**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский Нижегородский государственный университет им. Н.И. Лобачевского»**

**(ННГУ)**

**Институт информационных технологий, математики и механики**

Направление подготовки: «Фундаментальная информатика и информационные технологии»

Отчёт по лабораторной работе

**Реализация и сравнение алгоритмов сортировки**

Выполнил:

студент ИИТММ гр. 381906-2

Челебаев А.А.

Проверил:

ассистент каф. математического обеспечения и суперкомпьютерных технологий

Лебедев И.Г.

Нижний Новгород

2019 г.

Содержание

[Введение 3](#_Toc25003585)

[1. Постановка задачи 4](#_Toc25003586)

[2. Руководство пользователя 5](#_Toc25003587)

[3. Руководство программиста 6](#_Toc25003589)

[3.1. Описание структуры программы 6](#_Toc25003590)

[3.2. Описание структур данных 6](#_Toc25003591)

[3.3. Описание алгоритмов](#_Toc25003592) 6

[4. Эксперименты 7](#_Toc25003593)

[Заключение 8](#_Toc25003594)

[Литература 9](#_Toc25003595)

[Приложение 10](#_Toc25003596)

# Введение

Развитие современного общества предполагает широкое использование компьютерных и информационных технологий для работы с получаемой информацией. Иногда при работе с большими объёмами данных бывает необходимо отсортировать данные по возрастанию или убыванию; специально для таких случаев программистами было придумано множество алгоритмов сортировки данных, три из которых мы рассмотрим и применим в этой работе.

# Постановка задачи

Реализовать сортировки массивов данных, 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.

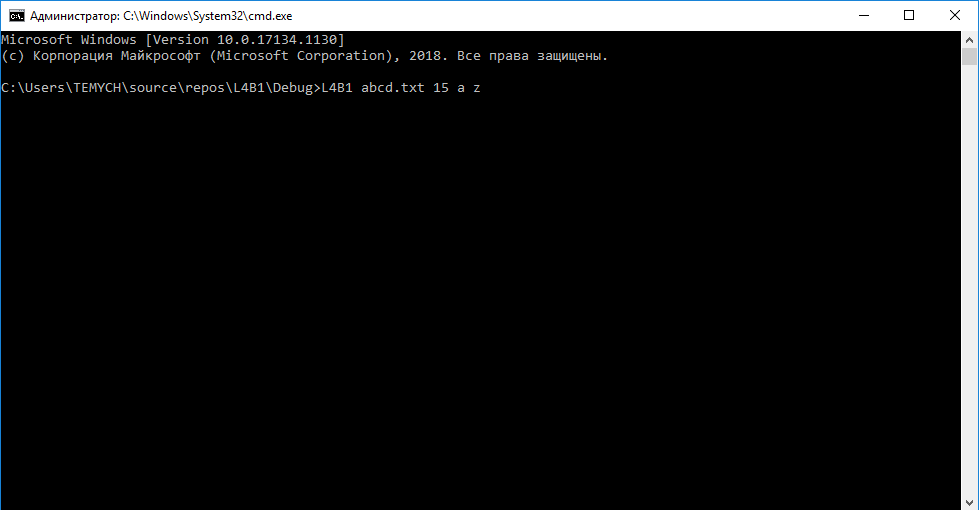
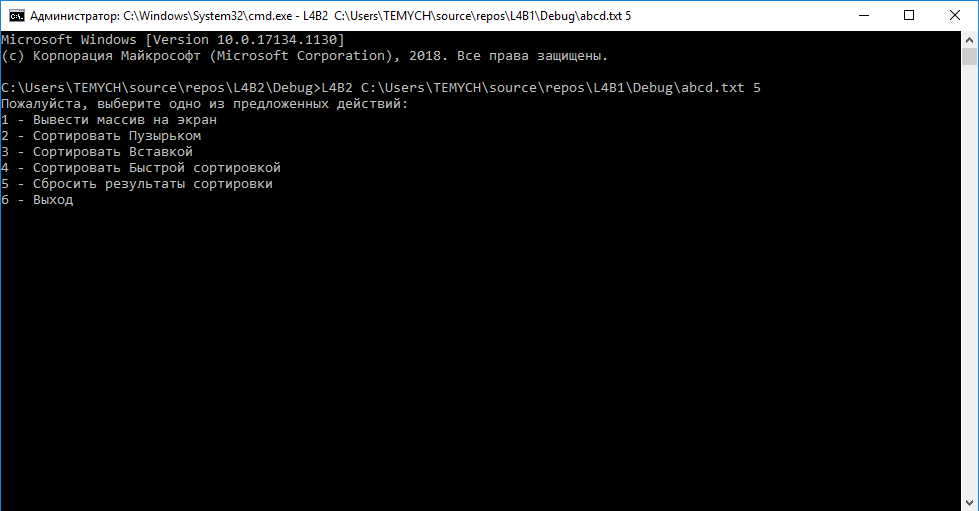
# Руководство пользователя

Использование программы пользователем:

Для того, чтобы запустить программу, перейдите по следующему пути:

**C:\Users\<Имя\_пользователя>\source\repos\<Название\_проекта>\Debug**,

Затем в строке состояния сотрите путь к каталогу и вместо него введите “**cmd**”. Откроется системная консоль. В командной строке введите название проекта и через пробел имя файла, в который хотите записать исходный массив, а также его размер и диапазон допустимых значений (Рис.1).

 ****

# Рисунок 1. Консольный интерфейс программы3. Руководство программиста

## 3.1. Описание структуры программы

1) Ввод данных;

2) Первая программа: создание массива и запись его в файл;

Вторая программа: Чтение массива из файла и различные действия с ним.

## 3.2. Описание структур данных

В работе используются переменные типа int(целочисленный) и char(символьный тип).

Программа содержит главную функцию void main(), являющуюся начальной точкой программы.

К программе подключены следующие библиотеки:

stdio.h–нужна для ввода/вывода

stdlib.h – нужна для использования функции rand()

В данной работе используется принцип модульного программирования: функции, вызываемые по ходу программы, описаны в отдельном исходном файле и подключены в заголовочном.

## 3.3. Описание алгоритмов

Первая программа:

В начале её работы открывается файл с указанным именем. Затем создаётся массив случайных символов на основе введённых пользователем данных и записывается в этот файл. На этом работа первой программы заканчивается.

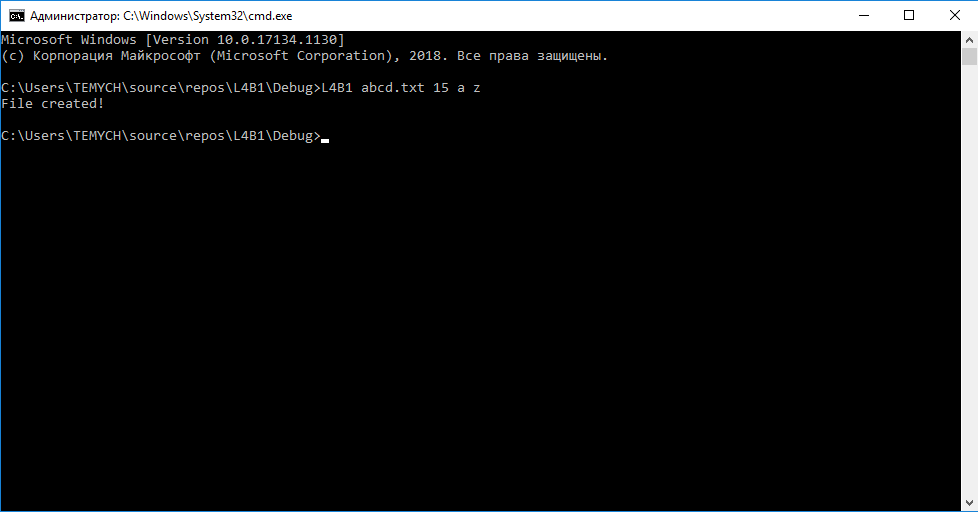
Вторая программа:

В начале её работы также открывается файл с уже записанным в него массивом случайных символов. Затем программа читает массив из файла и выводит на экран меню, в котором пользователю предлагаются различные действия.

# 

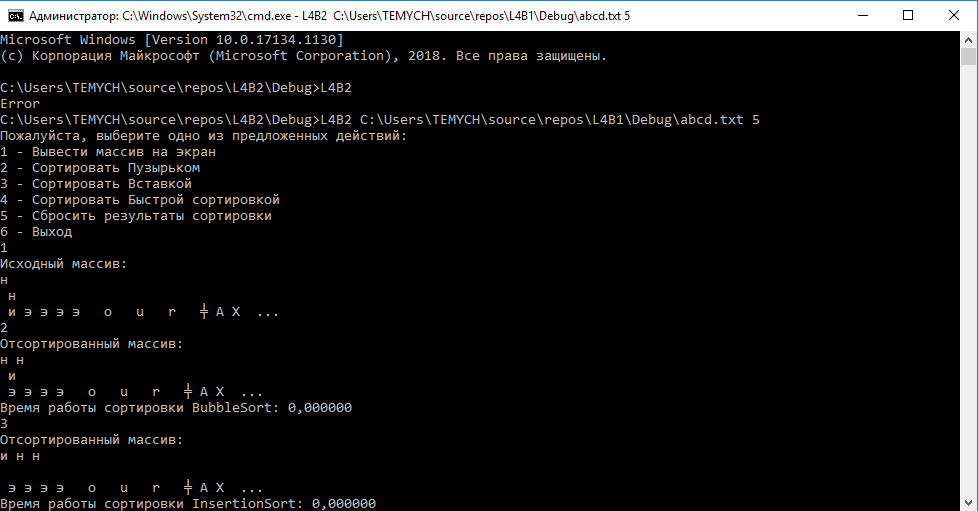
# 4. Эксперименты

Эксперимент №1 (Рис. 2)



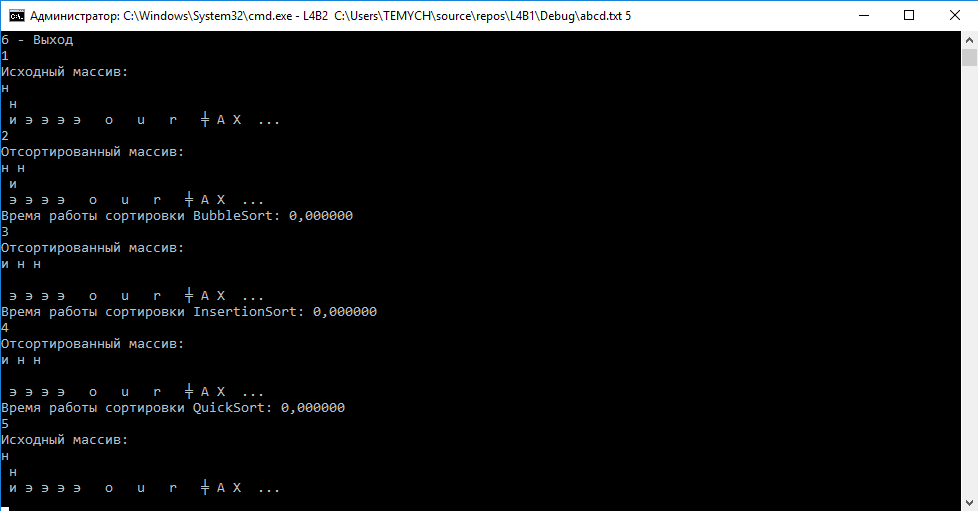
**Рисунок 2. Выполнение и результат эксперимента 1**

Эксперимент №2 (Рис. 3)



**Рисунок 3. Выполнение и результат эксперимента 2**

Эксперимент №3 (Рис.4)



**Рисунок 4. Выполнение и результат эксперимента 3**

# 

# Заключение

Таким образом, в рамках данной лабораторной работы была создана программа, которая применяет к исходному массиву данных три различных алгоритма сортировки: пузырьком, вставками и быструю сортировку.

# Литература

1. Павловская Т.А. C/C++, Программирование на языке высокого уровня, 2003.
2. Официальный сайт Habr. – <https://habr.com/ru/post/339656/>
3. Официальный сайт Microsoft. – <https://docs.microsoft.com/ru-ru/cpp/c-language/main-function-and-program-execution?view=vs-2019>
4. Брайан Керниган, Деннис Ритчи. Язык программирования С. – Москва: Вильямс, 2015. – 304 с.
5. Грег Перри, Дин Миллер. Программирование на С для начинающих. – Москва: Эксмо, 2015. – 368 с.: ил.

# Приложение

***Код программы 1***

main.c

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdlib.h>

#include <stdio.h>

#include "1.h"

void main(int argc, char\* argv[])

{

if (argc < 4)

{

printf("Error");

return 1;

}

FILE\* f = fopen(argv[1], "w");

if (f == NULL)

{

printf("Error");

return 1;

}

char min = argv[3];

char max = argv[4];

int n = atoi(argv[2]);

Make(f, n, min, max);

printf("File created!\n");

fclose(f);

return 1;

}

prog.c

#include <stdlib.h>

#include <time.h>

#include <stdio.h>

void Make(FILE\* f, int size, char min, char max)

{

srand(time(NULL));

int i = 0;

char a;

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

{

a = (rand() % max) + (min + 1);

fprintf(f, "%c\n", a);

}

}

1.h

#ifndef \_1\_H\_

#define \_1\_H\_

#include <stdio.h>

void Make(FILE\* f, int size, char min, char max);

#endif

***Код программы 2***

main.c

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdlib.h>

#include <stdio.h>

#include <time.h>

#include "1.h"

void main(int argc, char\* argv[])

{

if (argc < 3)

{

printf("Error");

return 1;

}

FILE\* f = fopen(argv[1], "r+");

if (f == NULL)

{

printf("Error");

}

else

{

int n = atoi(argv[2]);

clock\_t t3;

int t = 0;

char\* arr1 = (char\*)malloc(sizeof(char) \* n);

char\* arr2 = (char\*)malloc(sizeof(char) \* n);

arr1 = MakeArr(f, n);

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

{

arr2[i] = arr1[i];

}

menu();

do {

t = command();

//-------------------------------------------------------------------------------------

if (t == 1)

{

printf("Исходный массив:\n");

PrintArray(arr1, 20);

}

//-------------------------------------------------------------------------------------

if (t == 2)

{

t3 = BubbleSort(arr1, n);

printf("Отсортированный массив:\n");

PrintArray(arr1, 20);

printf("Время работы сортировки BubbleSort: %lf\n", (double)t3 / CLK\_TCK);

}

//-------------------------------------------------------------------------------------

if (t == 3)

{

t3 = InsertionSort(arr1, n);

printf("Отсортированный массив:\n");

PrintArray(arr1, 20);

printf("Время работы cортировки InsertionSort: %lf\n", (double)t3 / CLK\_TCK);

}

//-------------------------------------------------------------------------------------

if (t == 4)

{

t3 = QuickSort(arr1, 0, n - 1);

printf("Отсортированный массив:\n");

PrintArray(arr1, 20);

printf("Время работы сортировки QuickSort: %lf\n", (double)t3 / CLK\_TCK);

}

//-------------------------------------------------------------------------------------

if (t == 5)

{

int i = 0;

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

{

arr1[i] = arr2[i];

}

printf("Исходный массив:\n");

PrintArray(arr1, 20);

}

} while (t != 6);

}

return 0;

}

prog.c

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <malloc.h>

#include <locale.h>

#include <time.h>

char\* MakeArr(FILE\* f, int size)

{

char\* arr1 = (char\*)malloc(sizeof(char) \* size);

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

fscanf(f, "%c", &arr1[i]);

}

return arr1;

}

void PrintArray(char\* Arr, int size)

{

int i = 0;

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

{

printf("%c ", Arr[i]);

}

printf("...\n");

}

int command()

{

int i = 0;

scanf("%d", &i);

return i;

}

void menu()

{

setlocale(LC\_ALL, "Rus");

printf("Пожалуйста, выберите одно из предложенных действий:\n1 - Вывести массив на экран\n");

printf("2 - BubbleSort\n3 - InsertionSort\n4 - QuickSort\n");

printf("5 - Сбросить результаты сортировки\n6 - Выход\n");

}

clock\_t BubbleSort(char\* Arr, int size)

{

clock\_t t1, t2;

t1 = clock();

char tmp = 0;

int i, j = 0;

t1 = clock();

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

{

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

{

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

{

tmp = Arr[j];

Arr[j] = Arr[j + 1];

Arr[j + 1] = tmp;

}

}

}

t2 = clock();

return (t2 - t1);

}

clock\_t InsertionSort(char\* Arr, int size)

{

clock\_t t1, t2;

t1 = clock();

char newElement;

int location;

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

{

newElement = Arr[i];

location = i - 1;

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

{

Arr[location + 1] = Arr[location];

location = location - 1;

}

Arr[location + 1] = newElement;

}

t2 = clock();

return t2 - t1;

}

clock\_t QuickSort(char\* Arr, int first, int last)

{

clock\_t t1, t2, t3;

t1 = clock();

int i = first, j = last;

char tmp, x = Arr[(first + last) / 2];

do {

while (Arr[i] < x)

i++;

while (Arr[j] > x)

j--;

if (i <= j)

{

if (i < j)

{

tmp = Arr[i];

Arr[i] = Arr[j];

Arr[j] = tmp;

}

i++;

j--;

}

} while (i <= j);

if (i < last)

t3 = QuickSort(Arr, i, last);

if (first < j)

t3 = QuickSort(Arr, first, j);

t2 = clock();

return (t2 - t1);

}

1.h

#ifndef \_1\_H\_

#define \_1\_H\_

#include <stdio.h>

#include <time.h>

float\* MakeArr(FILE\* f, int size);

void menu();

int command();

clock\_t BubbleSort(char\* Arr, int size);

clock\_t InsertionSort(char\* Arr, int size);

clock\_t QuickSort(char\* Arr, int left, int right);

void PrintArray(char\* Arr, int size);

#endif