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

(2023/2024 учебный год)

Кузнецов Кирилл Игоревич

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

Наименование профиля подготовки «Программное обеспечение средств вычислительной техники и автоматизированных систем»

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 к/н, доцент, Карамышева Н.С.

(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

(2023/2024 учебный год)

Кузнецов Кирилл Игоревич

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

Наименование профиля подготовки «Программное обеспечение средств вычислительной техники и автоматизированных систем»

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 к/н, доцент, Карамышева Н.С.

(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	25.06.24 - 25.06.24	
2	Подбор и изучение материала по теме работы	15	26.06.24 – 01.07.24	
3	Разработка алгоритма	43	01.07.24 – 02.07.24	
4	Описание алгоритма и программы	18	02.07.24 – 03.07.24	
5	Тестирование	5	03.07.24 – 05.07.24	
6	Получение и анализ результатов	10	05.07.24 – 06.07.24	
7	Оформление отчёта	15	06.07.24 – 08.07.24	
	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

(2023/2024 учебный год)

Кузнецов Кирилл Игоревич

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

Наименование профиля подготовки «Программное обеспечение средств вычислительной техники и автоматизированных систем»

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

Год обучения 1 семестр 2

Период прохождения практики с 25.06.2023 по 8.07.2024

Кафедра «Вычислительная техника»

Кузнецов К.И. выполнял практическое задание «Двоичная сортировка». На первоначальном этапе был изучен и проанализирован алгоритм двоичной сортировки, был выбран метод решения и язык программирования C++, а также библиотека SFML для реализации GUI. Также, реализовал работу с файлами. Оформила отчёт.

Бакалавр Кузнецов К.И. _____ "___"_____2024 г.

Руководитель практики Карамышева Н.С. _____ "___"_____2024 г.

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

ОТЗЫВ

О ПРОХОЖДЕНИИ УЧЕБНОЙ (ОЗНАКОМИТЕЛЬНОЙ) ПРАКТИКИ

(2023/2024 учебный год)

Кузнецов Кирилл Игоревич

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

Наименование профиля подготовки «Программное обеспечение средств вычислительной техники и автоматизированных систем»

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 Кузнецов К.И. решал следующую задачу: реализация работы с файлами.

За период выполнения практики были освоены основные понятия и технологии сортировки вставками, реализованы методы работы с файлами. Во время выполнения работы Кузнецов К.И. 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 Кузнецов К.И. заслуживает оценки «_____».

Руководитель практики к/н, доцент, Карамышева Н.С. « » 2024 г.

Содержание

Введение

В эпоху стремительного развития компьютерных технологий сортировка данных стала одним из ключевых процессов в обработке информации. Эта задача широко распространена в различных профессиональных областях.

Алгоритмы сортировки представляют собой особую категорию алгоритмов, которые находят применение практически во всех аспектах обработки информации. Их тесная взаимосвязь позволяет выделить их в отдельный класс. Основная цель применения алгоритмов сортировки - оптимизация последующего поиска. Например, использование словарей было бы затруднительно без алфавитного порядка слов.

Значимость сортировки заключается в том, что на ее примере можно продемонстрировать множество фундаментальных методов и приемов построения алгоритмов. Сортировка также иллюстрирует разнообразие алгоритмических подходов к решению одной задачи, причем некоторые из них имеют определенные преимущества перед другими. Усложнение алгоритма может значительно повысить его эффективность и быстродействие по сравнению с более простыми методами. В общем понимании, сортировка - это процесс упорядочивания элементов множества в определенной последовательности.

Сортировка с помощью двоичного дерева – это универсальный алгоритм сортировки, который заключается в построении двоичного дерева поиска по ключам массива и сборке результирующего массива путем обхода узлов построенного дерева в необходимом порядке следования ключей. Данная сортировка является оптимальной при получении данных путем непосредственного чтения из потока (файла, сокета, консоли). [1]

1 Постановка задачи

Поставленная задача: необходимо заполнить массив из n-ого количества элементов случайными числами, записать данные элементы в отдельный файл. После этого выполнить двоичную сортировку над данными,находящимися в массиве, записать отсортированные данные в другой файл, посчитать время выполнения.

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

Оформить отчет по проведенной практике.

1.1 Достоинства алгоритма бинарной сортировки

- алгоритм эффективен при поиске определенного элемента;
- экономия памяти;
- простая реализация алгоритма.

1.2 Недостатки алгоритма бинарной сортировки

- ограничение 2 дочерними узлами;
- высокая алгоритмическая сложность $O(n^2)$;
- сложный алгоритм балансировки.

1.3 Типичные сценарии применения данного алгоритма

- товары в магазине (сортировка по цене, году выпуска, габаритам,весу, срокам поставки);
- студенты в вузе (сортировка по среднему балу, кол-ву прогулов,уровню IQ, числу хвостов, ФИО);
- города/страны (сортировка по населению, рождаемости, ВВП, ВВПна душу населения);

2 Выбор решения

Нашей бригадой было выбрано вести разработку в среде Microsoft Visual Studio на языке C++.

Для написания данной программы будет использован язык программирования C++. Язык программирования C++ представляет высокоуровневый компилируемый язык программирования общего назначения со статической типизацией, который подходит для создания самых различных приложений. На сегодняшний день C++ является одним из самых популярных и распространенных языков. C++ является мощным языком, унаследовав от Си богатые возможности по работе с памятью. Поэтому нередко C++ находит свое применение в системном программировании, в частности, при создании операционных систем, драйверов, различных утилит, антивирусов и т.д. [2]

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 GitHub Desktop. GitHub Desktop — это приложение, которое помогает работать с файлами, размещенными на GitHub или других службах размещения Git. GitHub Desktop можно использовать вместе с любыми инструментами, которые необходимо внести в проект.

3 Описание программы

Для реализации работы с файлами в программе были разработаны 3 функции. Эти функции работают с файлами, содержащими числа, и выполняют различные операции по их чтению, записи и генерации.

Функция parseNumbersFromFile – реализует читает файл по указанному пути, построчно считывает строки, разделяет их по запятым, пытается преобразовать каждый элемент в число и добавляет его в вектор чисел. В случае ошибки открытия файла или преобразования числа выводит сообщение об ошибке.

Код функции parseNumbersFromFile:

```
void parseNumbersFromFile(const std::string& path, std::vector<int>& numbers)
{
    std::ifstream file(path);

    if (!file.is_open()) {
        std::cerr << "Could not open the file " << path << std::endl;
        return;
    }

    std::string line;
    while (std::getline(file, line)) {
        std::stringstream ss(line);
        std::string item;
        while (std::getline(ss, item, ',')) {
            try {
                numbers.push_back(std::stoi(item));
            }
            catch (const std::invalid_argument& e) {
                std::cerr << "Invalid number: " << item << std::endl;
            }
            catch (const std::out_of_range& e) {
                std::cerr << "Number out of range: " << item << std::endl;
            }
        }
    }

    file.close();
}
```

Функция generateNumbersFile – создает файл по указанному пути, генерирует заданное количество случайных чисел, записывает их в файл, разделяя запятыми. В случае ошибки открытия файла выводит сообщение об ошибке.

Код функции generateNumbersFile:

```
void generateNumbersFile(const std::string& path, int numElements) {

    std::ofstream outFile(path);
```

```
        if (!outFile) {


    std::cerr << "Error opening file: " << path << std::endl;
            return;
        }
    srand(time(NULL));
        for (int i = 0; i < numElements; ++i) {
            int number = (rand() % 10000) * 10 + (rand() % 10);
            if (rand() % 2 == 0) {
                number *= -1;
            }
            outFile << number;
            if (i < numElements - 1) {
                outFile << ",";
            }
        }
        outFile.close();}
```

Функция writeNumbersToFile – открывает файл по указанному пути и записывает в него числа из переданного вектора, разделяя их запятыми. В случае ошибки открытия файла выводит сообщение об ошибке.

Код функции openFile:

```
void writeNumbersToFile(const std::string& path, const std::vector<int>&
data) {
    std::ofstream outFile(path);
    if (!outFile.is_open()) {
        std::cerr << "Не удалось открыть файл для записи: " << path <<
std::endl;
        return;
    }
    for (size_t i = 0; i < data.size(); ++i) {
        outFile << data[i];
        if (i < data.size() - 1) {outFile << ",";}
    }
```

4. Схемы программы

На рисунке 1 представлена схема функции parseNumbersFromFile.

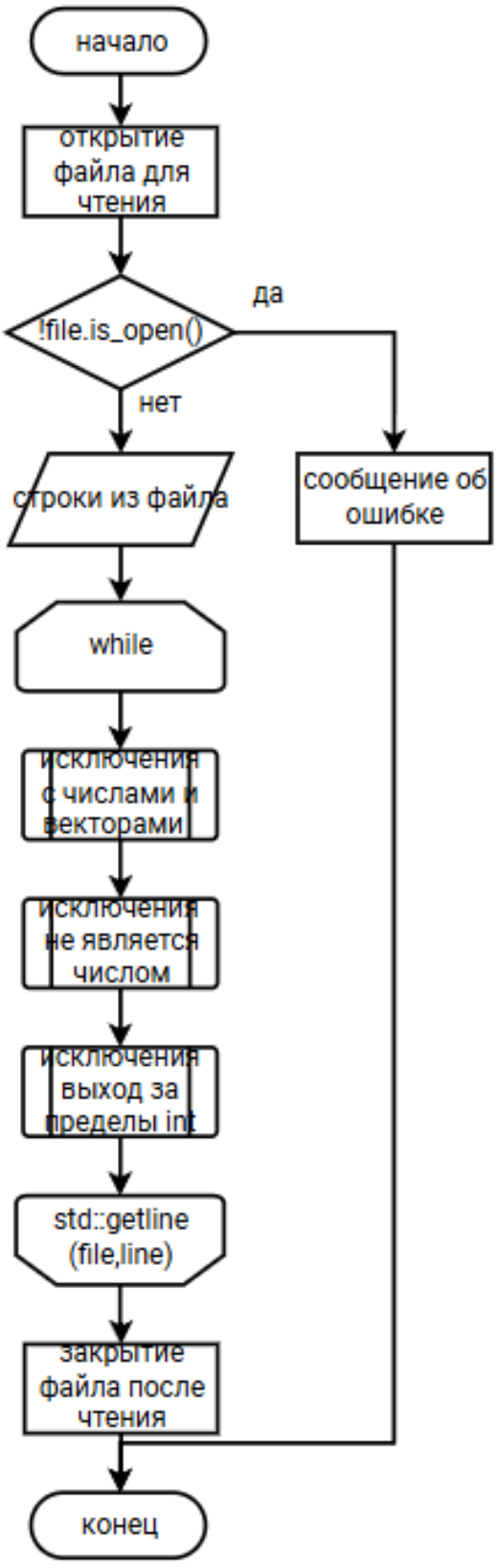

Рисунок 1 - Блок-схема функции parseNumbersFromFile

4.1 Блок-схема программы

На рисунке 2 представлена схема функции genereteNumbersFile.

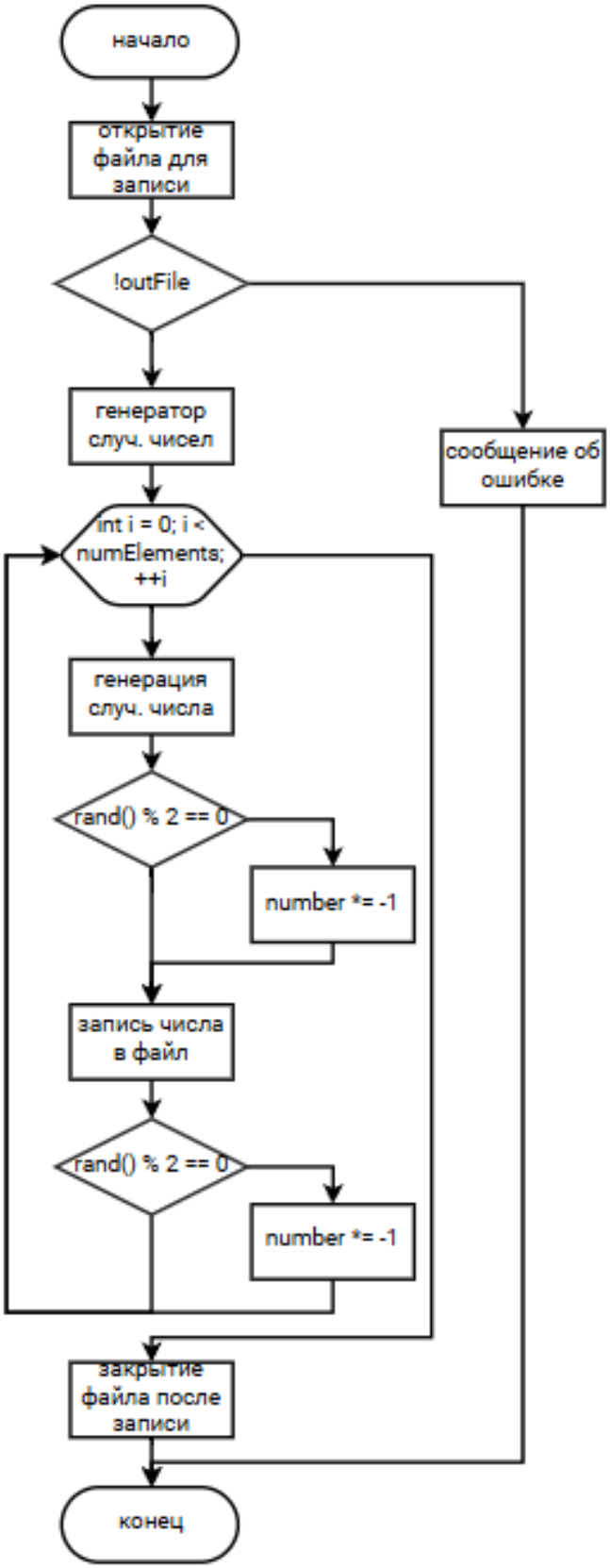

Рисунок 2 – Блок-схема функции genereteNumbersFile

4.2 Блок-схема программы

На рисунке 3 представлена схема функции writeNumbersToFile.

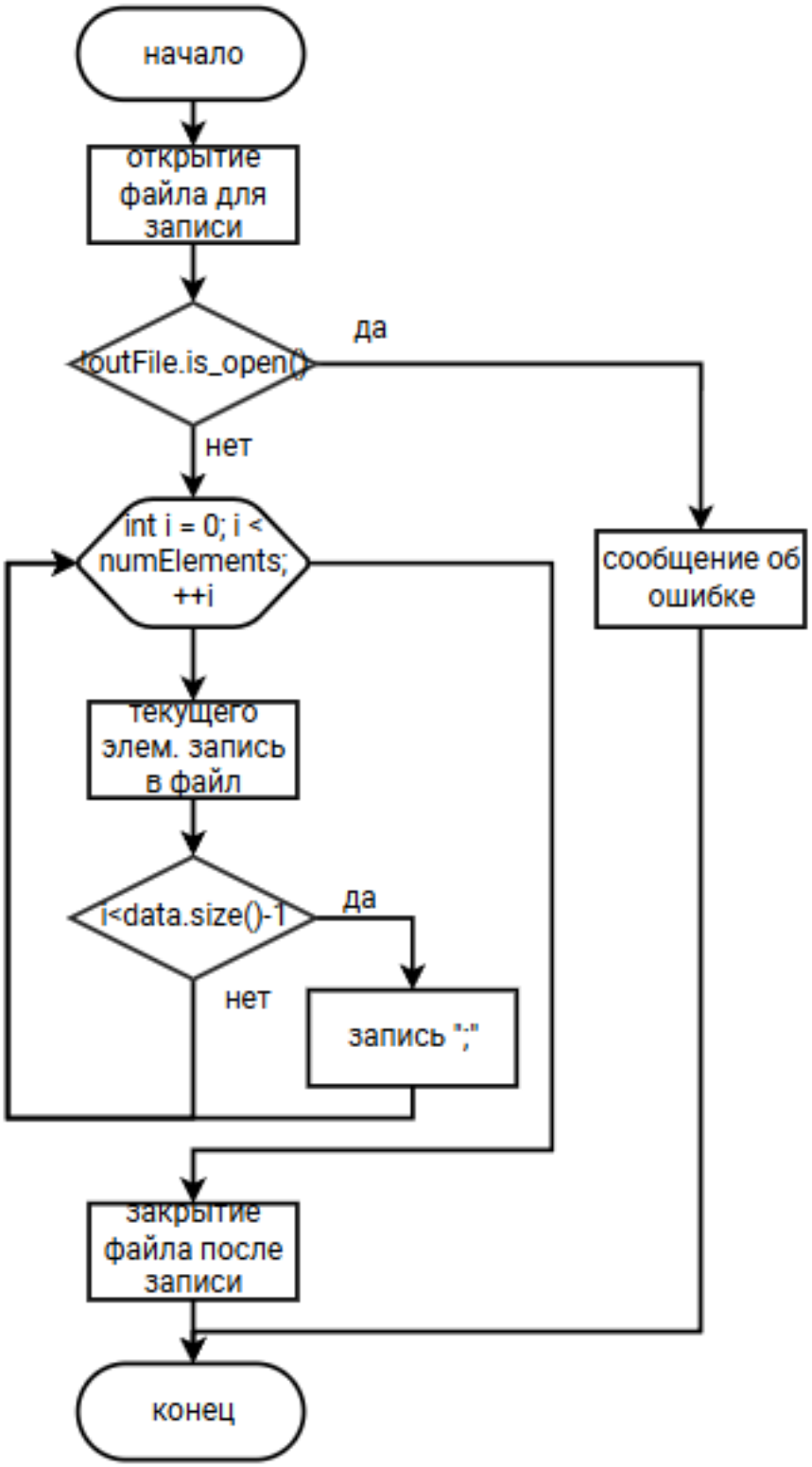

Рисунок 3 - Блок-схема функции writeNumbersToFile

5 Тестирование программы

Тестирование показало, что с увеличением количества элементов пропорционально увеличивается время работы программы, ниже представлен график результатов тестирования.

Рисунок 4 – Результаты тестирования пользовательских значений

Была выполнена проверка работоспособности кнопок меню, связанных с файлами.

При вводе значения программа выполняется (рис 5), в папке FILES создается два файла (рис 6).

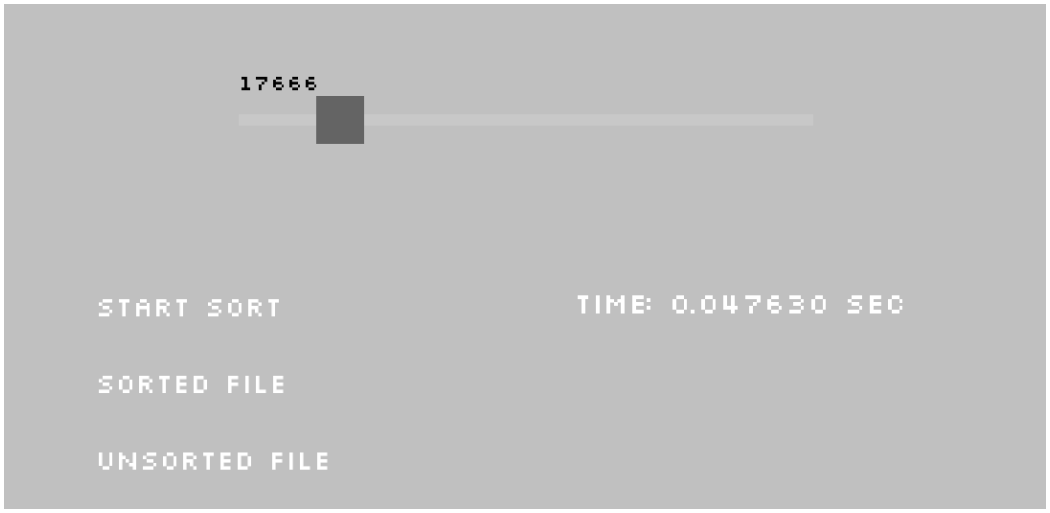

Рисунок 5 – Результат выполнения программы для значения пользователя

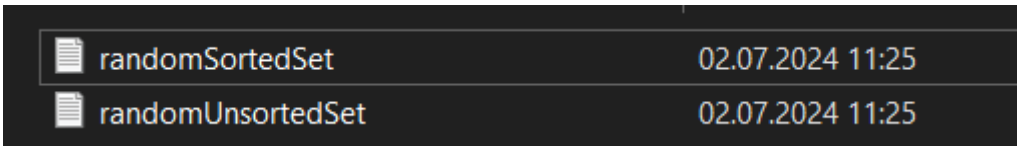

Рисунок 6 – Созданные файлы

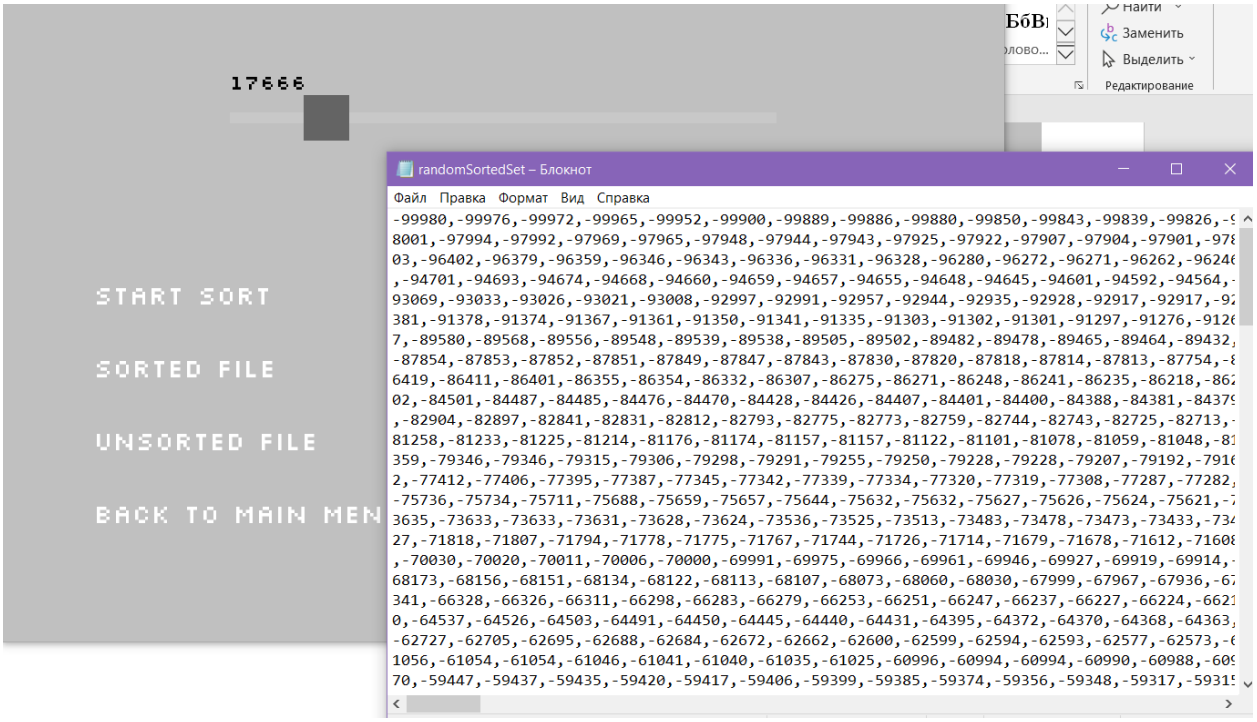

Рисунок 7 – Открытие файла при нажатии на кнопку «Sorted file»

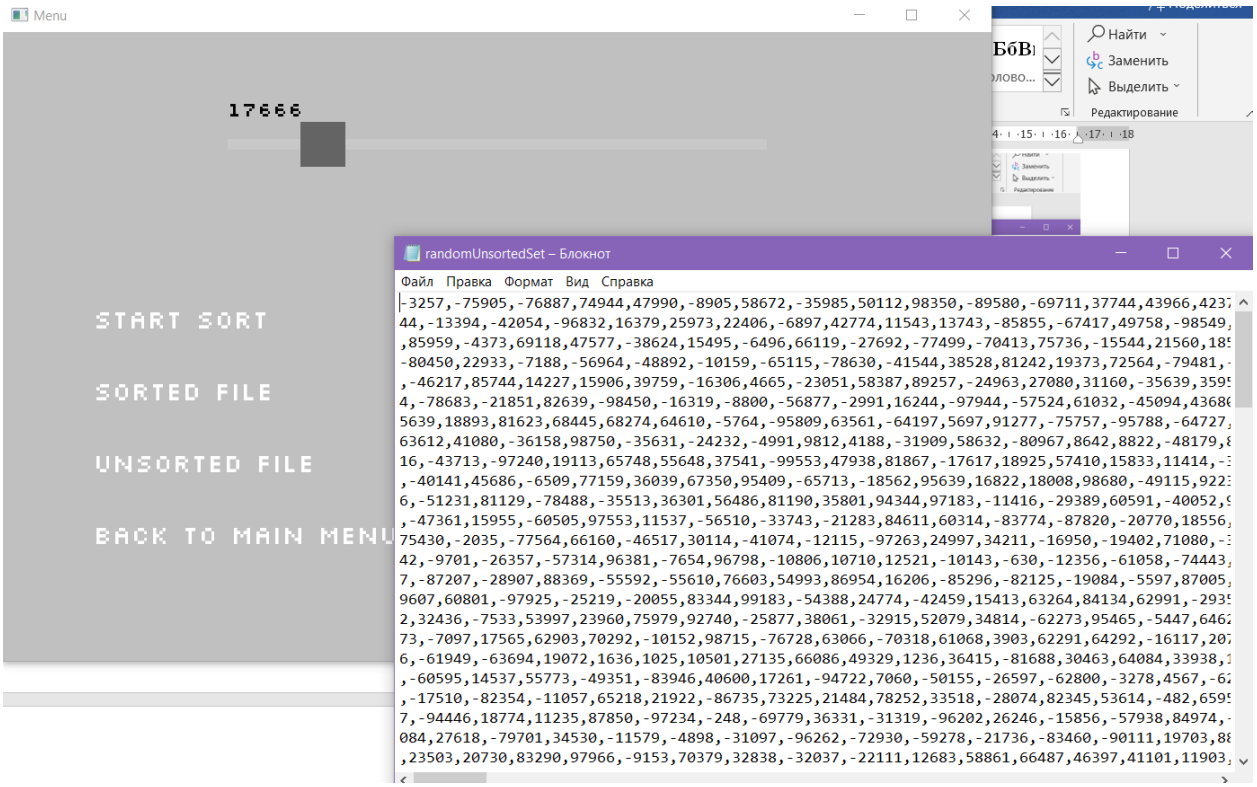

Рисунок 8 – Открытие файла при нажатии на кнопку «Unsorted file»

6 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

7 Совместная разработка

Для удобства совместной разработки был использован сервис GitHub Desktop.

Разделили роли, назначили исполнителей задачам.

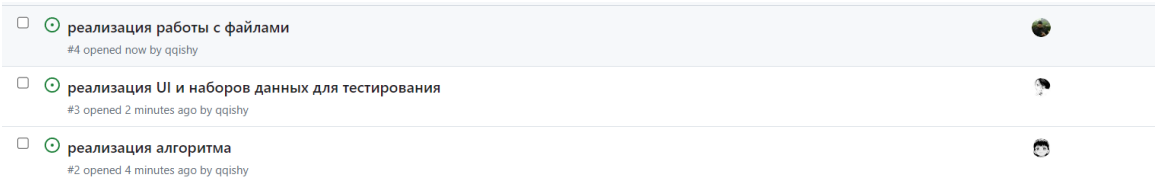

Рисунок 9 – Распределение задач

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

Мною были написаны функции для работы с файлами, которые осуществляют генерацию числе чтение и запись в файл.

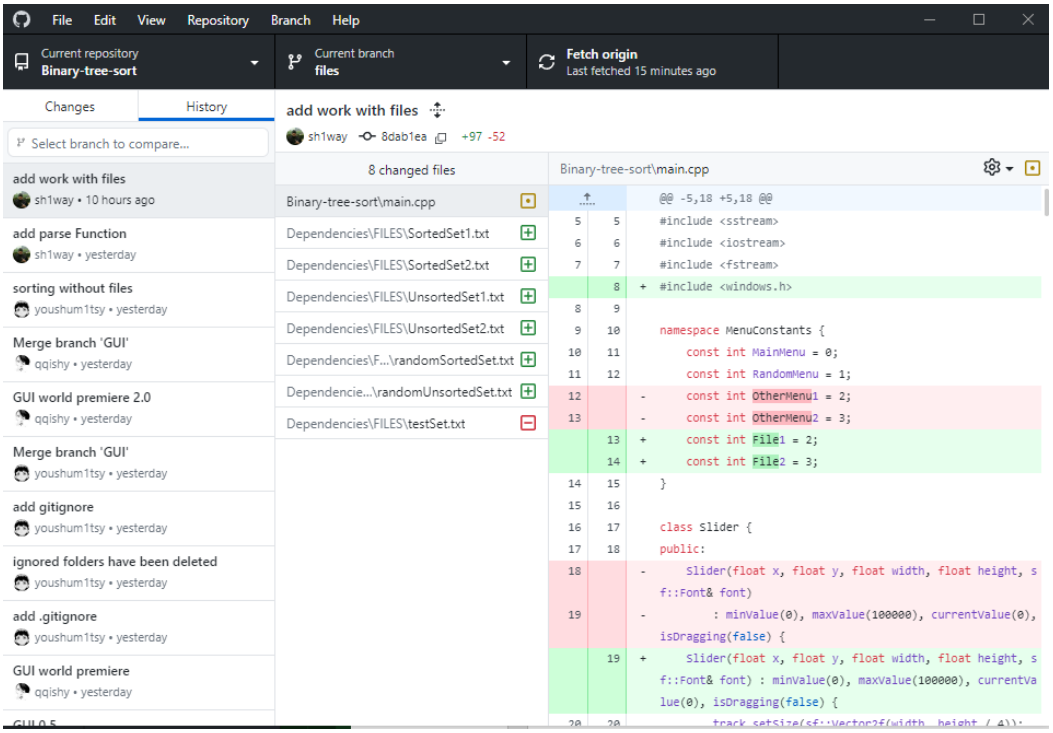

Рисунок 10 –изменения на ветке

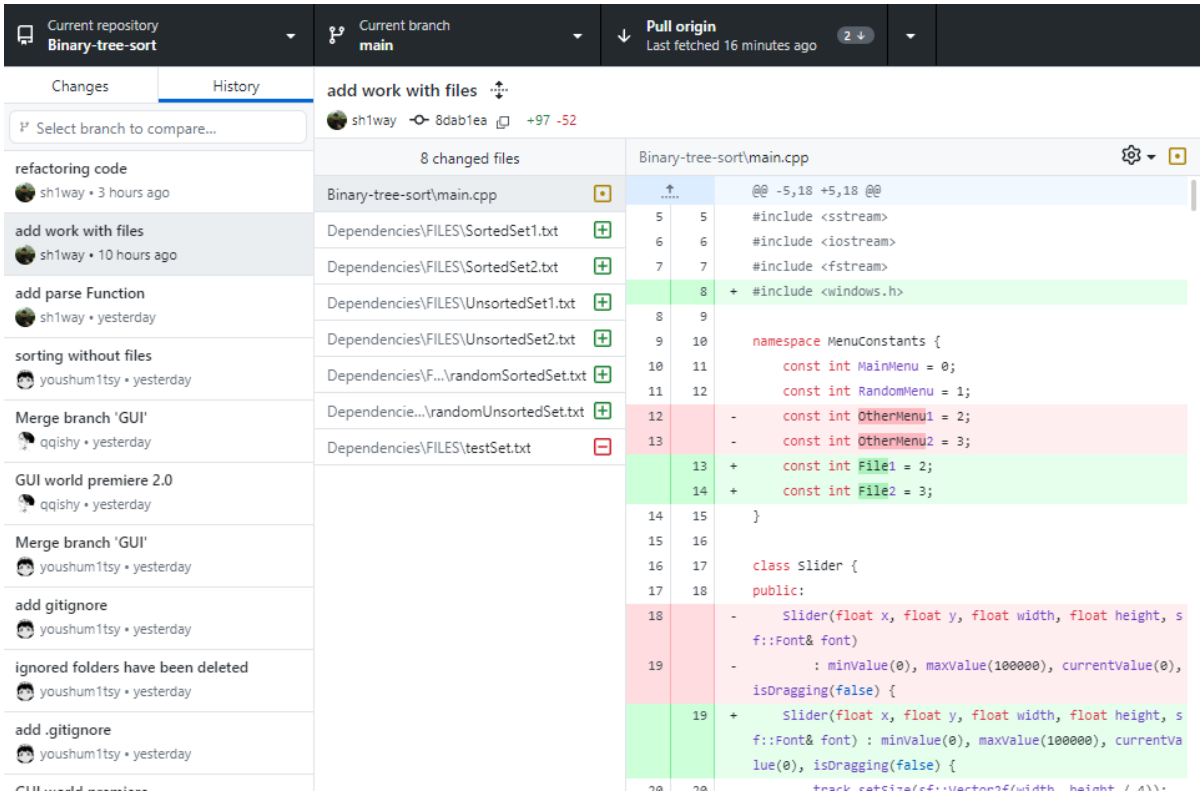

Рисунок 11 – слияние веток

Репозиторий находится на платформе GitHub в общественном доступе[3].

Заключение

При выполнении данной работы были получены навыки совместной работы с помощью сервисов GitHub. Был изучен алгоритм двоичной сортировки.

Мною были реализованы функции для работы с файлами, которые осуществляют запись генерируемых значений и их чтение.

При выполнении практической работы были улучшены базовые навыки программирования на языке C++. 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. К 78 Алгоритмы. Просто как дважды два / И. В. Красиков, И. Е. Красикова. — М. : Эксмо, 2007. — 256 с. — (Просто как дважды два).
2. Бьерн Страуструп. Язык программирования C++. Специальное издание = The C++ programming language. Special edition. — М.: Бином-Пресс, 2007. — 1104 с.
3. Youshum1tsu. Binary tree sort: репозиторий исходного кода [Электронный ресурс]. URL: https://github.com/youshum1tsy/Binary-tree-sort

Приложение А. Листинг программы

```
#include
<SFML/Graphics.hpp>

#include <string>

#include <cmath>

#include <iomanip>

#include <sstream>

#include <iostream>

#include <fstream>

#include <windows.h>



namespace MenuConstants {

    const int MainMenu = 0;

    const int RandomMenu =
1;

    const int File1 = 2;

    const int File2 = 3;

}



class Slider {

public:

    Slider(float x, float
y, float width, float
height, sf::Font& font) :
minValue(0),
maxValue(100000),
currentValue(0),
isDragging(false) {
```

```
track.setSize(sf::Vector2f(
width, height / 4));


track.setPosition(x, y +
height / 2 -
track.getSize().y / 2);


track.setFillColor(sf::Colo
r(200, 200, 200));


knob.setSize(sf::Vector2f(h
eight, height));


knob.setOrigin(knob.getSize
().x / 2, knob.getSize().y
/ 2);

        knob.setPosition(x,
y + height / 2);


knob.setFillColor(sf::Color
(100, 100, 100));


valueText.setFont(font);


valueText.setCharacterSize(
24);


valueText.setFillColor(sf::
Color::Black);


valueText.setPosition(x, y
- 30);

        updateValueText();
```

```
    }

    void handleEvent(const
sf::Event& event) {

        if (event.type ==
sf::Event::MouseButtonPress
ed) {

            sf::Vector2f
mousePos(event.mouseButton.
x, event.mouseButton.y);

            if
(knob.getGlobalBounds().con
tains(mousePos)) {

                isDragging
= true;

            }

        }

        else if (event.type
==
sf::Event::MouseButtonRelea
sed) {

            isDragging =
false;

        }

        else if (event.type
== sf::Event::MouseMoved) {

            if (isDragging)
{

                float newX
=
static_cast<float>(event.mo
useMove.x);

                newX =
```

```
std::max(track.getPosition(
).x, std::min(newX,
track.getPosition().x +
track.getSize().x));


knob.setPosition(newX,
knob.getPosition().y);


currentValue =
static_cast<int>(minValue +
(maxValue - minValue) *
((newX -
track.getPosition().x) /
track.getSize().x));


updateValueText();

            }

        }

    }

    void
draw(sf::RenderWindow&
window) const {

        window.draw(track);

        window.draw(knob);


window.draw(valueText);

    }

    int getValue() const {

        return
currentValue;

    }

    void setValue(int
```

```
value) {

        currentValue =
std::max(minValue,
std::min(maxValue, value));

        float newX =
track.getPosition().x +
track.getSize().x *
((currentValue - minValue)
/
static_cast<float>(maxValue
- minValue));


knob.setPosition(newX,
knob.getPosition().y);

        updateValueText();

    }

private:

    void updateValueText()
{

std::vector<int> data;


clock.restart();


parseNumbersFromFile("../De
pendencies/FILES/UnsortedSe
t2.txt", data);

        else if
(currentMenu ==
MenuConstants::File1) {

            for (int i = 0;
i < 4; ++i) {


window.draw(text);
```

```
window.draw(OtherMenu1[i]);

            }

        }

        else if
(currentMenu ==
MenuConstants::File2) {

            for (int i = 0;
i < 4; ++i) {


window.draw(text);


window.draw(OtherMenu2[i]);

            }

        }

        window.display();

    }



    return 0;
```