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 (ЭКСПЛУАТАЦИОННОЙ) ПРАКТИКЕ

(2023/2024 учебный год)

Монин Иван 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 25.06.2024 по 8.07.2024

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 (ЭКСПЛУАТАЦИОННОЙ) ПРАКТИКИ

(2023/2024 учебный год)

Монин Иван 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 25.06.2024 по 8.07.2024

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	26	20.06.24 – 24.06.24	
2	Подбор и изучение материала по теме работы	26	24.06.24 – 26.06.24	
3	Установка виртуальной машины	26	26.06.24 – 28.06.24	
4	Установка операционной системы	30	01.07.24 – 04.07.24	
5	Разработка программы на языке Си	30	04.07.24 – 09.07.24	
6	Тестирование и отладка	38	09.07.24 – 12.07.24	
7	Оформление отчёта	40	12.07.24 – 17.07.24	
	Общий объём часов	216		

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ
ФАКУЛЬТЕТ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

ОТЧЁТ

О ПРОХОЖДЕНИИ УЧЕБНОЙ (ЭКСПЛУАТАЦИОННОЙ) ПРАКТИКИ

(2023/2024 учебный год)

Монин Иван 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 25.06.2024 по 8.07.2024

Кафедра «Вычислительная техника»

Монин И.А. выполнял практическое задание «Сортировка выбором». На первоначальном этапе на первоначальном этапе был изучен и проанализирован алгоритм сортировки выбором, был выбран метод решения и язык программирования C++. Кроме того, осуществил подсчёт времени выполнения алгоритма и количества перестановок. Протестировала и отладил программу. Оформил отчёт.

Бакалавр Монин И.А. _____ "___" _____ 2024 г.

Руководитель практики Зинкин С.А._____ "___" _____ 2024 г.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ
ФАКУЛЬТЕТ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

ОТЗЫВ

О ПРОХОЖДЕНИИ УЧЕБНОЙ (ЭКСПЛУАТАЦИОННОЙ) ПРАКТИКИ

(2023/2024 учебный год)

Монин Иван 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 25.06.2024 по 8.07.2024

Кафедра «Вычислительная техника»

В процессе выполнения практики Монин И.А. решал следующие задачи: изучение алгоритма сортировки выбором, сравнение существующих методов сортировки, реализация подсчёта количества перестановок и времени выполнения алгоритма.

За период выполнения практики были освоены основные понятия и технологии сортировки выбором. Во время выполнения работы Монин И.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 Монин И.А. заслуживает оценки «_____».

Руководитель практики д.т.н., профессор, Зинкин С.А. « » 2024 г.

Содержание

Введение 7
1 Постановка задачи 8
1.1 Достоинства алгоритма сортировки выбором 8
1.2 Недостатки алгоритма сортировки выбором 8
1.3 Типичные сценарии применения данного алгоритма 8
2 Выбор решения 9
3 Описание программы 10
4 Схема программы 13
4.1 Блок-схема программы 13
4.2 Блок-схема алгоритма 14
5 Тестирование программы 15
5.1 Тестирование на разных наборах данных 15
5.2 Анализ полученных результатов тестирования (анализ работы алгоритма) 15
6 Отладка 16
7 Совместная работа 17

Введение

Сортировка — это процесс упорядочивания элементов в каком-то определенном порядке, таком как по возрастанию или убыванию их значений. Она является основополагающей задачей в компьютерных науках и программировании, поскольку позволяет эффективно управлять и обрабатывать данными. Выбор подходящего алгоритма сортировки зависит от конкретных требований задачи, объема данных и желаемой производительности.

Отсортированные данные обеспечивают более простой и быстрый доступ к информации. Например, для поиска элемента в отсортированном массиве можно использовать бинарный поиск, который значительно эффективнее линейного поиска в неупорядоченном массиве. Многие задачи в компьютерных науках требуют сортировки данных как этап решения. Например, сортировка может использоваться для анализа данных или агрегирования информации. Важность сортировки основана на том факте, что на ее примере можно показать многие основные фундаментальные приемы и методы построения алгоритмов.

Сортировка выбором — это один из простых алгоритмов сортировки, который проходит по массиву несколько раз, на каждом шаге находя минимальный элемент и перемещая его на начальную позицию текущего неотсортированного подмассива. Сортировка выбором имеет квадратичную сложность в худшем, лучшем и среднем случаях. Это делает ее не самым эффективным выбором для сортировки больших массивов данных, но она проста в реализации и может быть полезна для небольших массивов или в учебных целях.

1 Постановка задачи

Нужно создать массив из n элементов, заполнить его случайными числами, записать данные элементы в файл. Затем выполнить сортировку выбором для данных в массиве, записать отсортированные данные в другой файл, измерить время выполнения сортировки и подсчитать количество перестановок значений массива. Для совместной работы использовать сервис GitHub. Каждый участник бригады создает и выкладывает коммиты, которые отражают выполненные действия. После выполнения задачи необходимо оформить отчет, описывающий проведенную практику.

1.1 Достоинства алгоритма сортировки выбором

- эффективен для небольших наборов данных;
- простота реализации и понимания алгоритма;
- алгоритм эффективен для работы со списками.

1.2 Недостатки алгоритма сортировки выбором

- квадратичная временная сложность O(n2);
- неустойчивость к большим значениям;
- не сохраняет порядок равных элементов.

1.3 Типичные сценарии применения данного алгоритма

- книги в библиотеке (по автору, году издания, жанру);
- заказы в магазин (по дате, стоимости доставки, сумме покупки);
- задачи в проекте (по приоритету, срокам выполнения, статусу).

2 Выбор решения

Наша бригада выбрала среду Microsoft Visual Studio для разработки на языке C++.

Microsoft Visual Studio — это интегрированная среда разработки (IDE), созданная корпорацией Microsoft. Она предназначена для разработки программного обеспечения, включая консольные и графические приложения, веб-приложения, веб-сервисы и мобильные приложения.

Язык программирования C++ будет использован для написания данной программы. C++ является популярным языком программирования. При его создании был достигнут компромисс между низкоуровневыми возможностями ассемблера и высокоуровневыми функциями других языков. C++ – это язык общего назначения, известный своей эффективностью, экономичностью и переносимостью. Эти преимущества C++ обеспечивают высокое качество разработки практически любого вида программного обеспечения.

3 Описание программы

При запуске программы выводится меню из шести пунктов:

а) сортировка по возрастанию;

б) сортировка по убыванию;

в) ввести количество элементов массива;

г) заполнить массив вручную;

д) заполнить массив случайными числами;

е) выход.

```
std::cout << "\t    +Меню программы+\n" << std::endl;
std::cout << "_____________**********_____________\n\n\n";
std::cout << "1 - Сортировка по возрастанию" << std::endl;
std::cout << "2 - Сортировка по убыванию" << std::endl;
std::cout << "3 - Ввести количество элементов массива" << std::endl;
std::cout << "4 - Заполнить массив вручную" << std::endl;
std::cout << "5 - Заполнить массив случайными числами" << std::endl;
std::cout << "6 - Выйти\n\n" << std::endl;
std::cout << "_____________**********_____________\n\n\n";
std::cout << "Выберите желаемую опцию : ";
```

Пользователю нужно выбрать необходимый ему пункт. Для начала необходимо выбрать пункт В и ввести количество элементов массива.

```
std::cout << "Введите количество элементов массива: ";
std::cin >> size; // ввод количества элементов массива
```

Далее выделяется новый массив типа int размером size элементов. Оператор new[] используется для выделения динамической памяти под

массив. После выполнения этой строки переменная array указывает на новый массив, заменяя предыдущий. Затем пользователю выводится на экран количество элементов.

```
array = new int[size]; // динамическое выделение памяти

std::cout << "Создан массив из " << size << " элементов.\n" << std::endl;
```

После выбора определенного количества элементов массива, его необходимо заполнить: пункты Г (вручную) и Д (случайные числа).

```
// для случайного заполнения

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

   array[i] = rand() - rand(); // заполнение массива случайными числами

   }

//для ввода вручную

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

  std::cout << "Введите элемент " << i + 1 << ": ";

  std::cin >> array[i]; // ввод элементов массива вручную

  }
```

Затем можно выбрать сортировку по возрастанию или по убыванию в пунктах А и Б. В каждой итерации он сравнивает текущий элемент array[j] с элементом array[minMaxIndex] и, в зависимости от флага ascending, определяет, нужно ли обновить minMaxIndex. Если ascending равно true, выбирается минимальный элемент (меньше), иначе - максимальный элемент (больше). После завершения внутреннего цикла выполняется обмен между array[i] и элементом array[minMaxIndex], что обеспечивает правильную позицию текущего минимального/максимального элемента в сортированном порядке. Переменная ascending определяет режим сортировки (если равно true, то сортировка происходит по возрастанию, если false, то по убыванию).

```
for (int i = 0; i < size - 1; ++i) {

  int minMaxIndex = i;

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

    if (ascending ? array[j] < array[minMaxIndex] : array[j] > array[minMaxIndex]) {
```

```
            minMaxIndex = j;
          }
        } //алгоритм сортировки выбором
      int temp = array[minMaxIndex;
      array[minMaxIndex] = array[i];
      array[i] = temp;
    }
```

После сортировки отсортированный массив записывается в отдельный файл как и исходный массив.

```
std::ofstream file(filename); // открыте файла filename для записи
for (int i = 0; i < size; ++i) {
  file << array[i] << " "; // запись элементов массива в файл
}
file.close(); // закрытие файла
```

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

При выборе пункта меню под буквой E программа завершает выполнение.

Подробный алгоритм работы программы и функции сортировки представлен в подразделе 4.1 на рисунках 1, 2. Листинг программы приведен в приложении А.

4 Схема программы

4.1 Блок-схема программы

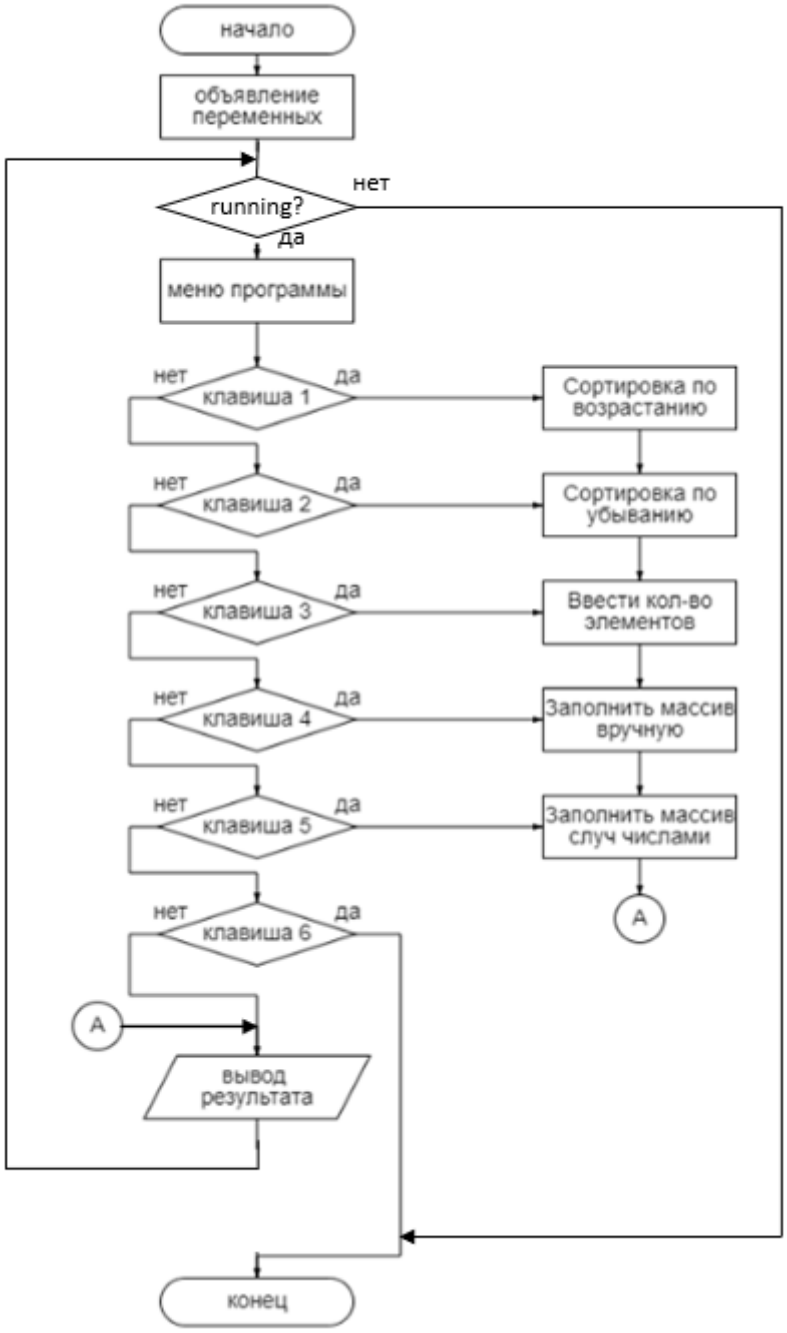

Рисунок 1 - Блок-схема программы

4.2 Блок-схема алгоритма

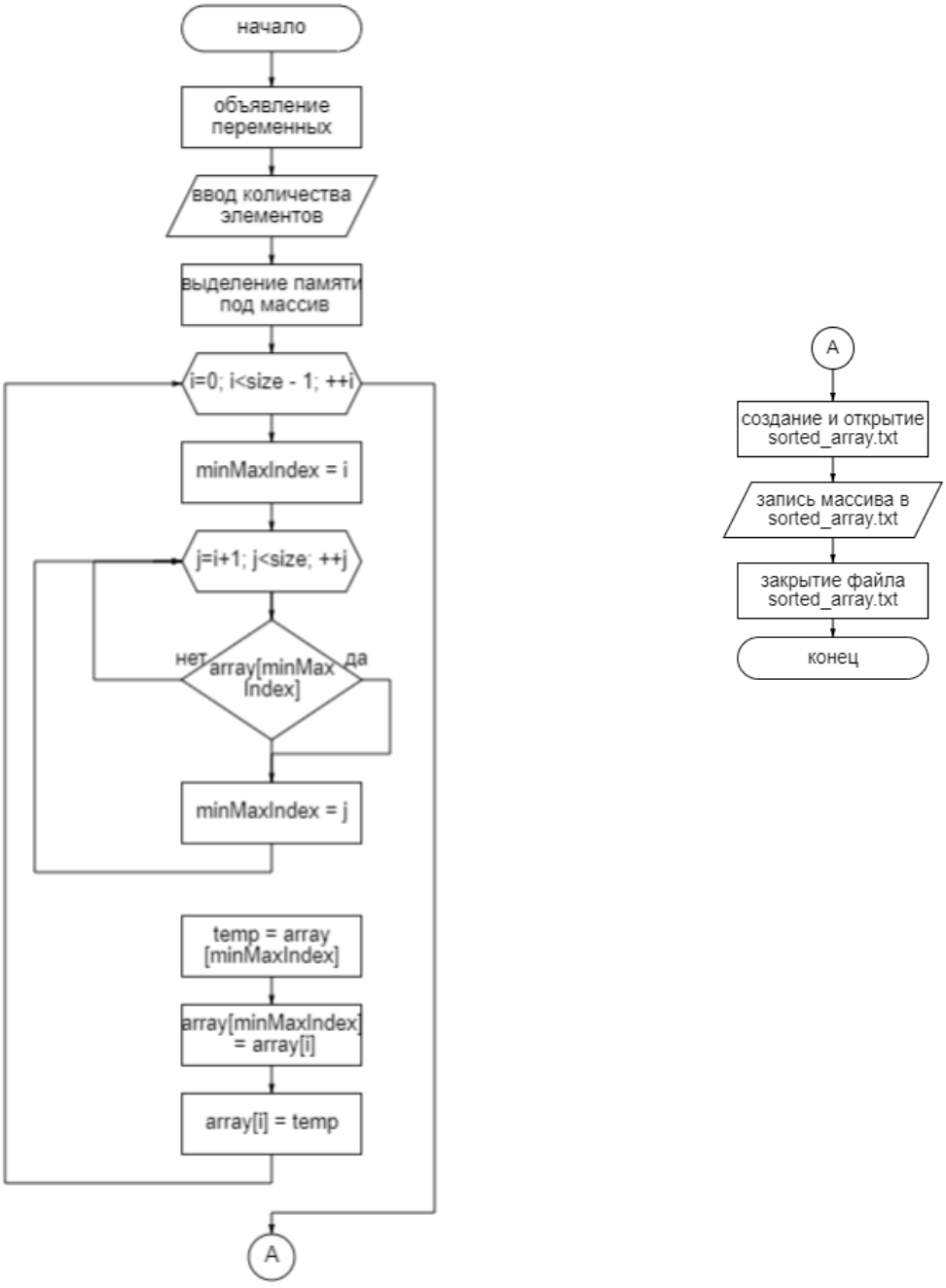

Рисунок 2 — Блок-схема алгоритма

5 Тестирование программы

5.1 Тестирование на разных наборах данных

Результаты тестирования приведены в приложении В.1-В.10, а тестовый набор данных представлен в таблице 1.

№	Размер массива	Время выполнения сортировки	Кол-во перестановок
1	10000	0.114 сек	9983
2	15000	0.259 сек	14989
3	20000	0.456 сек	19987
4	30000	1.028 сек	29987
5	50000	2.869 сек	49988
6	85000	8.268 сек	84993
7	100000	11.411 сек	99986
8	120000	16.476 сек	119982
9	150000	25.869 сек	149981
10	1000000	30 мин 25.81 сек	999992

Таблица 1 - Тестовый набор данных

5.2 Анализ полученных результатов тестирования (анализ работы алгоритма)

Когда количество элементов в массиве увеличивается, время выполнения программы также возрастает. Это было продемонстрировано в результатах тестирования, где проведено измерение времени сортировки при различных размерах массивов. График показал четкую зависимость: с увеличением числа элементов в массиве растет и время работы программы. Такое поведение вполне ожидаемо, так как алгоритмы сортировки обычно

имеют временную сложность, зависящую от количества элементов в сортируемом массиве.

Измерение времени сортировки позволяет оценить эффективность алгоритма и сравнить его производительность при различных условиях.

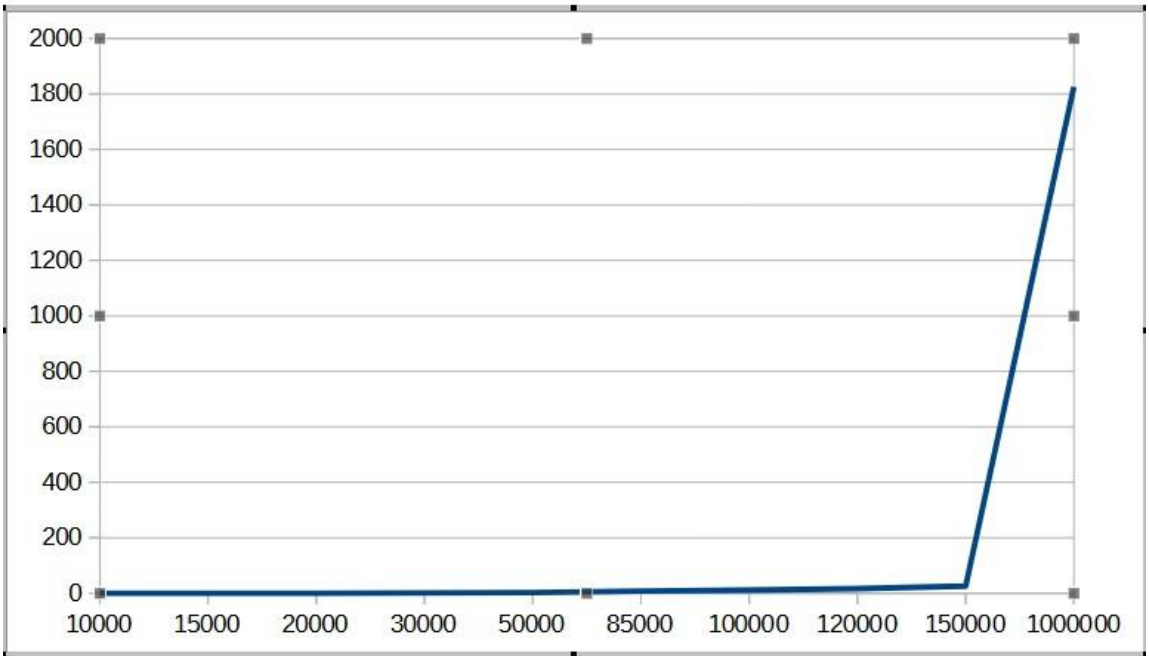

Рисунок 3 — Результаты тестирования сортировки

6 Отладка

В процессе разработки программы в качестве среды разработки была выбрана Microsoft Visual Studio. Этот выбор был обусловлен её обширным набором инструментов, необходимых для создания и отладки программного кода.

Для отладки использовались точки останова. Точки останова приводят к остановке выполнения программы, вызывая отладчик, который помогает находить и исправлять ошибки в программе, позволяя изучать текущее состояние программы. Устанавливались в ключевых местах кода для остановки выполнения программы. Это позволяло тщательно проверять текущие значения переменных и состояние программы на определенных

этапах выполнения. Точки останова использовались для контроля правильности логики программы и своевременного обнаружения ошибок.

Трассировка выполнения кода позволяла пошагово выполнять программу, анализируя каждую строку кода. Это помогало в выявлении логических ошибок и неправильного поведения программы. С помощью трассировки можно было наблюдать последовательность выполнения функций и точное место возникновения ошибки.

Анализ содержимого переменных использовался для проверки значений переменных в любой момент выполнения программы. Это помогало в обнаружении некорректных данных и проблем с инициализацией переменных.

7 Совместная работа

Для совместной разработки был разработан план для каждого участника бригады, распределены задачи, и установлены сроки их выполнения. Каждый участник команды получил чёткое представление о своей роли и обязанностях, что позволило обеспечить слаженную работу и высокую продуктивность.

На протяжении практики наша бригада использовала для совместной работы GitHub. Он позволил нам эффективно управлять версиями кода, отслеживать изменения организовывать совместную работу над проектом.

Была добавлена возможность сохранять в файлы для записи сгенерированных и отсортированных массивов, помимо этого мною был создан интерфейс, который облегчает навигацию и помогает лучше ориентироваться в программе. Эти изменения были зафиксированы и загружены на удалённый репозиторий GitHub в ветку main.

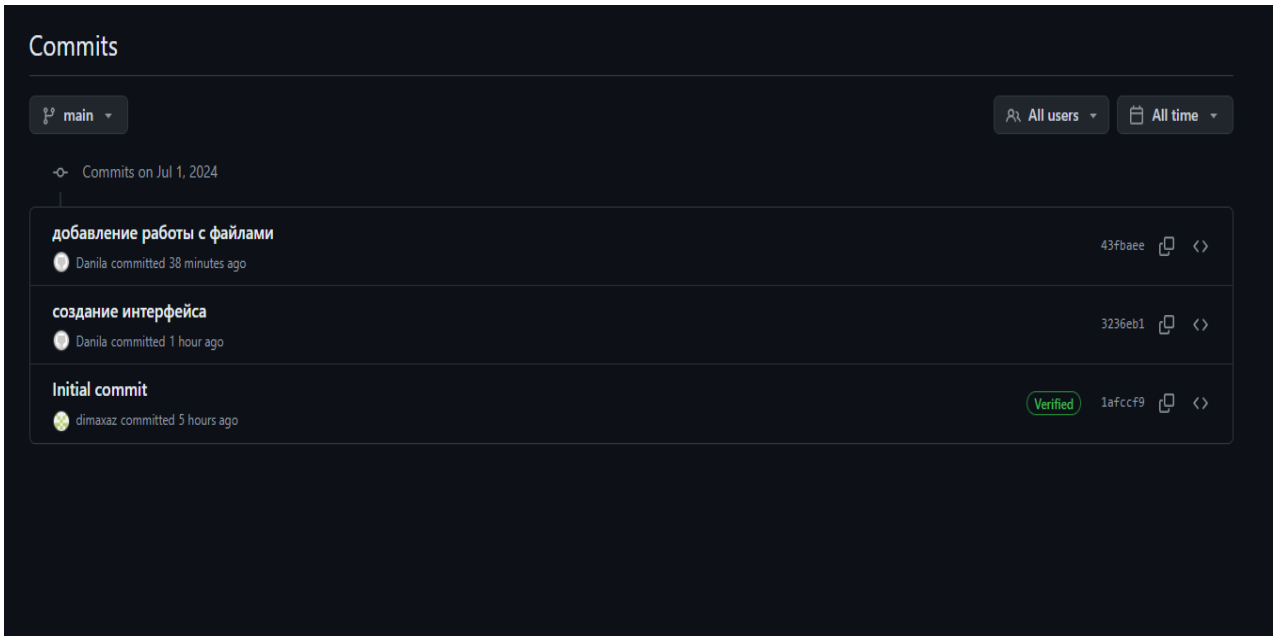

Рисунок 4 — Созданные коммиты

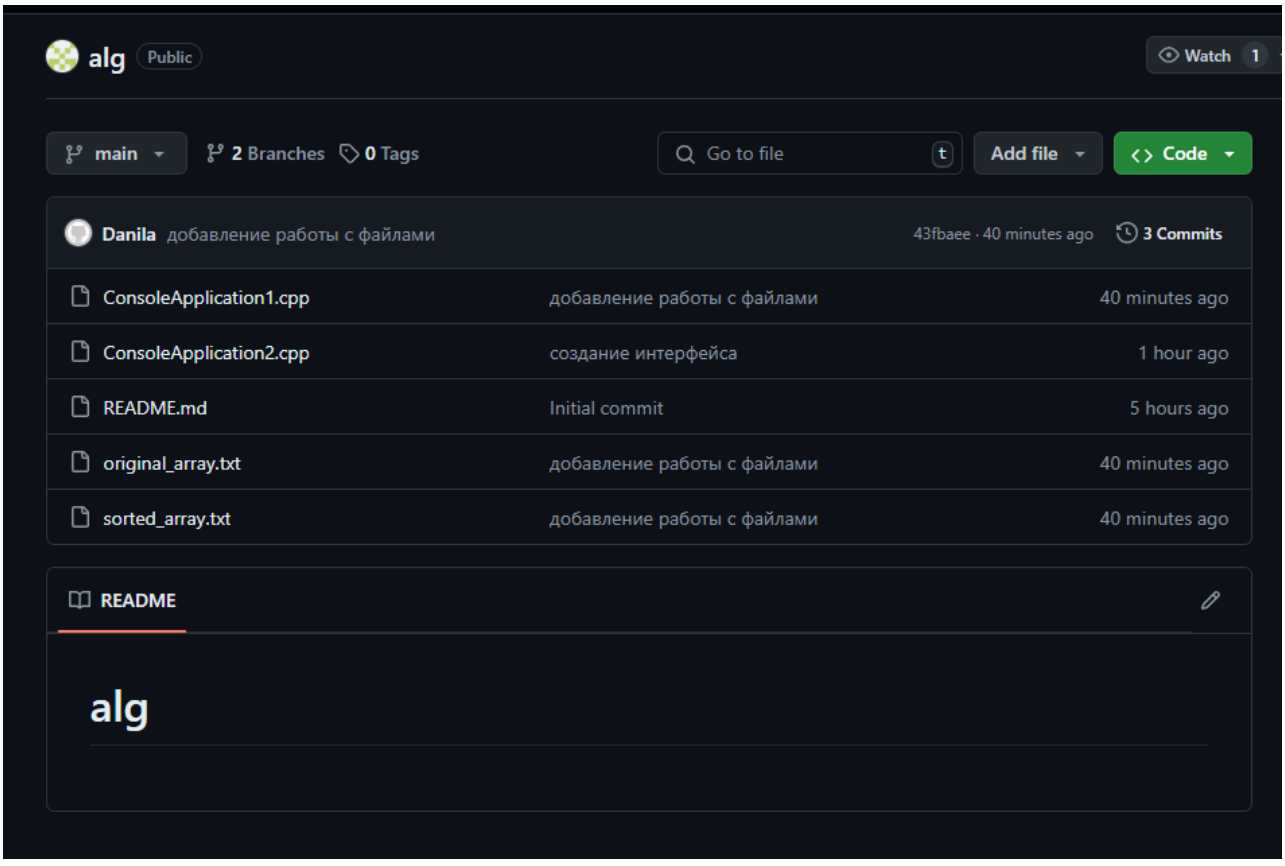

Рисунок 5 — Ветка main

Для загрузки данных в локальный репозиторий, а также отправки данных на удаленный репозиторий, были использованы основные команды Git

Bash, такие как git clone, git add, git commit, git push и git log. Команда git clone позволила клонировать удаленный репозиторий на локальный компьютер. С помощью git add были добавлены изменения в индекс, а git commit фиксировал их с сообщением о выполненных задачах. Команда git push отправляла локальные изменения в удаленный репозиторий, обеспечивая синхронизацию и актуальность проекта на GitHub. Для просмотра истории коммитов и отслеживания изменений использовалась команда git log, которая предоставляла детализированную информацию о всех предыдущих коммитах.

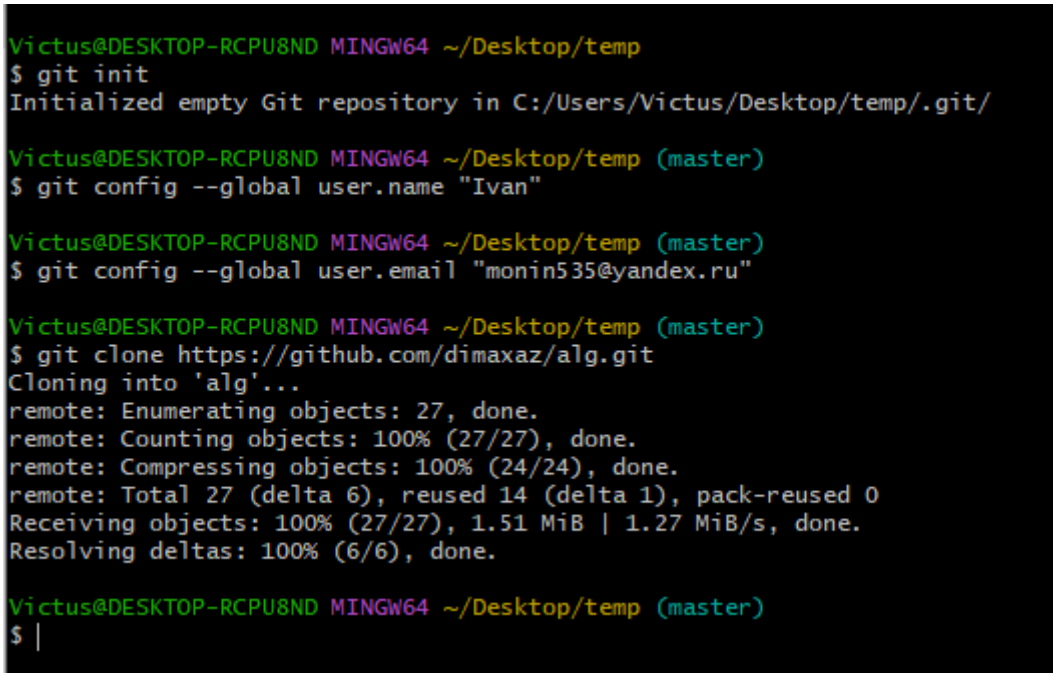

Рисунок 6 — Использование команды git clone

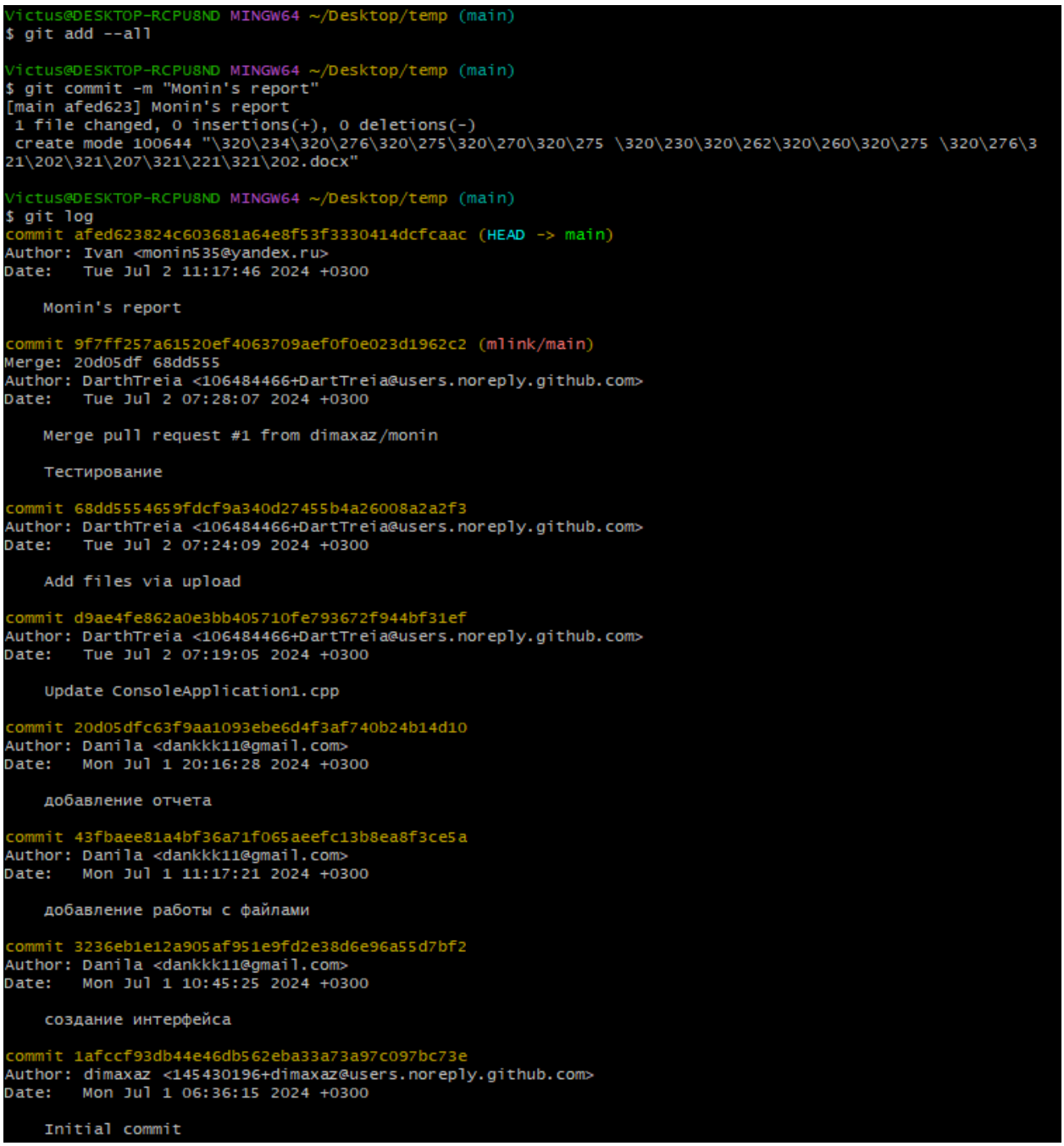

Рисунок 7 — Использование команд git commit и git log

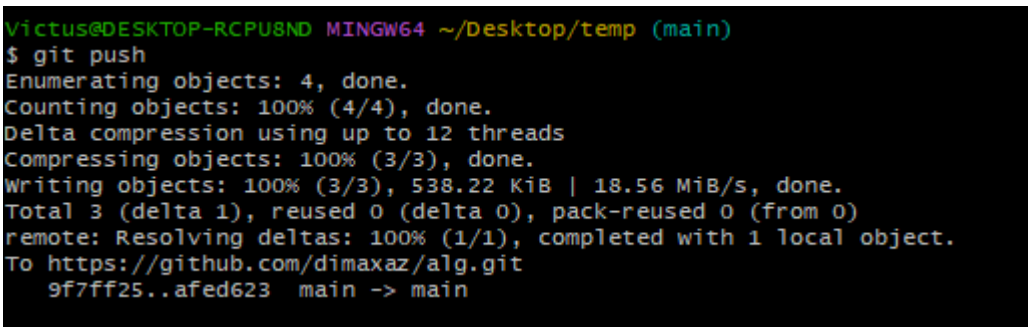

Рисунок 8 — Использование команды git push

Ссылка на удалённый репозиторий:

https://github.com/dimaxaz/alg.git

Заключение

В процессе выполнения задания был получен опыт совместной работы с использованием сервиса GitHub,а также навык использования инструмента Git Bash. Очень подробно был изучен алгоритм реализации сортировки выбором.

Мною было разработано меню программы, предлагающее выбор из нескольких опций. Помимо этого, я реализовал подключение файлов, в которых сохраняются результаты работы сгенерированных и отсортированных массивов.

В процессе практической работы были улучшены базовые навыки программирования на языке C++. Повысил уровень навыков отладки, умение находить и исправлять ошибки в коде, что повысило надежность и стабильность моих программных решений. Также я углубил знания о работе с разнообразными типами данных, что позволяет более гибко и эффективно использовать их в своих проектах.

Для дальнейшего улучшения программы можно реализовать внедрение библиотек, упрощающих реализацию алгоритма сортировки выбором в C++, что повысит производительность и читаемость кода. Также улучшение пользовательского опыта через разработку более интуитивного и привлекательного графического интерфейса, что сделает программу более доступной и удобной для использования.

Список используемой литературы

1. Язык программирования C++ Бьярне Страуструп 2013г.

2. ГОСТ 19.701 – 90 Схемы алгоритмов, программ, данных и систем.

3. Алгоритмы. Построение и анализ. Томас Х. Кормен, Чарльз Е. Лейзерсон, Рональд Л. Ривест, Клиффорд Штайн 2009г.

4. Git для профессионального программиста. Джон Лёлигер и Мэтью Маккалоу 2013г.

5. Программирование. Принципы и практика с использованием C++ Бьярне Страуструп 2018г.

Приложение А. Листинг программы

```
#include <iostream>

#include <fstream>

#include <conio.h>

#include <cstdlib> // Для rand() и srand()

#include <ctime>   // Для time(), clock_t, clock(),
CLOCKS_PER_SEC


void saveToFile(const std::string& filename, int* array, int
size) {

    std::ofstream file(filename);

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

        file << array[i] << " ";

    }

    file.close();

}


void selectionSort(int* array, int size, bool ascending, double&
duration, int& swapCount) {

    clock_t start = clock();

    swapCount = 0;


    for (int i = 0; i < size - 1; ++i) {

        int minMaxIndex = i;

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

            if (ascending ? array[j] < array[minMaxIndex] :
array[j] > array[minMaxIndex]) {

                minMaxIndex = j;

            }

        }
```

```
            if (minMaxIndex != i) {
                int temp = array[minMaxIndex];
                array[minMaxIndex] = array[i];
                array[i] = temp;
                ++swapCount;
            }
        }

        clock_t end = clock();
        duration = static_cast<double>(end - start) /
CLOCKS_PER_SEC;
}

void randomFillArray(int* array, int size) {
    for (int i = 0; i < size; ++i) {
        array[i] = rand() - rand();
    }
}

void manualFillArray(int* array, int size) {
    for (int i = 0; i < size; ++i) {
        std::cout << "Введите элемент " << i + 1 << ": ";
        std::cin >> array[i];
    }
}

void printMenu() {
    std::cout << "\t    +Меню программы+\n" << std::endl;
    std::cout <<
"_____________**********_____________\n\n\n";
```

```
    std::cout << "1 - Сортировка по возрастанию" << std::endl;
    std::cout << "2 - Сортировка по убыванию" << std::endl;
    std::cout << "3 - Ввести количество элементов массива" <<
std::endl;
    std::cout << "4 - Заполнить массив вручную" << std::endl;
    std::cout << "5 - Заполнить массив случайными числами" <<
std::endl;
    std::cout << "6 - Выйти\n\n" << std::endl;
    std::cout <<
"_____________***********_____________\n\n\n";
    std::cout << "Выберите желаемую опцию : ";
}


int main() {
    setlocale(LC_ALL, "RUS");
    srand(static_cast<unsigned>(time(0))); // Инициализация
генератора случайных чисел
    int size = 0;
    int* array = nullptr;
    bool running = true;
    while (running) {
        printMenu();
        int choice;
        std::cin >> choice;

        switch (choice) {
        case 1:
        case 2: {
            bool ascending = (choice == 1);
            if (array == nullptr) {
```

```
                std::cout << "Сначала создайте массив и
заполните его значениями.\n" << std::endl;

                std::cout << "Нажмите любую клавишу для
продолжения..." << std::endl;

                _getch();

                system("cls");

                break;

            }

            saveToFile("original_array.txt", array, size);

            std::cout << "Исходный массив сохранен в
original_array.txt\n";

            double duration;

            int swapCount;

            selectionSort(array, size, ascending, duration,
swapCount);



            saveToFile("sorted_array.txt", array, size);

            std::cout << "Массив отсортирован и сохранен в
sorted_array.txt\n" << std::endl;

            std::cout << "Время сортировки: " << duration << "
секунд\n";

            std::cout << "Количество перестановок: " <<
swapCount << "\n" << std::endl;

            std::cout << "Нажмите любую клавишу для
продолжения..." << std::endl;

            _getch();

            system("cls");

            break;

        }

        case 3: {

            std::cout << "Введите количество элементов массива:
";

            std::cin >> size;
```

```
            delete[] array;
            array = new int[size];
            std::cout << "Создан массив из " << size << "
элементов.\n" << std::endl;
            std::cout << "Нажмите любую клавишу для
продолжения..." << std::endl;
            _getch();
            system("cls");
            break;
        }
        case 4: {
            if (array == nullptr) {
                std::cout << "Сначала создайте массив, задав
количество элементов.\n" << std::endl;
                std::cout << "Нажмите любую клавишу для
продолжения..." << std::endl;
                _getch();
                system("cls");
                break;
            }
            manualFillArray(array, size);
            std::cout << "Массив заполнен вручную.\n" <<
std::endl;
            std::cout << "Нажмите любую клавишу для
продолжения..." << std::endl;
            _getch();
            system("cls");
            break;
        }
        case 5: {
            if (array == nullptr) {
```

```
                std::cout << "Сначала создайте массив, задав
количество элементов.\n" << std::endl;

                std::cout << "Нажмите любую клавишу для
продолжения..." << std::endl;

                _getch();

                system("cls");

                break;

            }

            randomFillArray(array, size);

            std::cout << "Массив заполнен случайными
значениями.\n" << std::endl;

            std::cout << "Нажмите любую клавишу для
продолжения..." << std::endl;

            _getch();

            system("cls");

            break;

        }

        case 6:

            running = false;

            break;

        default:

            std::cout << "Неверный выбор. Попробуйте снова.\n"
<< std::endl;

            std::cout << "Нажмите любую клавишу для
продолжения..." << std::endl;

            _getch();

            system("cls");

        }

    }

    delete[] array; // Освобождение выделенной памяти

    return 0;

}
```

Приложение В. Тестирование программы

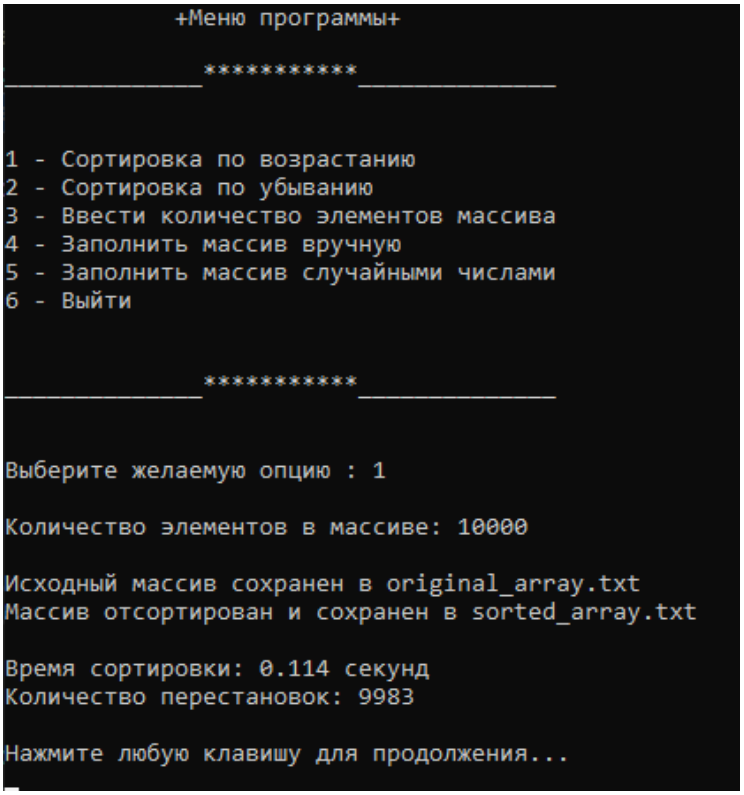

Рисунок В1

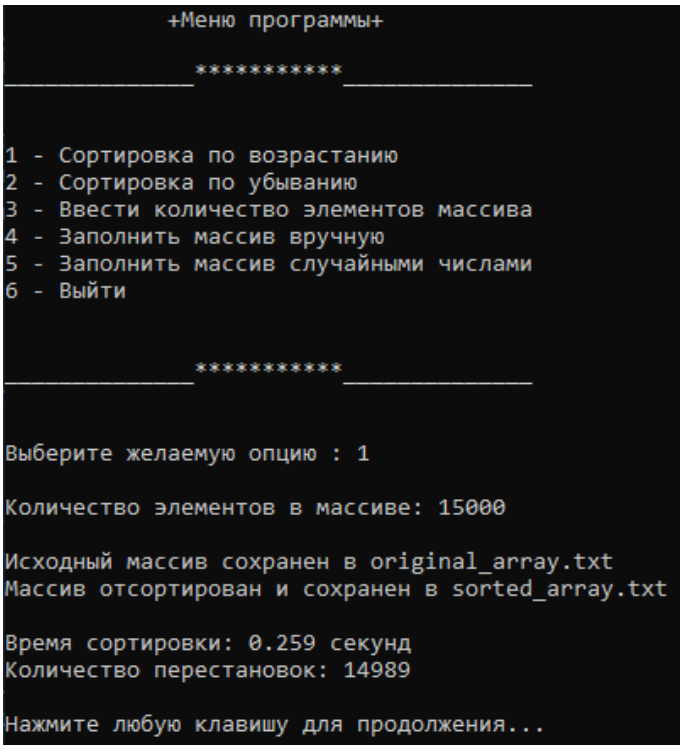

Рисунок В2

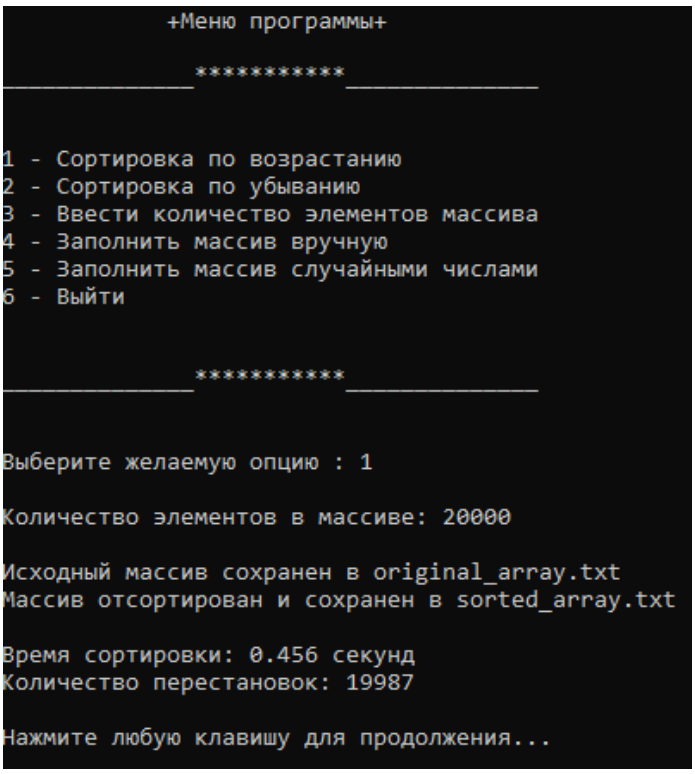

Рисунок В3

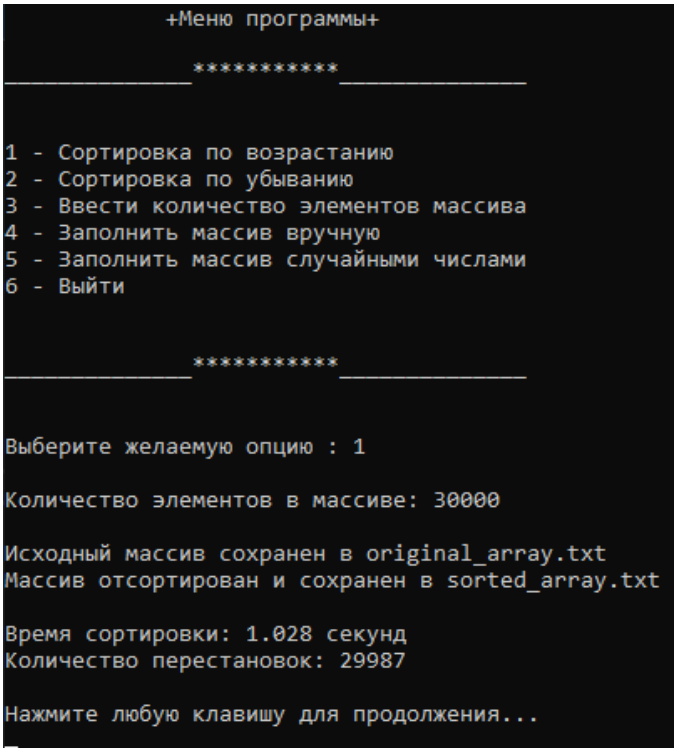

Рисунок В4

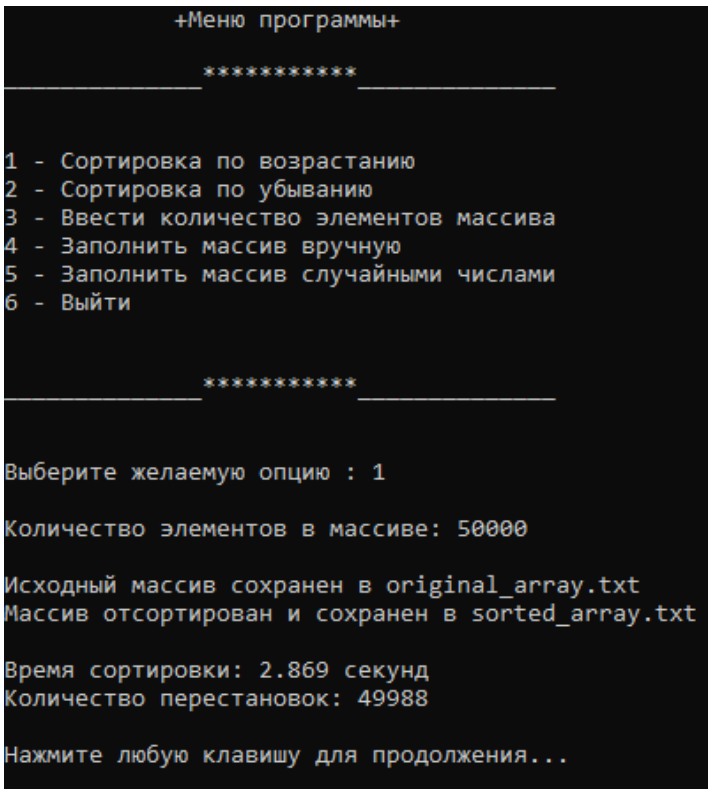

Рисунок В5

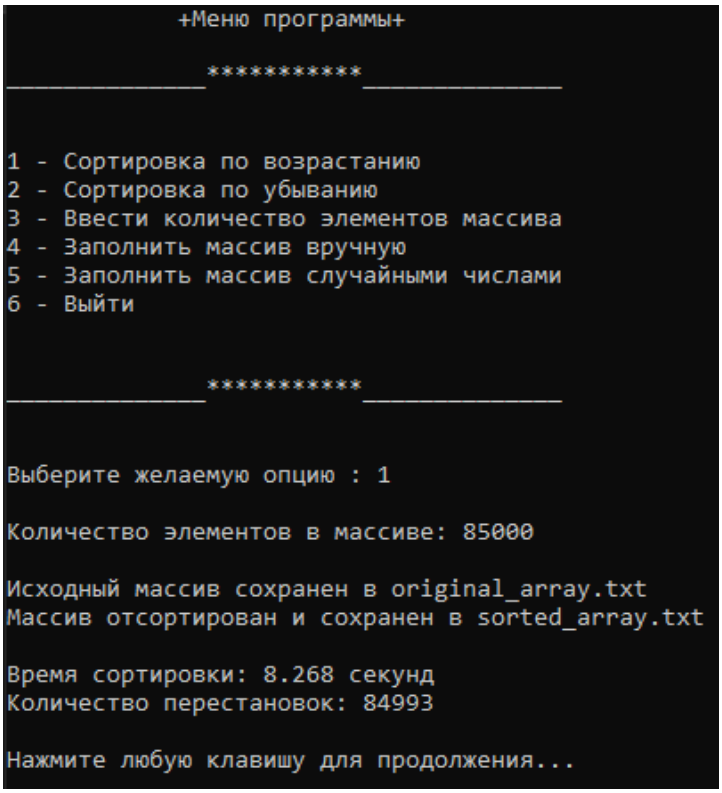

Рисунок В6

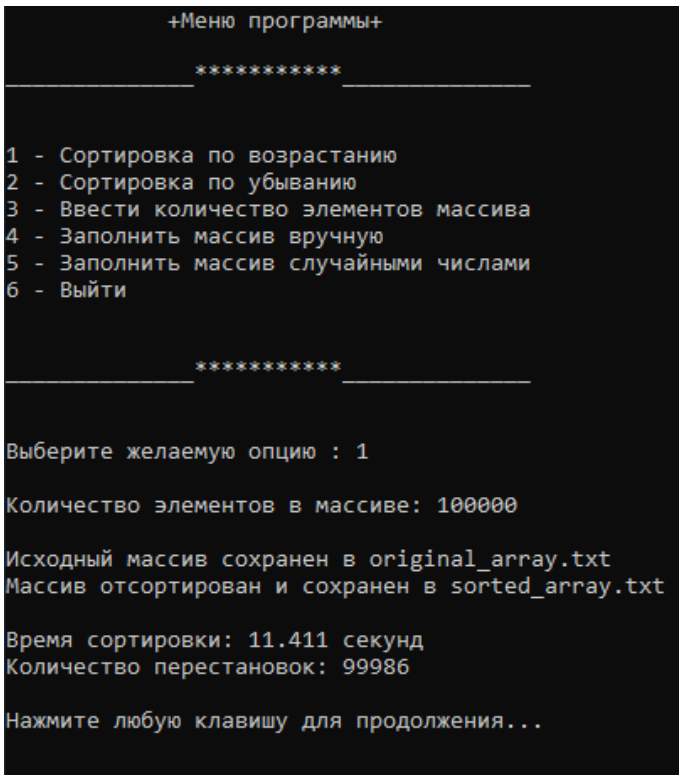

Рисунок В7

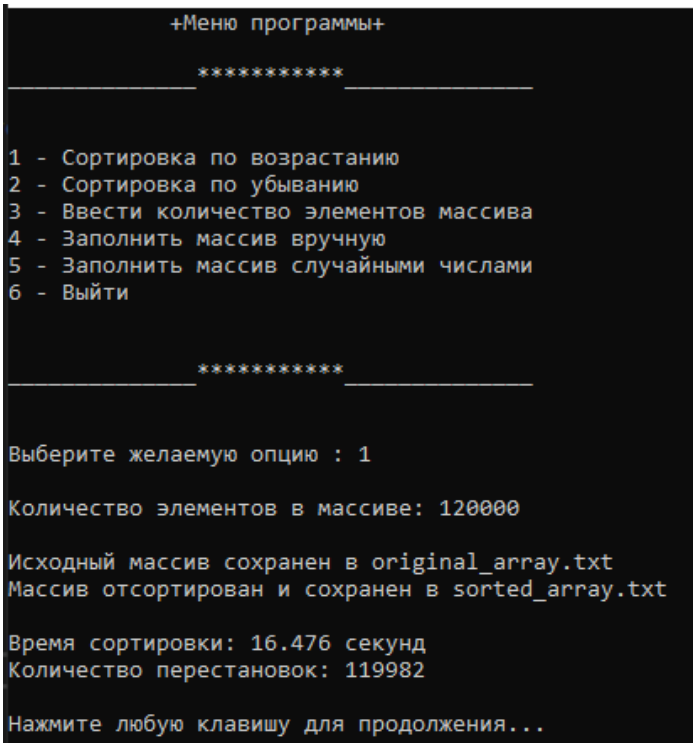

Рисунок В8

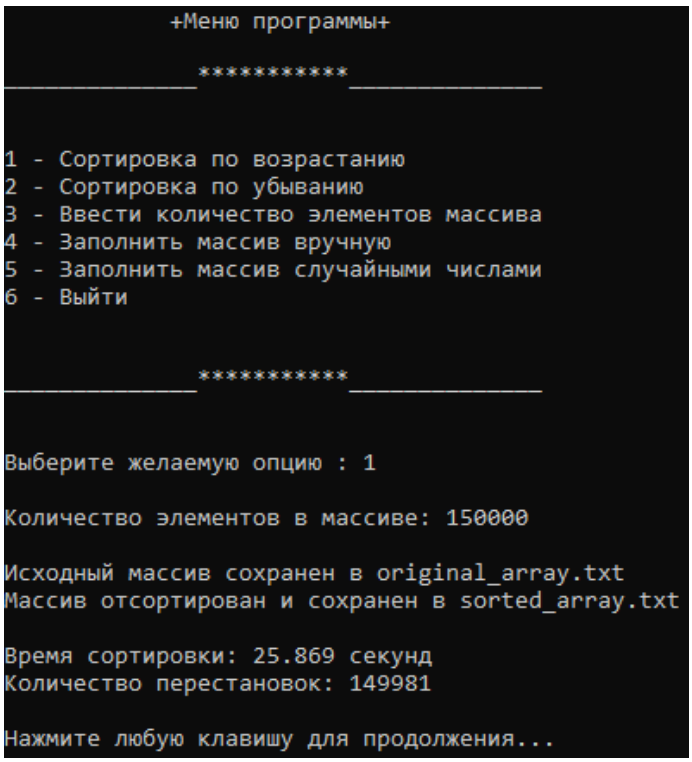

Рисунок В9

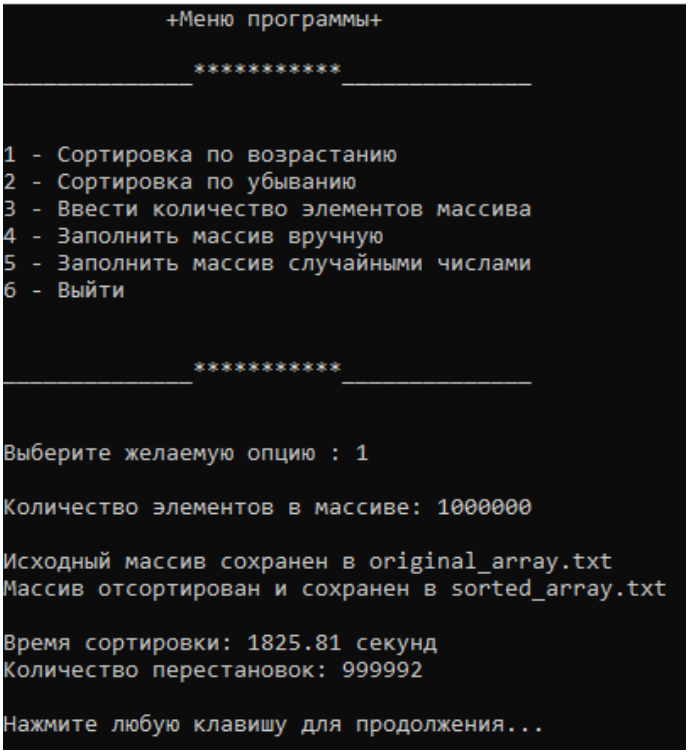

Рисунок В10