**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: студент учебной группы

ИСПк-101-51-00

Югрин Константин Дмитриевич

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2023

**Цель работы**

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

**Формулировка задания**

Вариант 14

1. Реализовать сортировку данных с помощью "пузырькового" алгоритма.

2. Реализовать сортировку данных с помощью быстрого 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).

**Описание алгоритма**

Данный алгоритм позволяет выбрать один из 3 массивов или ввести его самостоятельно. Далее из двух предложенных вариантов он нужно выбрать способ сортировки.

**Пузырьковый алгоритм с примером.**

Алгоритм пузырьковой сортировки работает путем многократной замены соседних элементов, если они расположены в неправильном порядке. Вот пример, иллюстрирующий процесс:

Допустим, у нас есть массив [5, 2, 8, 3, 1].

На первом проходе алгоритм сравнивает первые два элемента (5 и 2) и меняет их местами, поскольку 2 меньше. Массив становится [2, 5, 8, 3, 1].

Затем он сравнивает следующие два элемента (5 и 8), и ему не нужно менять их местами, потому что они уже расположены в правильном порядке. Массив остается [2, 5, 8, 3, 1].

Алгоритм продолжает этот процесс, пока не достигнет конца массива. После нескольких проходов самый большой элемент "всплывает" в конец массива.

В итоге отсортированный массив будет иметь вид [1, 2, 3, 5, 8].

**Быстрая сортировка с примером.**

Алгоритм быстрой сортировки работает следующим образом:

1)Выберите один элемент массива в качестве начального.

2)Разделите массив на две части: элементы, меньшие начального элемента, и элементы, большие начального элемента.

3)Сделайте то же самое с 2 получившимися массивами. Продолжайте разделять массивы до тех пор, пока их размер не станет равным 1 или 0.

4)Объедините отсортированные массивы и начальный элемент в отсортированный массив.

Давайте рассмотрим пример работы алгоритма на массиве [7, 2, 1, 6, 8, 5, 3]:

Выбор начального элемента. Для простоты возьмем первый элемент массива, т.е. опорный элемент = 7.

Разделение массива. После разделения получим два массива:

[2, 1, 6, 5, 3, 4] (меньшие элементы) и [8] (больший элемент).

Повторяем шаги 1 и 2 к массиву [2, 1, 6, 5, 3, 4].

Выбор опорного элемента. Возьмем первый элемент массива то есть 2.

Разделение массива [2, 1, 6, 5, 3, 4]. Получим массивы:

[1] (меньшие элементы) и [6, 5, 3, 4] (большие элементы).

Повторяем шаги 1 и 2 к массиву [1]. Базовый случай рекурсии: массив имеет размер 1.

Теперь то же самое c массивом [6, 5, 3, 4].

Снова повторяем те же действия и в итоге.

Объединяем отсортированные массивы и начальные элементы.

[1] + [2] + [3, 4, 5, 6] + [7] + [8] = [1, 2, 3, 4, 5, 6, 7, 8].

Таким образом, получаем отсортированный массив [1, 2, 3, 4, 5, 6, 7, 8].

**Схема алгоритма с комментариями**

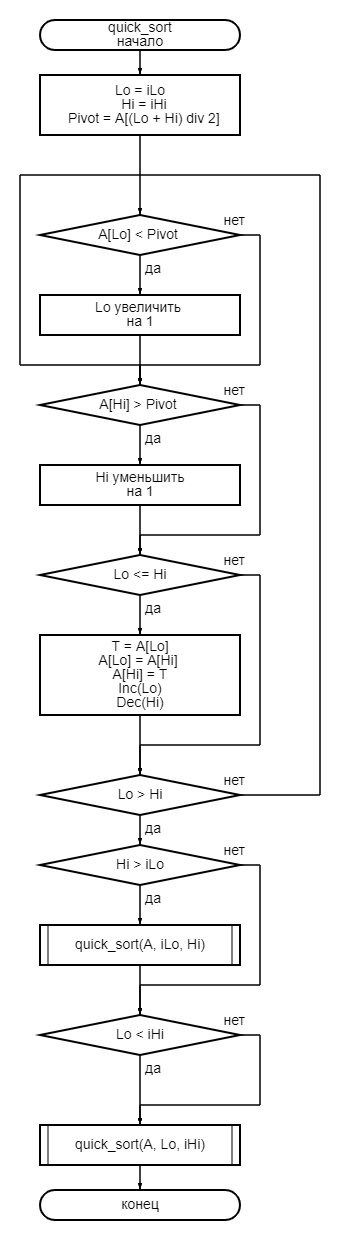


Рис. 1 «Схема алгоритма подпрограммы quick\_sort»

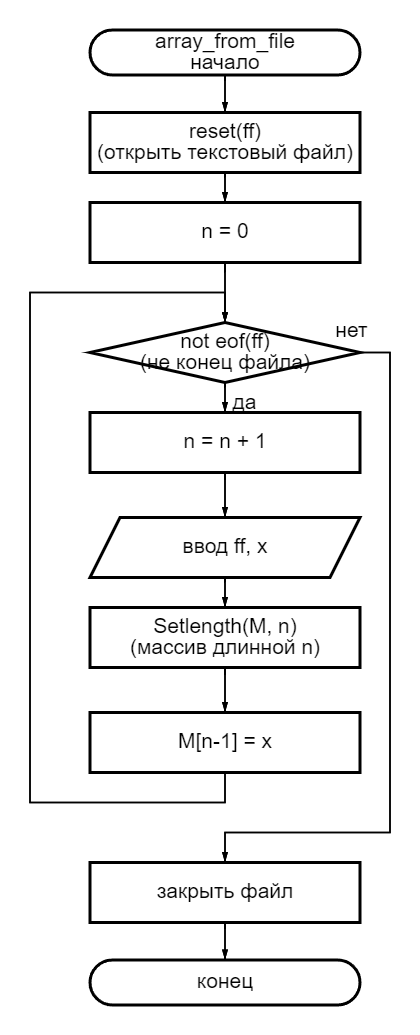
****

Рис. 2 «Схема алгоритма подпрограммы array\_from\_file»

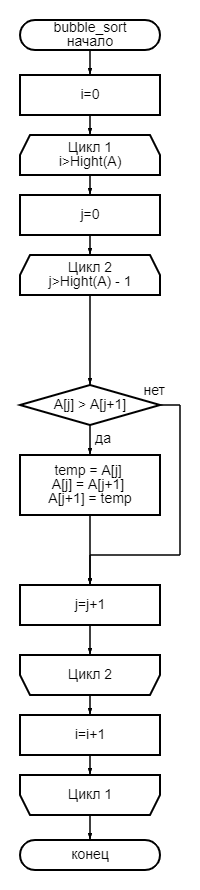


Рис. 3 «Схема алгоритма подпрограммы bubble\_sort»

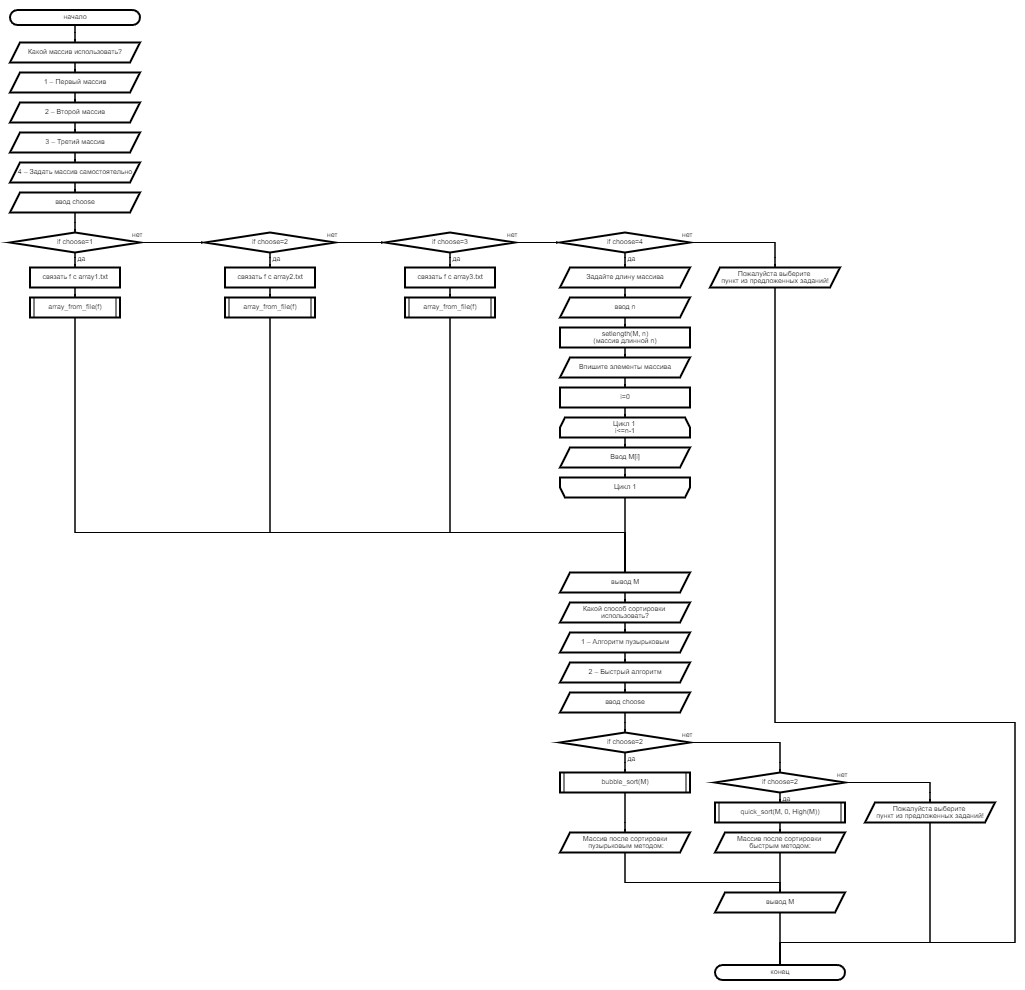
****

Рис. 4 «Схема основного алгоритма»

Рис. 4 «Схема алгоритма»

**Код программы**

**var**

choose, n, i, j, x : integer;

M : **array of** integer;

f : text;

**procedure** quick\_sort(**var** A: **array of** integer; iLo, iHi: integer);

**var**

Lo, Hi, Pivot, T: Integer;

**begin**

Lo := iLo;

Hi := iHi;

Pivot := A[(Lo + Hi) **div** 2];

**repeat**

**while** A[Lo] < Pivot **do** Inc(Lo);

**while** A[Hi] > Pivot **do** Dec(Hi);

**if** Lo <= Hi **then**

**begin**

T := A[Lo];

A[Lo] := A[Hi];

A[Hi] := T;

Inc(Lo);

Dec(Hi);

**end**;

**until** Lo > Hi;

**if** Hi > iLo **then** quick\_sort(A, iLo, Hi);

**if** Lo < iHi **then** quick\_sort(A, Lo, iHi);

**end**;

**procedure** array\_from\_file(ff : text);

**begin**

reset(ff);

n := 0; // Инициализация переменной "n"

**while not** Eof(ff) **do begin**

n := n + 1;

readln(ff,x);

Setlength(M, n);

M[n-1] := x;

**end**;

CloseFile(ff);

**end**;

**procedure** bubble\_sort(**var** A: **array of** integer);

**var**

i, j, temp : integer;

**begin**

**for** i := 0 **to** High(A) **do begin**

**for** j := 0 **to** High(A) - 1 **do begin**

**if** A[j] > A[j+1] **then begin**

temp := A[j];

A[j] := A[j+1];

A[j+1] := temp;

**end**;

**end**;

**end**;

**end**;

**begin**

Writeln('Какой массив использовать?');

Writeln('1 – Первый массив');

Writeln('2 – Второй массив');

Writeln('3 – Третий массив');

Writeln('4 – Задать массив самостоятельно');

readln(choose);

**case** choose **of**

1 : **begin** Assign(f, 'array1.txt'); array\_from\_file(f); **end**;

2 : **begin** Assign(f, 'array2.txt'); array\_from\_file(f); **end**;

3 : **begin** Assign(f, 'array3.txt'); array\_from\_file(f); **end**;

4 : **begin**

writeln('Задайте длину массива');

readln(n);

setlength(M, n);

writeln('Впишите элементы массива');

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

readln(M[i]);

**end**;

**else**

**begin**

writeln('Пожалуйста выберите пункт из предложенных заданий!');

Halt;

**end**;

**end**;

Writeln(M);

writeln('Какой способ сортировки использовать?');

writeln('1 – Алгоритм пузырьковым');

writeln('2 – Быстрый алгоритм');

readln(choose);

**case** choose **of**

1 : **begin** bubble\_sort(M); writeln('Массив после сортировки пузырьковым методом:'); writeln(M) **end**;

2 : **begin** quick\_sort(M, 0, High(M)); writeln('Массив после сортировки быстрым методом:'); Writeln(M) **end**;

**else**

**begin**

writeln('Пожалуйста выберите пункт из предложенных заданий!');

Halt;

**end**;

**end**;

**end**.

**Результат выполнения программы**

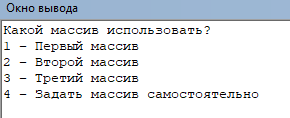


Рис. 5 «Результат выполнения программы»

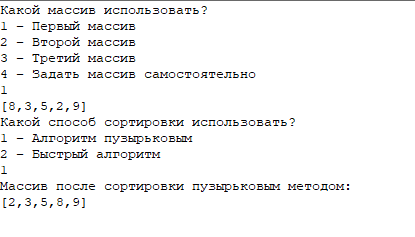


Рис. 6 «Результат выполнения процедуры bubble\_sort»

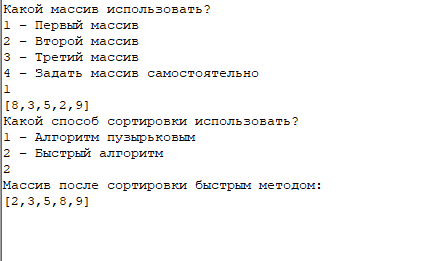


Рис. 7 «Результат выполнения процедуры quick\_sort»

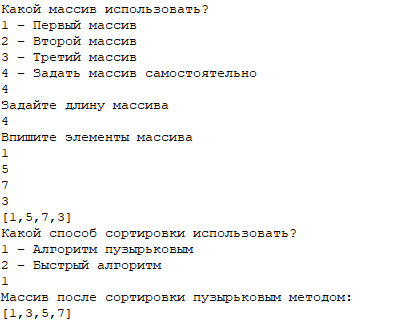
****

Рис. 8 «Результат выполнения choose = 4»

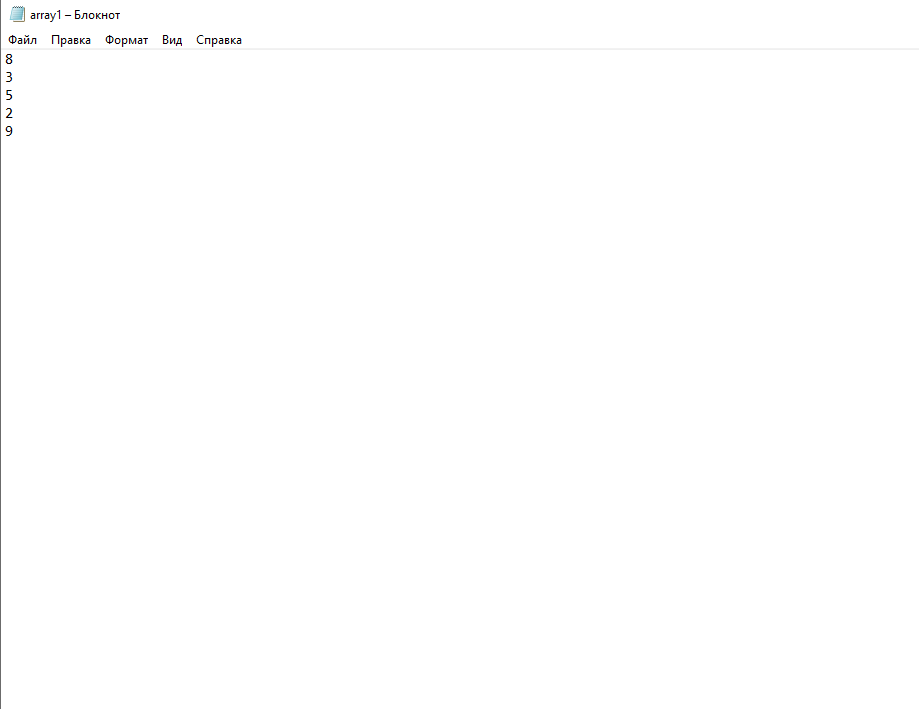
****

Рис. 9 «Файл с изначальным массивом»

**Вывод**

В процессе выполнения данной работы мы успешно применили изученный материал лабораторных занятий на практике, включая процедуры, условные операторы, циклы while, repeat, for, создание переменных и операторов ввода-вывода.

В результате выполнения данной работы мы получили базовые знания о наиболее известных алгоритмах сортировки, изучили принципы работы с текстовыми файлами и успешно выполнили данное нам практическое задание, применяя изученный материал на практике.