МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ ФАКУЛЬТЕТ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

Утвержден на заседании кафедры

«Вычислительная техника» " " 20 г. Заведующий кафедрой

М.А. Митрохин

**ОТЧЕТ ПО УЧЕБНОЙ (ОЗНАКОМИТЕЛЬНОЙ) ПРАКТИКЕ**

(2024/2025 учебный год)

Выдрин Дмитрий Алексеевич

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

Наименование профиля подготовки «Прикладной искусственный интеллект»

Форма обучения – очная Срок обучения в соответствии с ФГОС – 4 года

Год обучения 1 семестр 2

Период прохождения практики с 25.06.2025 по 08.07.2025

Кафедра «Вычислительная техника»

Заведующий кафедрой д.т.н., профессор, Митрохин М.А.

*(должность, ученая степень, ученое звание, Ф.И.О.)*

Руководитель практики д.т.н., профессор, Зинкин С.А.

*(должность, ученая степень, ученое звание)*

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ ФАКУЛЬТЕТ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

Утвержден на заседании кафедры

«Вычислительная техника» " " 20 г. Заведующий кафедрой

М.А. Митрохин

**ИНДИВИДУАЛЬНЫЙ ПЛАН ПРОХОЖДЕНИЯ УЧЕБНОЙ (ОЗНАКОМИТЕЛЬНОЙ) ПРАКТИКИ**

(2024/2025 учебный год)

Выдрин Дмитрий Алексеевич

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

Наименование профиля подготовки «Прикладной искусственный интеллект»

Форма обучения – очная Срок обучения в соответствии с ФГОС – 4 года Год обучения 1 семестр 2 Период прохождения практики с 25.06.2025 по 08.07.2025

Кафедра «Вычислительная техника»

Заведующий кафедрой д.т.н., профессор, Митрохин М.А.

*(должность, ученая степень, ученое звание, Ф.И.О.)*

Руководитель практики д.т.н., профессор, Зинкин С.А.

*(должность, ученая степень, ученое звание)*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № п/п | Планируемая форма работы во  время практики | Количество часов | Календарные сроки проведения работы | Подпись  руководителя практики от вуза |
| 1 | Выбор темы и разработка  индивидуального  плана проведения работ | 2 | 25.06.2025 -  25.06.2025 |  |
| 2 | Подбор и изучение материала по теме  работы | 15 | 30.06.2025 –  02.07.25 |  |
| 3 | Разработка  алгоритма | 43 | 02.07.25 –  03.07.25 |  |
| 4 | Описание алгоритма и  программы | 18 | 04.07.25 –  05.07.25 |  |
| 5 | Тестирование | 5 | 05.07.25 –  06.07.25 |  |
| 6 | Получение и  анализ результатов | 10 | 06.07.25 –  06.07.25 |  |
| 7 | Оформление  отчёта | 15 | 06.07.25 –  06.07.2025 |  |
|  | **Общий объём**  **часов** | 108 |  |  |

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

ФАКУЛЬТЕТ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

**ОТЧЁТ**

**О ПРОХОЖДЕНИИ УЧЕБНОЙ (ОЗНАКОМИТЕЛЬНОЙ) ПРАКТИКИ**

(2024/2025 учебный год)

Выдрин Дмитрий Алексеевич

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

Наименование профиля подготовки «Прикладной искусственный интеллект»

Форма обучения – очная Срок обучения в соответствии с ФГОС – 4 года

Год обучения 1 семестр 2

Период прохождения практики с 25.06.2025 по 08.07.2025

Кафедра «Вычислительная техника»

Выдрин Д.А. выполнял практическое задание «Сортировка Шелла». На первоначальном этапе были изучен и проанализирован алгоритм сортировки Шелла, был выбран метод решения и язык программирования С, на котором была написана программа сортировки массива методом Шелла. Также, осуществил подсчёт времени выполнения сортировки и количества перестановок. Протестировал и отладил программу. Оформил отчёт.

Бакалавр Выдрин Д.А. " " 2025 г.

Руководитель Зинкин С.А. " " 2025 г. практики

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

ФАКУЛЬТЕТ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

**ОТЗЫВ**

**О ПРОХОЖДЕНИИ УЧЕБНОЙ (ОЗНАКОМИТЕЛЬНОЙ) ПРАКТИКИ**

(2024/2025 учебный год)

Выдрин Дмитрий Алексеевич

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

Наименование профиля подготовки «Прикладной искусственный интеллект»

Форма обучения – очная Срок обучения в соответствии с ФГОС – 4 года

Год обучения 1 семестр 2

Период прохождения практики с 25.06.2025 по 08.07.2025

Кафедра «Вычислительная техника»

В процессе выполнения практики Выдрин Д.А. решал следующие задачи: изучение алгоритма сортировки Шелла, анализ работы алгоритма, сравнение существующих методов сортировки, произведение подсчёта времени выполнения сортировки и количества перестановок, тестирование и отладка программы.

За период выполнения практики были освоены основные понятия и технологии сортировки вставками. Во время выполнения работы Выдрин Д.А. показал себя ответственным, добросовестным учеником, знающим свой предмет, имеющим представление о современном состоянии науки, владеющим современными общенаучными знаниями по информатике и вычислительной технике, программированию и сортировке.

За выполнение работы Выдрин Д.А. заслуживает оценки « ».

Руководитель практики д.т.н., профессор, Зинкин С.А. « » 2025 г.

# Содержание

[Введение 2](#_bookmark0)

1. [Постановка задачи 3](#_bookmark1)
   1. [Достоинства алгоритма сортировки вставками 3](#_bookmark2)
   2. [Недостатки алгоритма сортировки вставками 3](#_bookmark3)
   3. [Типичные сценарии применения данного алгоритма 3](#_bookmark4)
2. [Выбор решения 4](#_bookmark5)
3. [Описание программы 5](#_bookmark6)
4. [Схемы программы 6](#_bookmark7)
   1. [Блок-схема программы 6](#_bookmark8)
5. [Тестирование программы 7](#_bookmark9)
   1. [Тестирование на разных наборах данных 7](#_bookmark10)
   2. [Анализ полученных результатов тестирования (анализ работы алгоритма) 7](#_bookmark11)
6. [Отладка 9](#_bookmark12)
7. [Совместная разработка 10](#_bookmark13)

[Заключение 14](#_bookmark14)

[Список используемой литературы 15](#_bookmark15)

[Приложение А 16](#_bookmark16)

[Приложение Б Листинг 19](#_bookmark17)

# Введение

Сортировка данных на сегодняшний день при современном развитии компьютерных технологий является одним из наиболее распространенных процессов современной обработки данных. Задачи на сортировку данных встречаются очень часто в различных профессиональных сферах деятельности.

Алгоритмы сортировки очень широко распространяются практически во всех задачах обработки информации. Они образуют отдельный класс алгоритмов, применяются с целью осуществления последующего более быстрого поиска.

Важность сортировки основана на том факте, что на ее примере можно показать многие основные фундаментальные приемы и методы построения алгоритмов. Сортировка является хорошим примером огромного разнообразия алгоритмов, которые выполняют одну и ту же задачу. Кроме того, многие из них имеют определенные преимущества друг перед другом. За счет усложнения алгоритма можно добиться существенного увеличения эффективности и быстродействия алгоритма по сравнению с более простыми методами. Как правило, термин сортировка понимают, как процесс перестановки объектов некоторого множества в определенном порядке.

Сортировка Шелла не является стабильной, так как она сравнивает и перемещает элементы с большими шагами, что может нарушить относительный порядок одинаковых элементов. Сортировка Шелла — это улучшенная версия сортировки вставками. Она устраняет один из главных недостатков вставками — медленную обработку далёких элементов. Используется в embedded-системах и случаях, где важна умеренная производительность при малом потреблении памяти. Иногда применяется в гибридных алгоритмах сортировки.

# Постановка задачи

Поставленная задача: необходимо заполнить массив из n-ого количества элементов случайными числами, записать данные элементы в отдельный файл. После этого выполнить сортировку Шелла над данными, находящимися в массиве, записать отсортированные данные в другой файл, посчитать время выполнения и количество перестановок значений массива при сортировке.

Использовать сервис GitHub для совместной работы. Создать и выложить коммиты, характеризующие действия, выполненные каждым участником бригады.

Оформить отчет по проведенной практике.

## Достоинства алгоритма сортировки Шелла

* + - Хорошо работает на средних и больших массивах;
    - Не требует дополнительной памяти;
    - Гибкость в выборе шагов.

## Недостатки алгоритма сортировки вставками

* Нестабильная сортировка;
* высокая алгоритмическая сложность О(n²);
* Сложность зависит от выбора шагов.

## Типичные сценарии применения данного алгоритма

* Сортировка средних по размеру массивов (когда быстрая сортировка или сортировка слиянием избыточны);
* Встраиваемые системы и устройства с ограниченной памятью (т. к. не требует рекурсии и дополнительной памяти);
* Предварительная сортировка в гибридных алгоритмах (например, перед финальной сортировкой вставками);
* Обработка частично упорядоченныхданных (например, логов, временных рядов, почти отсортированных массивов).

# Выбор решения

Для написания данной программы будет использован язык программирования Си. Этот язык является распространённым языком программирования. При разработке языка Си был принят компромисс между низким уровнем языка ассемблера и высоким уровнем других языков. Си – это язык программирования общего назначения, хорошо известный своей эффективностью, экономичностью и переносимостью. Указанные преимущества Си обеспечивают хорошее качество разработки почти любого вида программного продукта.

В качестве среды программирования была выбрана программа Microsoft Visual Studio. Microsoft Visual Studio — это программная среда по разработке приложений для ОС Windows, как консольных, так и с графическим интерфейсом.

Для удобства совместной разработки был использован сервис WEEK. WEEEK

— сервис для управления личными и командными проектами. В основе WEEEK лежит недельный планер и канбан-методология: доски, колонки и т. д. Проект динамично разрабатывается, регулярно расширяя функционал и возможности. Ведется активная работа с пожеланиями пользователей в еженедельном патчноте WEEEK Week.

# Описание программы

При запуске программы выводится меню из пяти пунктов: а) сортировка случайных значений по возрастанию;

б) сортировка случайных значений по убыванию;

в) сортировка возрастающих значений по убыванию;

г) ввод своего массива для сортировки;

д) ecs – выход.

Пользователю требуется выбрать тот пункт, который ему требуется. При выборе при выборе пунктов под буквами а-г выводится сообщение, в котором пользователю необходимо ввести количество значений для сортировки.

После того, как данные были введены, генерируется массив из случайных чисел, эти числа записываются в файл input.txt.

Процесс начинается с выбора начального интервала, который постепенно уменьшается. На каждом этапе элементы, находящиеся на данном интервале, сравниваются и распределяются по порядку, как в сортировке вставками. Этот процесс продолжается до тех пор, пока интервал не станет равным единице, после чего выполняется стандартная сортировка вставками для окончательной упорядоченности.

После этого отсортированный массив записывается в файл output.txt.

Программа так же осуществляет подсчет количества перестановок элементов массива и времени, которое заняла сортировка.

При выборе пункта меню под буквой д программа завершает выполнение.

Подробный алгоритм работы программы и функции сортировки представлен в подразделе 4.1 на рисунке 1.

Листинг программы приведен в приложении Б.

# Схемы программы

* 1. Блок-схема программы

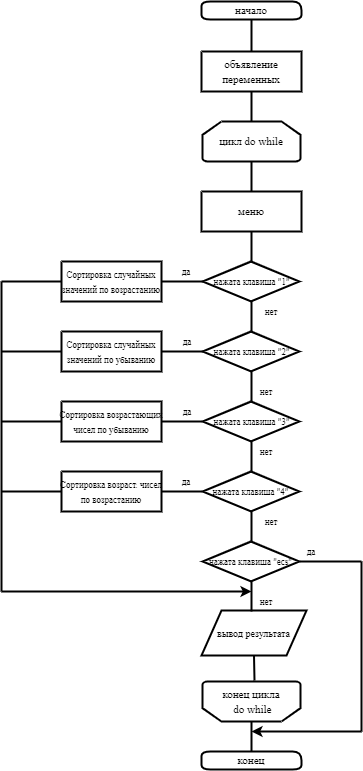
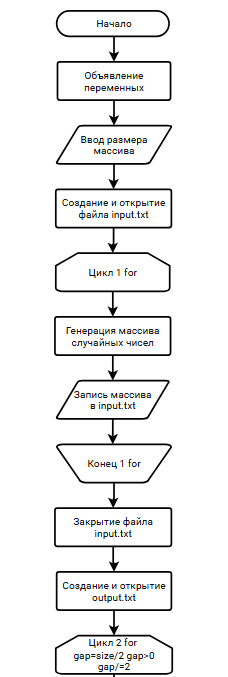
****

Рисунок 1 – Блок-схема программы

* 1. Блок-схема алгоритма



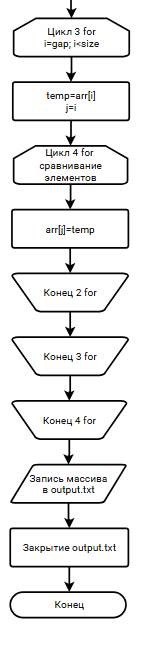


Рисунок 2 - Блок-схема программы

# Тестирование программы

## Тестирование на разных наборах данных

Тестовый набор данных представлен в таблице 1. Результаты тестирования приведены в Приложении А на рисунках А.1 - А.11.

Таблица 1 – Тестовый набор данных

|  |  |  |  |
| --- | --- | --- | --- |
|  | Размер массива size | Время выполнения  сортировки в секундах | Количество  перестановок |
| 1 | 10000 | 0.003 | 220119 |
| 2 | 20000 | 0.006 | 497574 |
| 3 | 30000 | 0.009 | 728162 |
| 4 | 40000 | 0.014 | 1153628 |
| 5 | 50000 | 0.022 | 1525393 |
| 6 | 60000 | 0.024 | 1850215 |
| 7 | 70000 | 0.026 | 2171615 |
| 8 | 80000 | 0.035 | 2824156 |
| 9 | 90000 | 0.036 | 3104400 |
| 10 | 100000 | 0.047 | 3863557 |
| 11 | 110000 | 0.053 | 3809754 |

## Анализ полученных результатов тестирования (анализ работы алгоритма)

На основании анализа данных, полученных в результате тестирования алгоритма сортировки Шелла, можно сделать вывод, что время, затраченное на работу программы относительно количества элементов увеличивается линейно, то есть с увеличением количества элементов пропорционально увеличивается время работы программы.

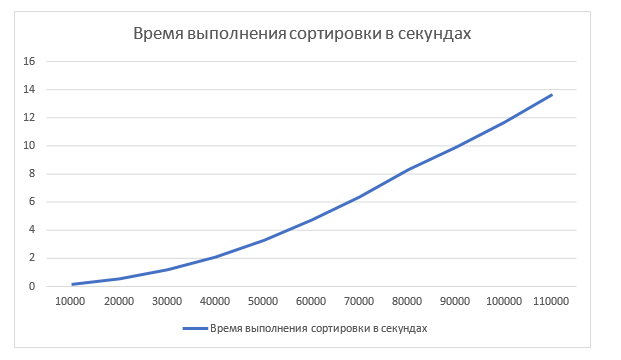


Рисунок 3 – Результаты тестирования

# Отладка

В качестве среды разработки была выбрана программа Microsoft Visual Studio, которая содержит в себе все необходимые средства для разработки и отладки модулей и программ.

Для отладки программы использовались точки остановки и пошаговое выполнение кода программы, анализ содержимого локальных переменных.

Точки останова – это прерывание выполнения программы, при котором выполняется вызов отладчика. Отладчик является инструментом для поиска и устранения ошибок в программе, с помощью которого можно исследовать состояние программы.

Был использован метод бинарного поиска, он включает в себя разделение частей кода для упрощения процесса отладки. Это может быть особенно полезно, если причина ошибки находится в начале языка программирования, а фактическая ошибка ближе к концу.

Команда шаг с заходом (step into) выполняет следующую инструкцию в обычном пути выполнения программы, а затем приостанавливает выполнение программы, чтобы мы могли проверить состояние программы с помощью отладчика. Если выполняемый оператор содержит вызов функции, шаг с заходом заставляет программу перескакивать в начало вызываемой функции, где она приостанавливается.

Тестирование проводилось в рабочем порядке, в процессе разработки, после завершения написания программы. После завершения написания программы, мною были выявлены и исправлены ошибки.

# Совместная разработка

Для удобства совместной разработки был использован сервис Git Hub. Определили задачи проекта, назначили приоритет задачам.

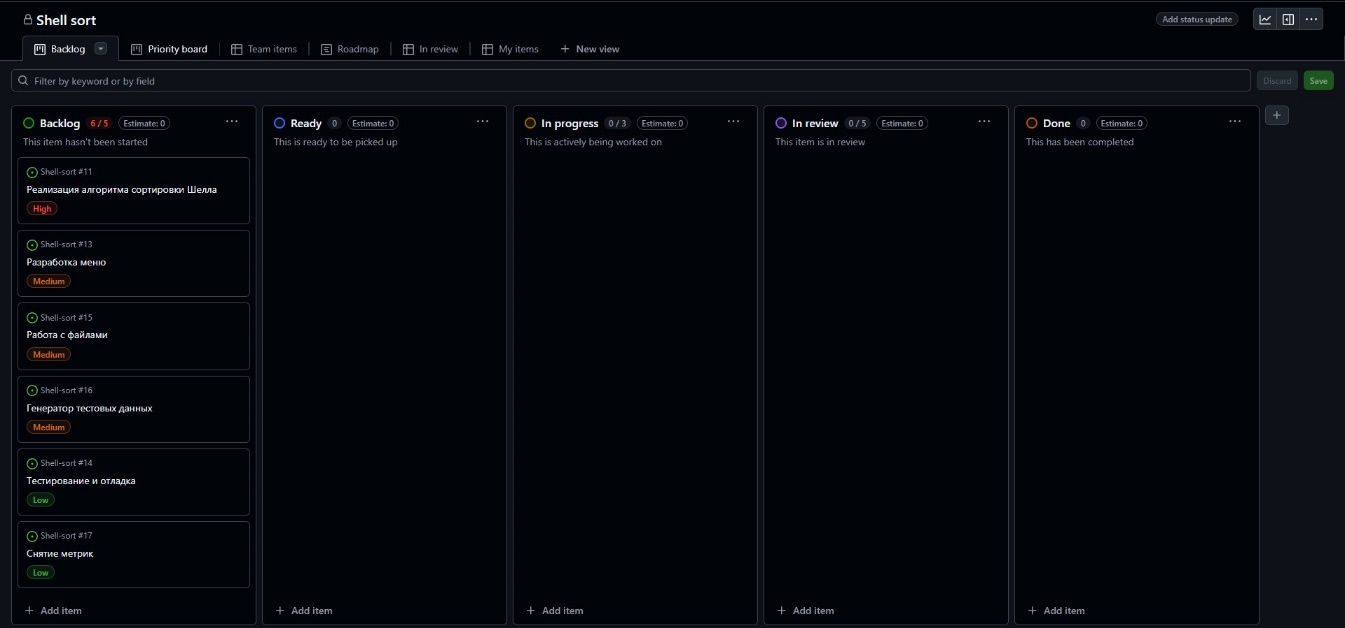


Рисунок 4 – Определение задач проекта Разделили роли, назначили исполнителей задачам.

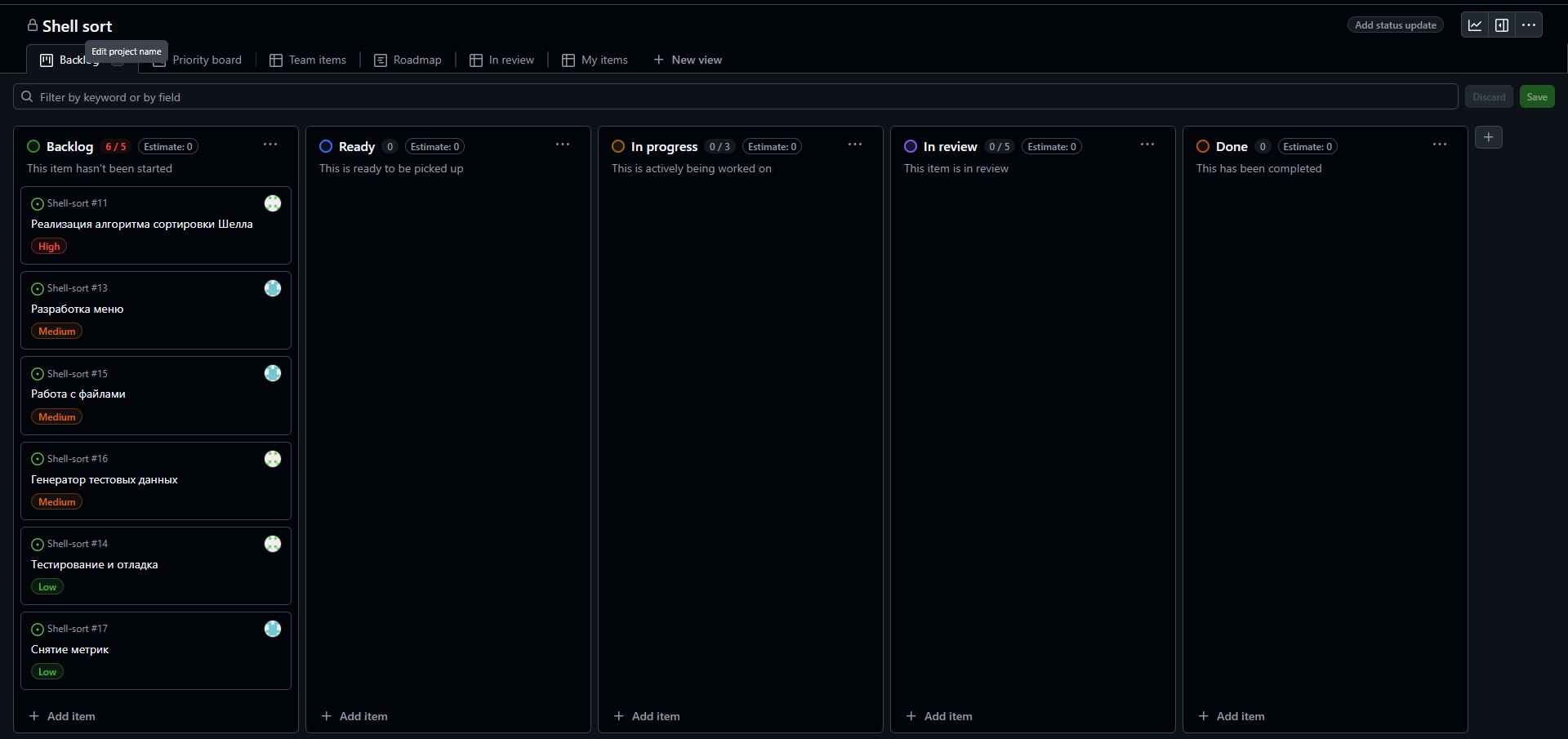


Рисунок 5 – Распределение задач проекта

Корректировали статус задач по мере выполнения.

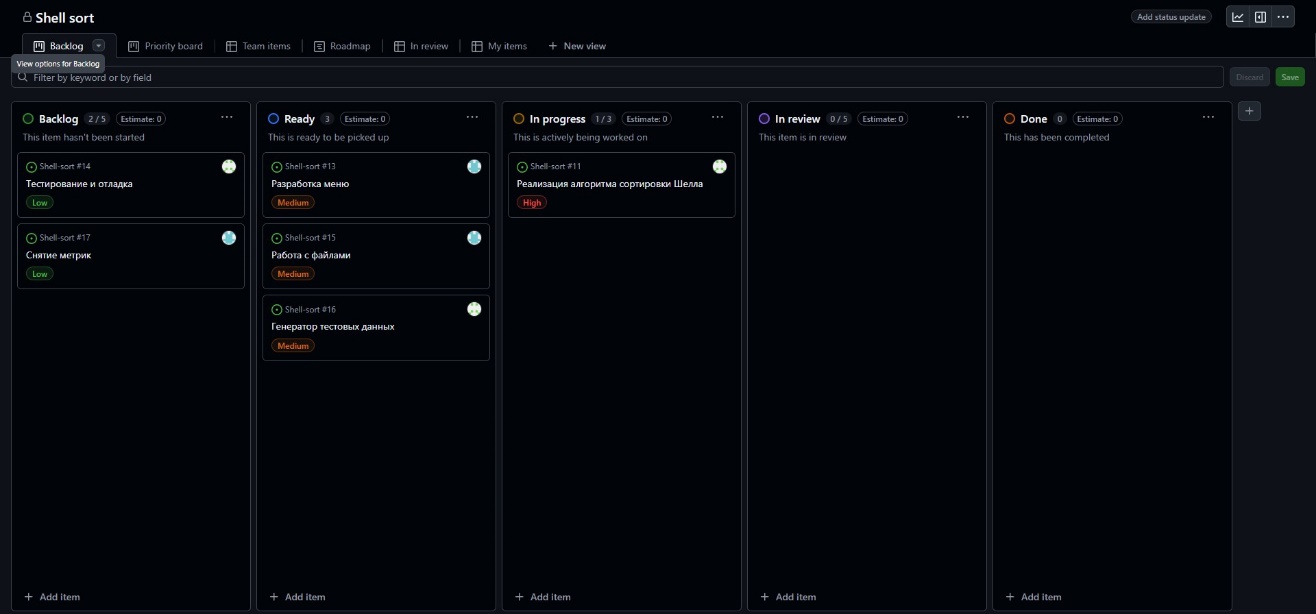


Рисунок 6 – Корректирование задач

Во время работы над данной практикой наша бригада осуществляла совместную работу в GitHub.

Данная программа была загружена на компьютер, с помощью git clone

<ссылка>.

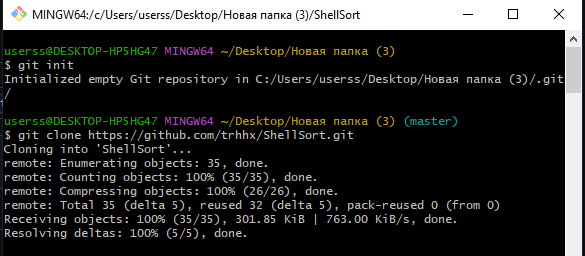


Рисунок 7 – Загрузка папки с удаленного репозитория

Мной было добавлено вывод массивов, подсчёт времени и перестановок.

Я загрузил обновленный код программы на удаленный репозиторий GitHub.

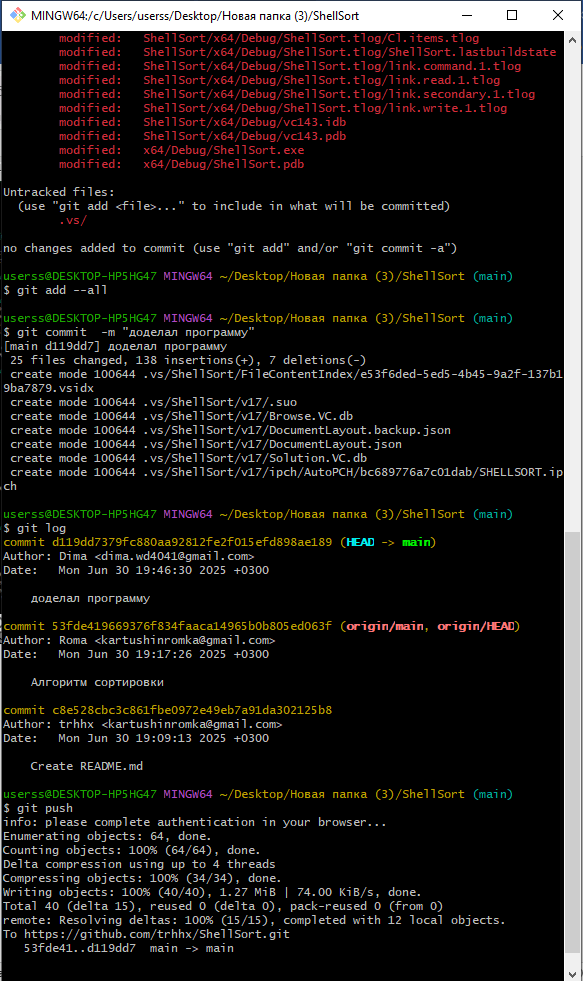


Рисунок 8 –Загрузка измененной программы на удаленный репозиторий

Ссылка на удаленный репозиторий: <https://github.com/trhhx/ShellSort.git>

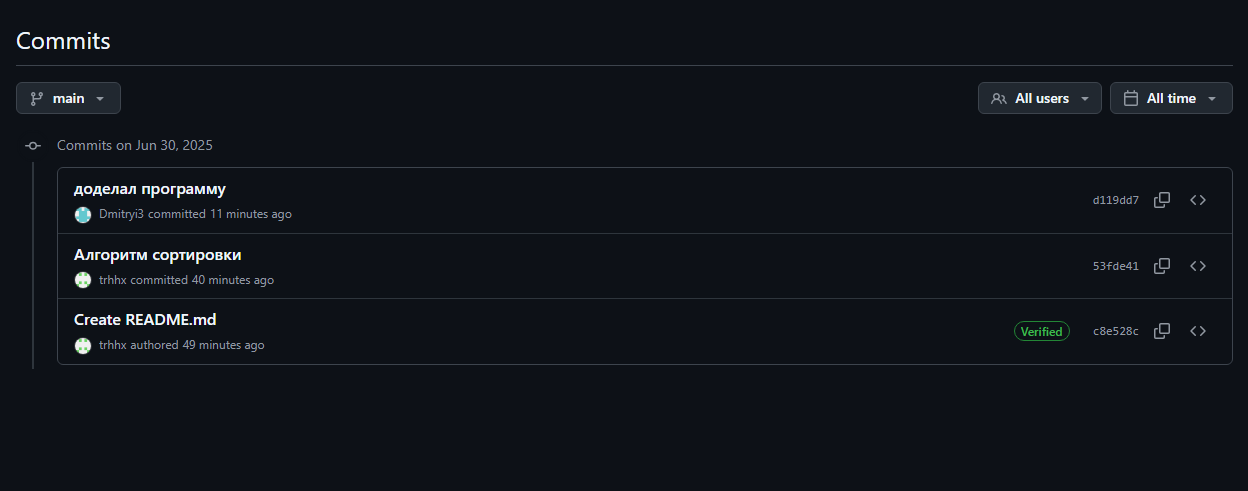


Рисунок 9 – История коммитов ветки main

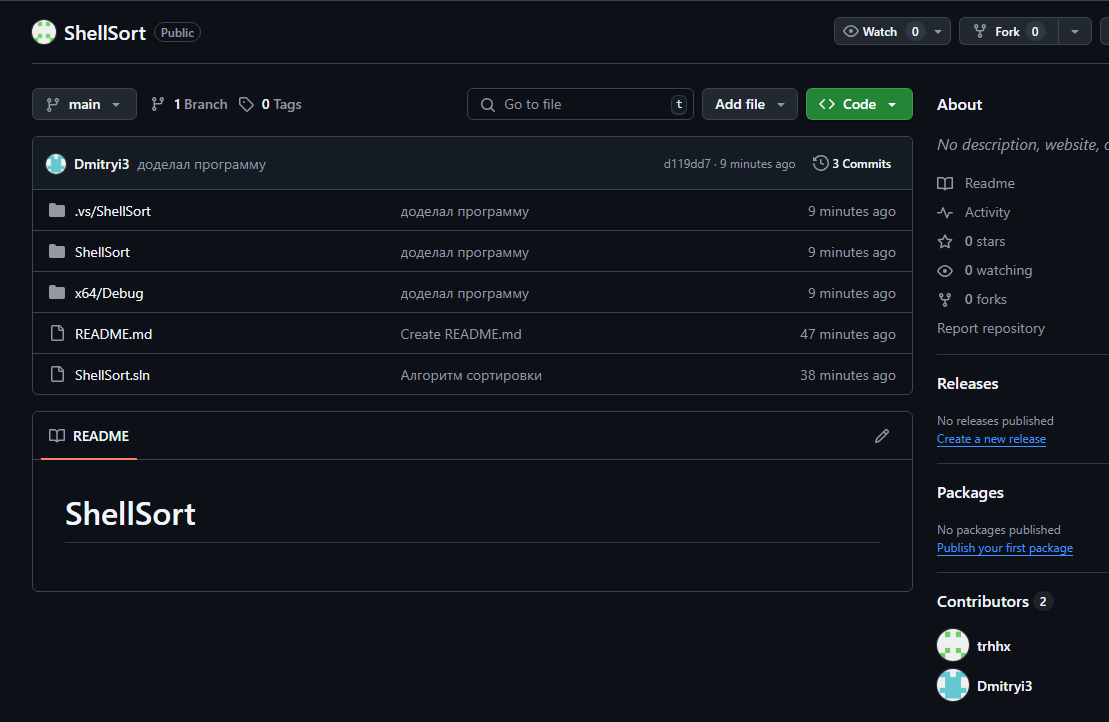


Рисунок 10 – Ветка main

# Заключение

При выполнении данной работы были получены навыки совместной работы с помощью сервисов GitHub и WEEK, навыки использования программы Git Bash. Был изучен алгоритм сортировки вставками.

Мною был написан алгоритм, считающий количество перестановок элементов массива во время сортировки и время работы программы в секундах. Было выполнено тестирование программы на разных наборах данных и отладка данной программы.

При выполнении практической работы были улучшены базовые навыки программирования на языке С. Улучшены навыки отладки, тестирования программ и работы со сложными типами данных.

В дальнейшем программу можно улучшить путем подключения упрощающих реализацию данной сортировки библиотек и улучшения графического интерфейса.

# Список используемой литературы

1. ГОСТ 19.701 – 90 Схемы алгоритмов, программ, данных и систем.
2. Керниган, Брайан У., Ритчи, Деннис М. Язык программирования С, 2-е издание.: Пер. с англ. – М.,2009.
3. Сортировка Шелла [Электронный ресурс] – URL: https://ru.wikipedia.org (дата обращения: 28.06.2025 г)

# Приложение А. Результаты тестирования программы

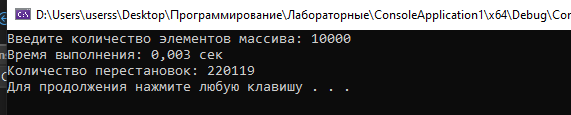


Рисунок А.1

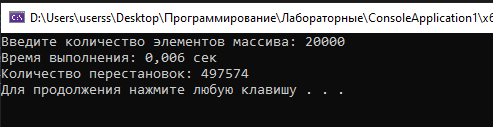


Рисунок А.2

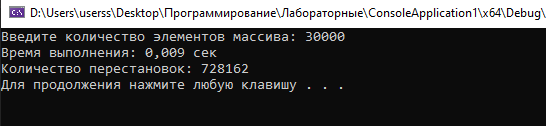


Рисунок А.3

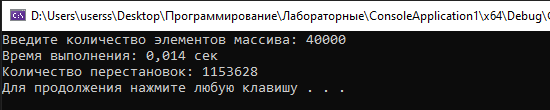


Рисунок А.4

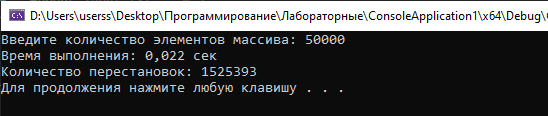


Рисунок А.5

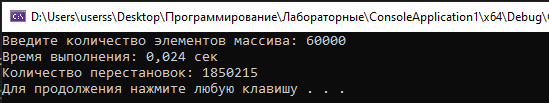


Рисунок А.6

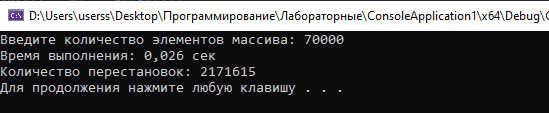


Рисунок А.7

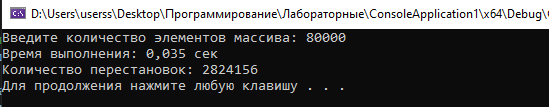


Рисунок А.8

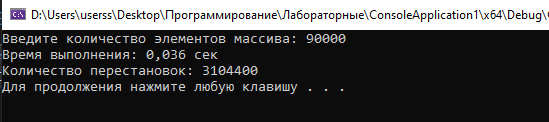


Рисунок А.9

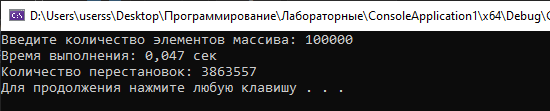


Рисунок А.10

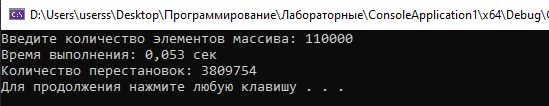


Рисунок А.11

# Приложение Б. Листинг программы

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <locale.h>

#include <stdlib.h>

#include <time.h>

#include <conio.h>

// Функция вывода массива

void printArray(int\* arr, int size) {

for (int i = 0; i < size; i++) {

printf("%d ", arr[i]);

}

printf("\n");

}

// Функция сортировки Шелла

void shellSort(int\* arr, int size, int srt, long\* count) {

\*count = 0; // Обнуляем счетчик перестановок

// Основной цикл с уменьшающимся шагом

for (int gap = size / 2; gap > 0; gap /= 2) {

for (int i = gap; i < size; i++) {

int temp = arr[i];

int j;

// Сортировка по возрастанию

if (srt) {

for (j = i; j >= gap && arr[j - gap] > temp; j -= gap) {

arr[j] = arr[j - gap];

(\*count)++;

}

}

// Сортировка по убыванию

else {

for (j = i; j >= gap && arr[j - gap] < temp; j -= gap) {

arr[j] = arr[j - gap];

(\*count)++;

}

}

arr[j] = temp;

if (j != i) (\*count)++;

}

}

}

int main() {

setlocale(LC\_ALL, "Russian");

srand(time(NULL));

FILE\* f;

int size;

int\* arr;

char ch;

long count = 0;

do {

system("cls");

printf("МЕНЮ:\n");

printf("1 - Сортировка случайных значений по возрастанию\n");

printf("2 - Сортировка случайных значений по убыванию\n");

printf("3 - Сортировка возрастающих значений по убыванию\n");

printf("4 - Ввод своего массива для сортировки\n");

printf("esc - выход\n");

ch = \_getch();

switch (ch) {

case '1': {

// Случайные числа по возрастанию

system("cls");

printf("Введите размер массива: ");

scanf("%d", &size);

arr = (int\*)malloc(size \* sizeof(int));

// Генерация и запись случайных чисел

f = fopen("input.txt", "w");

printf("Исходный массив:\n");

for (int i = 0; i < size; i++) {

arr[i] = rand() - rand();

fprintf(f, "%d ", arr[i]);

}

printArray(arr, size);

fclose(f);

// Сортировка и замер времени

time\_t start = clock();

shellSort(arr, size, 1, &count);

time\_t stop = clock();

// Запись результата

f = fopen("output.txt", "w");

printf("\nОтсортированный массив:\n");

for (int i = 0; i < size; i++) {

fprintf(f, "%d ", arr[i]);

}

printArray(arr, size);

fclose(f);

printf("\nВремя: %.3lf сек\n", (stop - start) / 1000.0);

printf("Перестановок: %ld\n", count);

free(arr);

system("pause");

break;

}

case '2': {

// Случайные числа по убыванию (аналогично case 1, но srt=0)

system("cls");

printf("Введите размер массива: ");

scanf("%d", &size);

arr = (int\*)malloc(size \* sizeof(int));

f = fopen("input.txt", "w");

printf("Исходный массив:\n");

for (int i = 0; i < size; i++) {

arr[i] = rand() - rand();

fprintf(f, "%d ", arr[i]);

}

printArray(arr, size);

fclose(f);

time\_t start = clock();

shellSort(arr, size, 0, &count);

time\_t stop = clock();

f = fopen("output.txt", "w");

printf("\nОтсортированный массив:\n");

for (int i = 0; i < size; i++) {

fprintf(f, "%d ", arr[i]);

}

printArray(arr, size);

fclose(f);

printf("\nВремя: %.3lf сек\n", (stop - start) / 1000.0);

printf("Перестановок: %ld\n", count);

free(arr);

system("pause");

break;

}

case '3': {

// Упорядоченный массив по убыванию

system("cls");

printf("Введите размер массива: ");

scanf("%d", &size);

arr = (int\*)malloc(size \* sizeof(int));

// Генерация возрастающего массива

f = fopen("input.txt", "w");

printf("Исходный массив:\n");

arr[0] = rand() % 1000;

for (int i = 1; i < size; i++) {

arr[i] = arr[i - 1] + rand() % 100 + 1;

}

for (int i = 0; i < size; i++) {

fprintf(f, "%d ", arr[i]);

}

printArray(arr, size);

fclose(f);

// Сортировка по убыванию

time\_t start = clock();

shellSort(arr, size, 0, &count);

time\_t stop = clock();

f = fopen("output.txt", "w");

printf("\nОтсортированный массив:\n");

for (int i = 0; i < size; i++) {

fprintf(f, "%d ", arr[i]);

}

printArray(arr, size);

fclose(f);

printf("\nВремя: %.3lf сек\n", (stop - start) / 1000.0);

printf("Перестановок: %ld\n", count);

free(arr);

system("pause");

break;

}

case '4': {

// Ручной ввод массива

system("cls");

printf("Введите размер массива: ");

scanf("%d", &size);

arr = (int\*)malloc(size \* sizeof(int));

printf("Введите %d элементов:\n", size);

for (int i = 0; i < size; i++) {

scanf("%d", &arr[i]);

}

// Выбор направления сортировки

printf("Выберите направление (1 - возр., 2 - убыв.): ");

int srt1 = \_getch() == '1' ? 1 : 0;

f = fopen("input.txt", "w");

printf("\nИсходный массив:\n");

for (int i = 0; i < size; i++) {

fprintf(f, "%d ", arr[i]);

}

printArray(arr, size);

fclose(f);

time\_t start = clock();

shellSort(arr, size, srt1, &count);

time\_t stop = clock();

f = fopen("output.txt", "w");

printf("\nОтсортированный массив:\n");

for (int i = 0; i < size; i++) {

fprintf(f, "%d ", arr[i]);

}

printArray(arr, size);

fclose(f);

printf("\nВремя: %.3lf сек\n", (stop - start) / 1000.0);

printf("Перестановок: %ld\n", count);

free(arr);

system("pause");

break;

}

}

} while (ch != 27);

return 0;

}