# Сортировка бинарными вставками.

Сортировка бинарными вставками является улучшением сортировки простыми вставками: поиск "подходящего места" в упорядоченной последовательности можно вести более экономичным способом, который называется *Двоичный поиск в упорядоченной последовательности*. Он напоминает детскую игру "больше-меньше": после каждого сравнения обрабатываемая последовательность сокращается в два раза.

Пусть, к примеру, нужно найти место для элемента **7** в таком массиве:

**[2 4 6 8 10 12 14 16 18]**

Найдем средний элемент этой последовательности (**10**) и сравним с ним семерку. После этого все, что больше **10** (да и саму десятку тоже), можно смело исключить из дальнейшего рассмотрения:

**[2 4 6 8] 10 12 14 16 18**

Снова возьмем середину в отмеченном куске последовательности, чтобы сравнить ее с семеркой.

Однако здесь нас поджидает небольшая проблема: точной середины у новой последовательности нет, поэтому нужно решить, который из двух центральных элементов станет этой "серединой". От того, к какому краю будет смещаться выбор в таких "симметричных" случаях, зависит окончательная реализация нашего алгоритма.

Новой "серединой" последовательности всегда будет становиться левый центральный элемент. Это соответствует вычислению номера "середины" по формуле

***nomer_sred:= (nomer_lev + nomer_prav) div 2***

Итак, отсечем половину последовательности:

**2 4 [6 8] 10 12 14 16 18**

И снова:

**2 4 6 [8] 10 12 14 16 18**

**2 4 6] [8 10 12 14 16 18**

Таким образом, мы нашли в исходной последовательности место, "подходящее" для нового элемента.

Если бы в той же самой последовательности нужно было найти позицию не для семерки, а для девятки, то последовательность границ рассматриваемых промежутков была бы такой:

**[2 4 6 8] 10 12 14 16 18**

**2 4 [6 8] 10 12 14 16 18**

**2 4 6 [8] 10 12 14 16 18**

**2 4 6 8] [10 12 14 16 18**

Из приведенных примеров уже видно, что поиск ведется до тех пор, пока левая граница не окажется правее **(!)** правой границы.

Кроме того, по завершении этого поиска последней левой границей окажется как раз тот элемент, на котором необходимо закончить сдвиг "хвоста" последовательности.

**Фрагмент кода** (сортировка по возрастанию массива a размера n):

```
{сортировка бинарными вставками}
for i:=2 to n do
  begin
   x:=a[i];{запомним элемент}
   l:=1;{левый край}
   r:=i-1;{правый}
   while l<=r do {пока левый не больше правого}
    begin
     m:=(l+r) div 2;{находим середину}
     if x<a[m] then r:=m-1{если элемент меньше среднего,
                           правый край левее середины}
     else l:=m+1{иначе левый правее середины}
    end;
   for j:=i-1 downto l do a[j+1]:=a[j];{сдвигаем массив вправо на 1}
   a[l]:=x{вставляем элемент на место}
  end; { Окончание алгоритма сортировки}
```

**Задание на пару:**

1. Отсортировать элементы массива по возрастанию и по убыванию.
2. Если в массиве есть число 5, то выполнить сортировку его элементов по убыванию, иначе ничего не делать.
3. Если сумма элементов массива больше 60, то отсортировать элементы массива по возрастанию.

**Подготовиться к контрольной работе**: уметь писать программы, в которых сортировка происходит, если выполняется некоторое условие.