_1 == 1) ? 2 : 1;
    void *target_handle_1 = (current_impl_1 == 1) ? lib_handle_1_v1 : lib_handle_1_v2;
    current_convert = (ConvertFunc)dlsym(target_handle_1, "convert");

    current_impl_2 = (current_impl_2 == 1) ? 2 : 1;
    void *target_handle_2 = (current_impl_2 == 1) ? lib_handle_2_v1 : lib_handle_2_v2;
    current_sort = (SortFunc)dlsym(target_handle_2, "sort");

    if (!current_convert || !current_sort) {
        char err_msg[MAX_OUTPUT_BUFFER];
        snprintf(err_msg, sizeof(err_msg), "Ошибка переключения: не удалось найти символ.\n");
        write(STDERR_FILENO, err_msg, strlen(err_msg));
        return;
    }

    custom_write(STDOUT_FILENO, "--- РЕАЛИЗАЦИИ ПЕРЕКЛЮЧЕНЫ ---\n");

    char output[MAX_OUTPUT_BUFFER];
    snprintf(output, sizeof(output), "Функция 1: %s\n", (current_impl_1 == 1) ? "Двоичная (V1)" : "Троичная (V2)");
    write(STDOUT_FILENO, output, strlen(output));

    snprintf(output, sizeof(output), "Функция 2: %s\n", (current_impl_2 == 1) ? "Пузырьковая (V1)" : "Хоара (V2)");
    write(STDOUT_FILENO, output, strlen(output));
}

static void handle_function_1(const char *arg_str) {
    if (!current_convert) {
        custom_write(STDOUT_FILENO, "Ошибка: Функция 1 не загружена.\n");
        return;
    }

    int x;
    if (sscanf(arg_str, "%d", &x) != 1) {
        custom_write(STDOUT_FILENO, "Ошибка ввода: требуется целое число для функции 1.\n");
        return;
    }

    char *result = current_convert(x);

    if (result) {
        char output[MAX_OUTPUT_BUFFER];
        snprintf(output, sizeof(output), "Результат (%s): %s\n",
            (current_impl_1 == 1) ? "Двоичный" : "Троичный", result);
```

```
        write(STDOUT_FILENO, output, strlen(output));
        free(result);
    } else {
        custom_write(STDOUT_FILENO, "Ошибка выделения памяти.\n");
    }
}

static void handle_function_2(const char *arg_str) {
    if (!current_sort) {
        custom_write(STDOUT_FILENO, "Ошибка: Функция 2 не загружена.\n");
        return;
    }

    int temp_array[100];
    size_t count = 0;
    const char *p = arg_str;

    while (sscanf(p, "%d", &temp_array[count]) == 1) {
        count++;
        while (*p && !isspace(*p)) p++;
        while (*p && isspace(*p)) p++;
        if (count >= 100) break;
    }

    if (count == 0) {
        custom_write(STDOUT_FILENO, "Ошибка ввода: требуется список целых чисел для функции 2.\n");
        return;
    }

    int *array_to_sort = (int*)malloc(count * sizeof(int));
    if (!array_to_sort) {
        custom_write(STDOUT_FILENO, "Ошибка выделения памяти.\n");
        return;
    }
    memcpy(array_to_sort, temp_array, count * sizeof(int));

    char output[MAX_OUTPUT_BUFFER];
    size_t offset = 0;

    offset += snprintf(output + offset, sizeof(output) - offset, "Исходный массив: ");
    for(size_t i = 0; i < count; i++) {
        offset += snprintf(output + offset, sizeof(output) - offset, "%d ", temp_array[i]);
    }
```

```
    offset += snprintf(output + offset, sizeof(output) - offset, "\n");
    write(STDOUT_FILENO, output, strlen(output));

    int *sorted_array = current_sort(array_to_sort, count);

    offset = 0;
    offset += snprintf(output + offset, sizeof(output) - offset, "Результат (%s): ",
        (current_impl_2 == 1) ? "Пузырьковая" : "Хоара");
    for(size_t i = 0; i < count; i++) {
        offset += snprintf(output + offset, sizeof(output) - offset, "%d ", sorted_array[i]);
    }
    offset += snprintf(output + offset, sizeof(output) - offset, "\n");
    write(STDOUT_FILENO, output, strlen(output));

    free(array_to_sort);


int main() {
    if (!load_libraries()) {
        custom_write(STDERR_FILENO, "Критическая ошибка инициализации. Выход.\n");
        unload_libraries();
        return EXIT_FAILURE;
    }

    char line[512];

    custom_write(STDOUT_FILENO, "--- Программа 2: Динамическая загрузка ---\n");
    custom_write(STDOUT_FILENO, "Начальные реализации: Ф1 - Двоичная, Ф2 - Пузырьковая.\n");
    custom_write(STDOUT_FILENO, "Введите команду (0 - переключить, 1 [число], 2 [массив чисел], q - выход):\n");

    while (custom_read_line(line, sizeof(line))) {
        if (line[0] == 'q' || line[0] == 'Q') break;

        int cmd = line[0] - '0';
        char *args = line + 1;
        while (*args == ' ' || *args == '\t') args++;

        switch (cmd) {
            case 0:
                switch_implementations();
                break;
            case 1:
                handle_function_1(args);
```

```
                break;
            case 2:
                handle_function_2(args);
                break;
            default:
                custom_write(STDOUT_FILENO, "Неверная команда. Используйте 0, 1 или 2.\n");
                break;
        }
    }

    unload_libraries();
    return 0;
}
```

Протокол работы программы

Тестирование:

```
armani@LAPTOP-F5TL4SI7:~/OS_Labs/build$ ./bin/lab4_prog1
Программа 1: Статическая линковка
Функция 1: Перевод в двоичную систему счисления
Функция 2: Пузырьковая сортировка
Введите команду: (1 [число] или 2 [массив чисел] или 'q' для выхода)
1 6
Результат(двоичный) : 110
Введите команду: (1 [число] или 2 [массив чисел] или 'q' для выхода)
2 4 5 3 6 2
Исходный массив: 4 5 3 6 2

Результат (Пузырьковая): 2 3 4 5 6

Введите команду: (1 [число] или 2 [массив чисел] или 'q' для выхода)
q
armani@LAPTOP-F5TL4SI7:~/OS_Labs/build$ ./bin/lab4_prog2
--- Программа 2: Динамическая загрузка ---
Начальные реализации: Ф1 - Двоичная, Ф2 - Пузырьковая.
Введите команду (0 - переключить, 1 [число], 2 [массив чисел], q - выход):
1 8
Результат (Двоичный): 1000
2 3 4 7 6 4 5 3
Исходный массив: 3 4 7 6 4 5 3
Результат (Пузырьковая): 3 3 4 4 5 6 7
0
--- РЕАЛИЗАЦИИ ПЕРЕКЛЮЧЕНЫ ---
Функция 1: Троичная (V2)
Функция 2: Хоара (V2)
1 9
Результат (Троичный): 100
2 5 4 6 4 6 4 6 4 7 4
Исходный массив: 5 4 6 4 6 4 6 4 7 4
Результат (Хоара): 4 4 4 4 4 5 6 6 6 7
q
```

Strace:

armani@LAPTOP-F5TL4SI7:~/OS_Labs/build$ strace -f ./bin/lab4_prog1

execve("./bin/lab4_prog1", ["./bin/lab4_prog1"], 0x7ffd3b973738 /* 37 vars */) = 0

brk(NULL) = 0x5c3f5284b000

arch_prctl(0x3001 /* ARCH_??? */, 0x7ffe52ff6b60) = -1 EINVAL (Invalid argument)

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x768d9f2c6000

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

openat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/glibc-hwcaps/x86-64-v3/libconvert_bin.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

newfstatat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/glibc-hwcaps/x86-64-v3", 0x7ffe52ff5d80, 0) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/glibc-hwcaps/x86-64-v2/libconvert_bin.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

newfstatat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/glibc-hwcaps/x86-64-v2", 0x7ffe52ff5d80, 0) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/tls/x86_64/x86_64/libconvert_bin.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

newfstatat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/tls/x86_64/x86_64", 0x7ffe52ff5d80, 0) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/tls/x86_64/libconvert_bin.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

newfstatat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/tls/x86_64", 0x7ffe52ff5d80, 0) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/tls/x86_64/libconvert_bin.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

newfstatat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/tls/x86_64", 0x7ffe52ff5d80, 0) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/tls/libconvert_bin.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

newfstatat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/tls", 0x7ffe52ff5d80, 0) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/x86_64/x86_64/libconvert_bin.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

newfstatat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/x86_64/x86_64", 0x7ffe52ff5d80, 0) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/x86_64/libconvert_bin.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

newfstatat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/x86_64", 0x7ffe52ff5d80, 0) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/x86_64/libconvert_bin.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

newfstatat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/x86_64", 0x7ffe52ff5d80, 0) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/libconvert_bin.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=15600, ...}, AT_EMPTY_PATH) = 0

mmap(NULL, 16448, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x768d9f2c1000

mmap(0x768d9f2c2000, 4096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x768d9f2c2000

mmap(0x768d9f2c3000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x768d9f2c3000

mmap(0x768d9f2c4000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x768d9f2c4000

close(3) = 0

```
openat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/libsort_bubble.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|0755, st_size=15136, ...}, AT_EMPTY_PATH) = 0

mmap(NULL, 16424, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x768d9f2bc000

mmap(0x768d9f2bd000, 4096, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x768d9f2bd000

mmap(0x768d9f2be000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x2000) = 0x768d9f2be000

mmap(0x768d9f2bf000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x768d9f2bf000

close(3)                        = 0

openat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/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

newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=17368, ...}, AT_EMPTY_PATH) = 0

mmap(NULL, 17368, PROT_READ, MAP_PRIVATE, 3, 0) = 0x768d9f2b7000

close(3)                        = 0

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

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

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

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

pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0O{\f\225\\=\201\327\312\301P\32$\230\266\235"..., 68, 896)
= 68

newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=2220400, ...}, 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"..., 784, 64) = 784

mmap(NULL, 2264656, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x768d9f000000

mprotect(0x768d9f028000, 2023424, PROT_NONE) = 0

mmap(0x768d9f028000, 1658880, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x768d9f028000

mmap(0x768d9f1bd000, 360448, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x1bd000) = 0x768d9f1bd000

mmap(0x768d9f216000, 24576, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x215000) = 0x768d9f216000

mmap(0x768d9f21c000, 52816, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x768d9f21c000

close(3)                        = 0

mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x768d9f2b4000
```

```
arch_prctl(ARCH_SET_FS, 0x768d9f2b4740) = 0

set_tid_address(0x768d9f2b4a10)         = 60825

set_robust_list(0x768d9f2b4a20, 24)     = 0

rseq(0x768d9f2b50e0, 0x20, 0, 0x53053053) = 0

mprotect(0x768d9f216000, 16384, PROT_READ) = 0

mprotect(0x768d9f2bf000, 4096, PROT_READ) = 0

mprotect(0x768d9f2c4000, 4096, PROT_READ) = 0

mprotect(0x5c3f1bc96000, 4096, PROT_READ) = 0

mprotect(0x768d9f300000, 8192, PROT_READ) = 0

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

munmap(0x768d9f2b7000, 17368)           = 0

write(1, "\320\237\321\200\320\276\320\263\321\200\320\260\320\274\320\274\320\260 1:
\320\241\321\202\320\260\321\202\320\270"..., 62Программа 1: Статическая линковка

) = 62

write(1, "\320\244\321\203\320\275\320\272\321\206\320\270\321\217 1:
\320\237\320\265\321\200\320\265\320\262\320\276\320\264"..., 87Функция 1: Перевод в двоичную
систему счисления

) = 87

write(1, "\320\244\321\203\320\275\320\272\321\206\320\270\321\217 2:
\320\237\321\203\320\267\321\213\321\200\321\214\320\272"..., 62Функция 2: Пузырьковая сортировка

) = 62

write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265
\320\272\320\276\320\274\320\260\320\275\320\264\321\203: ("..., 114Введите команду: (1 [число] или 2
[массив чисел] или 'q' для выхода)

) = 114

read(0, 1 6

"1 6\n", 511)                = 4

getrandom("\xee\x6d\x81\xae\x04\xb0\x25\xb2", 8, GRND_NONBLOCK) = 8

brk(NULL)                           = 0x5c3f5284b000

brk(0x5c3f5286c000)                 = 0x5c3f5286c000

write(1,
"\320\240\320\265\320\267\321\203\320\273\321\214\321\202\320\260\321\202(\320\264\320\262\320\276\32
0\270\321\207\320\275\321"..., 39Результат(двоичный) : ) = 39

write(1, "110", 3110)                = 3

write(1, "\n", 1

)                   = 1

write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265
\320\272\320\276\320\274\320\260\320\275\320\264\321\203: ("..., 114Введите команду: (1 [число] или 2
[массив чисел] или 'q' для выхода)
```

) = 114

read(0, 2 5 6 4 6 3 8

"2 5 6 4 6 3 8\n", 511) = 14

write(1, "\320\230\321\201\321\205\320\276\320\264\320\275\321\213\320\271
\320\274\320\260\321\201\321\201\320\270\320\262: ", 31Исходный массив:) = 31

write(1, "5 ", 25) = 2

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

write(1, "4 ", 24) = 2

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

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

write(1, "8 ", 28) = 2

write(1, "\n\n", 2

) = 2

write(1, "\320\240\320\265\320\267\321\203\320\273\321\214\321\202\320\260\321\202
(\320\237\321\203\320\267\321\213\321\200\321\214"..., 45Результат (Пузырьковая):) = 45

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

write(1, "4 ", 24) = 2

write(1, "5 ", 25) = 2

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

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

write(1, "8 ", 28) = 2

write(1, "\n\n", 2

) = 2

write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265
\320\272\320\276\320\274\320\260\320\275\320\264\321\203: ("..., 114Введите команду: (1 [число] или 2
[массив чисел] или 'q' для выхода)

) = 114

read(0, q

"q\n", 511) = 2

exit_group(0) = ?

+++ exited with 0 +++

```
armani@LAPTOP-F5TL4SI7:~/OS_Labs/build$ strace -f ./bin/lab4_prog2

execve("./bin/lab4_prog2", ["./bin/lab4_prog2"], 0x7ffc6e0effd8 /* 37 vars */) = 0

brk(NULL)                               = 0x55f16b72d000

arch_prctl(0x3001 /* ARCH_??? */, 0x7ffda921c050) = -1 EINVAL (Invalid argument)

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7faa24a40000

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

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

newfstatat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/glibc-hwcaps/x86-64-v3", 0x7ffda921b270,
0) = -1 ENOENT (No such file or directory)

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

newfstatat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/glibc-hwcaps/x86-64-v2", 0x7ffda921b270,
0) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/tls/x86_64/x86_64/libc.so.6",
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

newfstatat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/tls/x86_64/x86_64", 0x7ffda921b270, 0) =
-1 ENOENT (No such file or directory)

openat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/tls/x86_64/libc.so.6",
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

newfstatat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/tls/x86_64", 0x7ffda921b270, 0) = -1
ENOENT (No such file or directory)

openat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/tls/x86_64/libc.so.6",
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

newfstatat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/tls/x86_64", 0x7ffda921b270, 0) = -1
ENOENT (No such file or directory)

openat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/tls/libc.so.6", O_RDONLY|O_CLOEXEC) = -
1 ENOENT (No such file or directory)

newfstatat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/tls", 0x7ffda921b270, 0) = -1 ENOENT
(No such file or directory)

openat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/x86_64/x86_64/libc.so.6",
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

newfstatat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/x86_64/x86_64", 0x7ffda921b270, 0) = -1
ENOENT (No such file or directory)

openat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/x86_64/libc.so.6",
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

newfstatat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/x86_64", 0x7ffda921b270, 0) = -1
ENOENT (No such file or directory)

openat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/x86_64/libc.so.6",
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

newfstatat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/x86_64", 0x7ffda921b270, 0) = -1
ENOENT (No such file or directory)
```

```
openat(AT_FDCWD, "/home/armani/OS_Labs/build/bin/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

newfstatat(AT_FDCWD, "/home/armani/OS_Labs/build/bin", {st_mode=S_IFDIR|0755, st_size=4096, ...}, 0) = 0

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

newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=17368, ...}, AT_EMPTY_PATH) = 0

mmap(NULL, 17368, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7faa24a3b000

close(3)                                = 0

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

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

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

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

pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0O{\f\225\\=\201\327\312\301P\32$\230\266\235"..., 68, 896) = 68

newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=2220400, ...}, 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"..., 784, 64) = 784

mmap(NULL, 2264656, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7faa24800000

mprotect(0x7faa24828000, 2023424, PROT_NONE) = 0

mmap(0x7faa24828000, 1658880, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x7faa24828000

mmap(0x7faa249bd000, 360448, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1bd000) = 0x7faa249bd000

mmap(0x7faa24a16000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x215000) = 0x7faa24a16000

mmap(0x7faa24a1c000, 52816, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7faa24a1c000

close(3)                                = 0

mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7faa24a38000

arch_prctl(ARCH_SET_FS, 0x7faa24a38740) = 0

set_tid_address(0x7faa24a38a10)         = 61027

set_robust_list(0x7faa24a38a20, 24)     = 0

rseq(0x7faa24a390e0, 0x20, 0, 0x53053053) = 0

mprotect(0x7faa24a16000, 16384, PROT_READ) = 0

mprotect(0x55f12c39e000, 4096, PROT_READ) = 0

mprotect(0x7faa24a7a000, 8192, PROT_READ) = 0

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

munmap(0x7faa24a3b000, 17368)           = 0
```

```
getrandom("\xc5\xf2\xb9\x9a\x40\x7a\xf7\x93", 8, GRND_NONBLOCK) = 8

brk(NULL)                               = 0x55f16b72d000

brk(0x55f16b74e000)                     = 0x55f16b74e000

openat(AT_FDCWD, "bin/libconvert_bin.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=15600, ...}, AT_EMPTY_PATH) = 0

getcwd("/home/armani/OS_Labs/build", 128) = 27

mmap(NULL, 16448, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7faa24a3b000

mmap(0x7faa24a3c000, 4096, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x7faa24a3c000

mmap(0x7faa24a3d000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x2000) = 0x7faa24a3d000

mmap(0x7faa24a3e000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7faa24a3e000

close(3)                                = 0

mprotect(0x7faa24a3e000, 4096, PROT_READ) = 0

openat(AT_FDCWD, "bin/libconvert_tri.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=15552, ...}, AT_EMPTY_PATH) = 0

getcwd("/home/armani/OS_Labs/build", 128) = 27

mmap(NULL, 16440, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7faa24a33000

mmap(0x7faa24a34000, 4096, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x7faa24a34000

mmap(0x7faa24a35000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x2000) = 0x7faa24a35000

mmap(0x7faa24a36000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7faa24a36000

close(3)                                = 0

mprotect(0x7faa24a36000, 4096, PROT_READ) = 0

openat(AT_FDCWD, "bin/libsort_bubble.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=15136, ...}, AT_EMPTY_PATH) = 0

getcwd("/home/armani/OS_Labs/build", 128) = 27

mmap(NULL, 16424, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7faa24a2e000

mmap(0x7faa24a2f000, 4096, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x7faa24a2f000

mmap(0x7faa24a30000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x2000) = 0x7faa24a30000
```

```
	mmap(0x7faa24a31000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7faa24a31000

	close(3)                        = 0

	mprotect(0x7faa24a31000, 4096, PROT_READ) = 0

	openat(AT_FDCWD, "bin/libsort_hoare.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=15240, ...}, AT_EMPTY_PATH) = 0

	getcwd("/home/armani/OS_Labs/build", 128) = 27

	mmap(NULL, 16424, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7faa24a29000

	mmap(0x7faa24a2a000, 4096, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x7faa24a2a000

	mmap(0x7faa24a2b000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x2000) = 0x7faa24a2b000

	mmap(0x7faa24a2c000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7faa24a2c000

	close(3)                        = 0

	mprotect(0x7faa24a2c000, 4096, PROT_READ) = 0

	write(1, "--- \320\237\321\200\320\276\320\263\321\200\320\260\320\274\320\274\320\260 2:
\320\224\320\270\320\275"..., 72--- Программа 2: Динамическая загрузка ---

	) = 72

	write(1, "\320\235\320\260\321\207\320\260\320\273\321\214\320\275\321\213\320\265
\321\200\320\265\320\260\320\273\320\270\320\267\320"..., 95Начальные реализации: Ф1 - Двоичная, Ф2 -
Пузырьковая.

	) = 95

	write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265
\320\272\320\276\320\274\320\260\320\275\320\264\321\203 (0"..., 121Введите команду (0 - переключить, 1
[число], 2 [массив чисел], q - выход):

	) = 121

	read(0, 1 8

	"1 8\n", 511)                   = 4

	write(1, "\320\240\320\265\320\267\321\203\320\273\321\214\321\202\320\260\321\202
(\320\224\320\262\320\276\320\270\321\207\320\275"..., 44Результат (Двоичный): 1000

	) = 44

	read(0, 2 6 5 3 5 8

	"2 6 5 3 5 8\n", 511)           = 12

	write(1, "\320\230\321\201\321\205\320\276\320\264\320\275\321\213\320\271
\320\274\320\260\321\201\321\201\320\270\320\262: 6"..., 42Исходный массив: 6 5 3 5 8

	) = 42
```

```
write(1, "\320\240\320\265\320\267\321\203\320\273\321\214\321\202\320\260\321\202
(\320\237\321\203\320\267\321\213\321\200\321\214"..., 56Результат (Пузырьковая): 3 5 5 6 8

) = 56

read(0, 0

"0\n", 511)                     = 2

write(1, "--- \320\240\320\225\320\220\320\233\320\230\320\227\320\220\320\246\320\230\320\230
\320\237\320\225\320\240\320"..., 52--- РЕАЛИЗАЦИИ ПЕРЕКЛЮЧЕНЫ ---

) = 52

write(1, "\320\244\321\203\320\275\320\272\321\206\320\270\321\217 1:
\320\242\321\200\320\276\320\270\321\207\320\275\320\260"..., 40Функция 1: Троичная (V2)

) = 40

write(1, "\320\244\321\203\320\275\320\272\321\206\320\270\321\217 2:
\320\245\320\276\320\260\321\200\320\260 (V2"..., 34Функция 2: Хоара (V2)

) = 34

read(0, 1 9

"1 9\n", 511)                   = 4

write(1, "\320\240\320\265\320\267\321\203\320\273\321\214\321\202\320\260\321\202
(\320\242\321\200\320\276\320\270\321\207\320\275"..., 43Результат (Троичный): 100

) = 43

read(0, 2 5 7 3 5 8

"2 5 7 3 5 8\n", 511)           = 12

write(1, "\320\230\321\201\321\205\320\276\320\264\320\275\321\213\320\271
\320\274\320\260\321\201\321\201\320\270\320\262: 5"..., 42Исходный массив: 5 7 3 5 8

) = 42

write(1, "\320\240\320\265\320\267\321\203\320\273\321\214\321\202\320\260\321\202
(\320\245\320\276\320\260\321\200\320\260):"..., 44Результат (Хоара): 3 5 5 7 8

) = 44

read(0, q

"q\n", 511)                     = 2

munmap(0x7faa24a3b000, 16448)           = 0

munmap(0x7faa24a33000, 16440)           = 0

munmap(0x7faa24a2e000, 16424)           = 0

munmap(0x7faa24a29000, 16424)           = 0

exit_group(0)                           = ?

+++ exited with 0 +++
```

Вывод

В ходе выполнения лабораторной работы были успешно освоены принципы организации динамической загрузки кода и изучены различия между статической и динамической линковкой. Основные трудности возникли при корректной инициализации и управлении жизненным циклом загруженных разделяемых объектов, а также при реализации механизма динамического переключения между версиями функций (convert и sort) с сохранением целостности данных. В процессе работы был детально изучен и применён набор системных вызовов и функций: dlopen() для загрузки разделяемых библиотек, dlsym() для получения указателей на функции в памяти, и dlclose() для выгрузки. Также были использованы read() и write() для низкоуровневого ввода-вывода. Это позволило получить глубокое понимание механизмов гибкого конфигурирования программ в операционной системе, научиться корректно управлять модульностью кода и его версиями во время исполнения. Полученные навыки будут полезны для выполнения последующих, более сложных задач, требующих создания расширяемых архитектур приложений с подключаемыми модулями.