Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Колледж ВятГУ

**ОТЧЕТ**

**ПО ДОМАШНЕЙ КОНТРОЛЬНОЙ РАБОТЕ №5**

**«ИССЛЕДОВАНИЕ АЛГОРИТМОВ СОРТИРОВКИ»**

**ПО ДИСЦИПЛИНЕ «ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ»**

Выполнила: студентка учебной группы

ИСПк-203-52-00

Бочкарёва Виктория Дмитриевна

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2023

**Цель работы**

Получить базовые сведения о наиболее известных алгоритмах сортировки, изучить принцип работы с текстовыми файлами.

**Формулировка задания**

Фамилия: Бочкарева

Вариант: 3

Задание:

1. Реализовать сортировку данных с помощью алгоритма подсчетом.
2. Реализовать сортировку данных с помощью алгоритма слияния.
3. В обоих случаях предусмотреть возможность изменения компаратора (реализация компаратора в виде передаваемой в подпрограмму функции).
4. Считывание и вывод данных необходимо производить из текстового файла.
5. Для демонстрации работы программных реализаций самостоятельно подготовить варианты входных данных (при этом объем текстовых файлов должен позволять оценить скорость работы программ).

**Описание алгоритма**

Для решения поставленных задач, в коде будут использоваться подпрограммы, а именно – *procedure,* для каждой из заданных сортировок.

Чтобы пользователь мог выбрать вариант сортировки, используется оператор выбора *Case..of*.

Для работы с текстовыми файлами используются процедуры и функции для работы с файлами: *Assign* (связывает файловую переменную с файлом на диске), *Reset* (открывает файл для чтения), *Rewrite* (открывает/создает файл на запись), *Close* (закрывает файл), *Readln* (для ввода значений из файла), *Writeln* (для вывода значений в файл), *EOf* (возвращает true, когда достигнут конец файла).

Для удобства пользования программой подключается консоль с помощью модуля *CRT*.

Процедура для сортировки счетчиком работает по принципу: посчитать, сколько раз элемент встречается в массиве, а потом заполнить массив результатами этого подсчета.

Процедура для сортировки слиянием работает по принципу: массив разбивается на две одинаковых части, и каждая из получившихся частей сортируется отдельно тем же самым алгоритмом, все упорядоченные массивы сливаются в один.

В результате выполнения программы данные, расположенные в файле, считываются программой, сортируются в зависимости от выбранных пользователем пунктов и помещаются в другой файл.

**Схема алгоритма**

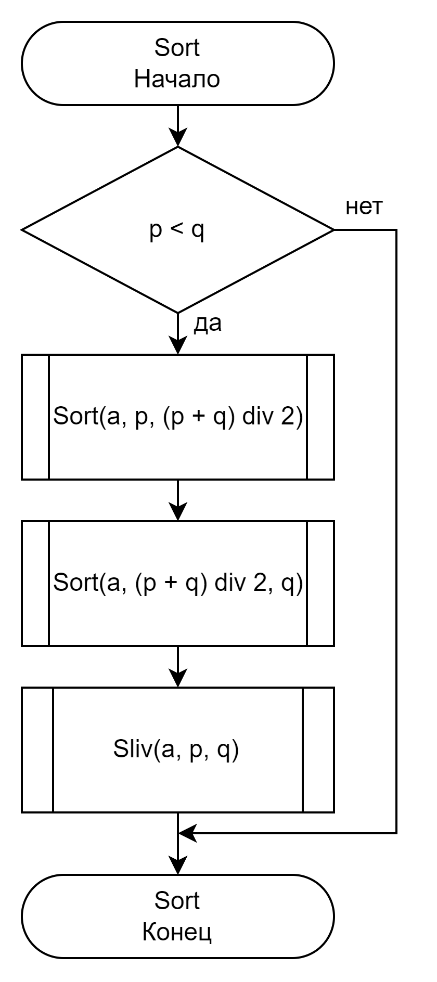


Рис. 1 – Схема процедуры для сортировки частей массива в сортировке алгоритмом слияния

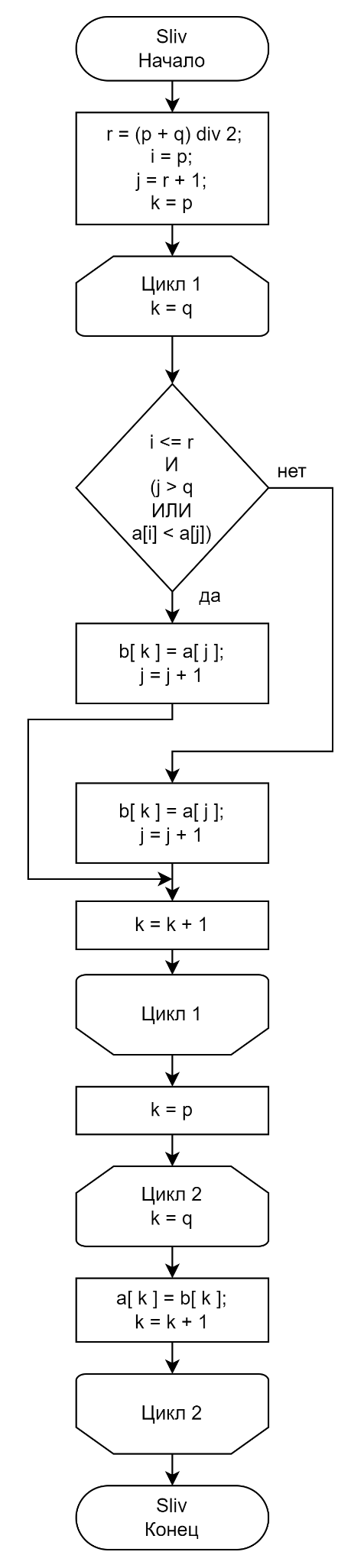


Рис. 2 – Схема процедуры для слияния массива в один в сортировке алгоритмом слияния

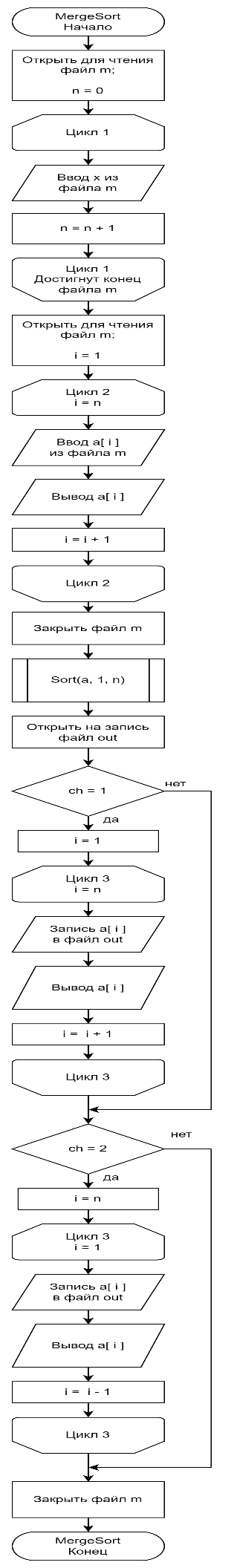


Рис. 3 – Схема процедуры для сортировки алгоритмом слияния

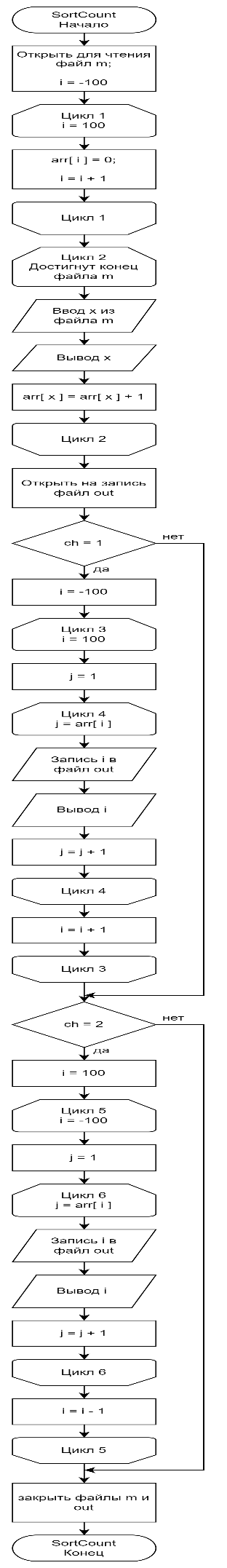


Рис. 4 – Схема алгоритма для сортировки алгоритмом подсчета

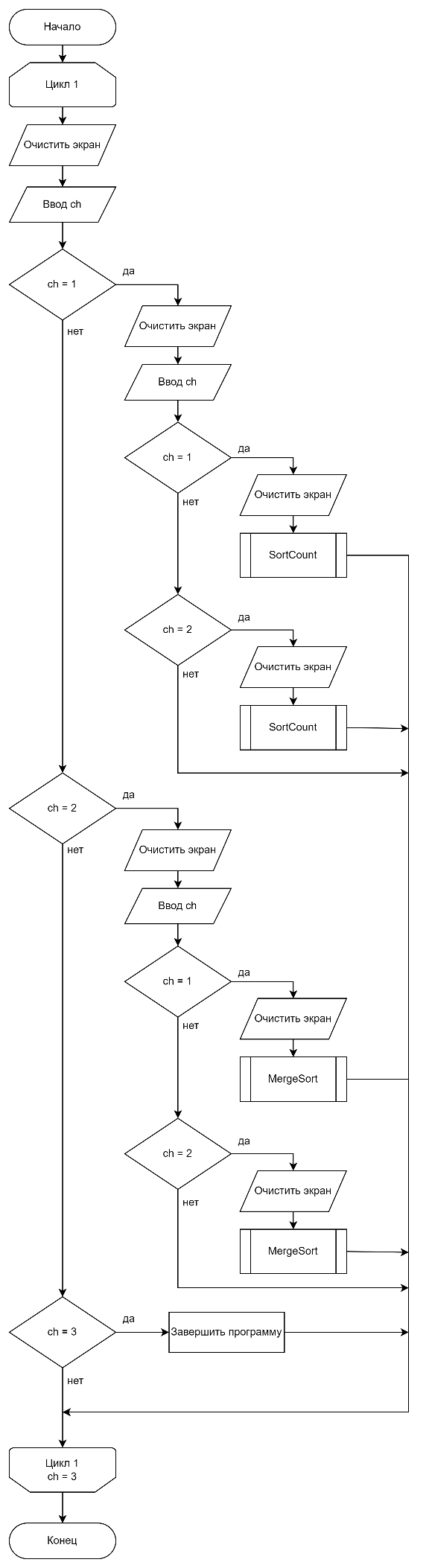


Рис. 5 – Схема основной программы

**Код программы**

{Assign(m, 'c:\маленбкий файл.txt'); 12 чисел

Assign(m, 'c:\файл поболбше.txt'); 50 чисел

Assign(m, 'c:\ваще болбшой файл жесь.txt'); 200 чисел}

**program** DKR\_5;

**Uses** CRT;

**type**

mas = **array** [1..1000] **of** integer;

**var**

m, out: text;

ch: integer;

**procedure** SortCount;

**var**

arr: **array**[-100..100] **of** integer; //arr[i] - количество чисел i в массиве

i, j, x: integer;

**begin**

Assign(m, 'c:\маленбкий файл.txt');

Reset(m);

Writeln('Исходный массив: ');

**for** i := -100 **to** 100 **do**

arr[i] := 0;

**while not** eof(m) **do**

**begin**

readln(m, x);

write(x, ' ');

inc(arr[x]);

**end**;

Assign(out, 'c:\output.txt');

ReWrite(out);

Writeln;

Writeln('Отсортированный массив: ');

**if** ch = 1 **then**

**begin**

**for** i := -100 **to** 100 **do**

**for** j := 1 **to** arr[i] **do**

**begin**

writeln(out, i);

print(i);

**end**;

**end**;

**if** ch = 2 **then**

**begin**

**for** i := 100 **downto** -100 **do**

**for** j := 1 **to** arr[i] **do**

**begin**

writeln(out, i);

print(i);

**end**;

**end**;

writeln;

writeln;

writeln('Отсортированные данные записаны в c:\output.txt');

close(out);

close(m);

**end**;

**procedure** Sliv(**var** a: mas; p, q: integer);

**var**

r, i, j, k: integer;

b: mas;

**begin**

r := (p + q) **div** 2;

i := p;

j := r + 1;

**for** k := p **to** q **do**

**if** (i <= r) **and** ((j > q) **or** (a[i] < a[j])) **then**

**begin**

b[k] := a[i];

i := i + 1;

**end**

**else**

**begin**

b[k] := a[j];

j := j + 1;

**end**;

**for** k := p **to** q **do**

a[k] := b[k];

**end**;

**procedure** Sort(**var** a: mas; p, q: integer);

**begin**

**if** p < q **then**

**begin**

Sort(a, p, (p + q) **div** 2);

Sort(a, (p + q) **div** 2 + 1, q);

Sliv(a, p, q);

**end**;

**end**;

**procedure** MergeSort;

**var**

a: mas;

n, i, x: integer;

m, inp: text;

**begin**

clrscr;

Writeln('Исходный массив: ');

Assign(m, 'c:\маленбкий файл.txt');

Reset(m);

n := 0;

**repeat**

readln(m, x);

inc(n);

**until** eof(m);

Reset(m);

**for** i := 1 **to** n **do**

**begin**

readln(m, a[i]);

print(a[i]);

**end**;

close(m);

writeln;

Sort(a, 1, N);

writeln('Отсортированный массив:');

Assign(out, 'c:\output.txt');

ReWrite(out);

**if** ch = 1 **then**

**begin**

**for** i := 1 **to** n **do**

**begin**

Writeln(out, a[i]);

print(a[i]);

**end**;

**end**;

**if** ch = 2 **then**

**begin**

**for** i := n **downto** 1 **do**

**begin**

Writeln(out, a[i]);

print(a[i]);

**end**;

**end**;

writeln;

writeln;

writeln('Отсортированные данные записаны в c:\output.txt');

close(out);

**end**;

**begin**

**repeat**

ClrScr;

Writeln('1. Сортировка подсчетом');

Writeln('2. Сортировка слиянием');

Writeln('3. Выход');

Write('Выберите пункт: ');

Readln(ch);

**case** ch **of**

1:

**begin**

clrscr;

Writeln('1. Отсортировать по возрастанию');

Writeln('2. Отсортировать по убыванию');

Write('Выберите пункт: ');

Readln(ch);

**case** ch **of**

1:

**begin**

ClrScr;

SortCount;

Readln;

**end**;

2:

**begin**;

ClrScr;

SortCount;

Readln;

**end**;

**end**;

**end**;

2:

**begin**

clrscr;

Writeln('1. Отсортировать по возрастанию');

Writeln('2. Отсортировать по убыванию');

Write('Выберите пункт: ');

Readln(ch);

**case** ch **of**

1:

**begin**

ClrScr;

MergeSort;

Readln;

**end**;

2:

**begin**;

ClrScr;

MergeSort;

Readln;

**end**;

**end**;

**end**;

3: **exit**;

**end**;

**until** ch = 3;

**end**.

**Результат выполнения программы**

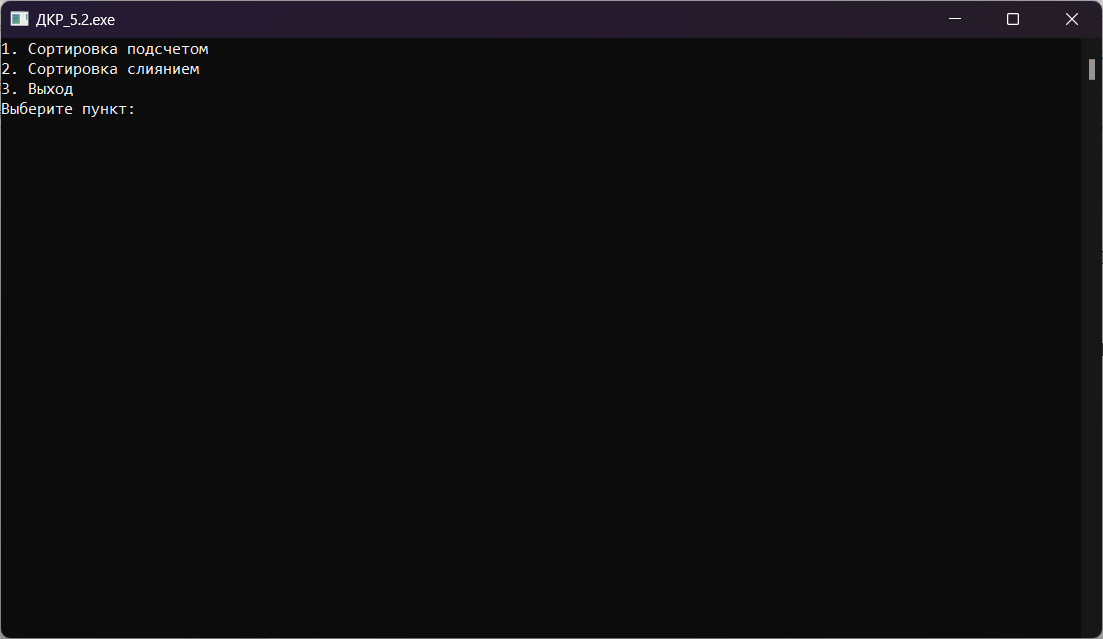


Рис. 6 – Меню выбора

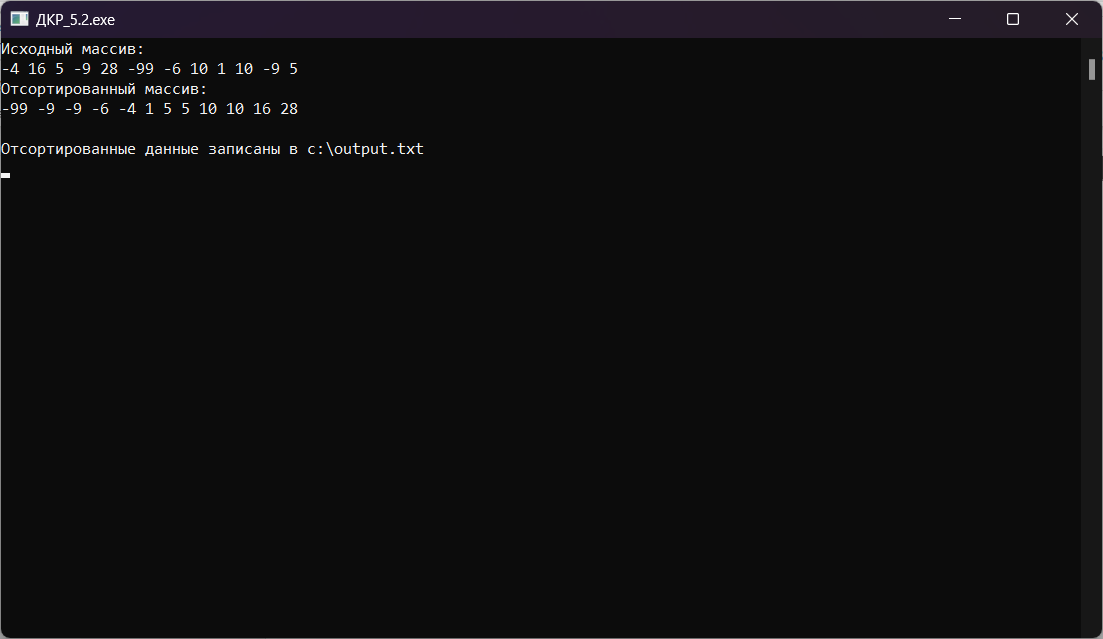


Рис. 7 – Результат сортировки

**Вывод**

В результате выполнения домашней контрольной работы поставленная цель была достигнута. Получены базовые сведения о наиболее известных алгоритмах сортировки, изучен принцип работы с текстовыми файлами.

В процессе работы было столкновение с небольшими сложностями, так, например, реализация компаратора была осуществлена не с помощью функций, а с помощью условия, для упрощения работы программы и сокращения кода. Тем не менее, программа работает так, как задумывалась, есть возможность выбора между сортировками, а также выбор: будет отсортирован массив по возрастанию или по убыванию.

Также, при работе с алгоритмом слияния, пришлось столкнуться с подобием рекурсии, потому что массив должен делиться на пополам, и сортироваться по тому же принципу, что вызвало небольшие сложности, которые были без проблем преодолены.

Таким образом, справившись со всеми сложностями, а также получив новые знания, были достигнута поставленная цель, выполнены задачи, получены удовлетворяющие результаты.