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)

Куриленко Фёдор Дмитриевич

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 08.07.2024

Кафедра «Вычислительная техника»

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

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

Руководитель практики к.т.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)

Куриленко Фёдор Дмитриевич

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 08.07.2024

Кафедра «Вычислительная техника»

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

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

Руководитель практики к.т.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 –  28.06.24 |  |
| 3 | Разработка  алгоритма | 43 | 28.06.24 –  02.07.24 |  |
| 4 | Описание алгоритма и  программы | 18 | 02.07.24 – 04.07.24 |  |
| 5 | Тестирование | 5 | 04.07.24 –  04.07.24 |  |
| 6 | Получение и  анализ результатов | 10 | 04.07.24 –  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)

Куриленко Фёдор Дмитриевич

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 08.07.2024

Кафедра «Вычислительная техника»

Куриленко Ф. Д. выполнял практическое задание «Сортировка выбором». На первоначальном этапе был изучен и проанализирован алгоритм сортировки выбором, был выбран метод решения и язык программирования С++, на котором написана программа сортировки массива выбором. Написал, отладил и протестировал программу. Оформил отчет.

Бакалавр Куриленко Ф.Д. " " 2024 г.

Руководитель Карамышева Н.С. " " 2024 г. практики

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)

                                            Куриленко Фёдор Дмитриевич

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 25.06.2024 по 08.07.2024

Кафедра «Вычислительная техника»

В процессе выполнения практики Куриленко Ф. Д. решал следующие задачи: разработка блок-схем, разработка интерфейса, разработка алгоритма сортировки, разработка api работы с файлами, тестирование алгоритма.

За период выполнения практики было освоено несколько алгоритмов сортировки, различные варианты реализации пользовательских интерфейсов, методы ручного тестирования. Во время выполнения работы Куриленко Ф. Д. показал себя ответственным, добросовестным учеником, знающим свой предмет, имеющим представление о современном состоянии науки, владеющим современными общенаучными знаниями по информатике и вычислительной технике.

За выполнение работы Куриленко Ф. Д. заслуживает оценки «\_\_\_\_\_\_».

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

**Содержание**

[**Введение** 2](#_Toc170772209)

[**1.** **Разработка и описание алгоритма** 4](#_Toc170772210)

[**2.** **Описание программы** 4](#_Toc170772211)

[**3.** **Схема программы** 8](#_Toc170772212)

[**3.1.** **Блок-схема программы** 9](#_Toc170772213)

[**3.2.** **Блок-схема алгоритма сортировки** 9](#_Toc170772214)

[**3.3.** **Блок-схема функции создания файла** 10](#_Toc170772215)

[**3.4.** **Блок-схема функции меню** 10](#_Toc170772216)

[**4.** **Тестирование программы** 12](#_Toc170772217)

[**4.1.** **Тестирование на разных наборах данных** 12](#_Toc170772218)

[**4.2.** **Анализ полученных результатов тестирования** 12](#_Toc170772219)

[**5.** **Отладка** 13](#_Toc170772220)

[**6.** **Совместная разработка** 14](#_Toc170772221)

[**Заключение** 16](#_Toc170772222)

[**Список используемой литературы** 17](#_Toc170772223)

[**Приложение А. Результаты тестирование программы** 18](#_Toc170772224)

[**Приложение Б. Листинг программы** 22](#_Toc170772225)

**Введение**

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

Функциональная структура среды включает в себя:

1) редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинг кода;

2) отладчик кода;

3) редактор форм, предназначенный для упрощённого конструирования графических интерфейсов;

4) веб-редактор;

5) дизайнер классов;

6) дизайнер схем баз данных.

Visual Studio также позволяет создавать и подключать сторонние дополнения (плагины) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем контроля версий исходного кода, добавление новых наборов инструментов (для редактирования и визуального проектирования кода на предметно-ориентированных языках программирования или инструментов для прочих аспектов процесса разработки программного обеспечения).

Интегрированная среда разработки (IntegratedDevelopmentEnvironment – IDE) Visual Studio предлагает ряд высокоуровневых функциональных возможностей, которые выходят за рамки базового управления кодом.

Ниже перечислены основные преимущества IDE-среды Visual Studio:

1. Встроенный Web-сервер;
2. Поддержка множества языков при разработке;
3. Интуитивный стиль кодирования;
4. Более высокая скорость разработки;
5. Возможности отладки.

Язык программирования С++ представляет высокоуровневый компилируемый язык программирования общего назначения со статической типизацией, который подходит для создания самых различных приложений. На сегодняшний день С++ является одним из самых популярных и распространенных языков.

Своими корнями он уходит в язык Си, который был разработан в 1969—1973 годах в компании Bell Labs программистом Деннисом Ритчи. В начале 1980-х годов датский программист Бьерн Страуструп, который в то время работал в компании Bell Labs, разработал С++ как расширение к языку Си. Фактически вначале C++ просто дополнял язык Си некоторыми возможностями объектно-ориентированного программирования. И поэтому сам Страуструп вначале называл его как "C with classes" ("Си с классами").

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.

1. **Разработка и описание алгоритма**

Сортировка выбором – один из самых известных методов сортировки массива. Алгоритм был разработан в 1950 году и отличается простотой и легкостью в реализации.

Общая идея алгоритма состоит в следующем:

* Находим номер минимального значения в текущем списке.
* Производим обмен этого значения со значением первой неотсортированной позиции (обмен не нужен, если минимальный элемент уже находится на данной позиции).
* Теперь сортируем хвост списка, исключив из рассмотрения уже отсортированные элементы.

**Достоинства алгоритма сортировки выбором:**

* Прост в реализации.
* Эффективен для небольших массивов.
* Не требует дополнительной памяти для сортировки.

**Недостатки алгоритма:**

* Неэффективен для больших списков из-за квадратичной скорости.
* Неустойчив.
* При худшем, среднем и лучшем случае время остается постоянным.

1. **Описание программы**

В программе для сортировки выбором подключены следующие заголовочные файлы: *iostream* – заголовочный файл с классами, функциями и переменными для организации ввода-вывода; ***fstream* – заголовочный файл,** предоставляющий функционал для считывания данных из файла и для записи в файл; *Using namespace std* – использование стандартного пространства имен.

#include <iostream>

#include <fstream>

using namespace std;

Далее идет текст функции choice\_sort. В первом цикле выбирается крайний левый элемент неотсортированного массива. Далее во вложенном цикле мы сравниваем элементы массива с крайним левым элементом неотсортированного массива. Если сравниваемый элемент массива меньше чем минимальный найденный элемент массива, то найденный элемент становится минимальным, а индекс запоминается. После вложенного цикла мы сравниваем минимальный индекс с индексом крайнего правого элемента. Если они не равны, то меняем местами элемент массива с минимальным индексом с крайним правым элементом неотсортированного массива.

void choice\_sort(int n, int mass[]) {

int buf = 0;

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

int min\_i = i;

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

if (mass[j] < mass[min\_i]) { min\_i = j; }

}

if (min\_i != i) {

buf = mass[i];

mass[i] = mass[min\_i];

mass[min\_i] = buf;

}

}

}

Далее располагается текст функции создания и заполнение неотсортированного массива. Вначале создается файл, далее запускается цикл, который записывает случайные числа в файл. Далее файл закрывается.

void create\_file(int n) {

ofstream fout("no\_sort\_array.txt");

srand(time(0));

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

fout << rand() % 20001 - 10000 << endl;

}

fout.close();

}

Далее идет текст программы меню. Вначале выводим пункты меню. Потом запускаем бесконечный цикл с предусловием и запрашиваем в нем число, которое является пунктом меню.

void menu() {

int n, count;

string name, t = ".txt";

cout << "\n\t1. Случайный ввод\n\t2. Ручной ввод\n\t3. Ввод с файла\n\t4. Выход\n\n";

while (true){

cout << "\tВыберите пункт: ";

cin >> n;

cout << endl;

Далее используем конструкцию switch case. Выбор происходит от номера пункта. В первом случае мы сначала запрашиваем количество элементов массива и создаем массив. Далее вызываем функцию создания файла create\_file. Затем с помощью цикла считываем числа из файла в массив. Потом запускается функция сортировки массива с замером времени сортировки. Далее выводим сообщение о времени сортировки массива, а после с помощью цикла массив записывается в новый файл.

switch (n) {

case 1: {

int n, buf = 0;

cout << "\tВведите количество элементов: ";

cin >> n;

int\* mass = new int[n];

create\_file(n);

ifstream fin("no\_sort\_array.txt");

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

fin >> mass[i];

}

fin.close();

time\_t start = clock();

choice\_sort(n, mass);

time\_t stop = clock();

double time = (stop - start) / 1000.0;

cout << "\n\tВремя сортировки массива состовляет: " << time << " секунд\n";

ofstream fout("sort\_array.txt");

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

fout << mass[i] << endl;

}

fout.close();

exit(0);

}

Во втором случае мы сначала запрашиваем количество элементов массива и создаем массив. Далее поэлементно вводим массив с клавиатуры. Затем запускается функция сортировки массива с замером времени сортировки. Далее выводим сообщение о времени сортировки массива, а после с помощью цикла выводим отсортированный массив на экран.

case 2: {

int n;

cout << "\tВведите количество элементов: ";

cin >> n;

cout << endl;

int\* mass = new int[n];

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

cout << "\tВведите " << i + 1 << " элемент массива: ";

cin >> mass[i];

cout << endl;

}

time\_t start = clock();

choice\_sort(n, mass);

time\_t stop = clock();

double time = stop - start;

cout << "\tВремя сортировки массива состовляет: " << time << " милисекунд\n\n" << "Сортированный массив: ";

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

cout << mass[i] << " ";

}

exit(0);

}

В третьем случае мы сначала запрашиваем имя заранее подготовленного файла. В файле сначала идет количество элементов, а далее уже сами числа. Далее считываем первый элемент, который является количеством элементов. Затем создаем массив из считанного количества элементов. После с помощью цикла считываем числа из файла в массив. Потом запускается функция сортировки массива с замером времени сортировки. Далее выводим сообщение о времени сортировки массива, а после с помощью цикла массив записывается в новый файл.

case 3: {

cout << "\tВведите имя файла: ";

cin >> name;

name += t;

ifstream fin(name);

fin >> count;

int\* mass = new int[count];

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

fin >> mass[i];

}

fin.close();

time\_t start = clock();

choice\_sort(count, mass);

time\_t stop = clock();

double time = (stop - start) / 1000.0;

cout << "\n\tВремя сортировки массива состовляет: " << time << " секунд";

ofstream fout("sort\_array.txt");

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

fout << mass[i] << endl;

}

fout.close();

exit(0);

}

В четвёртом случае происходит завершение программы, а в остальных случаях выводится сообщение об ошибке и цикл начинается заново.

case 4: { exit(0); }

default: { cout << "Такого пункта нет, попробуйте ещё раз" << endl;}

}

Затем идет текст главной программы. Сначала выполняется русификация, а после запускается функция меню.

int main(){

setlocale(0, "");

menu();

return 0;

}

1. **Схема программы**
   1. **Блок-схема программы**

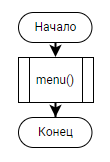


Рисунок 1 – Блок-схема программы

* 1. **Блок-схема алгоритма сортировки**

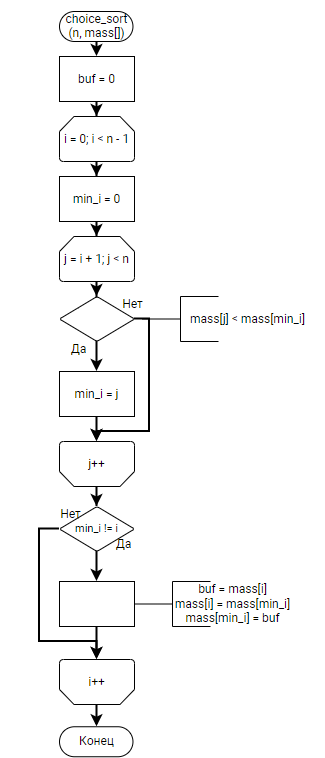
****

Рисунок 2 – блок-схема алгоритма сортировки

* 1. **Блок-схема функции создания файла**

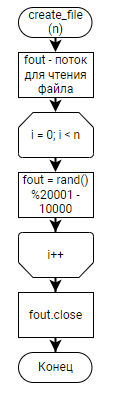
****

Рисунок 3 – блок-схема функции создания файла

* 1. **Блок-схема функции меню**

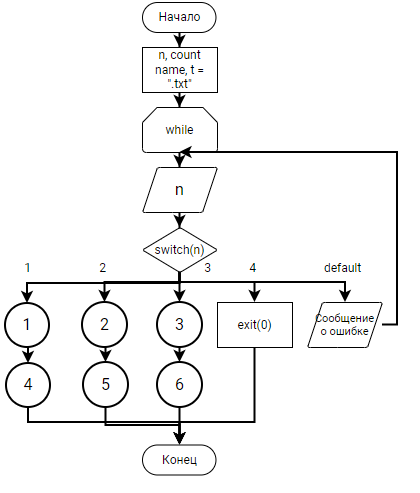


Рисунок 4 – часть блок-схемы функции меню

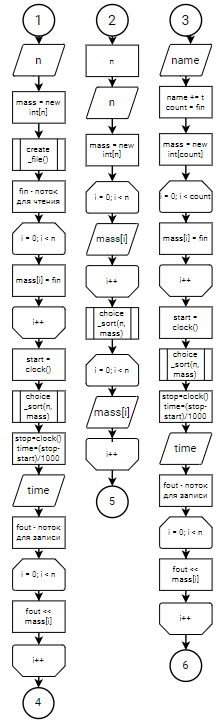


Рисунок 5 – оставшаяся часть блок-схемы функции меню

1. **Тестирование программы**
   1. **Тестирование на разных наборах данных**

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

Таблица 1.

|  |  |  |
| --- | --- | --- |
|  | Размер массива size | Время выполнения сортировки в секундах |
| 1 | 10000 | 0.034 |
| 2 | 20000 | 0.145 |
| 3 | 30000 | 0.31 |
| 4 | 40000 | 0.552 |
| 5 | 50000 | 0.864 |
| 6 | 60000 | 1.287 |
| 7 | 70000 | 1.736 |
| 8 | 80000 | 2.245 |
| 9 | 90000 | 2.827 |
| 10 | 100000 | 3.451 |
| 11 | 110000 | 4.243 |

* 1. **Анализ полученных результатов тестирования**

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 6 – Результаты тестирования

1. **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.

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.

1. **Совместная разработка**

Для удобства был создан репозиторий. Ссылка на него: <https://github.com/FuneTome/praktika.git>

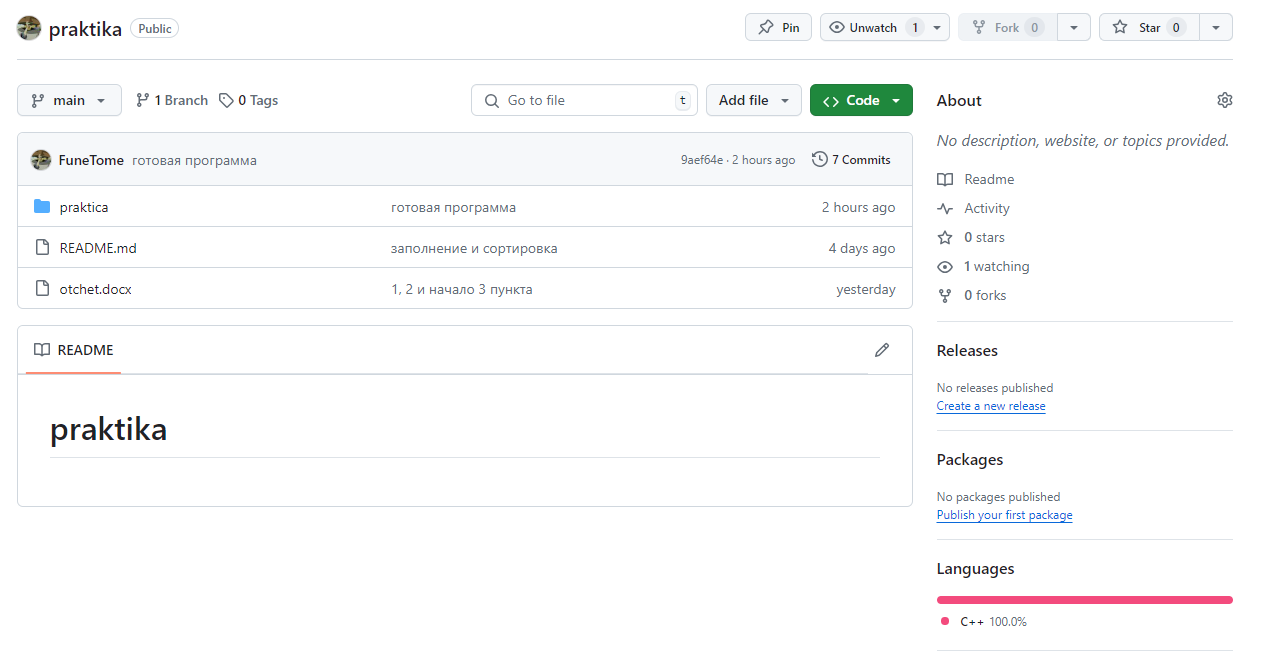


Рисунок 7 – репозиторий и ветка main

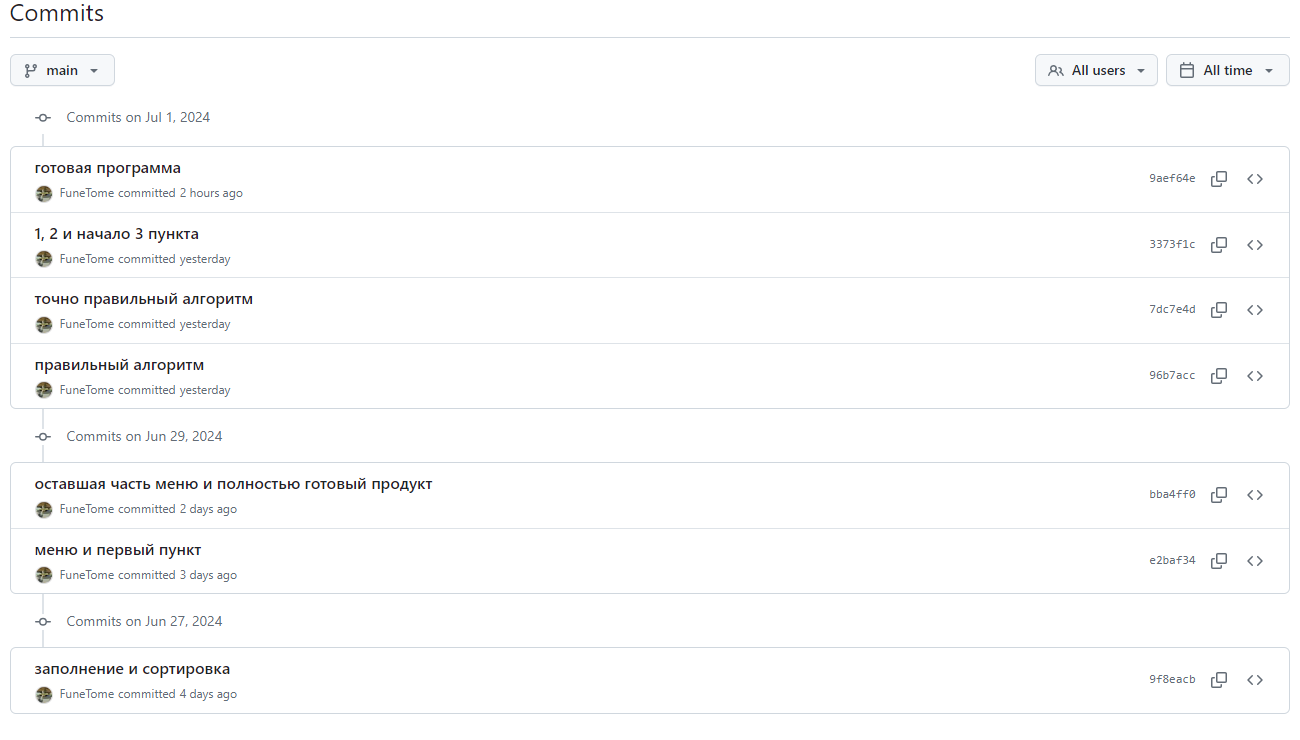


Рисунок 8 – созданные коммиты

Для загрузки на локальный и глобальный репозиторий были использованы основные команды git bash.



Рисунок 9 – команды add, commit и push

# **Заключение**

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

Мною был написан алгоритм сортировки выбором, алгоритм создания и заполнения файла, а так же было написано меню.

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

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

# **Список используемой литературы**

1. ГОСТ 19.701 – 90 Схемы алгоритмов, программ, данных и систем.
2. Керниган, Брайан У., Ритчи, Деннис М. Язык программирования С, 2-е издание.: Пер. с англ. – М.,2009.
3. Сортировка выбором [электронный ресурс] – URL: <https://ru.wikipedia.org/wiki/Сортировка_выбором>

**Приложение А. Результаты тестирование программы**

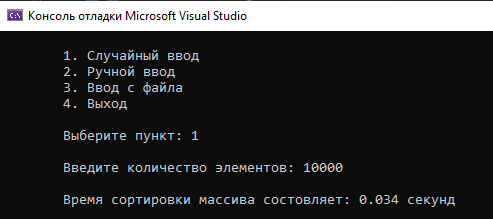
****

Рисунок А.1

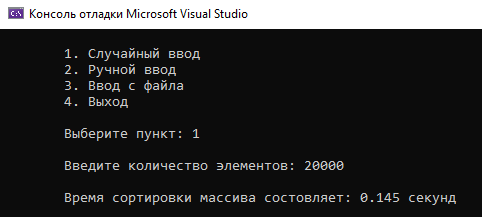
****

Рисунок А.2

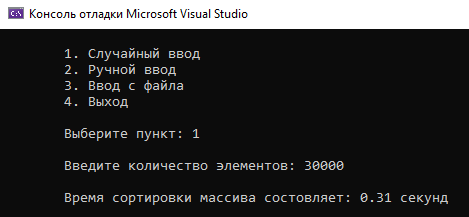
****

Рисунок А.3

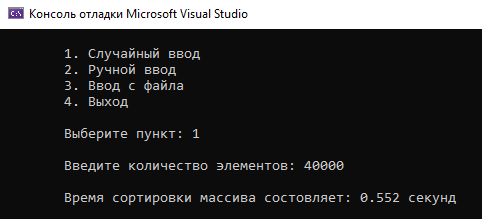
****

Рисунок А.4

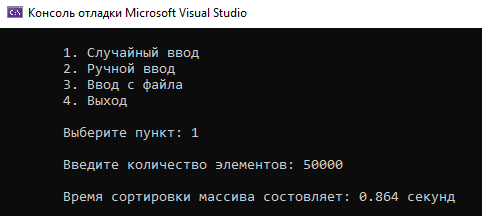
****

Рисунок А.5

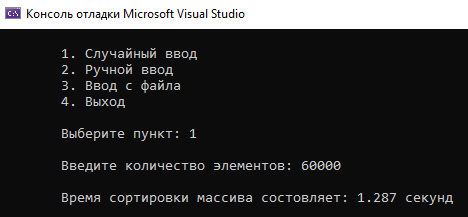
****

Рисунок А.6

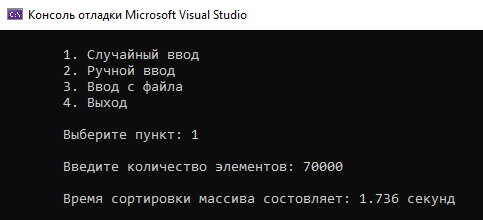
****

Рисунок А.7

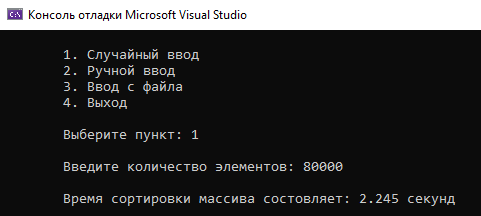
****

Рисунок А.8

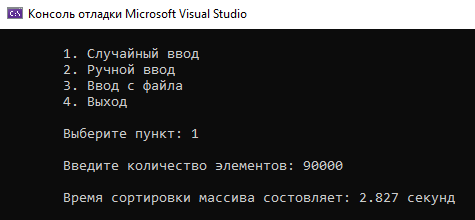
****

Рисунок А.9

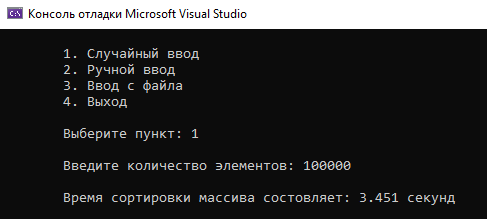
****

Рисунок А.10

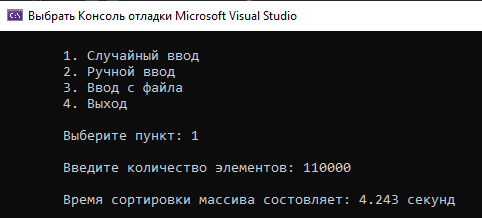
****

Рисунок А.11

**Приложение Б. Листинг программы**

#include <iostream>

#include <fstream>

using namespace std;

void choice\_sort(int n, int mass[]) {

int buf = 0;

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

int min\_i = i;

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

if (mass[j] < mass[min\_i]) { min\_i = j; }

}

if (min\_i != i) {

buf = mass[i];

mass[i] = mass[min\_i];

mass[min\_i] = buf;

}

}

}

void create\_file(int n) {

ofstream fout("no\_sort\_array.txt");

srand(time(0));

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

fout << rand() % 20001 - 10000 << endl;

}

fout.close();

}

void menu() {

int n, count;

string name, t = ".txt";

cout << "\n\t1. Случайный ввод\n\t2. Ручной ввод\n\t3. Ввод с файла\n\t4. Выход\n\n";

while (true){

cout << "\tВыберите пункт: ";

cin >> n;

cout << endl;

switch (n) {

case 1: {

int n, buf = 0;

cout << "\tВведите количество элементов: ";

cin >> n;

int\* mass = new int[n];

create\_file(n);

ifstream fin("no\_sort\_array.txt");

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

fin >> mass[i];

}

fin.close();

time\_t start = clock();

choice\_sort(n, mass);

time\_t stop = clock();

double time = (stop - start) / 1000.0;

cout << "\n\tВремя сортировки массива состовляет: " << time << " секунд\n";

ofstream fout("sort\_array.txt");

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

fout << mass[i] << endl;

}

fout.close();

exit(0);

}

case 2: {

int n;

cout << "\tВведите количество элементов: ";

cin >> n;

cout << endl;

int\* mass = new int[n];

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

cout << "\tВведите " << i + 1 << " элемент массива: ";

cin >> mass[i];

cout << endl;

}

choice\_sort(n, mass);

for (int i = 0; i < n; i++) {cout << mass[i] << " ";}

exit(0);

}

case 3: {

cout << "\tВведите имя файла: ";

cin >> name;

name += t;

ifstream fin(name);

fin >> count;

int\* mass = new int[count];

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

fin >> mass[i];

}

fin.close();

time\_t start = clock();

choice\_sort(count, mass);

time\_t stop = clock();

double time = (stop - start) / 1000.0;

cout << "\n\tВремя сортировки массива состовляет: " << time << " секунд";

ofstream fout("sort\_array.txt");

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

fout << mass[i] << endl;

}

fout.close();

exit(0);

}

case 4: { exit(0); }

default: { cout << "Такого пункта нет, попробуйте ещё раз" << endl;}

}

}

}

int main(){

setlocale(0, "");

menu();

return 0;}