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)

                                               Серякова Анастасия Сергеевна

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 28.06.2022 по 11.07.2022

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)

                                            Серякова Анастасия Сергеевна

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 –4года

Год обучения                  1                 семестр                 2

Период прохождения практики с 28.06.2022 по 11.07.2022

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 | 25.06.24 –  26.06.24 |  |
| 3 | Разработка алгоритма | 43 | 27.06.24 –  29.06.24 |  |
| 4 | Описание алгоритма и программы | 18 | 29.06.24 –  30.06.24 |  |
| 5 | Тестирование | 5 | 01.07.24 –  01.07.24 |  |
| 6 | Получение и анализ результатов | 10 | 01.07.24 –  02.07.24 |  |
| 7 | Оформление отчёта | 15 | 02.07.24 –  02.07.24 |  |
|  | **Общий объём часов** | 108 |  |  |

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

ФАКУЛЬТЕТ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

**ОТЧЁТ**

**ОПРОХОЖДЕНИИУЧЕБНОЙ (ОЗНАКОМИТЕЛЬНОЙ) ПРАКТИКИ**

(2023/2024 учебный год)

                                           Серякова Анастасия Сергеевна

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 28.06.2022 по 11.07.2022

Кафедра «Вычислительная техника»

Серякова А.С. выполняла практическое задание «Сортировка слияниями». На первоначальном этапе были изучен и проанализирован материал сортировки слияниями, также, осуществила работу с тестированием программы и разработала графический интерфейс, отвечающий за меню программы. Оформил отчёт.

Бакалавр Серякова А.С. \_\_\_\_\_\_\_\_\_\_\_\_ "\_\_\_" \_\_\_\_\_\_\_\_\_\_\_ 2022 г.

Руководитель Карамышева Н.С. \_\_\_\_\_\_\_\_\_\_\_\_ "\_\_\_" \_\_\_\_\_\_\_\_\_\_\_ 2022 г.

практики

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

ФАКУЛЬТЕТ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

**ОТЗЫВ**

**ОПРОХОЖДЕНИИ УЧЕБНОЙ (ОЗНАКОМИТЕЛЬНОЙ) ПРАКТИКИ**

(2023/2024 учебный год)

                                           Серякова Анастасия Сергеевна

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 28.06.2022 по 11.07.2022

Кафедра «Вычислительная техника»

За выполнение работы Серякова А.С. заслуживает оценки « ».

Руководитель практики к.т.н., доцент, Карамышева Н.С. «» 2022 г.

Оглавление

[Введение 7](#_Toc170808782)

[1 Описание алгоритма 9](#_Toc170808783)

[1.1 Достоинства алгоритма быстрой сортировки: 10](#_Toc170808784)

[1.2 Недостатки алгоритма: 10](#_Toc170808785)

[2 Описание программы 11](#_Toc170808786)

[3 Тестирование на разных наборах данных 13](#_Toc170808787)

[4 Осуществление командной работы в сфере GitHub 14](#_Toc170808788)

[Заключение 15](#_Toc170808789)

[Список литературы 16](#_Toc170808790)

[КОД ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 17](#_Toc170808791)

[Приложение А 17](#_Toc170808792)

[Приложение B 23](#_Toc170808793)

# 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 [рефакторинга кода](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D1%84%D0%B0%D0%BA%D1%82%D0%BE%D1%80%D0%B8%D0%BD%D0%B3);

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) VisualStudio предлагает ряд высокоуровневых функциональных возможностей, которые выходят за рамки базового управления кодом.

Ниже перечислены основные преимущества IDE-среды Visual Studio:

* Встроенный Web-сервер;
* Поддержка множества языков при разработке;
* Интуитивный стиль кодирования;
* Более высокая скорость разработки;
* Возможности отладки.

Язык программирования С — компилируемый статически типизированный язык программирования общего назначения, разработанный в 1969—1973 годах сотрудником Bell Labs Деннисом Ритчи как развитие языка Би.

Первоначально был разработан для реализации операционной системы UNIX, но впоследствии был перенесён на множество других платформ. Согласно дизайну языка, его конструкции близко сопоставляются типичным машинным инструкциям, благодаря чему он нашёл применение в проектах, для которых был свойственен язык ассемблера, в том числе как в операционных системах, так и в различном прикладном программном обеспечении для множества устройств — от суперкомпьютеров до встраиваемых систем. Язык программирования Си оказал существенное влияние на развитие индустрии программного обеспечения, а его синтаксис стал основой для таких языков программирования, как C++, C#, Java и Objective-C.

Язык Си разрабатывался как язык системного программирования, для которого можно создать однопроходный компилятор. Стандартная библиотека также невелика. Как следствие данных факторов — компиляторы разрабатываются сравнительно легко. Поэтому данный язык доступен на самых различных платформах. К тому же, несмотря на свою низкоуровневую природу, язык ориентирован на переносимость. Программы, соответствующие стандарту языка, могут компилироваться под различные архитектуры компьютеров.

Целью языка было облегчение написания больших программ с минимизацией ошибок по сравнению с ассемблером, следуя принципам процедурного программирования, но избегая всего, что может привести к дополнительным накладным расходам, специфичным для языков высокого уровня.

# 1 Описание алгоритма

Сортировка слиянием (Mergesort) — алгоритм сортировки, который упорядочивает списки (или другие структуры данных, доступ к элементам которых можно получать только последовательно, например — потоки) в определённом порядке. Эта сортировка — хороший пример использования принципа «разделяй и властвуй». Сначала задача разбивается на несколько подзадач меньшего размера. Затем эти задачи решаются с помощью рекурсивного вызова или непосредственно, если их размер достаточно мал. Наконец, их решения комбинируются, и получается решение исходной задачи.

Эффективно заранее создать временный массив и передать его в качестве аргумента функции. Эта сортировка рекурсивна, а потому возможен переход на квадратичную при небольшом числе элементов.

Общая идея алгоритма состоит в следующем:

1. Сортируемый массив разбивается на две части примерно одинакового размера;
2. Каждая из получившихся частей сортируется отдельно, например — тем же самым алгоритмом;
   1. . Рекурсивное разбиение задачи на меньшие происходит до тех пор, пока размер массива не достигнет единицы (любой массив длины 1 можно считать упорядоченным);
3. Два упорядоченных массива половинного размера соединяются в один;
   1. . Слияние двух подмассивов в третий результирующий массив;
   2. . «Прицепление» остатка. (Когда один из подмассивов закончился, добавляются все оставшиеся элементы второго подмассива в результирующий массив.)

## Достоинства алгоритма быстрой сортировки:

1. Работает даже на структурах данных последовательного доступа.
2. Неплохо работает в параллельном варианте: легко разбить задачи между процессорами поровну, но трудно сделать так, чтобы другие процессоры взяли на себя работу, в случае если один процессор задержится.
3. Не имеет «трудных» входных данных.
4. Устойчивая - сохраняет порядок равных элементов (принадлежащих одному классу эквивалентности по сравнению).

## Недостатки алгоритма:

1. На «почти отсортированных» массивах работает столь же долго, как на хаотичных. Существует вариант сортировки слиянием, который работает быстрее на частично отсортированных данных, но он требует дополнительной памяти, в дополнении ко временному буферу, который используется непосредственно для сортировки.
2. Требует дополнительной памяти по размеру исходного массива.

# 2 Описание программы

В программе для сортировки слиянием подключены следующий заголовочный файлы: *stdio.h* – заголовочный файл с классами, функциями и переменными для организации ввода-вывода

#include<stdio.h>

Далее идет текст функции *menu* отвечающее за меню главной программы *main*.

void menu(void)

{

printf("Выберите действие:\n");

printf("1.Сортировка предварительных отсортированных чисел\n");

printf("2.Сортировка случайного набора чисел\n");

printf("3.Выход\n");

}

Программа main.

int main(void)

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

srand(time(NULL));

int s = 1;

while (s != 0)

{

menu();

s = \_getch();

switch (s)

{

case '1': system("cls"); sort(); break;

case '2': system("cls"); random(); break;

case '3': system("cls"); exit(0); break;

default: system("cls");

}

}

return 0;

}

На рисунке 1 представлено главное меню программы.

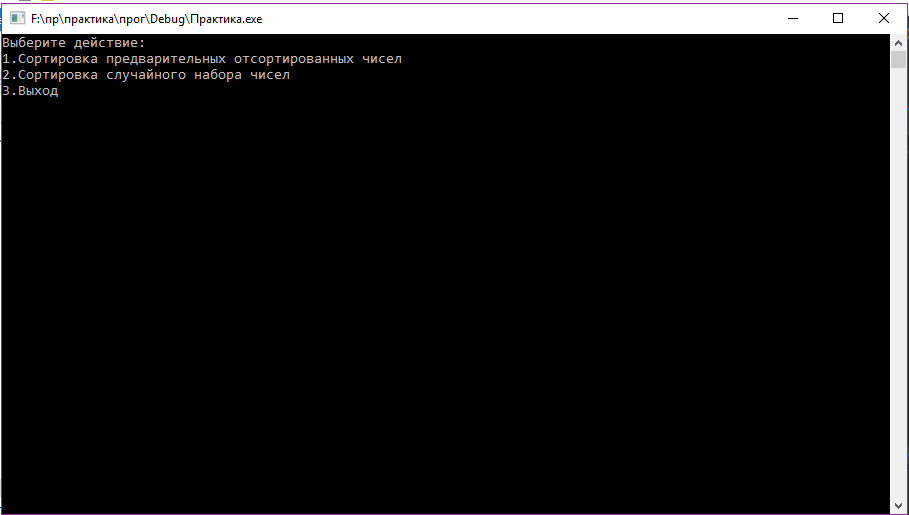


Рисунок 1 – Меню программы

# 3 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 – Тестовый набор данных

|  |  |  |
| --- | --- | --- |
| № теста | Размер массива size | Время выполнения сортировки в секундах |
| 1 | 10000 | 0.0013 |
| 2 | 20000 | 0.026 |
| 3 | 30000 | 0.038 |
| 4 | 40000 | 0.051 |
| 5 | 50000 | 0.063 |
| 6 | 60000 | 0.076 |
| 7 | 70000 | 0.089 |
| 8 | 80000 | 0.103 |
| 9 | 90000 | 0.115 |
| 10 | 100000 | 0.129 |

# 

# 4 Осуществление командной работы в сфере GitHub

GitHub — это многофункциональная платформа для командной работы над проектом, состоящая из:

• облачного хранилища, в котором размещаются все файлы;

• социальной сети для разработчиков;

• системы контроля версий, фиксирующей все редакции, вносимые в код тем или иным участником.

Простыми словами GitHub — площадка, где можно разместить общий проект, совместно управлять всеми изменениями, а в случае неудачи — быстро вернуться к исправной версии.

Действия, совершённые мной в GitHub:

1. Создал репозиторий и пригласил членов бригады

2. Создал проект Microsoft Visual Studio и загрузил его в репозиторий

3. Загрузил блок-схемы своего кода

4. Загрузил отчёт практики

# Заключение

На основании анализа данных, полученных в результате тестирования алгоритма сортировкой слиянием, 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. Данный анализ подтверждается просмотром графика, либо в «Приложении А» наглядно показано, как при увеличении массива, также увеличивается время работы.

Рисунок 1 – Результаты тестирования

# Список литературы

1. Красиков И.В. – Алгоритмы. Просто как дважды два. 2007 г.

2. Дейтел и Пол Харви. Как программировать на С.

3.<https://en.wikipedia.org/wiki/Merge_sort> - Википедия. Сортировка слиянием.

4. Язык программирования С. Лекции и упражнения | Прата Стивен

# КОД ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

# Приложение А

(обязательное)

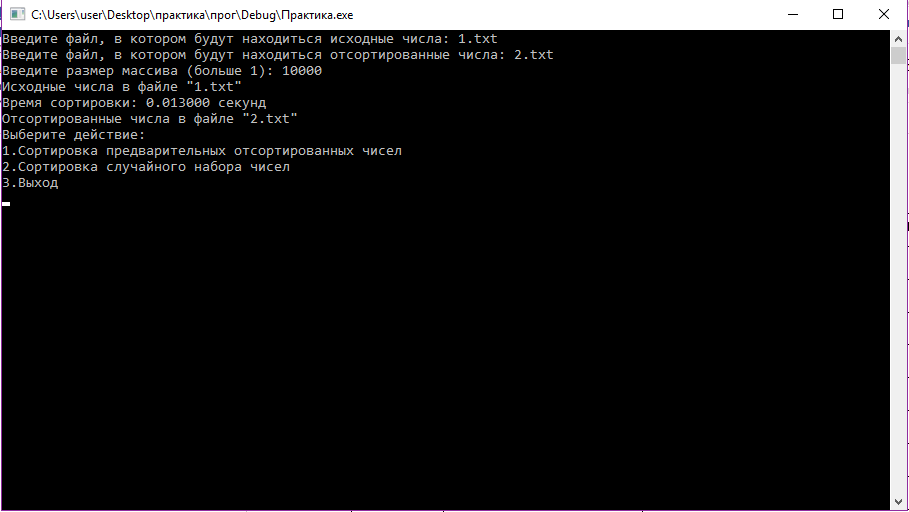


Рисунок А.1

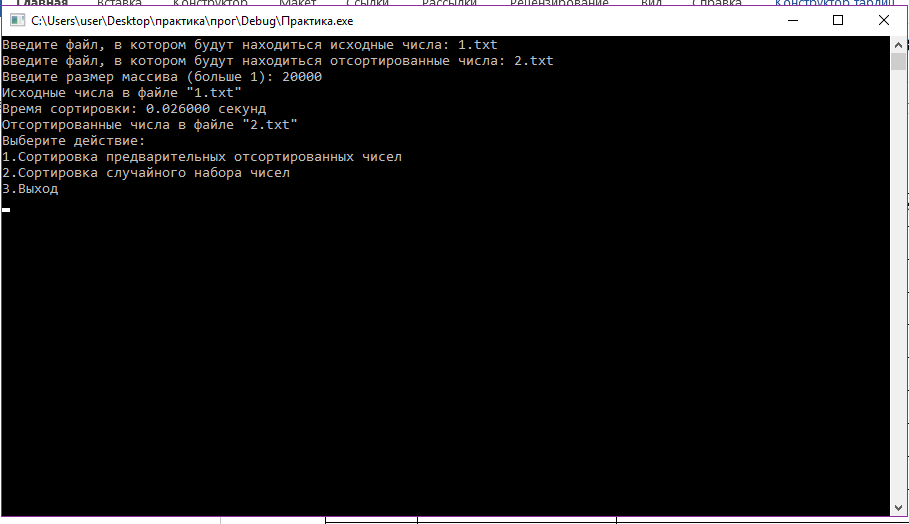


Рисунок А.2

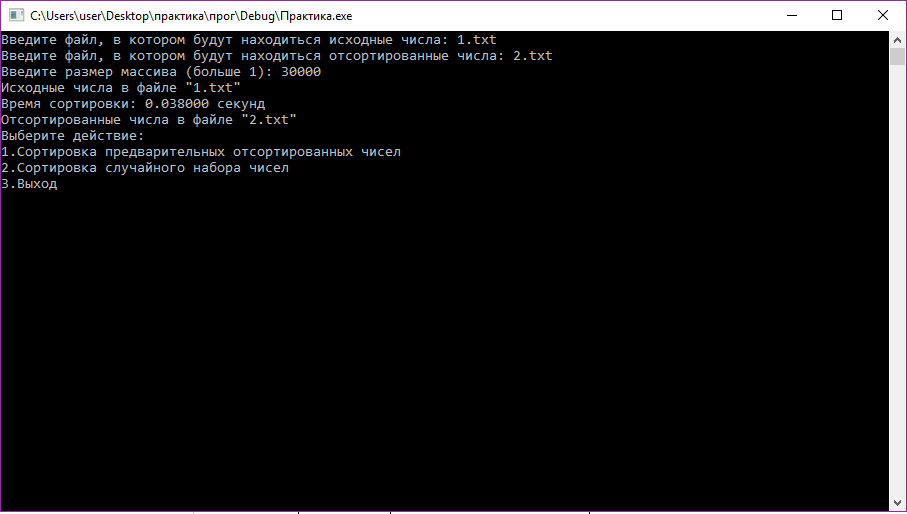


Рисунок А.3

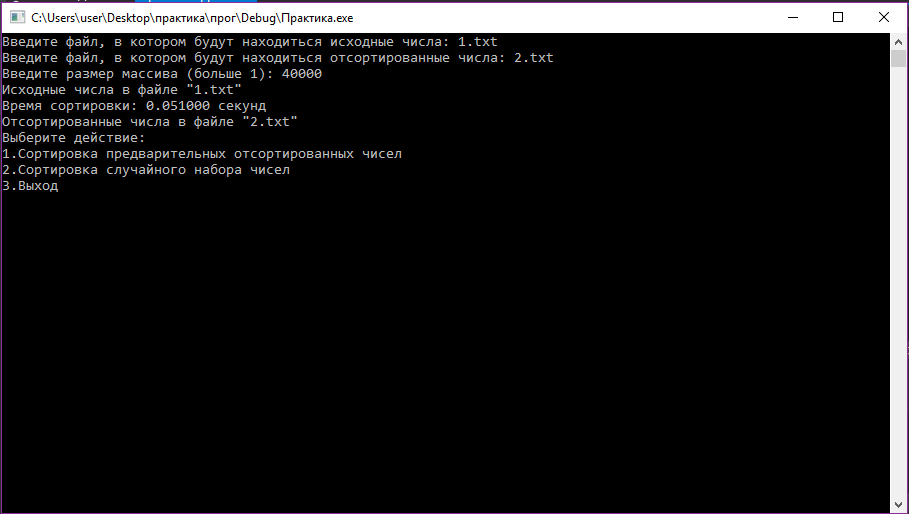


Рисунок А.4

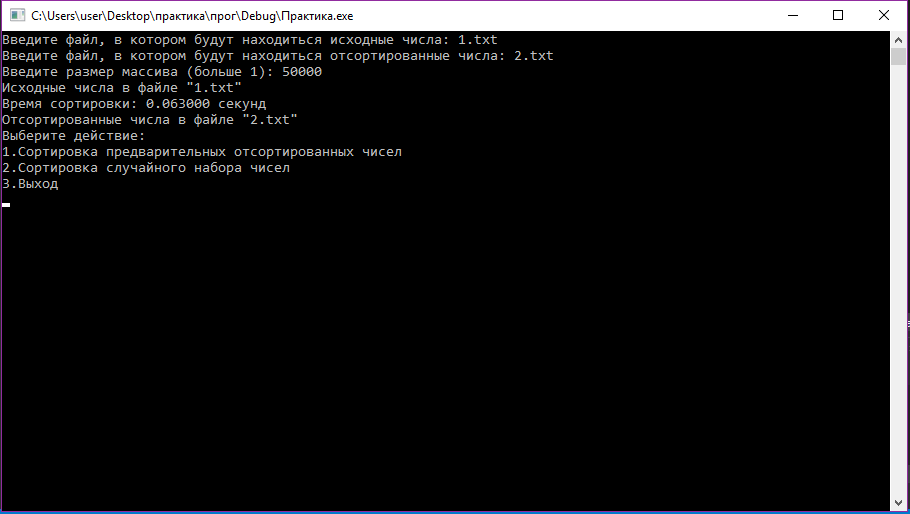


Рисунок А.5

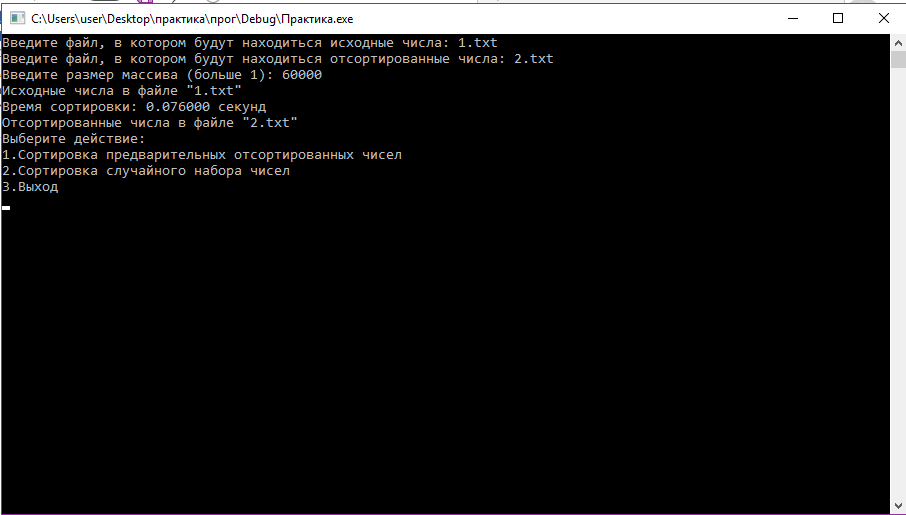


Рисунок А.6

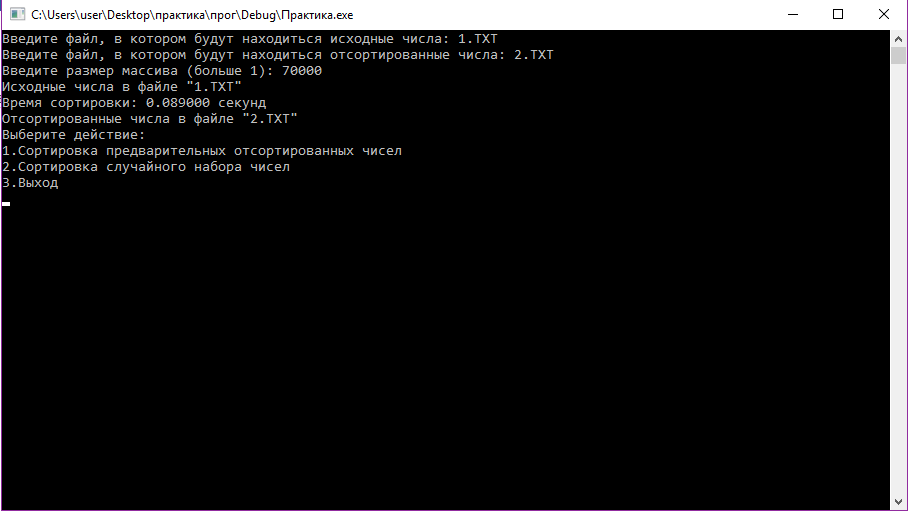


Рисунок А.7

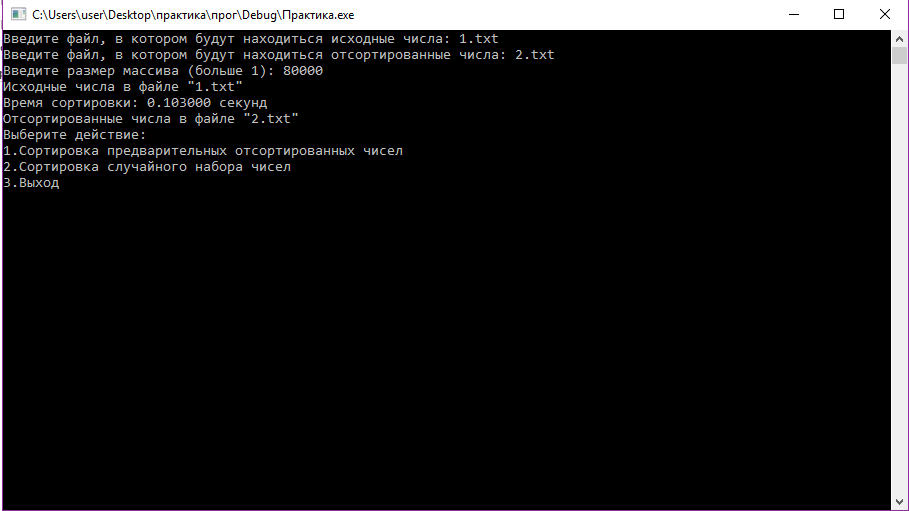


Рисунок А.8

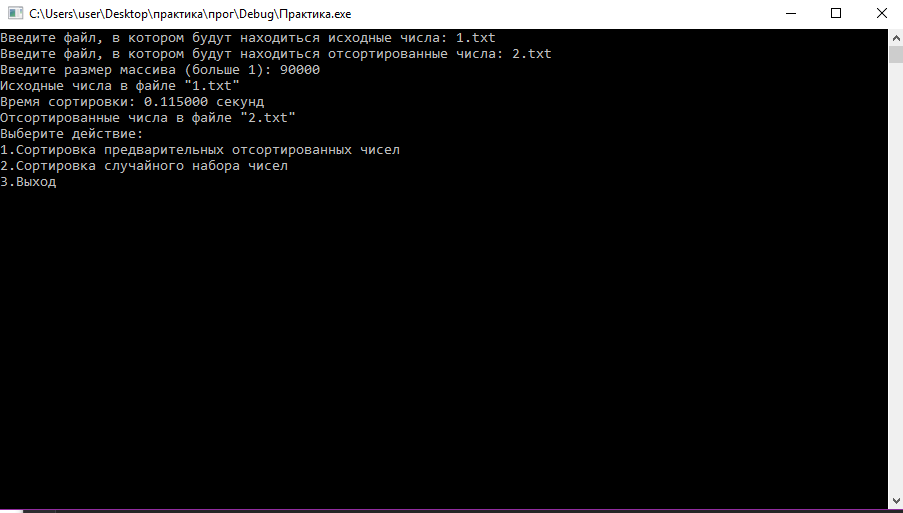


Рисунок А.9

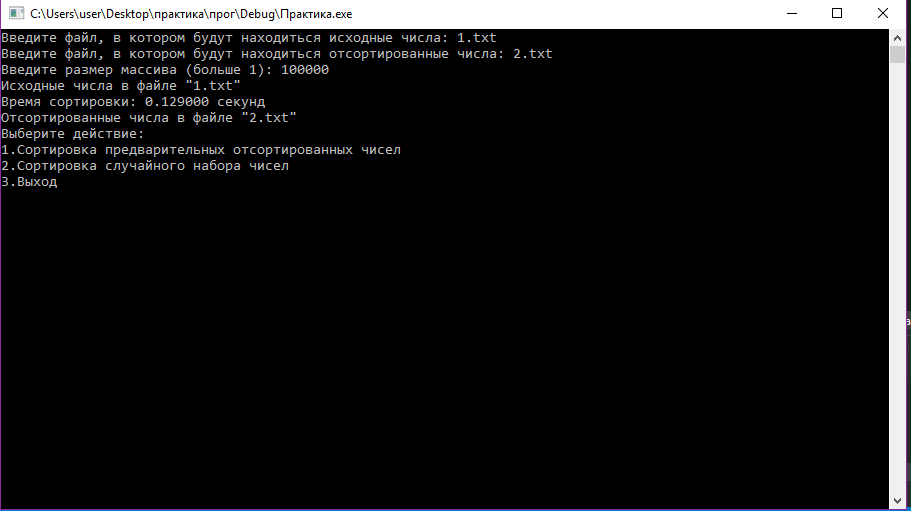


Рисунок А.10

# Приложение B

Листинг программы

#pragmawarning(disable : 4996)

#include"menu.h"

#include<time.h>

#include<conio.h>

#include<stdio.h>

#include<Windows.h>

void Merge(int\* Ap, intp, intq, intr)

{

int n1 = q - p + 1;

int n2 = r - q;

int\* L; int\* R;

L = newint[n1 + 1];

R = newint[n2 + 1];

for (int i = 1; i <= n1; i++)

{

L[i-1] = Ap[p + i - 1];

}

for (int j = 1; j <= n2; j++)

{

R[j-1] = Ap[q + j];

}

for (int i = 0, j = 0, k = p; k <= r; k++) {

if (i < n1 && j < n2) {

if (L[i] <= R[j]) {

Ap[k] = L[i];

i++;

}

else {

Ap[k] = R[j];

j++;

}

}

elseif (i < n1) {

Ap[k] = L[i];

i++;

}

else {

Ap[k] = R[j];

j++;

}

}

}

void Merge\_Sort(int\* Ap, intp, intr)

{

int q;

if (p<r)

{

q = (p + r) / 2;

Merge\_Sort(Ap, p, q);

Merge\_Sort(Ap, q + 1, r);

Merge(Ap, p, q, r);

}

}

void save1(FILE\* F1, char\* F1name, int\* A, intsize)

{

F1 = fopen(F1name, "w");

if (F1 != NULL)

{

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

{

fprintf(F1, "%d\n", A[i]);

}

}

else

{

printf("Ошибка записи в файл");

}

fclose(F1);

}

void save2(FILE\* F2, char\* F2name, int\* A, intsize)

{

F2 = fopen(F2name, "w");

if (F2 != NULL)

{

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

{

fprintf(F2, "%d\n", A[i]);

}

}

else

{

printf("Ошибка записи в файл");

}

fclose(F2);

}

void random(void)

{

int size;

FILE\* F1, \* F2;

char F1name[20];

char F2name[20];

printf("Введите файл, в котором будут находиться исходные числа: ");

gets\_s(F1name);

F1 = fopen(F1name, "w");

printf("Введите файл, в котором будут находиться отсортированные числа: ");

gets\_s(F2name);

F2 = fopen(F2name, "w");

printf("Введите размер массива (больше 1): ");

do

{

scanf\_s("%d", &size);

if (size < 2)

{

printf("Неверно ввёден размер массива\n");

printf("Введите размер массива (больше 1): ");

}

} while (size < 2);

int\* A = newint[size];

for (int i = 0; i < size; i++) //Заполнение массива А случайными числами

{

A[i] = (rand() % 20001) - 10000;

}

printf("Исходные числа в файле \"%s\" \n", F1name);

save1(F1, F1name, A, size);

time\_t start = clock();

Merge\_Sort(A, 0, (size - 1));

time\_t stop = clock();

int time = (stop - start) / 1000;

printf("Время сортировки: %i мс\n", time);

save2(F2, F2name, A, size);

printf("Отсортированные числа в файле \"%s\" \n", F2name);

}

void sort(void)

{

FILE \*F3,\* F4, \*F5, \*F6;

char F3name[20], F4name[20];

printf("Введите название файла: ");

gets\_s(F3name);

F3 = fopen(F3name, "r");

if (F3 == NULL)

{

printf("Невозможно открыть файл \"%s\"\n", &F3name);

}

else

{

int size = 0;

int curDigit = 0;

while (fscanf(F3, "%d", &curDigit) != EOF) {

size++;

}

fseek(F3, 0, SEEK\_SET);

int\* A = newint [size];

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

{

fscanf(F3, "%d", &A[i]);

}

fclose(F3);

time\_t start = clock();

Merge\_Sort(A, 0, size - 1);

time\_t stop = clock();

int time = (stop - start) / 1000;

printf("Время сортировки: %i мс\n", time);

printf("Напишите, в какой файл сохранить отсортированные файлы (текстовый файл!): ");

gets\_s(F4name);

F4 = fopen(F4name, "w");

if (F4 != NULL)

{

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

{

fprintf(F4, "%d\n", A[i]);

}

}

else

{

printf("Ошибка записи в файл");

}

fclose(F4);

}

}

int main(void)

{

SetConsoleCP(1251); //русификация

SetConsoleOutputCP(1251);

srand(time(NULL));

int s = 1;

while (s != 0)

{

menu();

s = \_getch();

switch (s)

{

case'1': system("cls"); sort(); break;

case'2': system("cls"); random(); break;

case'3': system("cls"); exit(0); break;

default: system("cls");

}

}

return 0;

}