Федеральное государственное образовательное бюджетное учреждение  
высшего профессионального образования  
«Нижегородский Государственный Университет им.  
Н.И.Лобачевского» (ННГУ)

Национальный исследовательский Университет  
Институт Информационных Технологий Математики и Механики

Отчёт по лабораторной работе  
Генерация случайных чисел. Выполнение операций с  
ними.

Выполнил:  
студент группы 3821Б1ФИ3  
Мажара А. И.  
Проверил:  
Заведующий лабораторией, каф. МО СТ, ИИТММ

Лебедев И.Г

Нижний Новгород  
2021 г.

**Содержание.**

[Введение. 3](#_Введение.)

[Постановка задачи. 4](#_Toc86237500)

[Руководство пользователя. 5](#_Toc86237501)

[Руководство программиста. 8](#_Toc86237502)

[Эксперименты. 14](#_Toc86237503)

[Заключение. 19](#_Toc86237504)

[Литература. 20](#_Toc86237505)

[Приложение. 21](#_Toc86237506)

# Введение.

Программирование - это интересный, полезный и увлекательный процесс, благодаря которому мы, с помощью специальных команд, заставляем компьютер, выполнять для нас различные задачи, от выполнения операций с числами и навигации, до управления самолетами, спутниками и прочей техникой.

Случайные числа являются одной из основных составляющих любого языка программирования, на них строятся различные алгоритмы.

В данной лабораторной работе для изучения методов и особенностей работы со случайными числами была поставлена задача, которую нужно было выполнить, используя язык программирования «С».

# 1.Постановка задачи.

Сравнение сортировок.

Реализовать сортировки массивов данных (тип данных “double”) задаваемых: обязательно случайно, дополнительно с клавиатуры или из файла.

Реализовать сортировки: пузырьком, вставкой, быстрая.

Сравнить время работы, сделать выводы.

Первая программа создает текстовый файл с записанными в него числами. Программа принимает количество чисел n, максимальное и минимальное значение.

Вторая программа читает текстовый файл с набором чисел, выводит консольный интерфейс (печать, сортировка, сброс, выход), выполняет выбранные действия.

# 2.Руководство пользователя.

Используется две программы. После запуска первой программы выводится сообщение на экран: «Введите n:». (см. рис. 1)

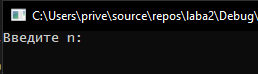


Рисунок 1. Ввод кол-ва случайных чисел.

Следует ввести с клавиатуры кол-во случайных чисел, которые в дальнейшем будут записаны в файл. После ввода необходимо нажать клавишу «Enter». Далее на экран выводится сообщение: «Введите min (мин. значение диапазона)». Пользователь должен ввести минимальное значение диапазона случайных чисел и нажать «Enter». Следующим шагом выводиться сообщение: «Введите max (макс. значение диапазона)». Необходимо ввести с клавиатуры максимальное значение диапазона случайных чисел и снова нажать клавишу «Enter». (см. рис. 2)

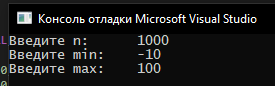


Рисунок 2. Ввод значений.

Если пользователь введет отрицательное значение для кол-ва случайных чисел, на экран будет выведено сообщение: «ERROR n <= 0 !!!!». (см. рис. 3)

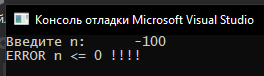


Рисунок 3. Ошибка в кол-ве случайных чисел.

Если же пользователь введет такие значения для минимального и максимального чисел, что максимальное будет меньше минимального, то на экран будет выведено следующее сообщение: «ERROR max < min !!!!». (см. рис. 4)

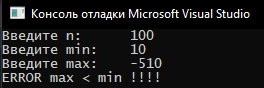


Рисунок 4. Ошибка в диапазоне (min, max).

Если все данные введены верно, то программа завершится и в файле появятся числа.

(см. рис. 5, 6)

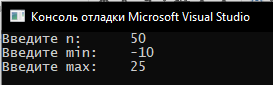


Рисунок 5. Завершение первой программы.

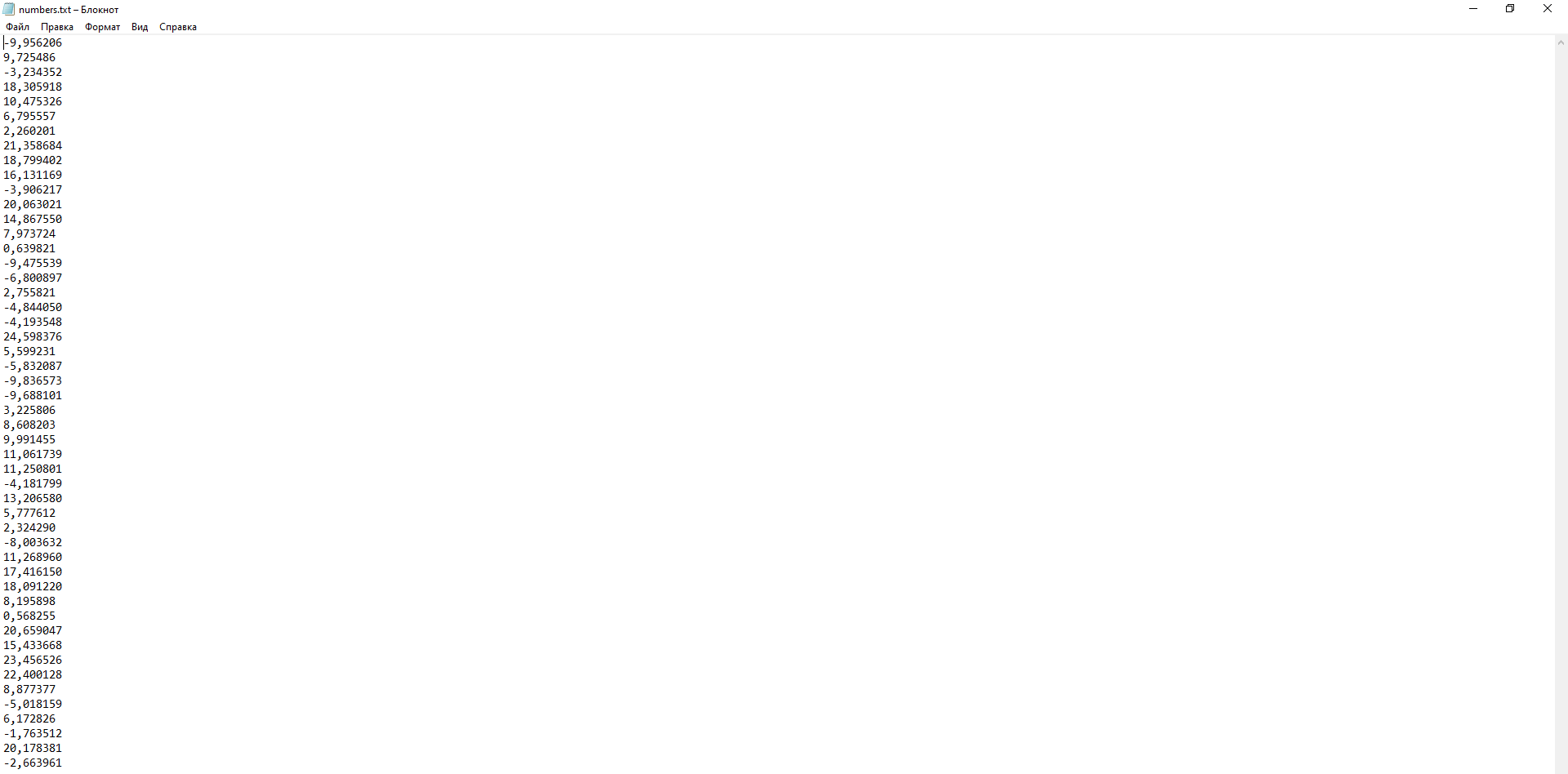


Рисунок 6. Записанные числа в файле.

После завершения первой программы следует запустить вторую. После ее запуска на экран будет выведен консольный интерфейс. (см. рис. 7)

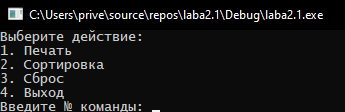


Рисунок 7. Консольный интерфейс.

Консольный интерфейс представляет собой набор из четырех команд:

1. Печать

2. Сортировка

3. Сброс

4. Выход

* Команда «Печать» осуществляет вывод на экран чисел, записанных в файле.
* Команда «Сортировка» сортирует числа.
* Команда «Сброс» сбрасывает сортировку.
* Команда «Выход» завершает программу.

Чтобы вывести на экран числа, отсортированные по возрастанию, следует сначала выполнить «Сортировку» и после «Печать».

При выборе команды «Сортировка» пользователю предоставляется выбор типа сортировки. (см. рис. 8)

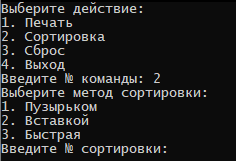


Рисунок 8. Типы сортировки.

После выполнения какой-либо сортировки на экран будет выведено сообщение о времени занимаемом этим типом сортировки. (см. рис. 9)

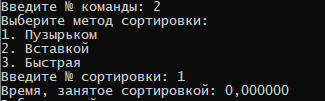


Рисунок 9. Время сортировки.

Чтобы выполнить другую сортировку нужно сбросить предыдущую. Для этого следует воспользоваться командой «Сброс».

Таким образом, мы можем разными способами сортировать числа в файле, сравнивать скорость сортировок и выводить отсортированные по возрастанию числа на экран.

(см. рис. 10)

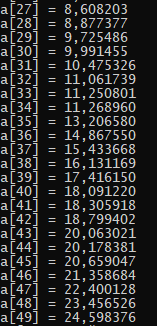


Рисунок 10. Отсортированные числа.

**3.Руководство программиста.**

***Описание структуры программы.***

Первая программа состоит из одного модуля int main() {…}, в котором находиться код создания чисел и записи их в файл. Вторая программа также состоит из одного модуля int main() {…}, в котором находится код сортировки чисел.

***Описание алгоритмов.***

**1. Алгоритм записи случайных чисел в файл.**

a = (double\*)malloc(n \* sizeof(double));//массив для рандомных чисел

if (a == 0)

{

printf("ERROR 1!\n");

return 0;

}

fnumbers = fopen("C:\\Users\\prive\\Desktop\\numbers.txt", "w");

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

{

a[i] = (((double)rand()) / RAND\_MAX) \* (max - min) + min;//забиваю массив

fprintf(fnumbers, "%lf\n", a[i]);

}**2. Алгоритм считывания чисел из файла.**

fnumbers = fopen("C:\\Users\\Administrator\\Desktop\\numbers.txt", "r");

fnumbers = fopen("C:\\Users\\prive\\Desktop\\numbers.txt", "r");

n = sc(fnumbers); //подсчет кол-ва строк

a = (double\*)malloc(n \* sizeof(double));

if (a == 0)

{

printf("Ошибка 1!\n");

return 0;

}

rewind(fnumbers);

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

{

fscanf\_s(fnumbers, "%lf", &a[i]);

}

**3. Алгоритм консольного интерфейса.**

fnumbers = fopen("C:\\Users\\prive\\Desktop\\numbers.txt", "r");

n = sc(fnumbers); //подсчет кол-ва строк

a = (double\*)malloc(n \* sizeof(double));

if (a == 0)

{

printf("Ошибка 1!\n");

return 0;

}

rewind(fnumbers);

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

{

fscanf\_s(fnumbers, "%lf", &a[i]);

}

do

{

printf("Выберите действие:\n1. Печать\n2. Сортировка\n3. Сброс\n4. Выход\n");

printf("Введите № команды: ");

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

if (cmd == 1)

{

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

{

printf("a[%d] = %lf\n", i, a[i]);

}

}

else if (cmd == 2)

{

printf("Выберите метод сортировки:\n1. Пузырьком\n2. Вставкой\n3. Быстрая\n");

printf("Введите № сортировки: ");

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

if (sort == 1) //сортировка пузырьком

{

clock\_t start = clock();

BubbleSort(n, a);

clock\_t end = clock();

t = ((double)(end - start)) / CLOCKS\_PER\_SEC;

printf("Время, занятое сортировкой: %lf\n", t);

}

else if (sort == 2) //сортировка вставкой

{

rewind(fnumbers);

clock\_t start = clock();

InsertionSort(n, a);

clock\_t end = clock();

t = ((double)(end - start)) / CLOCKS\_PER\_SEC;

printf("Время, занятое сортировкой: %lf\n", t);

}

else if (sort == 3) //быстрая сортировка

{

clock\_t start = clock();

qs(a, 0, n - 1);

clock\_t end = clock();

t = ((double)(end - start)) / CLOCKS\_PER\_SEC;

printf("Время, занятое сортировкой: %lf\n", t);

}

}

else if (cmd == 3)

{

rewind(fnumbers);

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

{

fscanf\_s(fnumbers, "%lf", &a[i]);

}

printf("Массив сброшен к исходному.\n");

}

else if (cmd == 4)

{

printf("Завершение работы.\n");

}

else

{

printf("Error.\n");

}

} while (cmd != 4);

**4. Сортировки.**

**4.1. Сортировка пузырьком.**

void BubbleSort(int n, double a[]) {

int i, j;

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

{

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

{

if (a[j] > a[j + 1])

{

double tmp = a[j];

a[j] = a[j + 1];

a[j + 1] = tmp;

}

}

}

}

**4.2. Сортировка вставкой.**

void InsertionSort(int n, double a[]) {

int i;

int location;

double newElement;

for (i = 1; i < n; i++)

{

newElement = a[i];

location = i - 1;

while (location >= 0 && a[location] > newElement)

{

a[location + 1] = a[location];

location = location - 1;

}

a[location + 1] = newElement;

}

}

**4.3. Быстрая сортировка.**

void qs(double\* a, int left, int right) {

int i, j;

double x, y;

i = left;

j = right;

x = a[(left + right) / 2];

do

{

while ((a[i] < x) && (i < right))

i++;

while ((x < a[j]) && (j > left))

j--;

if (i <= j)

{

y = a[i];

a[i] = a[j];

a[j] = y;

i++;

j--;

}

} while (i <= j);

if (left < j)

qs(a, left, j);

if (i < right)

qs(a, i, right);

}

**Блок-схема 1. Алгоритм записи случайных чисел в файл.**

i++

fprintf(fnumbers, "%lf\n", a[i])

a[i] = (((double)rand()) / RAND\_MAX) \* (max - min) + min

True

i < n

i = 0

fnumbers = fopen("C:\\Users\\Administrator\\Desktop\\numbers.txt", "w")

False

True

printf("Ошибка 1!\n")

a = 0

a = (double\*)malloc(n \* sizeof(double))

**Блок-схема 2.** **Алгоритм считывания чисел из файла.**

i++

fscanf\_s(fnumbers, "%lf", &a[i])

i < n

i = 0

printf("Ошибка 1!\n")

False

rewind(fnumbers)

a == 0

a = (double\*)malloc(n \* sizeof(double))

# Эксперименты.

Для начала убедимся, что первая программа записывает числа в файл. Изначально «Блокнот» чист:

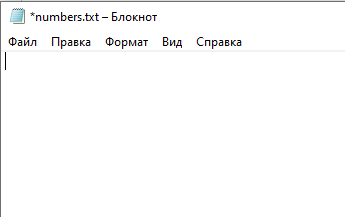


Рисунок 11. Чистый файл.

Далее запустим программу для записи чисел в файл и проверим его на наличие записанных чисел:

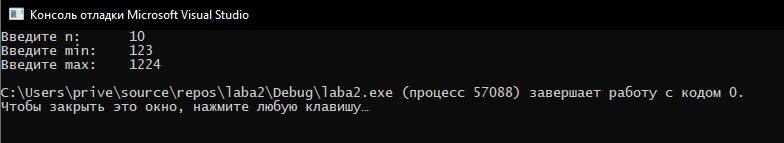


Рисунок 12. Выполнение программы.

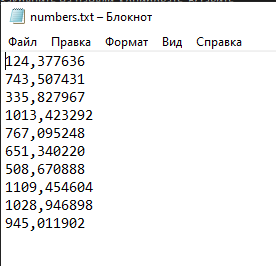


Рисунок 13. Записанные числа в файл.

Действительно, программа записала 10 случайных чисел в правильном диапазоне.

Заметим, если будут введены некорректные данные, то программа выведет на экран сообщение с указанием на конкретную ошибку при вводе:

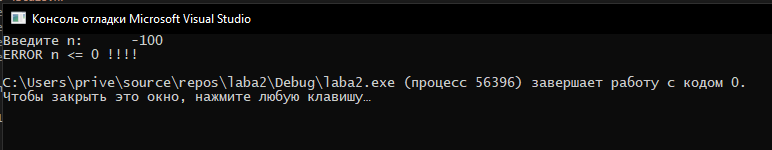


Рисунок 14. Некорректное кол-во чисел.

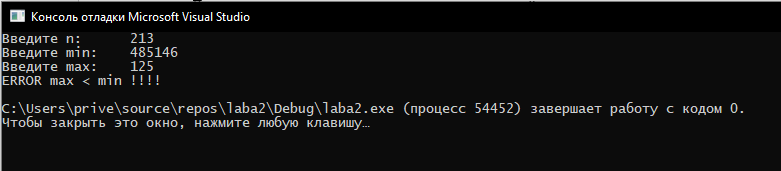


Рисунок 15. Некорректный диапазон.

Далее проверим правильность выполнения второй программы, которая представляет собой консольный интерфейс.

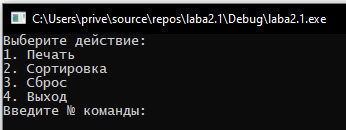


Рисунок 16. Консольный интерфейс.

Выберем команду «Печать». Для этого введем «1» и нажмем «Enter». На экран будут выведены числа, записанные в файле:

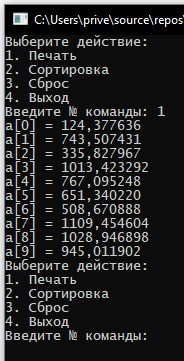


Рисунок 17. Печать.

Теперь отсортируем их методом пузырька и выведем результат:

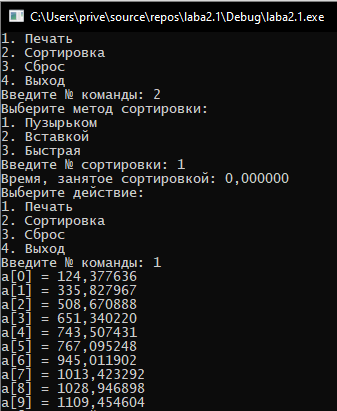


Рисунок 18. Сортировка пузырьком и вывод результата.

Проверим сброс сортировки. Выберем команду «Сброс», после «Печать»:

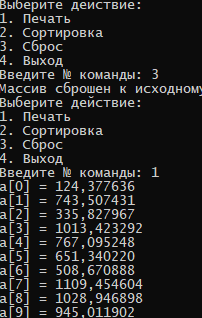


Рисунок 19. Сброс сортировки.

Проверим оставшиеся две сортировки. Для этого после каждой сортировки будем сбрасывать ее:

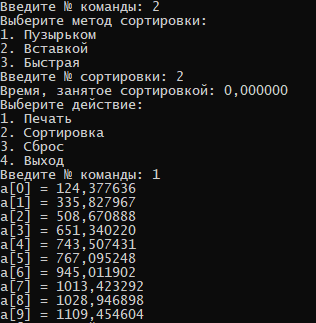


Рисунок 20. Сортировка вставками и вывод результата.

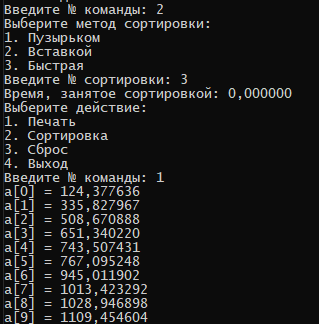


Рисунок 21. Быстрая сортировка и вывод результата.

Проверим программу на введение некорректного номера команды. Для этого введем номер, не присвоенный не одной из команд:

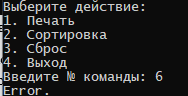


Рисунок 22. Некорректный номер команды.

Наконец, выберем команду «Выход» и завершим программу. На экран выведется сообщение о завершении.

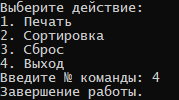


Рисунок 23. Завершение работы программы.

|  |  |  |  |
| --- | --- | --- | --- |
| **Тип сортировки** | **Сложность** | **Размер входных данных** | **Время работы, с** |
| Пузырьком |  | 100000 |  |
| Вставками |  | 100000 | 5,901000 |
| Быстрая |  | 100000 | 0,013000 |
| Пузырьком |  | 50000 |  |
| Вставками |  | 50000 |  |
| Быстрая |  | 50000 |  |

# Заключение.

В ходе лабораторной работы была написана программа на языке программирования «С», которая полностью выполняет поставленную задачу, а именно:

«Сравнение сортировок.

Реализовать сортировки массивов данных (тип данных “double”) задаваемых: обязательно случайно, дополнительно с клавиатуры или из файла.

Реализовать сортировки: пузырьком, вставкой, быстрая.

Сравнить время работы, сделать выводы.

Первая программа создает текстовый файл с записанными в него числами. Программа принимает количество чисел n, максимальное и минимальное значение.

Вторая программа читает текстовый файл с набором чисел, выводит консольный интерфейс (печать, сортировка, сброс, выход), выполняет выбранные действия».

Интерфейс программы простой, понятный и удобный в использовании, все сообщения выводятся на русском языке и при своих небольших размерах содержат нужную информацию, которую необходимо донести до пользователя.

В ходе сравнения сортировок можно сделать вывод: «быстрая сортировка (quick sort)» выполняет сортировку чисел быстрее остальных типов сортировки и очень быстро работает с любыми типами данных; сортировка «пузырьком» занимает большее время чем «quick sort», но тоже довольно быстро сортирует данные; «сортировка вставками» работает медленнее всех на большом диапазоне чисел, ее следует использовать на коротком отрезке не больше, чем десятки, или уже на частично отсортированном массиве данных.

Литература.  
1. Т.А. Павловская Учебник по программированию на языках высокого  
уровня(С/С++) – Режим доступа: http://cph.phys.spbu.ru/documents/First/books/7.pdf  
2. Бьерн Страуструп. Язык программирования С++ - Режим доступа:  
<http://8361.ru/6sem/books/Straustrup-Yazyk_programmirovaniya_c.pdf>

# Приложение.

Приложение 1.

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#include <malloc.h>

#include <math.h>

int main() {

setlocale(LC\_ALL, "Rus");

int i = 0, n = 0;

double min = 0.0, max = 0.0;

double\* a;

FILE\* fnumbers = 0;

printf("Введите n: \t"); //кол-во чисел

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

if (n <= 0)

{

printf("ERROR n <= 0 !!!!\n");

return 0;

}

printf("Введите min: \t"); //минимальное значение

scanf\_s("%lf", &min);

printf("Введите max: \t"); //максимальное значение

scanf\_s("%lf", &max);

if (max < min)

{

printf("ERROR max < min !!!!\n");

return 0;

}

a = (double\*)malloc(n \* sizeof(double));//массив для рандомных чисел

if (a == 0)

{

printf("ERROR 1!\n");

return 0;

}

fnumbers = fopen("C:\\Users\\prive\\Desktop\\numbers.txt", "w");

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

{

a[i] = (((double)rand()) / RAND\_MAX) \* (max - min) + min;//забиваю массив

fprintf(fnumbers, "%lf\n", a[i]);

}

fclose(fnumbers);

free(a);

return 0;

}

Приложение 2.

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#include <malloc.h>

#include <time.h>

#include "Header.h"

//счет строк в файле

int sc(FILE\* fnumbers) {

int result = 0;

rewind(fnumbers);

while (!ferror(fnumbers) && !feof(fnumbers)) {

if (fgetc(fnumbers) == '\n') ++result;

}

return result;

}

//сортировка пузырьком

void BubbleSort(int n, double a[]) {

int i, j;

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

{

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

{

if (a[j] > a[j + 1])

{

double tmp = a[j];

a[j] = a[j + 1];

a[j + 1] = tmp;

}

}

}

}

//сортировка вставкой

void InsertionSort(int n, double a[]) {

int i;

int location;

double newElement;

for (i = 1; i < n; i++)

{

newElement = a[i];

location = i - 1;

while (location >= 0 && a[location] > newElement)

{

a[location + 1] = a[location];

location = location - 1;

}

a[location + 1] = newElement;

}

}

//быстрая сортировка

void qs(double\* a, int left, int right) {

int i, j;

double x, y;

i = left;

j = right;

x = a[(left + right) / 2];

do

{

while ((a[i] < x) && (i < right))

i++;

while ((x < a[j]) && (j > left))

j--;

if (i <= j)

{

y = a[i];

a[i] = a[j];

a[j] = y;

i++;

j--;

}

} while (i <= j);

if (left < j)

qs(a, left, j);

if (i < right)

qs(a, i, right);

}

int main() {

setlocale(LC\_ALL, "Rus");

FILE\* fnumbers;

int i = 0, j = 0, n = 0, cmd = 0, sort = 0;

double b = 0.0;

double\* a;

double t = 0;

fnumbers = fopen("C:\\Users\\prive\\Desktop\\numbers.txt", "r");

n = sc(fnumbers); //подсчет кол-ва строк

a = (double\*)malloc(n \* sizeof(double));

if (a == 0)

{

printf("Ошибка 1!\n");

return 0;

}

rewind(fnumbers);

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

{

fscanf\_s(fnumbers, "%lf", &a[i]);

}

do

{

printf("Выберите действие:\n1. Печать\n2. Сортировка\n3. Сброс\n4. Выход\n");

printf("Введите № команды: ");

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

if (cmd == 1)

{

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

{

printf("a[%d] = %lf\n", i, a[i]);

}

}

else if (cmd == 2)

{

printf("Выберите метод сортировки:\n1. Пузырьком\n2. Вставкой\n3. Быстрая\n");

printf("Введите № сортировки: ");

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

if (sort == 1) //сортировка пузырьком

{

clock\_t start = clock();

BubbleSort(n, a);

clock\_t end = clock();

t = ((double)(end - start)) / CLOCKS\_PER\_SEC;

printf("Время, занятое сортировкой: %lf\n", t);

}

else if (sort == 2) //сортировка вставкой

{

rewind(fnumbers);

clock\_t start = clock();

InsertionSort(n, a);

clock\_t end = clock();

t = ((double)(end - start)) / CLOCKS\_PER\_SEC;

printf("Время, занятое сортировкой: %lf\n", t);

}

else if (sort == 3) //быстрая сортировка

{

clock\_t start = clock();

qs(a, 0, n - 1);

clock\_t end = clock();

t = ((double)(end - start)) / CLOCKS\_PER\_SEC;

printf("Время, занятое сортировкой: %lf\n", t);

}

}

else if (cmd == 3)

{

rewind(fnumbers);

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

{

fscanf\_s(fnumbers, "%lf", &a[i]);

}

printf("Массив сброшен к исходному.\n");

}

else if (cmd == 4)

{

printf("Завершение работы.\n");

}

else

{

printf("Error.\n");

}

} while (cmd != 4);

fclose(fnumbers);

free(a);

return 0;

}