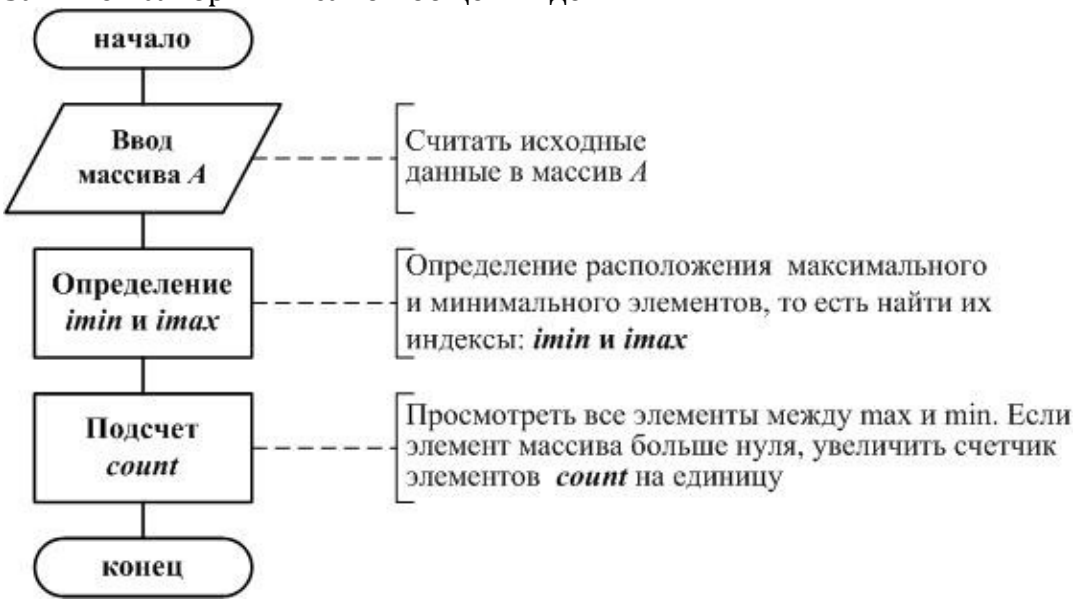
Задача 1. Написать программу, которая для 10 целочисленных элементов определяет, сколько положительных элементов располагается между максимальным и минимальным элементами.

Запишем алгоритм в самом общем виде:

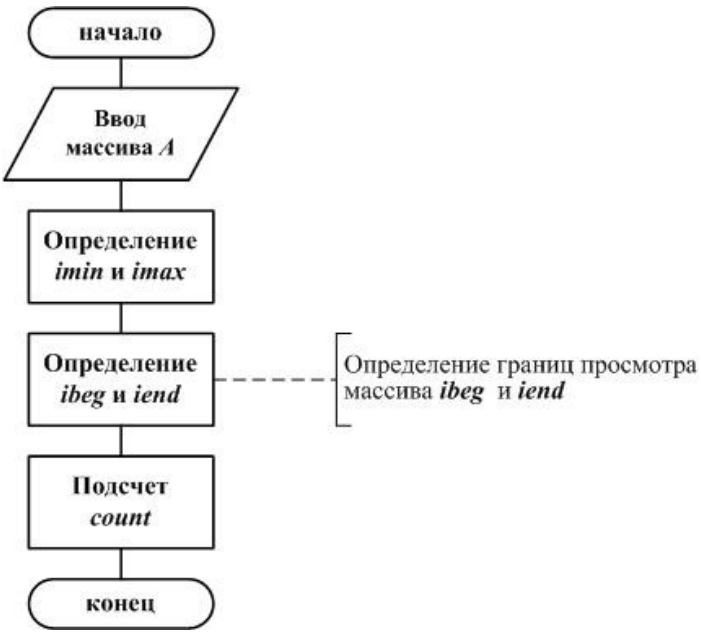


Перед написанием программы полезно составить тестовые примеры, чтобы более наглядно представить себе алгоритм. Ниже представлен массив из 10 чисел и обозначены искомые величины:

6	–8	15	9	–1	3	5	–10	12	2
		макс	+		+	+	мин		

Для этого примера программа должна вывести число 3.

Порядок расположения элементов в массиве заранее не известен — сначала может следовать как максимальный, так и минимальный элемент, более того, они могут совпадать. Поэтому прежде чем искать количество положительных элементов, требуется определить, какой из этих индексов больше, чтобы просматривать массив от меньшего номера к большему.



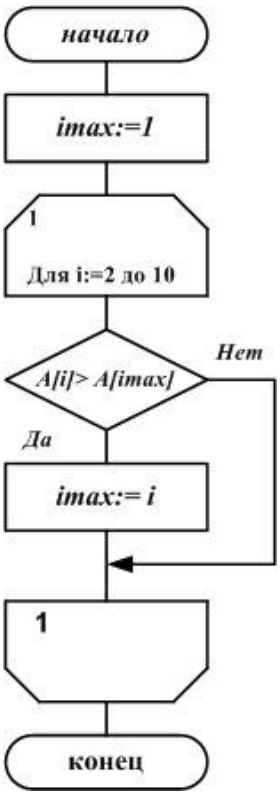
Рассмотрим подробно принцип поиска *максимального элемента* в массиве. Он весьма прост. Очевидно, что для его отыскания нужно сравнить между собой все элементы массива. Поскольку компьютер может сравнивать одновременно только два числа, элементы выбираются попарно.

Например, сначала первый элемент сравнивается со вторым, затем тот из них, который оказался больше — с третьим, тот, который оказался больше — с четвертым, и так далее до последнего элемента.

Иными словами, при каждом сравнении из двух чисел выбирается наибольшее. Поскольку его надо где-то хранить, в программе описывается переменная того же типа, что и элементы массива. После окончания просмотра массива в ней окажется самый большой элемент. Для того чтобы все элементы сравнивались единообразно, перед началом просмотра в эту переменную заносится первый элемент массива.
Сформулируем алгоритм поиска максимума:

1. Принять за максимальный первый элемент массива.
2. Просмотреть массив, начиная со второго элемента.
3. Если очередной элемент оказывается больше максимального, принять его за максимальный.

Можно просматривать массив не со второго, а с первого элемента. В этом случае за начальное значение максимума можно принять любой элемент массива.
Для решения поставленной задачи нам требуется знать не значение максимума, а его положение в массиве, то есть индекс.




```
program index_max_elem;
  const
    n = 10;
   var
     a : array [1..n] of integer; { массив }
     i : integer;      { номер текущего элемента }
     imax : integer:  {индекс максимального элемента }
 begin
    writeln('Введите ', n, ' элементов массива');
    for i  := 1 to n do read(a[i]);
    imax := 1;
    for i := 2 to n do
      if a[i] > a[imax]     then imax := i:
   writeln('Номер максимального элемента: ', imax)
   writeln('Максимальный элемент:    '. a[imax])
 end.
```

Индекс минимального элемента определяется аналогично.

Запишем уточненный алгоритм решения нашей задачи:

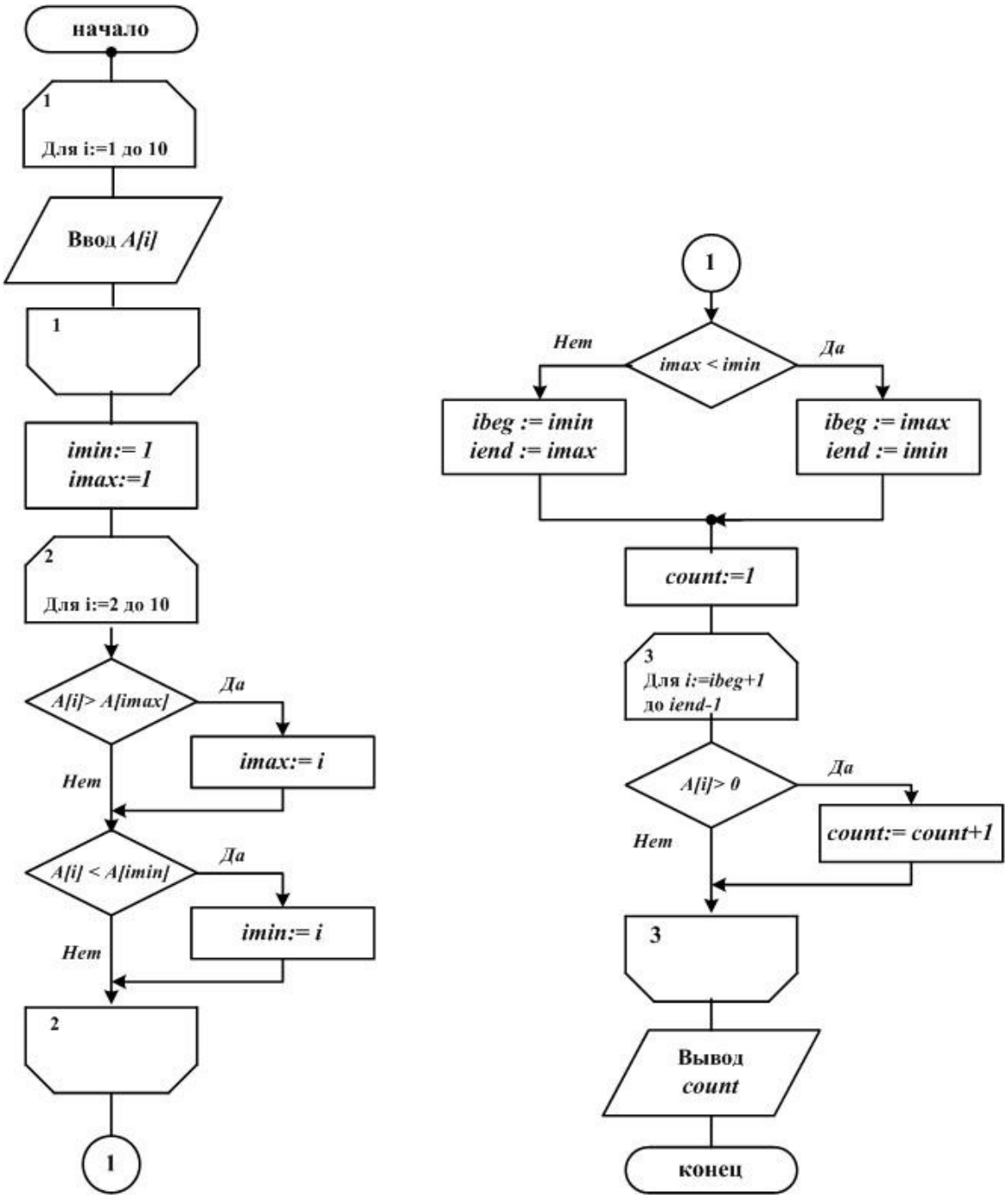
1. Определить, где в массиве расположены его максимальный и минимальный элементы:
   - задать начальные значения индексов искомых максимального и минимального элементов ***imax := 1; imin :=1***;
   - просмотреть массив, поочередно сравнивая каждый его элемент с ранее найденными максимумом и минимумом. Если очередной элемент больше ранее найденного максимума, принять этот элемент за максимум (то есть запомнить его индекс). Если очередной элемент меньше ранее найденного минимума, принять этот элемент за минимум.
2. Определить границы просмотра массива ***ibeg*** и ***iend*** для поиска положительных элементов, находящихся между его максимальным и минимальным элементами:
   - если максимум расположен в массиве раньше, чем минимум, принять левую границу просмотра равной индексу максимума, иначе — индексу минимума;

- если максимум расположен в массиве раньше, чем минимум, принять правую границу просмотра равной индексу минимума, иначе — индексу максимума.

3. Определить количество положительных элементов в найденном диапазоне:
   - обнулить счетчик положительных элементов: ***count:=0***;
   - просмотреть массив в указанном диапазоне. Если очередной элемент больше нуля, увеличить счетчик на единицу.

Массив просматривается начиная с элемента, следующего за максимальным (минимальным), до элемента, предшествующего минимальному (максимальному).

Схема разработанного алгоритма имеет следующий вид:



```
program num_positive_l;
 const
    n = 10;
  var
    a      : array [1 .. n] of integer;
    i      : integer:    { индекс текущего элемента }
    imax : integer;   { индекс максимального элемента }
    imin : integer;   { индекс минимального элемента }
    ibeg : integer;    { начало интервала }
    iend : integer;    { конец интервала }
    count : integer;  { количество положительных элементов }
  begin
    writeln('  Введите ',n:2, ' элементов массива:  ');
    for i := 1 to n do read(a[i]);          {ввод массива }
    imax ;= 1; imin := 1; {начальные значения номеров макс.и мин.эл-тов}
    for i := 2  to n do
      begin
        if a[i] > a[imax] then imax := i;  { новый номер максимума }
        if a[i] < a[imin] then  imin := i;  { новый номер минимума }
     end;
    writeln('max= '. a[imax]. ' min= '. a[imin]);  { отладочная печать }
    if imax < imin
       then begin
              ibeg := imax  { левая граница }
              iend := imin   { правая граница }
            end
      else begin
              ibeg := imin;  { левая граница }
              iend := imax; { правая граница }
          end;
    writeln('ibeg = ', ibeg, ' iend= ', iend);  { отладочная печать }
    count := 0:
    for i  := ibeg + 1 to iend-1 do  { подсчет количества положительных }
       if a[i] > 0 then inc(count);
    writeln('  Количество положительных:   '. count);
 end.
```

СОВЕТ. После нахождения каждой величины вставлена отладочная печать. Рекомендуется никогда не пренебрегать этим способом отладки и не жалеть времени, стараясь сделать эту печать максимально понятной, то есть содержащей необходимые пояснения и хорошо отформатированной. Кроме того, полезно выводить на печать исходные данные.

Тестовых примеров для этой задачи должно быть по крайней мере три — для случаев, когда:

- элемент *a[imin]* расположен левее элемента *a[imax]*;
- элемент *a[imin]* расположен правее элемента *a[imax]*;
- элементы *a[imin]* и *a[imax]* совпадают.

Последняя ситуация имеет место, когда в массиве все элементы имеют одно и то же значение. Желательно также проверить, как работает программа, если элементы *a[imin]* и *a[imax]* расположены рядом, а также в начале и в конце массива (граничные случаи). В массиве должны присутствовать как положительные, так и отрицательные элементы.