Алгоритмы поиска и сортировки массива.

1. Поиск элемента массива с максимальным значением.

Пусть значения элементов линейного массива **x** сформированы. Требуется среди чисел x[1], x[2], …, x[n] найти такое, что

x[j] = max(x[1], x[2], …, x[n]).

Если в задаче требуется найти порядковый номер этого элемента, то значит необходимо найти еще и значение индекса j.

Основная идея алгоритма состоит в том, что переменной **max** присваивается значение любого элемента массива (чаще всего первого по порядку). В случае нахождения порядкового номера переменной **ind** присваивается значение индекса этого элемента (т.е. 1).

Далее просматриваются все элементы массива, значения которых сравниваются со значением переменной **max**. Если окажется, что значение какого-либо элемента массива превосходит значение переменной **max**, то переменная **max** меняет свое значение на значение большего элемента. В случае отыскания порядкового номера переменная **ind** запоминает значение индекса большего элемента.

Если поиск наибольшего элемента идет в двумерном массиве, то необходимо просматривать поочередно все элементы каждой из строк. Для этого можно воспользоваться вложенными циклами:

```
max:=x[1,1];
indstr:=1;
indcol_____________________________________________________________
_____________________________________________________________
_____________________________________________________________
_____________________________________________________________
_____________________________________________________________
_____________________________________________________________
_____________________________________________________________
_____insrsid16196444    if x[i,j] > max then
          begin
             max:=x[i,j];
             indstr:=i;
             indcol:=j;
           end;
```

2. Методы сортировки массивов.

Задача сортировки (упорядочения) элементов массива в соответствии с их значением – классическая задача, исследование которой началось еще с момента появления первых ЭВМ. Создано много различных алгоритмов сортировки, однако задача разработки такого метода сортировки, который

был бы эффективен для массивов с любым количеством элементов, т.е. упорядочивал массив за наименьшее количество времени при минимальном объеме затрачиваемой памяти, не потеряла своей актуальности. В последнее время появилось достаточно много эффективных алгоритмов сортировки, которые базируются на принципах рекурсии, динамического программирования.

Рассмотрим несколько типов сортировки.

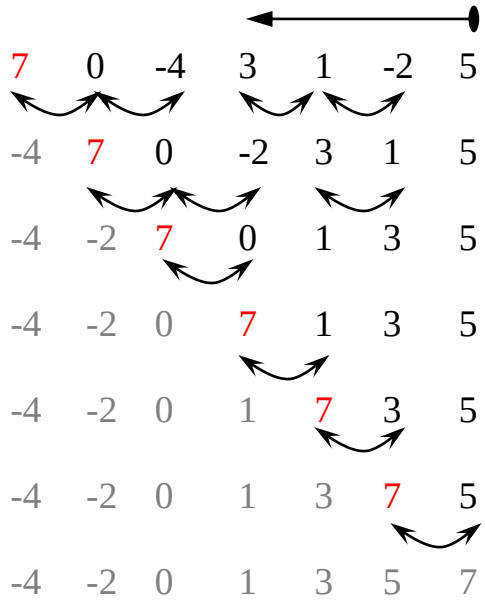
Договоримся для удобства, что в каждой задаче требуется упорядочить массив по возрастанию значений его элементов.

1. Метод «пузырька»

Идея метода:

весь массив просматривается несколько раз, причем при каждом просмотре сравниваются значения двух соседних элементов. Если эти значения следуют не в порядке возрастания, то производится их перестановка. Так происходит до тех пор, пока не будет сделано ни одной перестановки.

Этот метод называют «пузырьковой сортировкой» потому, что меньшие значения элементов массива постепенно «всплывают», как легкие пузырьки воздуха в воде, и перемещаются в начало массива, в то время, как б'ольшие значения «оседают на дно», т.е. перемещаются в конец массива.



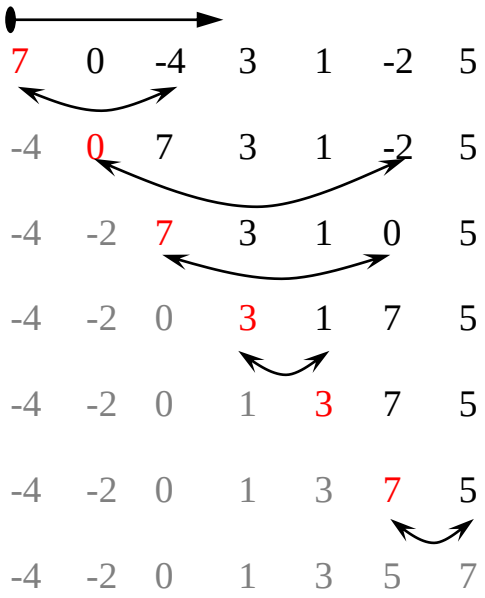

```
procedure float (k:integer; var t:mass);
   var i,j,h: integer;
begin
   for i:=2 to k do
     for j:=k downto i do
        if t[j]<t[j-1] then
           begin
              h:=t[j];
              t[j]:=t[j-1];
              t[j-1]:=h;
```

```
            end;
    end;
```

2. Метод простых обменов.

Идея метода:

весь массив просматривается несколько раз и при каждом просмотре ищется минимальный по значению элемент, который меняется местами с первым, вторым, третьим, …, предпоследним элементов массива и исключается из дальнейшего рассмотрения.

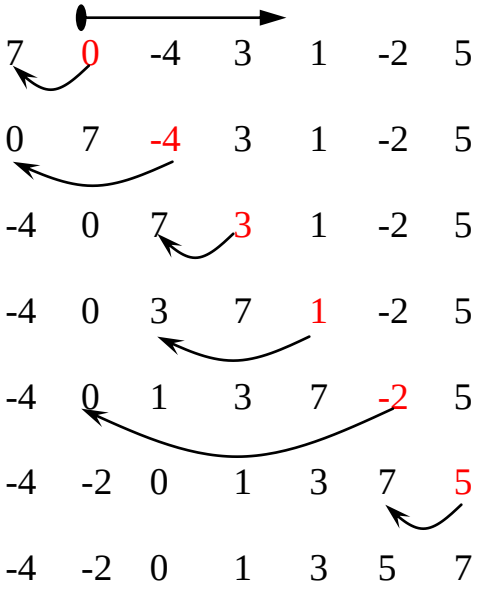



```
procedure obmen (k:integer; var t: mass);
   var i, j, min, ind, h: integer;
begin
   for i:=1 to k do
      begin
         min:=t[i];
         ind:=i;
         for j:=i+1 to k do
            if t[j] < min then
               begin
                  min:=t[j];
                  ind:=j;
               end;
         h:=t[i];
         t[i]:=t[ind];
         t[ind]:=h;
      end;
end;
```

3. Метод вставки и сдвига.

Идея метода:

делается предположение, что первые k элементов массива упорядочены, и если k+1-ый элемент меньше, чем какой-либо из первых k, то он записывается на «свое» место среди упорядоченных, т.е. меняется значением с тем из б’ольших, значение которого меньше, при этом «хвост» массива «сдвигается» к концу.




```
procedure vstavka (k:integer; var t:mass);
   var i,j: integer;
   procedure sdvig (p,g:integer; var tt:mass);
       var f,h:integer;
   begin
       f:=tt[p];
       for h:=p downto g+1 do
          tt[h]:=tt[h-1];
       tt[g]:=f;
   end;
begin
   for i:=2 to k do
       for j:=1 to i-1 do
          if t[i]<t[j] then sdvig(i,j,t);
end;
```