Министерство образования и науки РФ
Государственное образовательное учреждение
высшего профессионального образования

Тульский государственный университет

КАФЕДРА АВТОМАТИКИ И ТЕЛЕМЕХАНИКИ

ПОДПРОГРАММА-ПРОЦЕДУРА

Лабораторная работа № 1

по курсу «Программирование на ЯВУ»

Вариант № 4

Выполнил: студент группы 220601 _____ Белым А.А.
(подпись)

Проверил: _____ Сулимова В.В.
(подпись)

Тула 2011

Цель работы

Целью работы является изучение подпрограммам-процедур и написание программы с их использованием.

Задание

В заданной последовательности из N вещественных чисел определить, сколько чисел меньше данного K, равно K и больше K. Написать программу, решающую эту задачу с использованием подпрограмм-процедур.

Теоретическая справка

Схема алгоритма

На рисунке 1 представлена схема алгоритма ввода данных, анализа элементов массива относительно заданного числа и вывода результатов анализа.

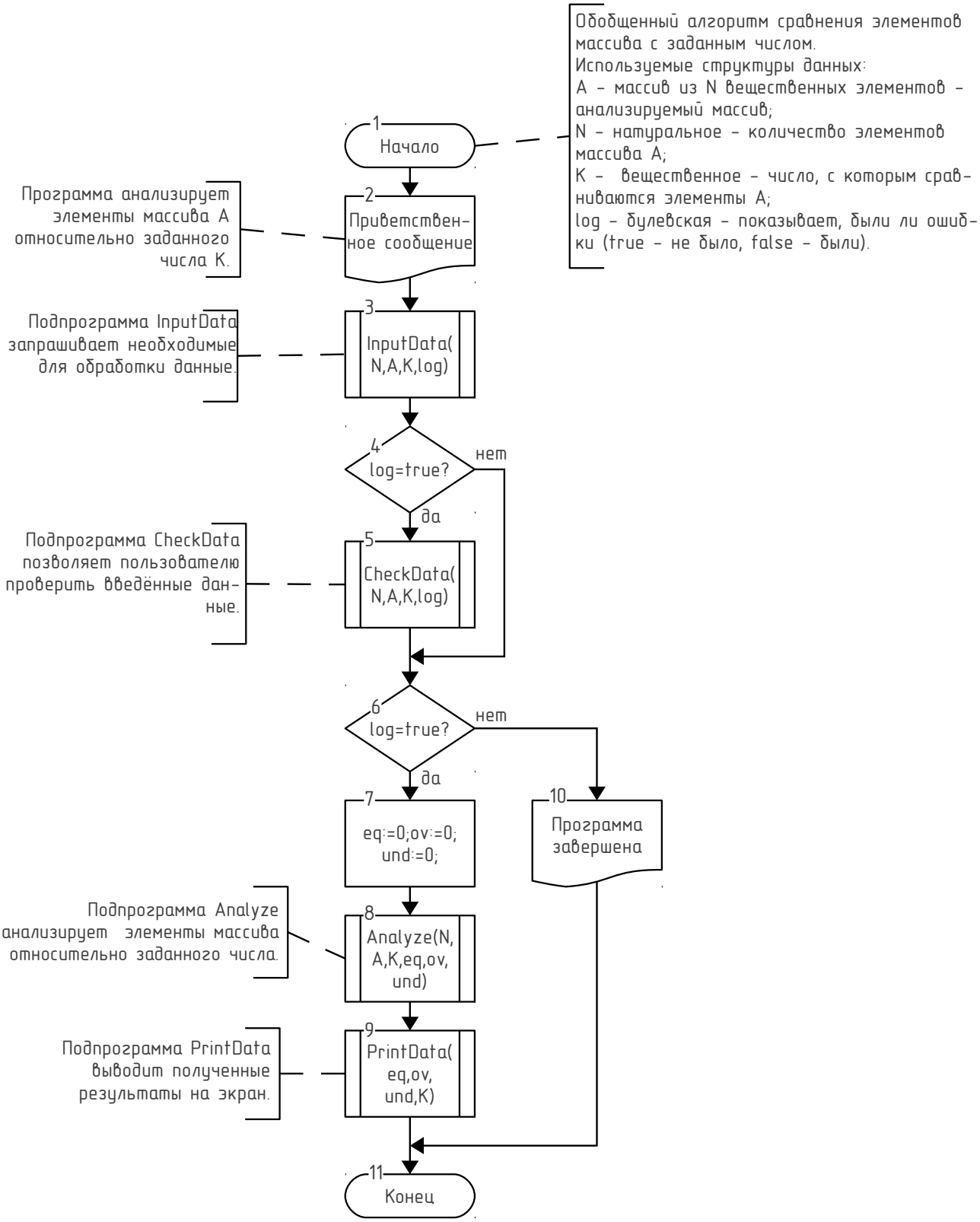

Рисунок 1 — Блок-схема обобщенного алгоритма анализа элементов векторов относительно заданного числа

На рисунках 2 и 3 представлена схема алгоритма ввода данных для анализа элементов массива относительно заданного числа.

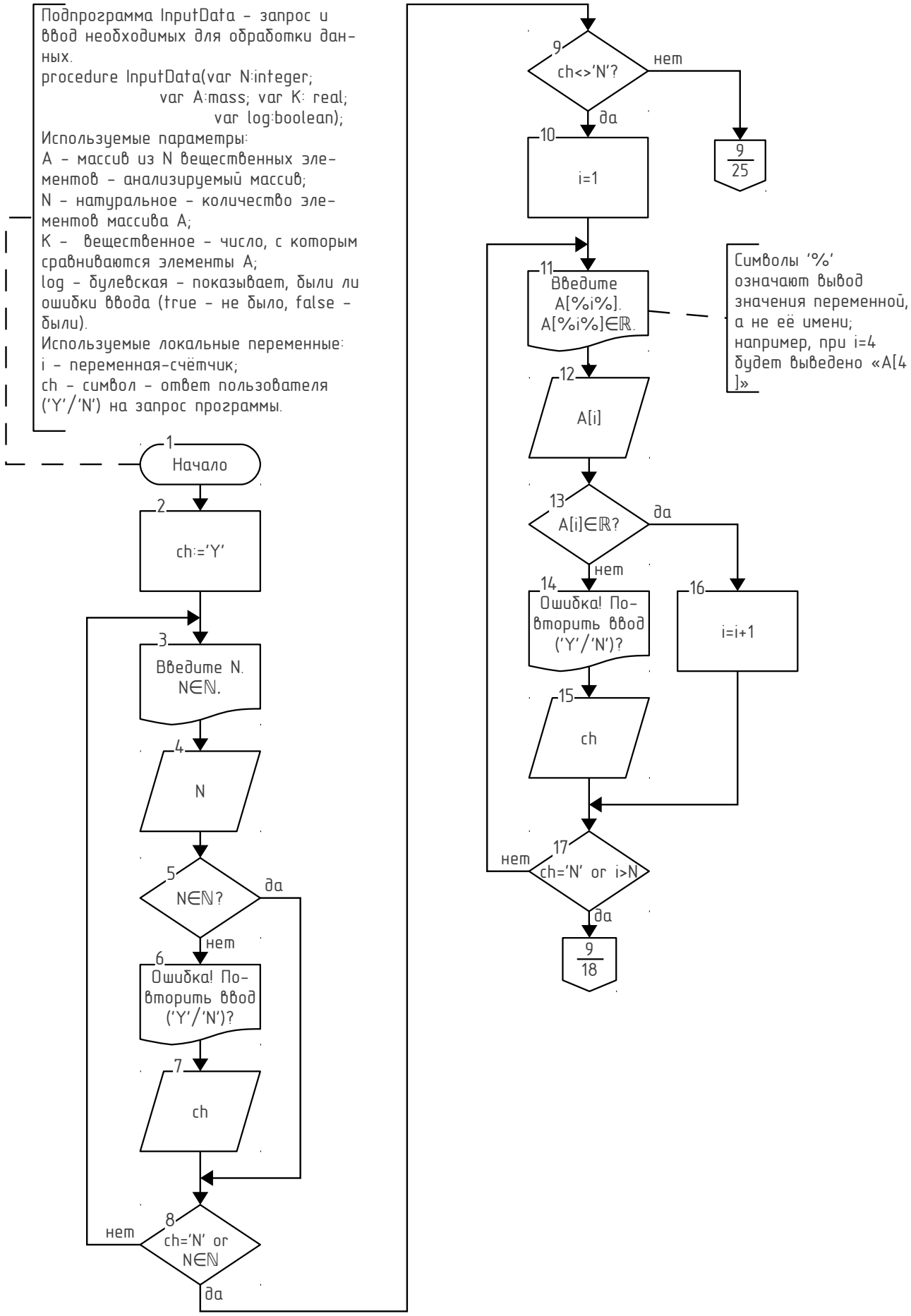

Рисунок 2 — Блок-схема алгоритма ввода данных для анализа массива(начало)

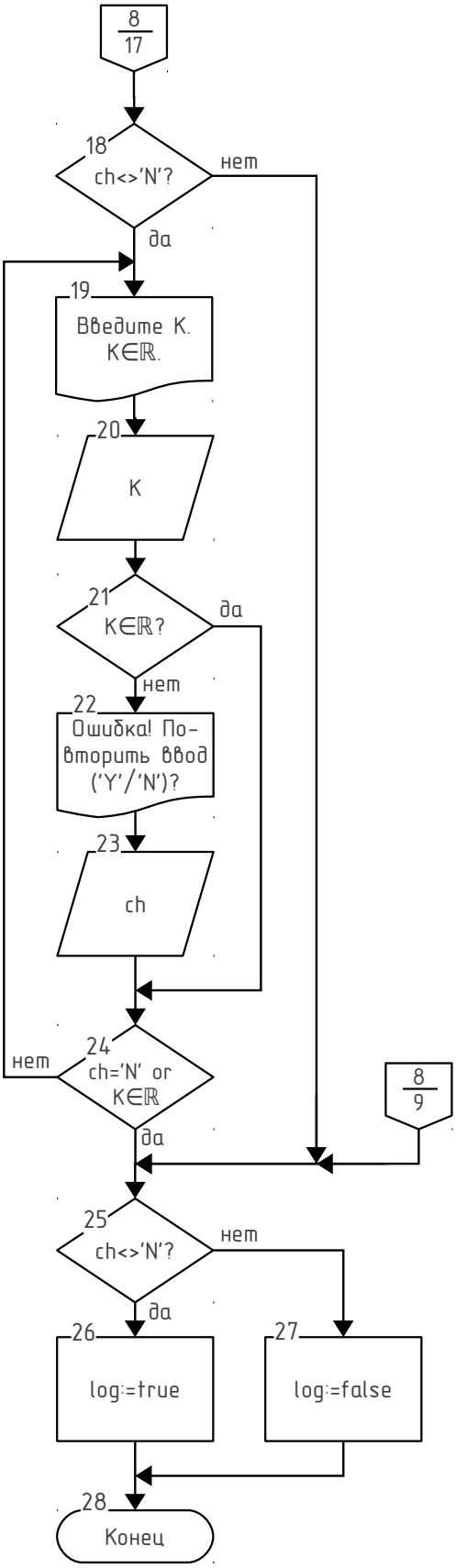

Рисунок 3 — Блок-схема алгоритма ввода данных для анализа массива(продолжение)

На рисунке 4 представлена схема алгоритма организации проверки пользователем введённых данных.

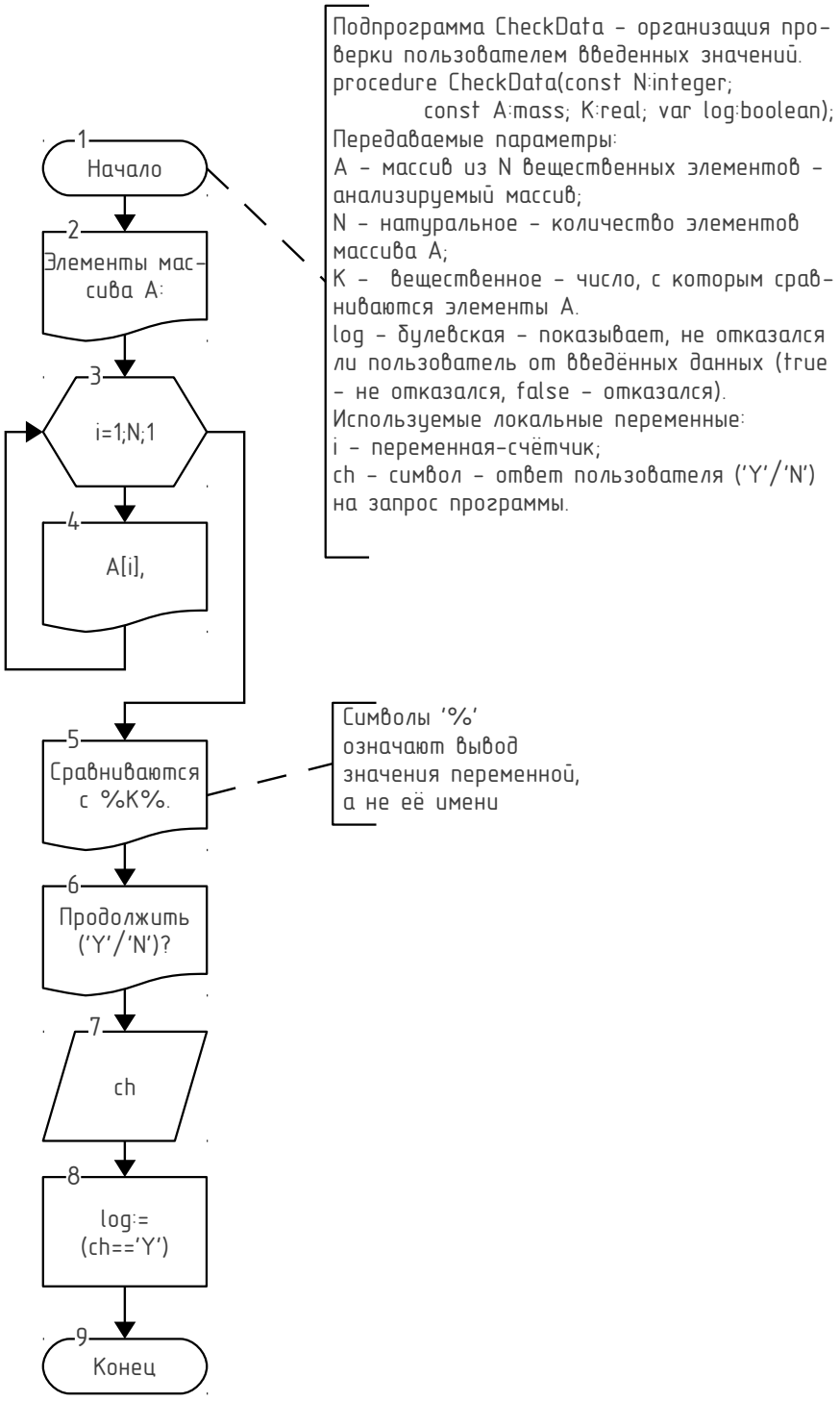

Рисунок 4 — Блок-схема алгоритма организации проверки пользователем введённых данных

На рисунке 5 представлена схема алгоритма анализа элементов массива относительно введенного числа.

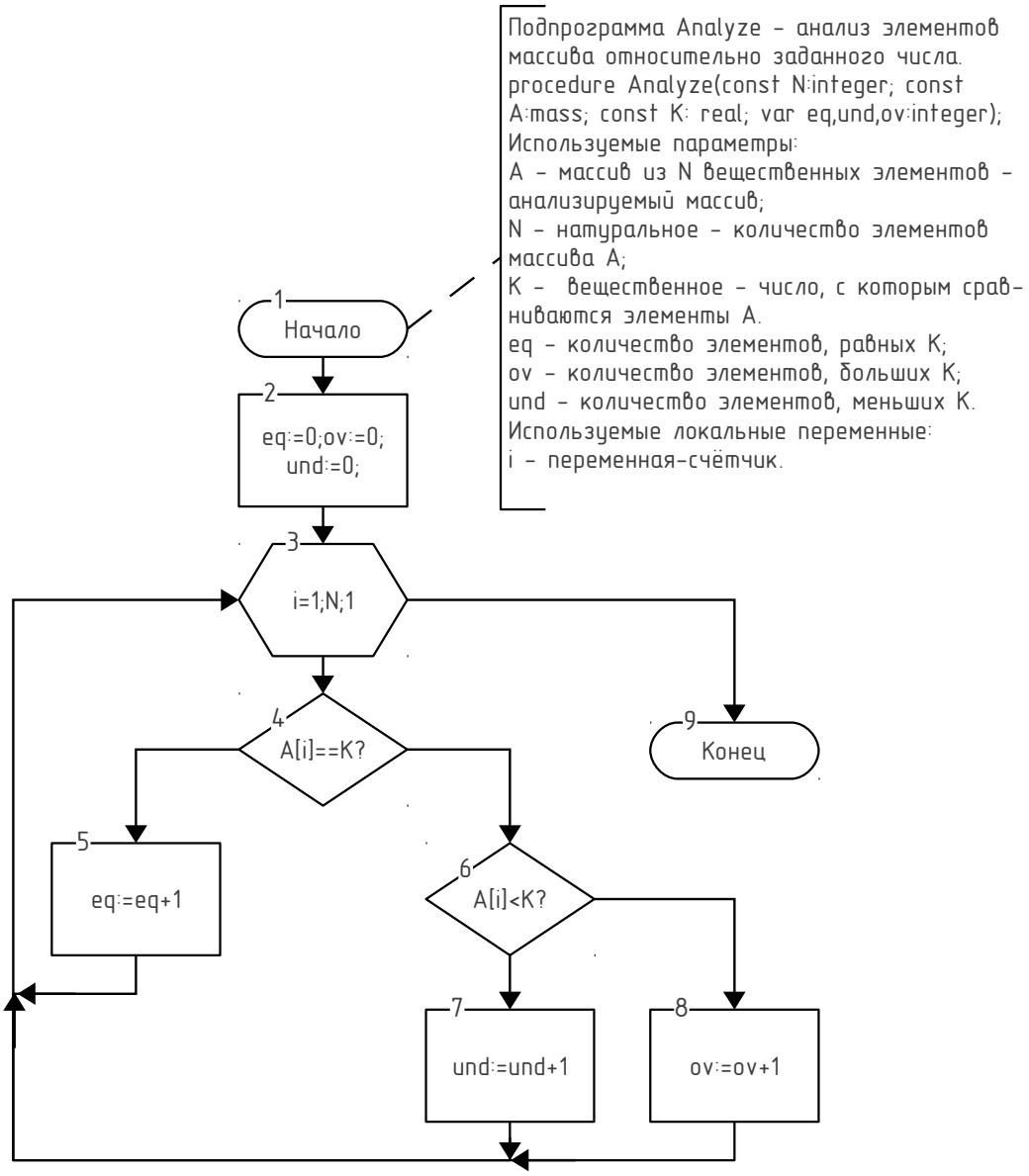

Рисунок 5 — Блок-схема алгоритма анализа элементов массива относительно введенного числа

На рисунке 6 представлена схема алгоритма вывода результатов анализа массива.

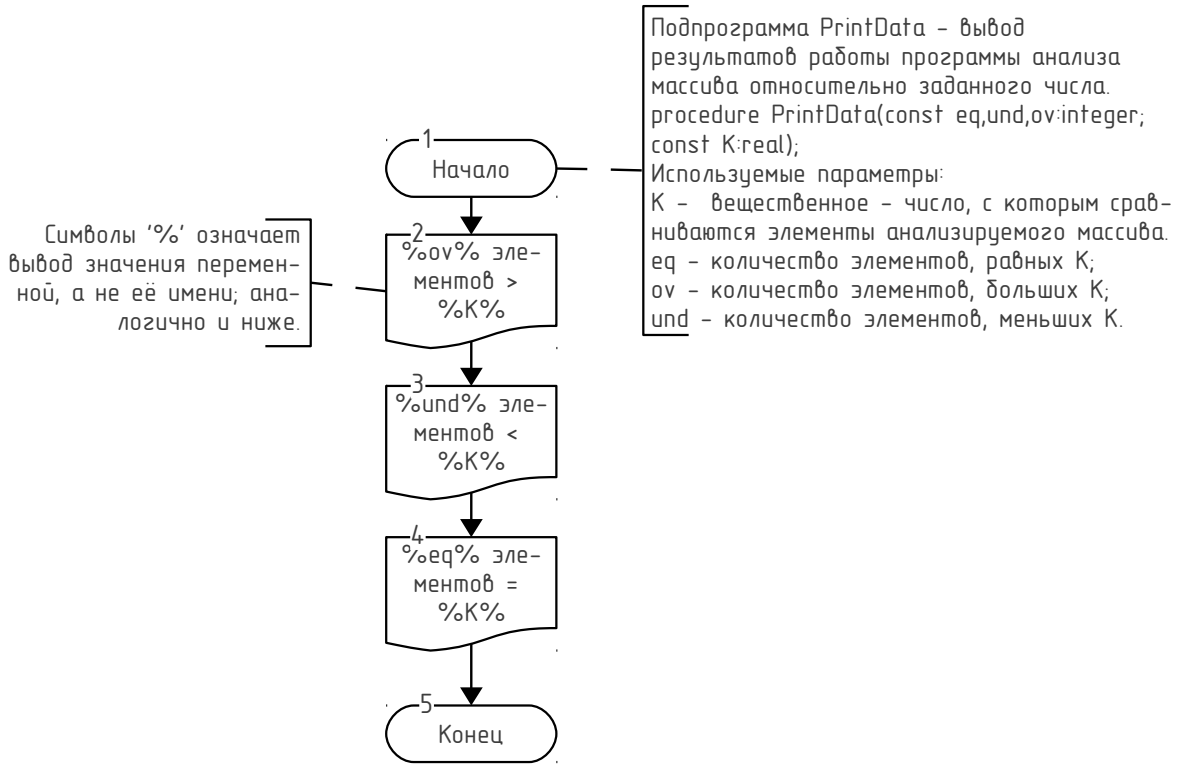

Рисунок 6 — Блок-схема алгоритма вывода результатов анализа массива

Инструкция пользователю

Данная программа позволяет узнать, сколько во введённом векторе элементов равно заданному числу, больше его и меньше этого числа.

Для работы программе необходимо передать количество элементов анализируемого вектора. Это натуральное число. Затем необходимо передать программе элементы введенного вектора. Это могут быть любые вещественные числа. После нужно указать вещественное число, с которым будут сравниваться данные элементы. Программа после ввода данных предлагает их проверить и предоставляет возможность отказаться от дальнейших вычислений. Для того, чтобы продолжить работу, нужно на запрос ответить 'y'.

Далее будут выведены три сообщения: сколько элементов вектора равно данному числу, больше его и меньше его. Чтобы покинуть программу, нужно нажать клавишу Enter.

Инструкция программисту

Для решения задачи поиска в массиве элементов, равных, больших и меньших данного числа были выполнены следующие действия.

В основной части программы определена костанта nn, равная 100, которая задет максимальный размер анализируемого массива.

Были введены структуры данных, описание которых представлено в таблице 1.

Таблица 1 - Структуры данных, используемые в в основной части программы оценки элементов массива

имя	тип	предназначение
N	longint	Размер вектора A.
A	array[1..nn] of real	Вектор A, элементы которого используются для оценки.
K	real	Число для оценки элементов вектора A.
log	boolean	Указывает на отсутствие ошибок (значение true) ввода данных или желания пользователя прервать программу.

Кроме того, для решения вышеуказанной задачи оценки элементов массива относительного заданного числа были определены следующие подпрограммы:

1. Процедура InputData вводит данные, необходимые для анализа элементов массива относительно заданного числа. Процедура также отслеживает ошибки ввода и уведомляет о них основную программу с помощью переменной log.

```
procedure InputData(var N:integer;var A:array of Real; var K: real;
                                    var log:boolean);
```

Используемые процедурой параметры-переменные приведены в таблице 2; локальные переменные - в таблице 3.

Таблица 2 - Параметры-переменные процедуры ввода данных, необходимых для анализа элементов массива относительно заданного числа

имя	тип	предназначение
N	longint	Размер вектора A.
A	array of real	Вектор A, элементы которого используются для оценки.
K	real	Число для оценки элементов вектора A.
log	boolean	Указывает на отсутствие ошибок (значение true) ввода данных

Таблица 3 - Локальные переменные процедуры ввода данных, необходимых для анализа элементов массива относительно заданного числа

имя	тип	предназначение
ch	char	Содержит ответ пользователя на запросы программы о повторении ввода данных.
i	integer	Переменная-счётчик в цикле, связанном с заполнением массива.
buf	strin	Буфер, который содержит значения в строковом формате на случай ошибочного ввода переменных.

2. Процедура CheckData - организация проверки пользователем введенных значений. Также позволяет уведомить с помощью переменной log основную программу об отказе пользователя продолжать работу.

```
procedure CheckData(const N:integer;const A:array of Real; const K: real;
                    var log:boolean);
```

Используемые процедурой параметры-константы приведены в таблице 4; параметры-переменные - в таблице 5; локальные переменные - в таблице 6.

Таблица 4 - Параметры-константы процедуры организации проверки пользователем введенных значений

имя	тип	предназначение
N	longint	Размер вектора A.
A	array of real	Вектор A, элементы которого используются для оценки.
K	real	Число для оценки элементов вектора A.

Таблица 5 - Параметры-переменные процедуры организации проверки пользователем введенных значений

имя	тип	предназначение
log	boolean	Указывает на желание пользователя продолжить работу программы(при значении true, false - завершение программы)

Таблица 6 - Локальные переменные процедуры организации проверки пользователем введенных значений

имя	тип	предназначение
ch	char	Содержит ответ пользователя на запрос программы о правильности данных.
i	integer	Переменная-счётчик в цикле, связанном с выводом массива.

3. Процедура Analyze - анализ элементов массива относительно заданного числа.

procedure Analyze(const N:integer;const A: array of Real;const K: real;

var eq,und,ov:integer);

Используемые процедурой параметры-константы приведены в таблице 7; параметры-переменные - в таблице 8; локальные переменные - в таблице 9.

Таблица 7 - Параметры-константы процедуры анализа элементов массива относительно заданного числа

имя	тип	предназначение
N	longint	Размер вектора A.
A	array of real	Вектор A, элементы которого используются для оценки.
K	real	Число для оценки элементов вектора A.

Таблица 8 - Параметры-переменные процедуры анализа элементов массива относительно заданного числа

имя	тип	предназначение
eq,ov,und	integer	Содержат, сколько элементов в векторе равно K, больше или меньше K.

Таблица 9 - Локальные переменные процедуры анализа элементов массива относительно заданного числа

имя	тип	предназначение
i	integer	Переменная-счётчик в цикле, связанном обработкой массива.

4. Процедура PrintData - вывод результатов работы программы анализа массива относительно заданного числа.

procedure PrintData(const eq,und,ov:integer; const K:real);

Используемые процедурой параметры-константы приведены в таблице 10.

Таблица 10 - Параметры-значения процедуры вывода результатов работы программы анализа массива относительно заданного числа

имя	тип	предназначение
eq,ov,und	integer	Содержат, сколько элементов в векторе равно K, больше или меньше K.
K	real	Число для оценки элементов вектора A.

Текст программы

Ниже представлен текст программы, оценивающей элементы массива относительно заданного числа, написанной на языке Turbo Pascal 7.

```
Program ArrayAnalyze;

{----------------ПРОЦЕДУРА ВВОДА ДАННЫХ-----------------}
{Параметры: кол-во элементов массива; массив для анализа; число для анализа;
            переменная состояния}
procedure InputData(var N:integer;var A:array of Real; var K: real;
                    var log:boolean);
var i:integer; buf:string; code:integer; ch:char;
begin
    ch:='Y';   {на случай, если пользователь ни разу не ошибётся}
    repeat
        WriteLn('Введите N - целое число. N>=1 и N<=',High(A)+1);
        Write('N:');ReadLn(buf); Val(buf,N,code);
        if code<>0 then begin   {n-не целое?}
            WriteLn('Недопустимое значение N.');
            WriteLn('Повторить ввод?(y/n)');
            ReadLn(ch);
        end
        else
          if (N<1) or (N>High(A)+1) then begin {N-не слишком маленькое или слишком
                                                большое?}
              code:=-1;
              WriteLn('N<1 или N>',High(A)+1,'!');
              WriteLn('Повторить ввод?(y/n)');
              ReadLn(ch);
          end;
    until (Upcase(ch)='N') or (code=0); {закончить, когда правильно, или
                                         пользователь отказался}

    if UpCase(ch)<>'N' then begin {если не отказался, то}
        i:=1;
        repeat
            WriteLn('Ввод вектора A. Введите элемент A[',i,'] - вещественное
                     число');
            Write('A[',i,']:');ReadLn(buf); Val(buf,A[i-1],code); {индекс первого
                                          элемента открытого массива равен 0}
            if code<>0 then begin {A[i] - не вещественное?}
                WriteLn('Недопустимое значение элемента.');
                WriteLn('Повторить ввод?(y/n)');
                ReadLn(ch);
```

```
            end else
                inc(i); code:=1;
        until (Upcase(ch)='N') or (i>N);{закончить, когда все правильные, или
                                пользователь отказался}
    end;

    if UpCase(ch)<>'N' then begin {если не отказался, то}

        repeat
            WriteLn('Введите K - вещественное число.');
            Write('K:');ReadLn(buf); Val(buf,K,code);
            if code<>0 then begin {K - не вещественное?}
                WriteLn('Недопустимое значение K.');
                WriteLn('Повторить ввод?(y/n)');
                ReadLn(ch);
            end;
        until (Upcase(ch)='N') or (code=0); {закончить, когда правильно, или
                                            пользователь отказался}
    end;
    log:=UpCase(ch)<>'N'; {не отказался ли пользователь?}
end;

{----------------ПРОЦЕДУРА ПРОВЕРКИ ДАННЫХ-----------------}
{Параметры: кол-во элементов массива; массив для анализа; число для анализа;
           переменная состояния}
procedure CheckData(const N:integer;const A:array of Real; const K: real;
                    var log:boolean);
var i:integer; ch:char;
begin
    WriteLn('Вектор A:');
    for i:=0 to N-1 do
        Write(A[i]:1:4,' ');
    WriteLn;
    WriteLn('Элементы A сравниваются с ',K:1:4);
    WriteLn('OK?(y/n)');
    ReadLn(ch);
    log:=UpCase(ch)='Y';{пользователь решил продолжить...}
end;

{----------------ПРОЦЕДУРА АНАЛИЗА МАССИВА-----------------}
{Параметры: кол-во элементов массива; массив для анализа; число для анализа;
          переменная состояния; кол-во элементов равных K; меньших K; больших K}
procedure Analyze(const N:integer;const A: array of Real;const K: real;
                  var eq,und,ov:integer);
```

```
var i:integer;
begin
    eq:=0;ov:=0;und:=0;
    for i:=0 to N-1 do {т.к. индексы открытого массива начинаются от 0}
        if A[i]=K then
            inc(eq)
        else
            if A[i]>K then
                inc(ov)
            else
                inc(und);
end;

{----------------ПРОЦЕДУРА ВЫВОДА РЕЗУЛЬТАТОВ НА ЭКРАН------------------}
{Параметры: кол-во элементов равных К; меньших К; больших К;
            К - число для анализа}
procedure PrintData(const eq,und,ov:integer;const K:real);
begin
    WriteLn('В векторе A ',eq,' элементов, равных ',K:1:4);
    WriteLn('В векторе A ',ov,' элементов, больших ',K:1:4);
    WriteLn('В векторе A ',und,' элементов, меньших ',K:1:4);
end;

{----------------ОСНОВНАЯ ЧАСТЬ------------------}
const nn=100; {макс. размер массива}
var N,eq,und,ov:integer;
{общее кол-во элементов; кол-во элементов равных К; меньших К; больших К}
    A:array [1..nn] of real; K:real;
{анализируемый массив; число для анализа}
    log:boolean;
{состояние программы(есть ошибки/нет ошибок}
BEGIN
WriteLn('Программа находит в векторе A длины N');
WriteLn('количество элементов A[i]>K, A[i]<K и A[i]=K.');
InputData(N,A,K,log); {вводим данные}
if log then {если все хорошо,то...}
    CheckData(N,A,K,log); {пусть пользователь их ещё проверит.}
if log then begin {и если опять всё хорошо}
    WriteLn;
    eq:=0;ov:=0;und:=0;
    Analyze(N,A,K,eq,ov,und); {анализируем массив}
    PrintData(eq,ov,und,K); {выводим везультаты}
end else
    WriteLn('Программа завершена');
```

```
WriteLn('Нажмите <Enter>...');
ReadLn;
END.
```

Тестовый пример

На рисунке 7 представлен результат работы программы оценки элементов векторов для вектора [1,2,3,4,5] и числа для оценки, равного 3.

```
Программа находит в векторе A длины N
количество элементов A[i]>K, A[i]<K и A[i]=K.
Введите N. N-целое, N>=1 и N<=5
N:t
Недопустимое значение N.
Повторить ввод?(y/n)
y
Введите N. N-целое, N>=1 и N<=5
N:5
Ввод вектора A. Введите элемент A[1] - вещественное число
A[1]:1
Ввод вектора A. Введите элемент A[2] - вещественное число
A[2]:2
Ввод вектора A. Введите элемент A[3] - вещественное число
A[3]:3
Ввод вектора A. Введите элемент A[4] - вещественное число
A[4]:4
Ввод вектора A. Введите элемент A[5] - вещественное число
A[5]:5
Введите K - вещественное число.
K:3
Вектор A:
1.0000 2.0000 3.0000 4.0000 5.0000
Элементы A сравниваются с 3.0000
OK?(y/n)
y

В векторе A 1 элементов, равных 3.0000
В векторе A 2 элементов, больших 3.0000
В векторе A 2 элементов, меньших 3.0000
Нажмите <Enter>...
```

Рисунок 7— Пример работы программы анализа вектора

Вывод

В ходе выполнения данной лабораторной работы я научился использовать подпрограммы-процедуры при написании программ. Процедуры позволяют организовать многократное использование частей программного кода, более оптимально использовать память и облегчают построение абстрактной модели решаемой задачи.