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**

(2022/2023 учебный год)

                                          Кузьмин Данила Владиславович

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 обучения1семестр 2

Период прохождения практики с 29.06.2023 по 12.07.2023

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**

(2022/2023 учебный год)

                                            Кузьмин Данила Владиславович

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 29.06.2023 по 12.07.2023

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 | 29.06.2023 -  29.06.2023 |  |
| 2 | Подбор и изучение материала по теме работы | 15 | 30.06.2023 –  02.07.23 |  |
| 3 | Разработка алгоритма | 43 | 02.07.23 –  06.07.23 |  |
| 4 | Описание алгоритма и программы | 18 | 6.07.23 –  08.07.23 |  |
| 5 | Тестирование | 5 | 08.07.23 –  08.07.23 |  |
| 6 | Получение и анализ результатов | 10 | 08.07.23 –  10.07.23 |  |
| 7 | Оформление отчёта | 15 | 10.07.23 –  12.07.2023 |  |
|  | **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**

(2022/2023 учебный год)

                                            Кузьмин Данила Владиславович

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 обучения1семестр 2

Период прохождения практики с 29.06.2023 по 12.07.2023

Кафедра «Вычислительная техника»

Кузьмин Д.В. выполнял практическое задание «Сортировка Шелла». На первоначальном этапе был изучен и проанализирован алгоритм сортировки Шелла, был выбран метод решения и язык программирования С++, на котором была написана программа сортировки массива методом Шелла. Протестировал и отладил программу. Оформил отчёт.

Бакалавр Кузьмин Д.В. \_\_\_\_\_\_\_\_\_\_\_\_ "\_\_\_" \_\_\_\_\_\_\_\_\_\_\_ 2023 г.

Руководитель Зинкин С.А. \_\_\_\_\_\_\_\_\_\_\_\_ "\_\_\_" \_\_\_\_\_\_\_\_\_\_\_ 2023 г.

практики

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

ФАКУЛЬТЕТ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

**ОТЗЫВ**

**О ПРОХОЖДЕНИИ УЧЕБНОЙ (ОЗНАКОМИТЕЛЬНОЙ) ПРАКТИКИ**

(2022/2023 учебный год)

                                            Кузьмин Данила Владиславович

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 обучения1семестр 2

Период прохождения практики с 29.06.2023 по 12.07.2023

Кафедра «Вычислительная техника»

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

За период выполнения практики были освоены основные понятия и технологии сортировки Шелла, реализован метод работы с файлами. Во время выполнения работы Кузьмин Д.В. 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 Кузьмин Д.В. заслуживает оценки «\_\_\_\_\_\_».

Руководитель практики д.т.н., профессор, Зинкин С.А. « » 2023 г.

Содержание

Введение.................................................................................................................. 2   
1 Постановка задачи............................................................................................... 3   
 1.1 Достоинства алгоритма сортировки вставками..................................... 3  
 1.2 Недостатки алгоритма сортировки вставками....................................... 3

1.3 Применение алгоритма сортировки Шелла............................................ 3  
2 Выбор решения.................................................................................................... 4  
3 Описание программы.......................................................................................... 5   
4. Схемы программы.............................................................................................. 6   
 4.1 Блок-схема программы............................................................................ 6   
5 Тестирование программы................................................................................... 7

5.1 Тестирование на разных наборах данных.............................................. 7

5.2 Анализ работы программы....................................................................... 7  
6 Отладка............................................................................................................... 9  
7 Совместная разработка .................................................................................... 10  
Заключение ........................................................................................................... 13   
Список используемой литературы ..................................................................... 14   
Приложение А. Результаты тестирования программы..................................... 15

Приложение Б. Листинг программы…………......……………………………. 17

**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 образуют отдельный класс алгоритмов, применяются практически во всех задачах обработки информации. При этом они настолько тесно связаны друг с другом, что образуют отдельный класс алгоритмов. Алгоритмы сортировки, как правило, применяются с целью осуществления последующего более быстрого поиска. Например, трудно пользоваться словарями, если бы слова в них не были бы упорядочены по алфавиту.

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 — алгоритм сортировки, являющийся усовершенствованным вариантом сортировки вставками. Идея метода Шелла состоит в сравнении элементов, стоящих не только рядом, но и на определённом расстоянии друг от друга. Иными словами — это сортировка вставками с предварительными «грубыми» проходами.

**1 Постановка задачи**

Поставленная задача: необходимо заполнить массив из n-ого количества элементов случайными числами, записать данные элементы в отдельный файл. После этого выполнить сортировку вставками 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.

* 1. **Достоинства алгоритма сортировки Шелла**

Как правило, для сортировки требуется меньше памяти и времени, чем при использовании алгоритмов устойчивой сортировки.

* 1. **Недостатки алгоритма сортировки Шелла**

При сортировке по одному полю данных, состоящих из нескольких полей, не сохраняется взаимное расположение равных элементов.

* 1. **Применение алгоритма сортировки Шелла**
* Замена сортировки вставками, когда выполнение заданной задачи занимает много времени.
* Для вызова служебных данных стека обычно используется сортировка Шелла.
* Для наборов данных среднего и большого размера.
* В совокупности с сортировкой вставками, чтобы уменьшить количество операций.

**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 С++. С++ является мощным языком, унаследовав от Си богатые возможности по работе с памятью. Поэтому нередко С++ находит свое применение в системном программировании, в частности, при создании операционных систем, драйверов, различных утилит, антивирусов и т.д. К слову сказать, ОС Windows большей частью написана на С++. Но только системным программированием применение данного языка не ограничивается. С++ можно использовать в программах любого уровня, где важны скорость работы и производительность. Нередко он применяется для создания графических приложений, различных прикладных программ. Также особенно часто его используют для создания игр с богатой насыщенной визуализацией. Кроме того, в последнее время набирает ход мобильное направление, где С++ тоже нашел свое применение. И даже в веб-разработке также можно использовать С++ для создания веб-приложений или каких-то вспомогательных сервисов, которые обслуживают веб-приложения. В общем С++ – язык широкого пользования, на котором можно создавать практически любые виды программ.

Microsoft Visual Studio — это программная среда по разработке приложений для ОС Windows, как консольных, так и с графическим интерфейсом.

**3 Описание программы**

В результате запуска исполняемого файла пользователю

предоставляется возможность ввести размер массива для сортировки. После

введения значения пользователем программа генерирует и записывает в файл

input.txt исходный массив.

Далее запускается непосредственно алгоритм сортировки Шелла и

отсортированный массив записывается в файл output.txt. Для расчета времени

необходимого на сортировку массива используется библотечная функция

clock() из ctime.h. Сразу до и после сортировки значения clock() сохраняется

в переменные, разность которых будет показывать необходимое нам

значение.

time\_t start = clock();

ShellSort(arr, n);

time\_t stop = clock();

double time = (stop - start) / 1000.0;

После завершения работы программа выводит значение time в консоль.

Данные в файлах записываются в формате – каждый элемент массива

на отдельной новой строчке.

Подробный алгоритм работы программы представлен в разделе 4 на рисунке 1.

Листинг программы приведен в приложении Б.

**4 Схемы программы**

**4.1 Блок-схема программы**

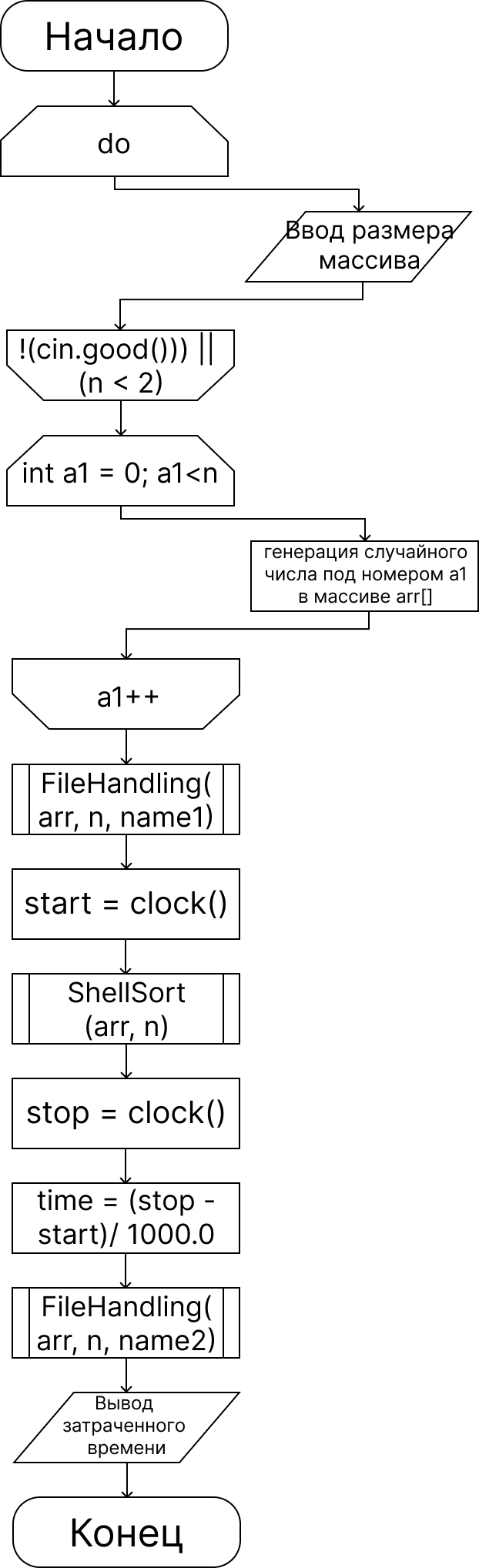


Рисунок 1 – Блок-схема программы

**5 Тестирование программы**

**5.1 Тестирование на разных наборах данных**

Тестовый набор данных представлен в таблице 1. Результаты тестирования приведены в Приложении А на рисунках А.1 – А.5.

Таблица 1 – Тестовый набор данных

|  |  |  |
| --- | --- | --- |
|  | Размер массива | Время выполнения сортировки в секундах |
| 1 | 10000 | 0,002 |
| 2 | 20000 | 0,005 |
| 3 | 30000 | 0,007 |
| 4 | 40000 | 0,01 |
| 5 | 50000 | 0,016 |
| 6 | 60000 | 0,018 |
| 7 | 70000 | 0,019 |
| 8 | 80000 | 0,022 |
| 9 | 90000 | 0,027 |
| 10 | 100000 | 0,03 |

**5.2 Анализ полученных результатов тестирования (анализ работы алгоритма)**

На основании анализа данных, полученных в результате тестирования алгоритма сортировки вставками, 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 2 – Результаты тестирования

**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.

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

**7 Совместная разработка**

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

Мною был загружен измененный код с удаленного репозитория. Удостоверившись в его правильной работе я приступил к тестирования с разными наборами данных.  
 По окончании тестирования примеры наборов данных в виде файлов ввода/вывода и справочный файл с информацией о времени работы алгоритма был загружен на GitHub. В коммите, мною была описана проделанная работа.

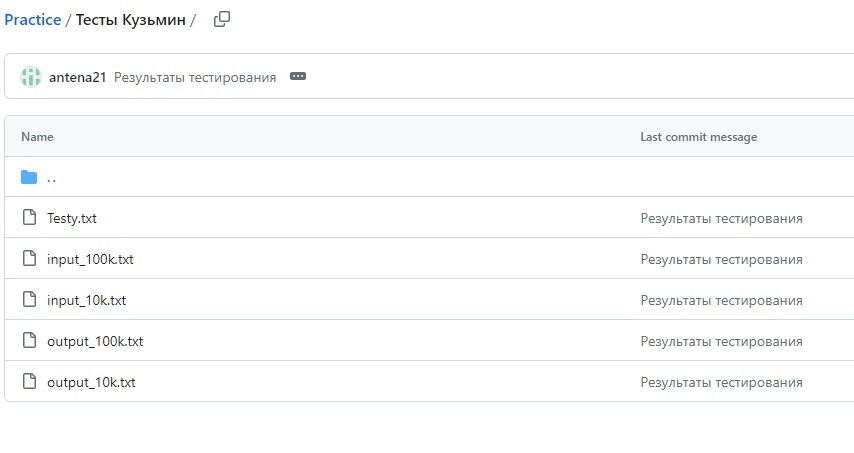


Рисунок 3 – Загруженные файлы

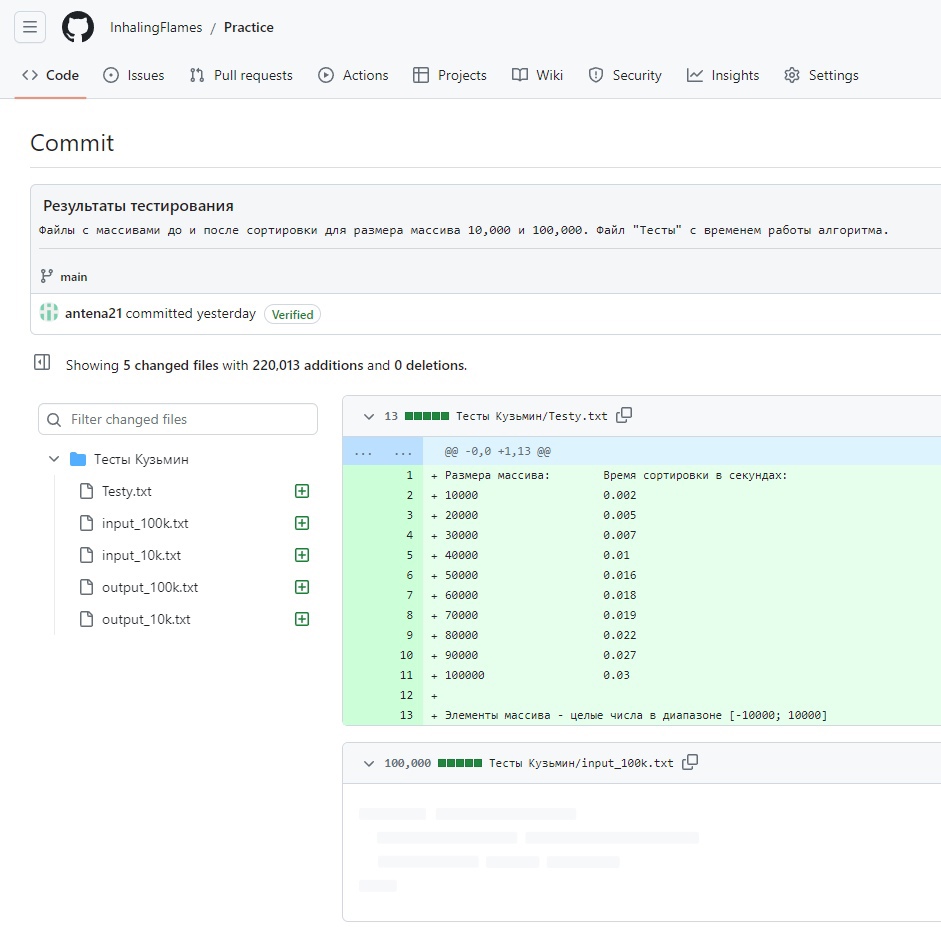


Рисунок 4 – Коммит с описанием проделанной работы

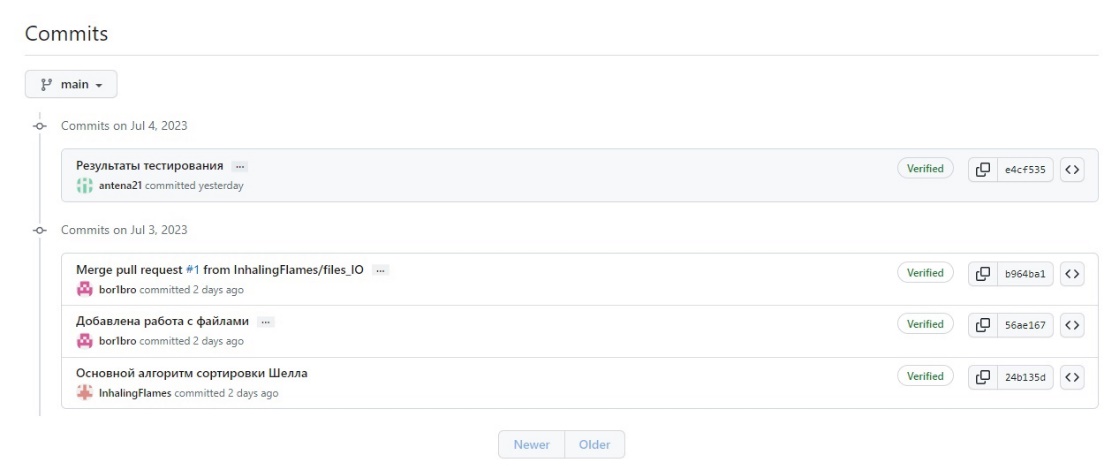


Рисунок 5 – Созданные коммиты

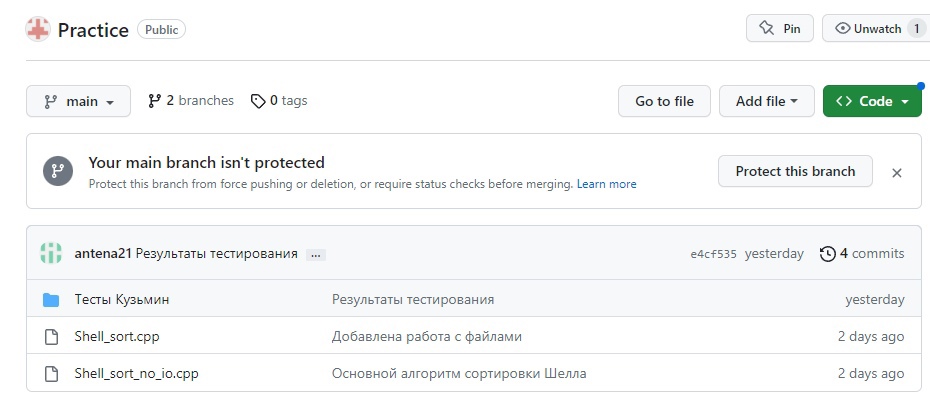


Рисунок 6 – Ветка main

Для загрузки данных на удаленный репозиторий использовали Web интерфейс Github.

Ссылка на удаленный репозиторий:

<http://github.com/InhalingFlames/Practice>

**Заключение**

При выполнении данной работы были получены навыки совместной работы с помощью сервиса GitHub, навыки использования программы Git Bash. Был изучен алгоритм сортировки Шелла.

Мною был написан алгоритм, считающий 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.

**Приложение А. Результаты тестирования программы**

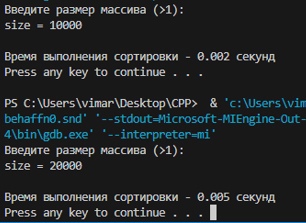


Рисунок А.1

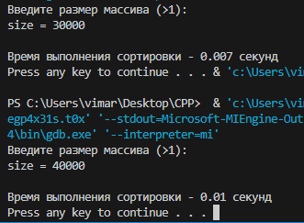


Рисунок А.2

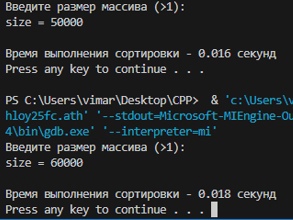


Рисунок А.3

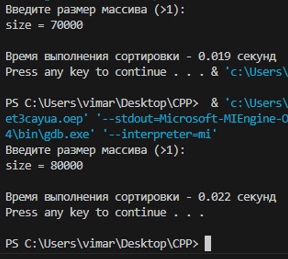


Рисунок А.4

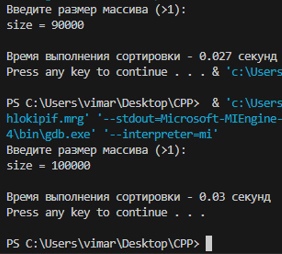


Рисунок А.5

**Приложение Б. Листинг программы**

#include <iostream> // test output to visually indicate the noraml flow of programm

#include <windows.h> // for pausing the console

#include <fstream> //работа с файлами

#include <ctime> // генератор случайных чисел

#include <iomanip>

using namespace std;

int ShellSort(int arr[], int n) {

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

for (int i = gap; i < n; ++i) { // circle thru elements from number [gap] to [n]

int temp = arr[i];

int j;

for (j = i; (j >= gap) && (arr[j - gap] > temp); j -=gap ) { // if position of current element >= gap AND element on the "left" of the gap is

arr[j] = arr[j - gap]; // Bigger than the current one THEN swap them IF NO --> choose element [j-gap]

}

arr[j] = temp; // actually moving stored content of the array's element

}

}

return 0;

}

void FileHandling(int array[], int n, string name) {

ofstream f(name.c\_str());

if (f.is\_open()) {

for (int i = 0; i < n; i++)

f << array[i] << endl;

f.close();

}

else cout << "Ошибка записи в файл " << name.c\_str() << endl;

}

int main() {

int i, n;

cout << "Введите размер массива (>1): " << endl;

do {

cout << "size = ";

cin >> n;

if ( (!(cin.good())) || (n < 2) ) {

cout << "Неверно введен размер массива size " << endl;

cin.clear();

cin.get();

}

} while ((!(cin.good())) || (n < 2));

int \*arr = new int[n];

string name1 = "input.txt";

string name2 = "output.txt";

// создание случайного массива

for( int a1=0; a1 < n; a1++) {

arr[a1] = (rand() % 2001) - 1000;

}

FileHandling(arr, n, name1); // запись в файл

// вычисление времени работы

time\_t start = clock();

ShellSort(arr, n);

time\_t stop = clock();

double time = (stop - start) / 1000.0;

FileHandling(arr, n, name2);

cout << endl << "Время выполнения сортировки - " << time << " секунд " << endl;

system("pause");

return 0;

}