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»**

Выполнил: студент учебной группы

ИСПк-205-52-00

Шипунова Валерия Борисовна

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2023

1. Цель работы: получить базовые сведения о наиболее известных алгоритмах сортировки, изучить принципы работы с текстовыми файлами.

2. Формулировка задания (с вариантом)

Вариант: 21

1) Реализовать сортировку данных с помощью алгоритма выбором.

2) Реализовать сортировку данных с помощью пирамидального алгоритма.

3) В обоих случаях необходимо 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 тестовых файлов должен позволять оценить скорость работы программ).

3. Описание алгоритма

Сделаны подпрограммы для сортировки:

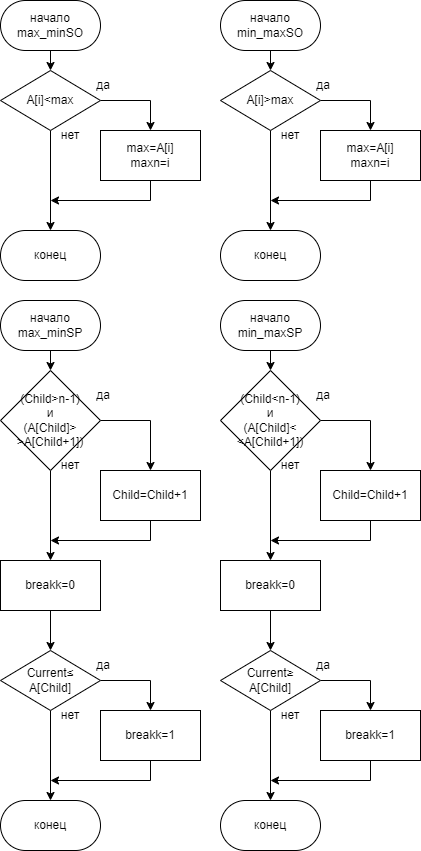
1. Методом выбора
2. Пирамидальным методом

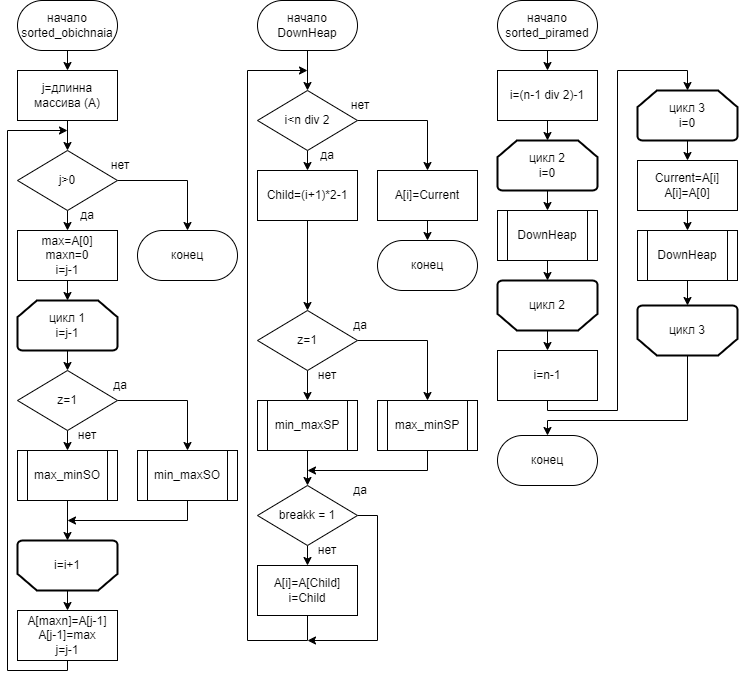
Для них были сделаны компараторы для сортировки от минимума к максимуму и наоборот.

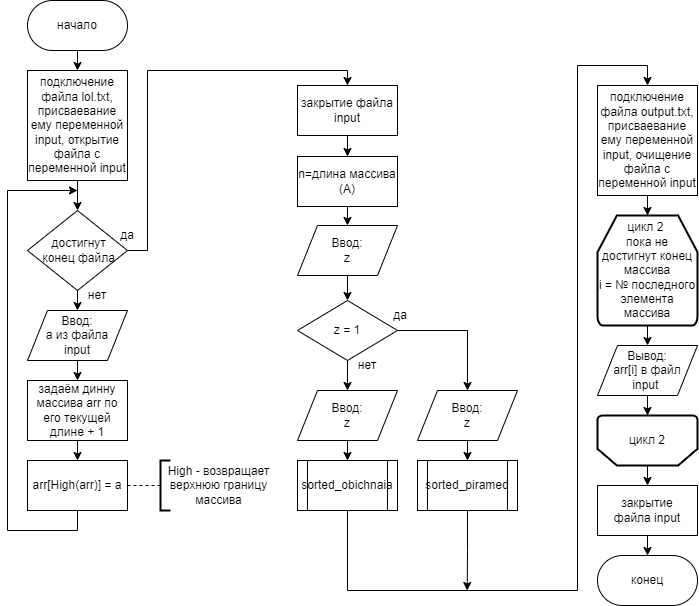
Затем было сделано case-меню для взаимодействия с пользователем.

Также были подключены файлы ввод и вывод.

4. Схема алгоритма с комментариями







5. Код программы

**program** zad\_1;

**var** A: **array of** Integer;

input,output: TextFile;

x,i,max,maxn,n,z: integer;

/// Реализовать сортировку данных с помощью алгоритма выбором

/// Реализовать сортировку данных с помощью пирамидального алгоритма

/// В обоих случаях необходимо предусмотреть возможность изменения компаратора

/// (реализация компаратора в виде передаваемой в подпрограмму функции)

/// Считывание и вывод данных необходимо производить из текстового файла

/// Для демонстрации работы программных реализаций самостоятельно подготовить

/// варианты входных данных (при этом объем тестовых файлов должен

/// позволять оценить скорость работы программ).

**procedure** max\_minSO(**var** max,maxn: integer; i: integer);

**begin**

// maxn - номер макс эл-та

**if** A[i]<max **then**

**begin**

max:=A[i];

maxn:=i

**end**;

**end**;

**procedure** min\_maxSO(**var** max,maxn: integer; i: integer);

**begin**

**if** A[i]>max **then**

**begin**

max:=A[i];

maxn:=i;

**end**;

**end**;

**procedure** max\_minSP(**var** Child,breakk: integer;Current: integer);

**begin**

**if** (Child>n-1) **and** (A[Child]>A[Child+1]) **then**// для смены последовательности заменить знаки

Child:=Child+1;

breakk:=0;

**if** Current<=A[Child] **then** breakk:=1;// и здесь

**end**;

**procedure** min\_maxSP(**var** Child,breakk: integer;Current: integer);

**begin**

**if** (Child<n-1) **and** (A[Child]<A[Child+1]) **then**// для смены последовательности заменить знаки

Child:=Child+1;

breakk:=0;

**if** Current>=A[Child] **then** breakk:=1;// и здесь

**end**;

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

**procedure** sorted\_obichnaia (**var** A: **array of** integer; z: integer);

**var** i,j,max,maxn: integer;

**begin**

j:=Length(A);

**while** j>0 **do**

**begin**

max:=A[0];

maxn:=0;

**for** i:=1 **to** j-1 **do**

**if** z=1 **then**

min\_maxSO(max,maxn,i)

**else**

max\_minSO(max,maxn,i);

A[maxn]:=A[j-1];

A[j-1]:=max;

j:=j-1;

**end**;

**end**;

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

**procedure** sorted\_piramed(**var** A: **array of** integer; n,z: integer);

**procedure** DownHeap(i,n: integer; Current: integer);

**var** Child,breakk: integer;

**begin**

// создание пирамиды // Current - Текущий

// Child - Ребенок

**while** i<n **div** 2 **do**

**begin**

Child:=(i+1)\*2-1;

**if** z=1 **then**

max\_minSP(Child,breakk,Current)

**else**

min\_maxSP(Child,breakk,Current);

**if** breakk = 1 **then**

**break**;

A[i]:=A[Child];

i:=Child;

**end**;

A[i]:=Current;

**end**;

**var** i: integer;

Current: integer;

**begin**

// сборка пирамиды

**for** i:=(n-1 **div** 2)-1 **downto** 0 **do**

DownHeap(i,n,A[i]);

// сортировка пирамиды

**for** i:=n-1 **downto** 0 **do**

**begin**

// перемещение верхушки в начало отсортированного списка

Current:=A[i];

A[i]:=A[0];

// нахождение нужного места в пирамиде для нового эл-та

DownHeap(0,i,Current);

**end**;

**end**;

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

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

**begin**

// AssignFile - подключение файла; Eof - конец файла

// SetLength - увеличивает длину массива; x - переменная

// Low - нижняя граница; High - верхняя граница массива

// Rewrite - запись и очищение файла

// файл

AssignFile(input,'lol.txt');

Reset(input);

**while not** Eof(input) **do**

**begin**

readln(input,x);

SetLength(A,Length(A)+1);

A[High(A)]:=x;

**end**;

writeln (A);

CloseFile(input);

writeln ('Введите 1 для sorted\_obichnaia, любое другое число для sorted\_piramed');

read (z);

n:=length(A);

**if** z=1 **then**

**begin**

writeln ('Введите 1 для max\_minSO, любое другое число для min\_maxSO');

read (z);

sorted\_obichnaia(A,z);

**end**

**else**

**begin**

writeln ('Введите 1 для max\_minSP, любое другое число для min\_maxSP');

read (z);

sorted\_piramed(A,n,z);

**end**;

AssignFile(input,'output.txt');

Rewrite(input);

**for** i:= Low(A) **to** High(A) **do**

writeln (input,A[i]);

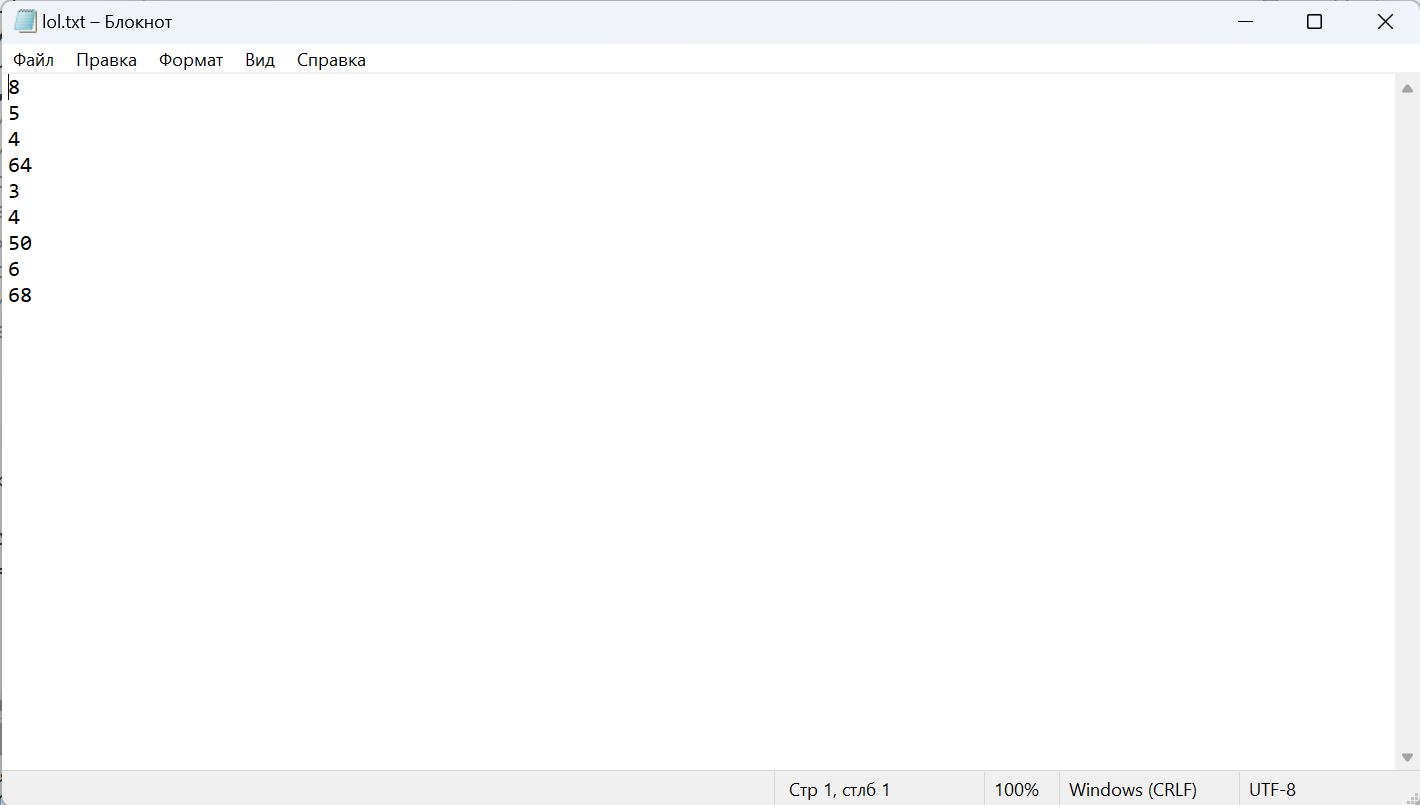
writeln (A);

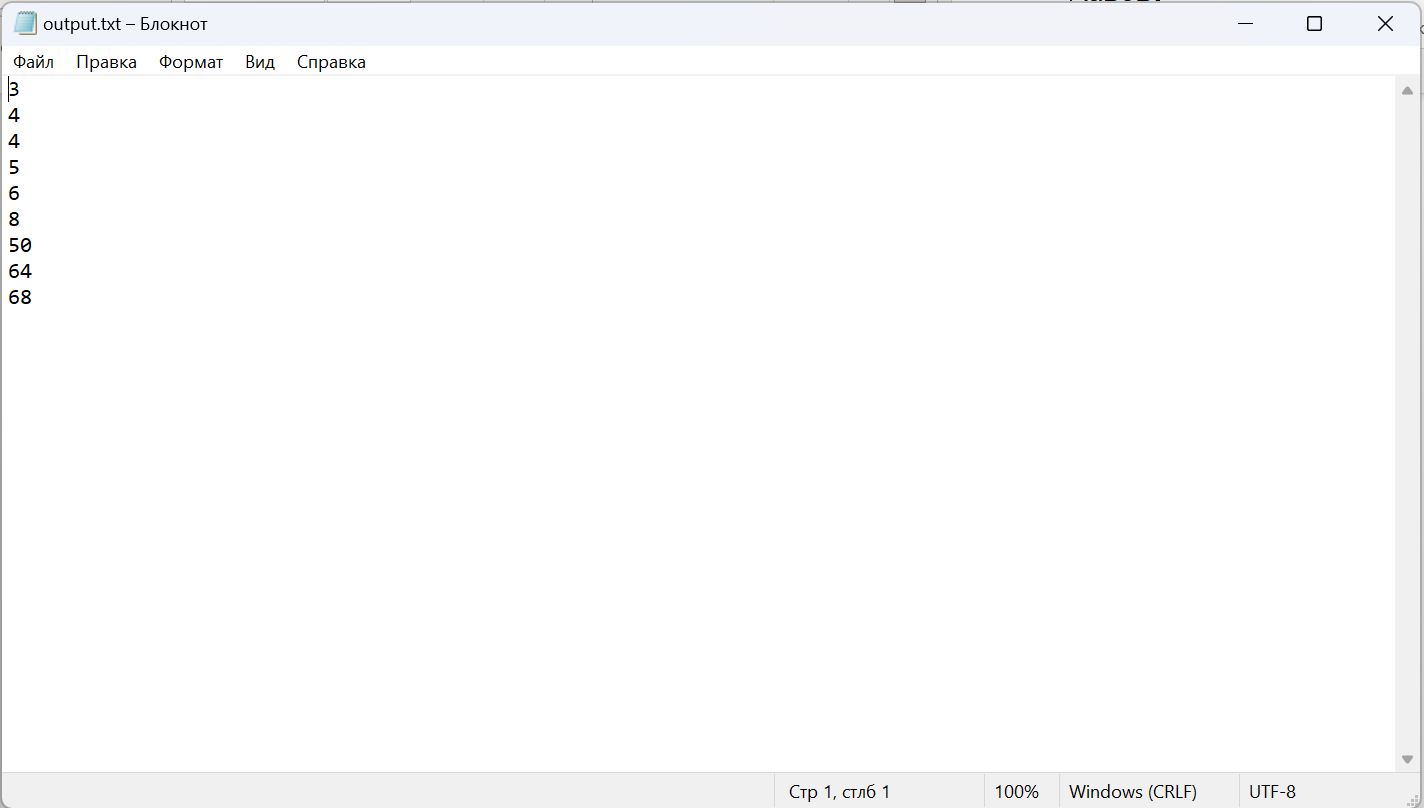
CloseFile(input);

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

**end**.

6. Результат выполнения программы





6. Вывод

Цель работы была достигнута.

Были изучены виды сортировки: сортировка выбором, пирамидальная сортировка, также их принципы и понятия.