**Insertion Sort:**

This is an in-place comparison based sorting algorithm. Here, a sub-list is maintained which is always sorted. For example, the lower part of an array is maintained to be sorted. An element which is to be 'insert'ed in this sorted sub-list, has to find its appropriate place and insert it there. Hence the name **insertion sort**.

The array is searched sequentially and unsorted items are moved and inserted into sorted sub-list (in the same array). This algorithm is not suitable for large data sets as its average and worst case complexity are of Ο(n2) where n is no. of items.

How insertion sort works?

We take an unsorted array first.

Insertion sort compares the first two elements.

JAVA Implementation

public static void InsertionSort( int [ ] num)  
{  
     int j;                     // the number of items sorted so far  
     int key;                // the item to be inserted  
     int i;    
  
     for (j = 1; j < num.length; j++)    // Start with 1 (not 0)  
    {  
           key = num[ j ];  
           for(i = j - 1; (i >= 0) && (num[ i ] < key); i--)   // Smaller values are moving up  
          {  
                 num[ i+1 ] = num[ i ];  
          }  
         num[ i+1 ] = key;    // Put the key in its proper location  
     }  
}