МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИПЕНЗЕНСКИЙ 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)

Мишанина Анна Алексеевна

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

Наименование профиля подготовки «Программное обеспечение средстввычислительной 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 ФЕДЕРАЦИИПЕНЗЕНСКИЙ 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)

Мишанина Анна Алексеевна

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

Наименование профиля подготовки «Программное обеспечение средстввычислительной техники и автоматизированных систем»

Форма обучения – очная Срок обучения в соответствии с

ФГОС – 4 годаГод

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)

Мишанина Анна Алексеевна

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

Наименование профиля подготовки «Программное обеспечение средстввычислительной 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»

Мишанина А.А. выполняла 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, осуществила работу с наборами данных для тестирования и создала меню. Оформила отчёт.

Бакалавр Мишанина А.А. _____ "___"_____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)

Мишанина Анна Алексеевна

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

Наименование профиля подготовки «Программное обеспечение средстввычислительной 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 Мишанина А.А. решала следующие задачи: реализация GUI, работа с наборами данных для тестирования.

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

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

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++ с использованием библиотеки SFML. [2]

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. [3]

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

При запуске программы выводится меню из четырех пунктов:

- Random – ввод своего значения;
- Open file 1 – открыть меню для работы с первым файлом с заданными значениями;
- Open file 2 – открыть меню для работы со вторым файлом с заданными значениями;
- Exit – выход.

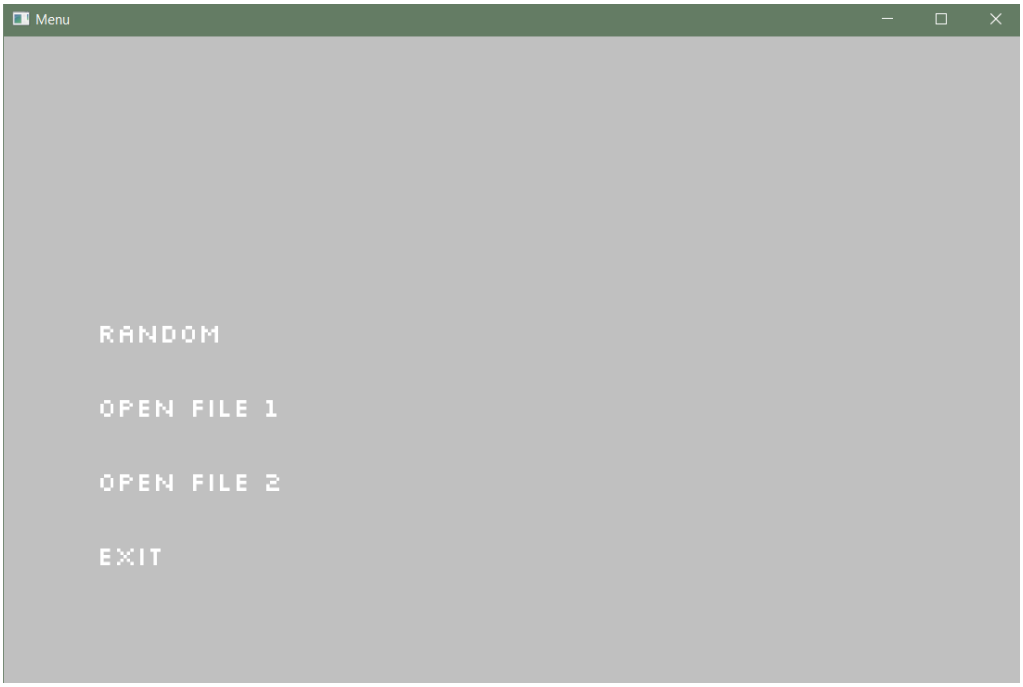

Рисунок 1 – главное меню

Код для главного меню:

```
if (currentMenu == MenuConstants::MainMenu) {
                if (event.type == sf::Event::MouseButtonPressed) {
                    sf::Vector2i mousePos = sf::Mouse::getPosition(window);
                    for (int i = 0; i < 4; ++i) {
                        if (mainMenu[i].getGlobalBounds().contains(mousePos.x,
mousePos.y)) {
                            if (i == 0) {
                                currentMenu = MenuConstants::RandomMenu;
                            }
                            else if (i == 1) {
                                currentMenu = MenuConstants::OtherMenu1;
                            }
                            else if (i == 2) {
```

```
                    currentMenu = MenuConstants::OtherMenu2;

                else if (i == 3) {
                    window.close();
                }
            }
        }
    }
}
```

Пользователю требуется выбрать тот пункт, который ему требуется. Привыборе варианта «Random» выводится другое меню.

В нем:

- Слайдер сверху для ввода своего значения;
- Start sort – начать сортировку;
- Sorted file – открыть отсортированный файл;
- Unsorted file – открыть неотсортированный файл;
- Back to main menu – выход в главное меню;
- Time – время сортировки.

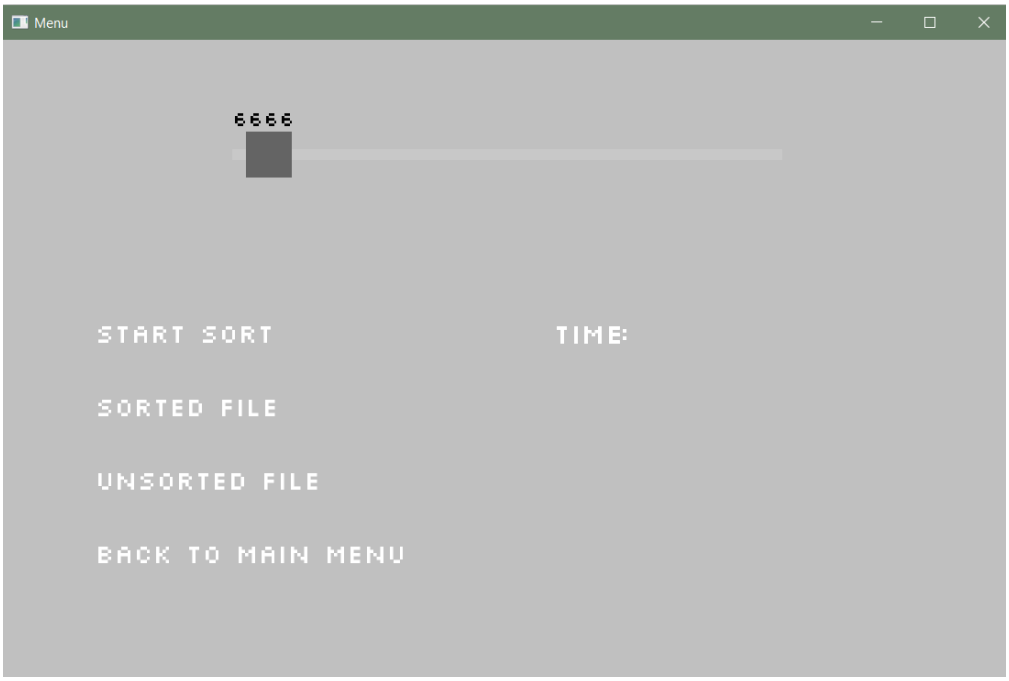

Рисунок 2 – меню при нажатии на «Random»

Код для меню «Random»:

```
else if (currentMenu == MenuConstants::RandomMenu) {
            std::string timeString = "Time:";
            text.setString(timeString);
            text.setPosition(600, 299);
            slider.handleEvent(event);
            if (event.type == sf::Event::MouseButtonPressed) {
                sf::Vector2i mousePos = sf::Mouse::getPosition(window);
                for (int i = 0; i < 4; ++i) {
                    if (mainMenu[i].getGlobalBounds().contains(mousePos.x,
mousePos.y)) {
                        if (i == 0) {
                            std::cout << "start" << "\n";
                        }
                        else if (i == 1) {
                            openFile("file1.txt");
                        }
                        else if (i == 2) {
                            openFile("file2.txt");
                        }
                        else if (i == 3) {
                            currentMenu = MenuConstants::MainMenu;
                        }
                    }
                }
            }
        }
```

При выборе варианта «Open file 1» выводится другой пункт меню.

В нем:

- Start sort – начать сортировку;
- Sorted file – открыть отсортированный файл;
- Unsorted file – открыть неотсортированный файл;
- Back to main menu – выход в главное меню;
- Time – время сортировки.

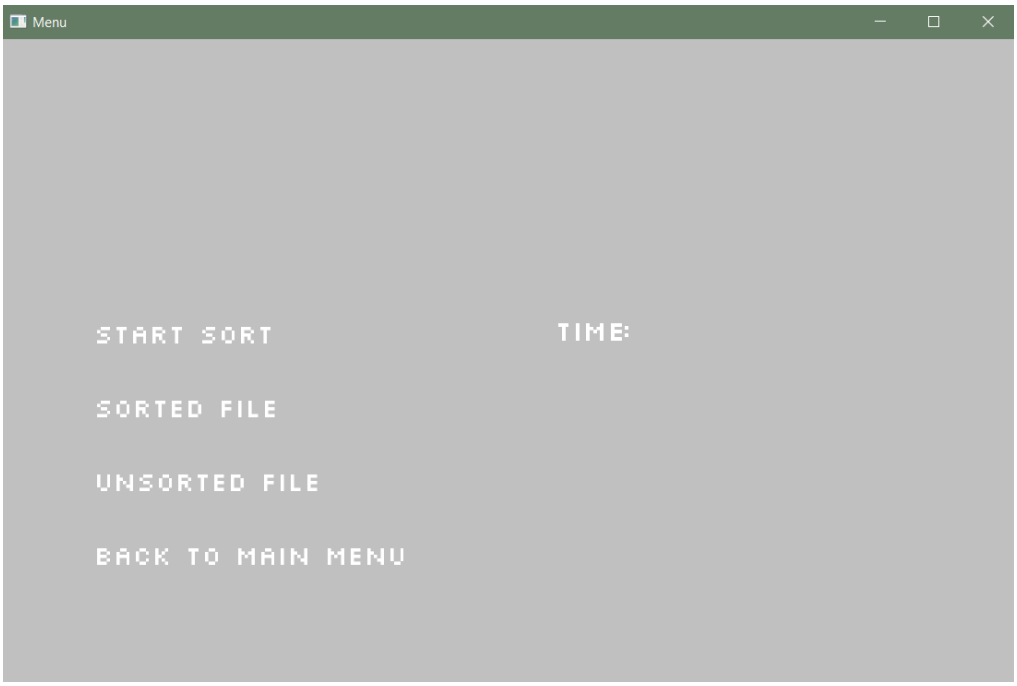

Рисунок 3 – код для меню «Open file 1»

Код для меню «Open file 1»/ «Open file 2»:

```
    else if (currentMenu == MenuConstants::OtherMenu1) {
                    std::string timeString = "Time:";
                    text.setString(timeString);
                    text.setPosition(600, 295);
                    slider.handleEvent(event);
                    if                (event.type                ==
sf::Event::MouseButtonPressed) {
                        sf::Vector2i          mousePos          =
sf::Mouse::getPosition(window);
                        for (int i = 0; i < 4; ++i) {
                            if
(mainMenu[i].getGlobalBounds().contains(mousePos.x, mousePos.y)) {
```

```
        if (i == 0) {
                                        std::cout << "start" << "\n";
                                    }
                                    else if (i == 1) {
                                        openFile("file1.txt");

                                    else if (i == 2) {
                                        openFile("file2.txt");
                                    }
                                    else if (i == 3) {
                                        currentMenu                    =
    MenuConstants::MainMenu;
                                    }
                                }
                            }
                        }

                    }
```

При выборе пункта меню «Open file 1» программа выглядит аналогично пункту под буквой «Open file 2».

При выборе пункта меню «Exit» программа завершает выполнение.

Подробный алгоритм работы программы и функции сортировки представлен в подразделе 4.1-4.3.

Листинг программы приведен в приложении А.

4. Схемы программы

На рисунке 4 представлен алгоритм работы главного меню.

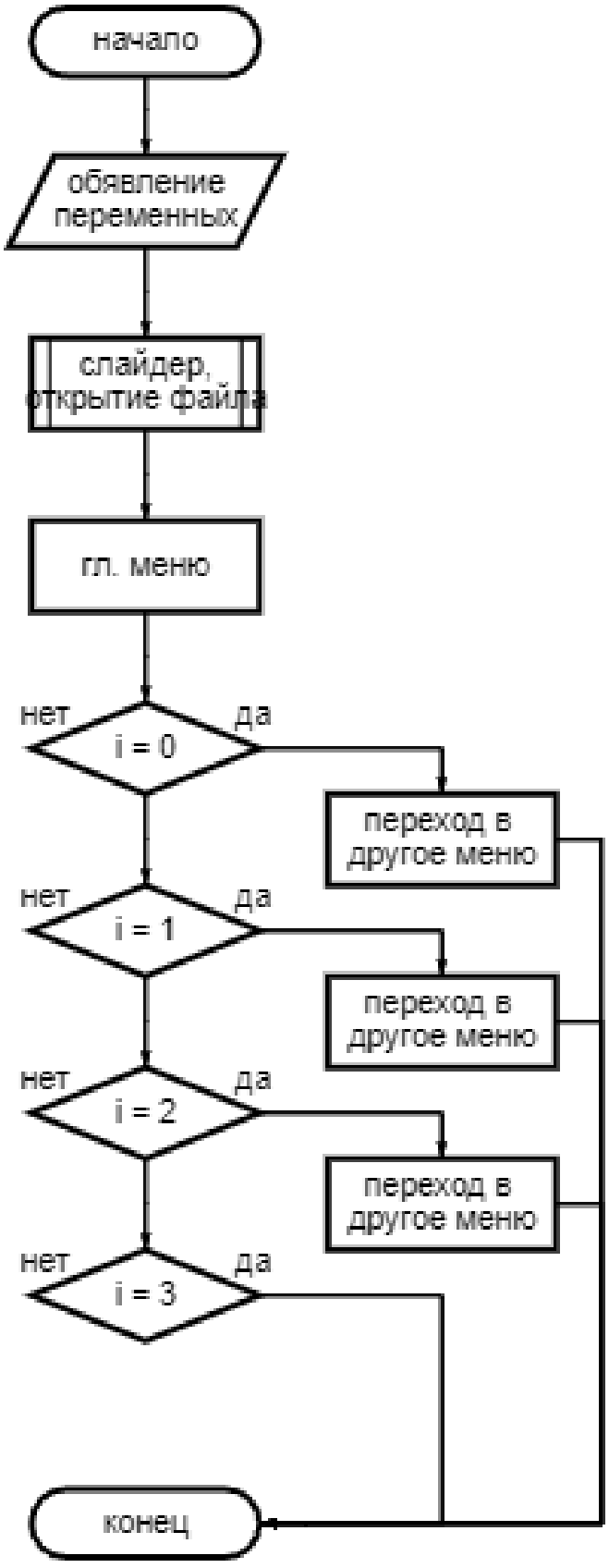

Рисунок 4 - Блок-схема главного меню

4.1 Блок-схема программы

На рисунке 5 представлен алгоритм работы меню с пользовательским значением.

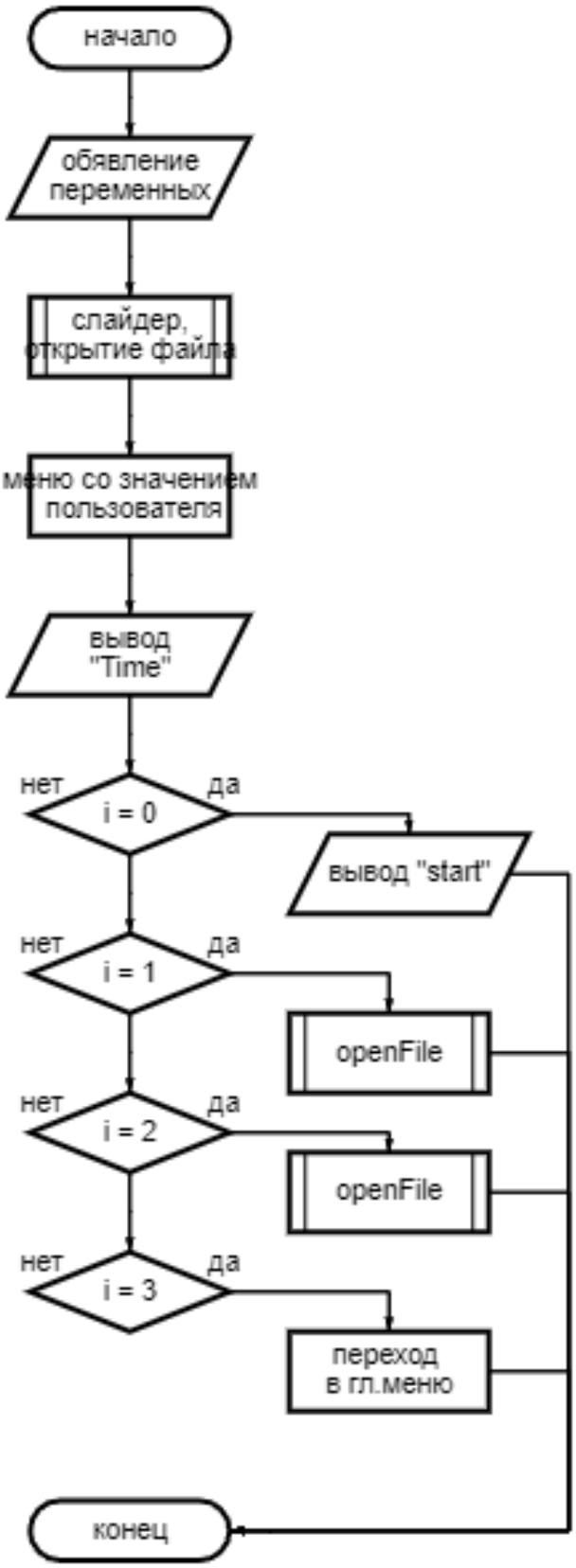

Рисунок 5 - Блок-схема меню с пользовательским значением

4.2 Блок-схема программы

На рисунке 6 представлен алгоритм работы меню с готовыми значениями (1 файл).

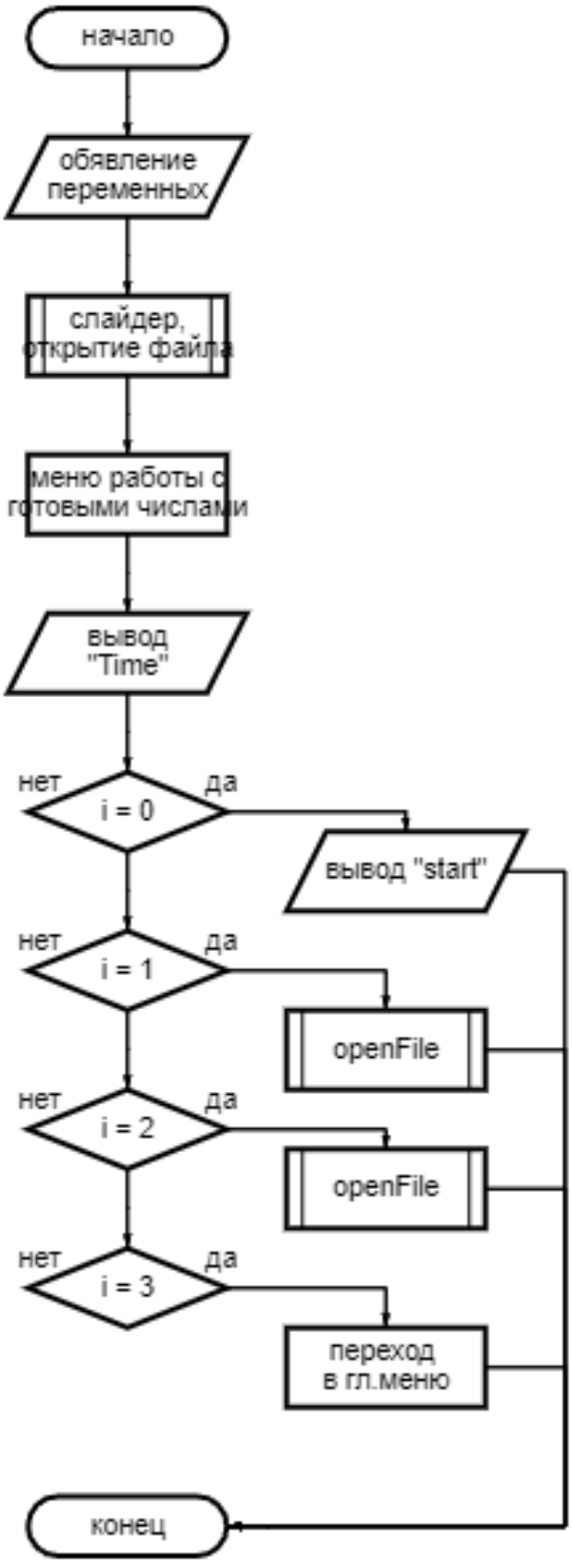

Рисунок 6 - Блок-схема меню с готовыми значениями (1 файл)

4.3 Блок-схема программы

На рисунке 7 представлен алгоритм работы меню с готовыми значениями (2 файл).

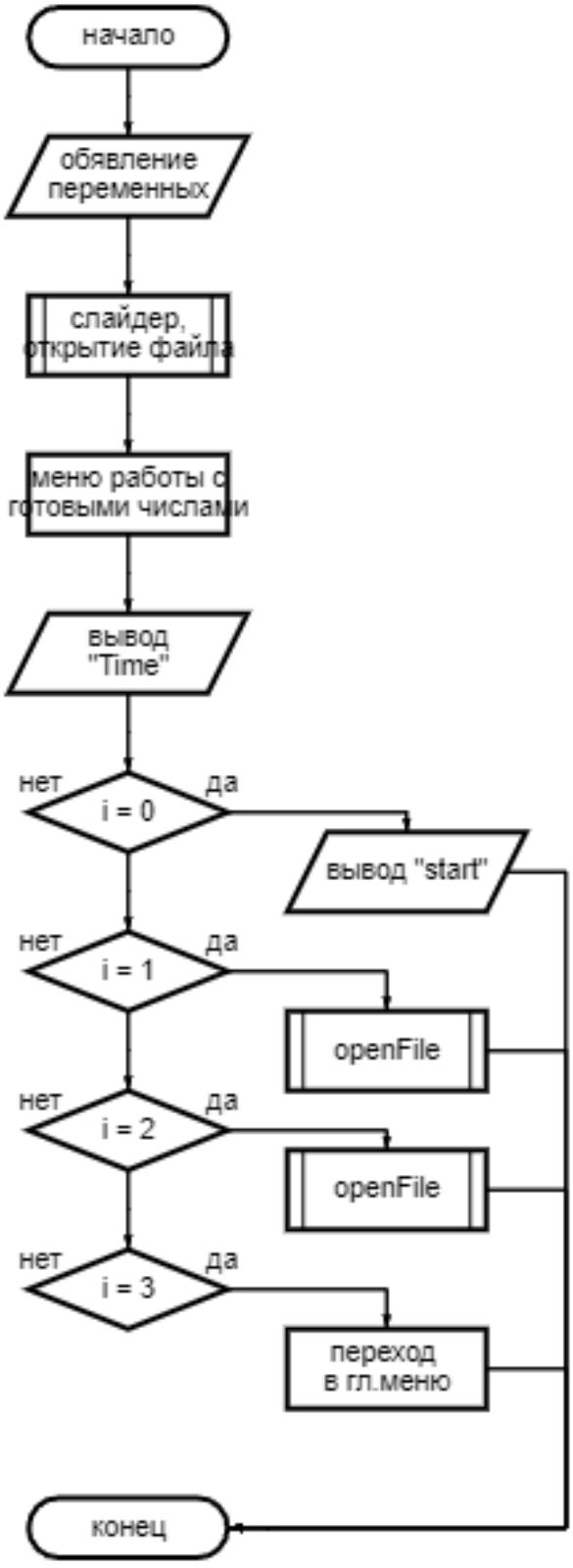

Рисунок 7 - Блок-схема меню с готовыми значениями (2 файл)

5 Тестирование программы

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

Рисунок 8 – Результаты тестирования пользовательских значений

Рисунок 9 – Результат выполнения программы для 20к элементов

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.

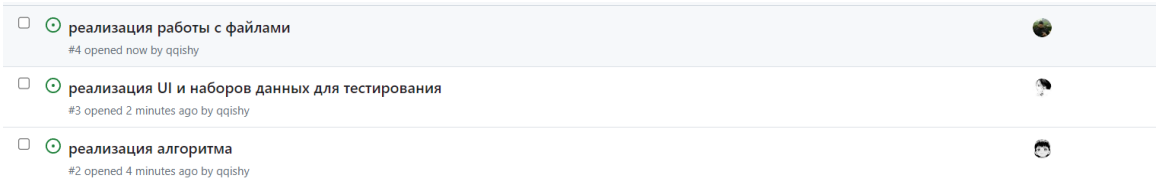

Рисунок 10 – распределение задач

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

Мною было написано меню, которое помогает ориентироваться в программе, это было зафиксировано и загруженона удаленный репозиторий Github, на ветку GUI, после было сделано слияние веток с main.

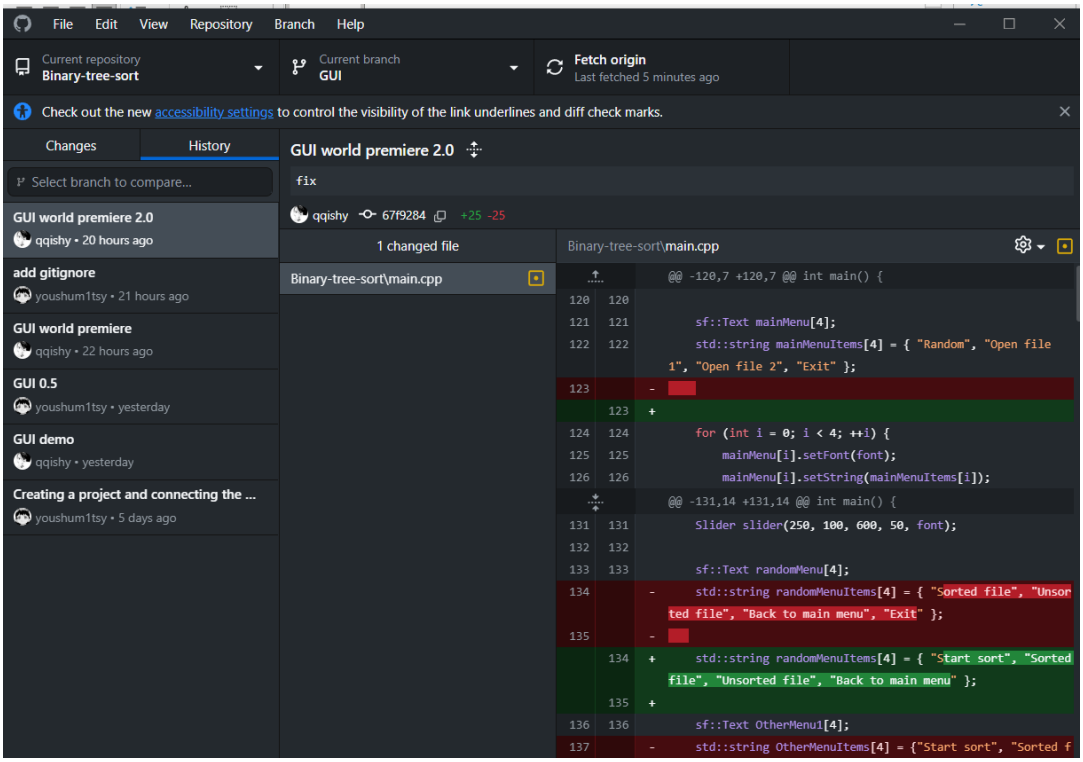

Рисунок 11 – изменения на ветке

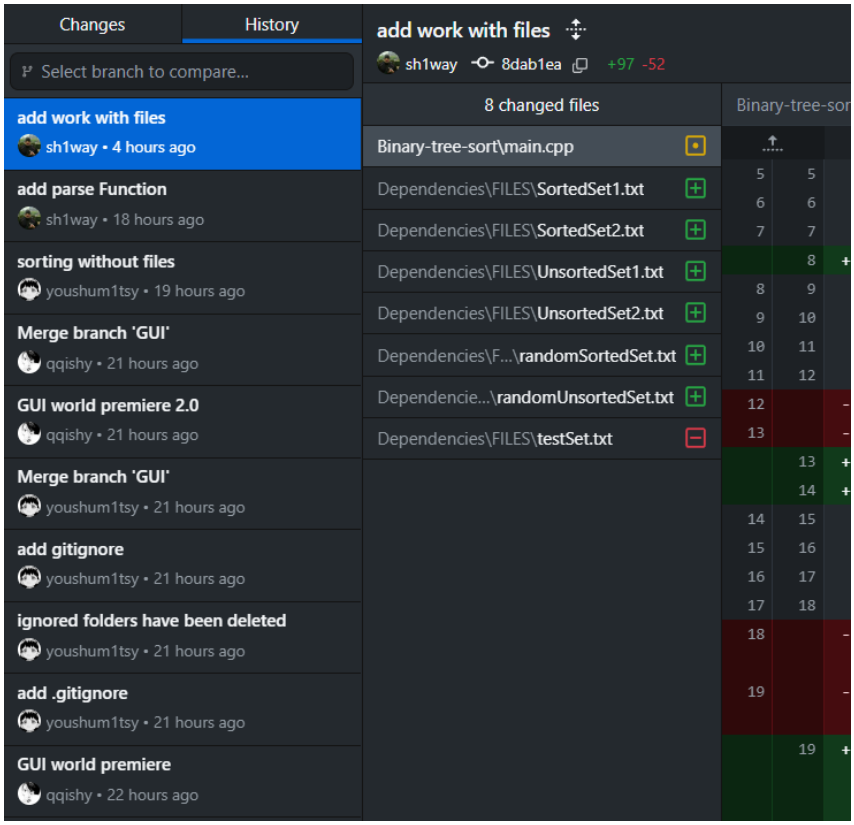

Рисунок 12 – слияние веток

Репозиторий находится на платформе GitHub в общественном доступе[4]

Заключение

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

Мною было создано меню программы, позволяющее выбрать одну из нескольких опций программы. Также я реализовала наборы данных для тестирования.

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. Артур Морейра, Ян Халлер, Хенрик Фогелиус Хансон. Разработка игр на языке SFML. — Издательство Packt, 2013 – 296р.

3. Бьерн Страуструп. Язык программирования C++. Специальное издание = The C++ programming language. Special edition. — М.: Бином-Пресс, 2007. — 1104 с.

4. 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;
```