# Algorithms

Solve any problem to achieve a rank

View Leaderboard

Topics: Insertion Sort

## Insertion Sort

TUTORIAL PROBLEMS VISUALIZER BETA

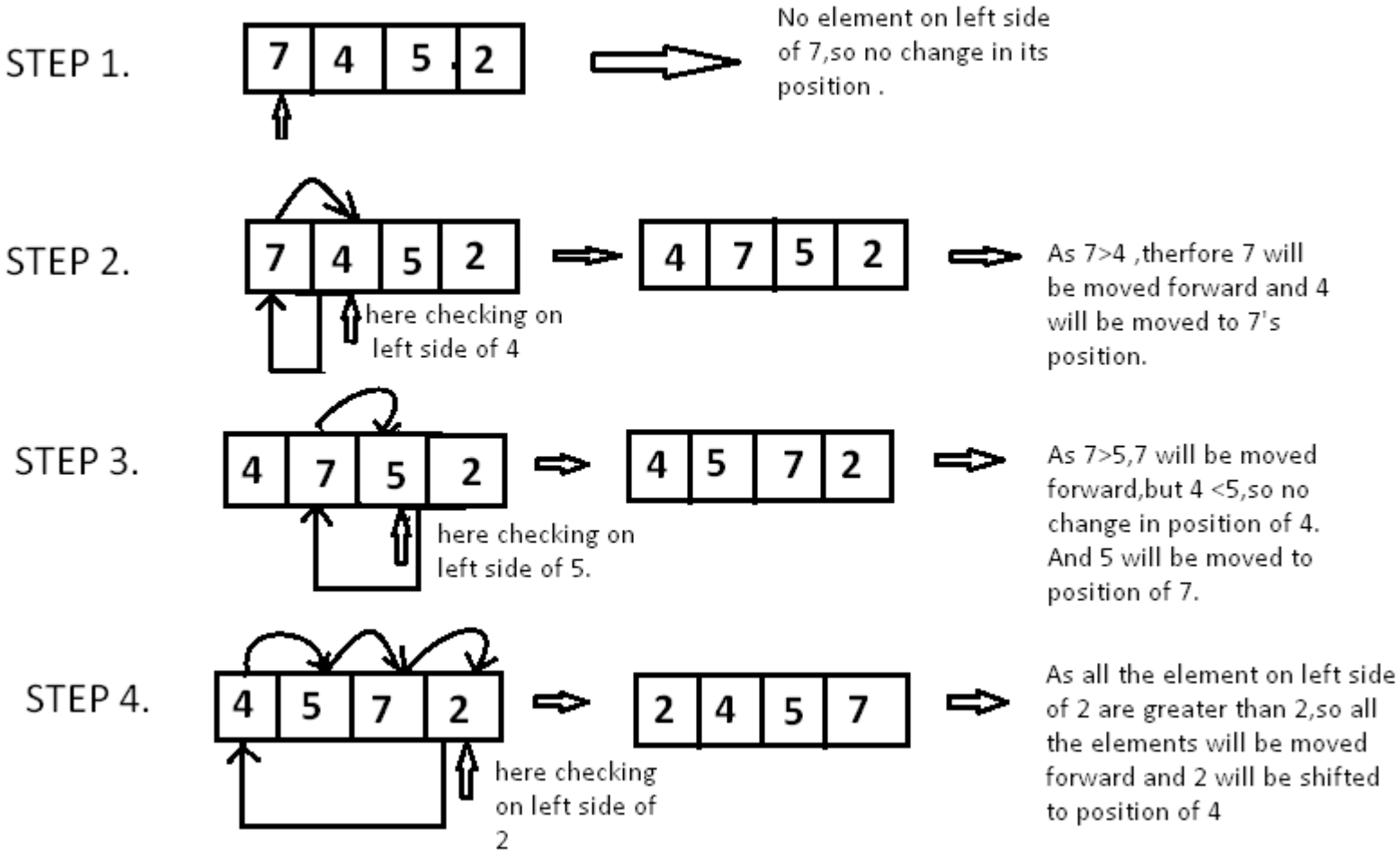
Insertion sort is based on the idea that one element from the input elements is consumed in each iteration to find its correct position i.e, the position to which it belongs in a sorted array.

It iterates the input elements by growing the sorted array at each iteration. It compares the current element with the largest value in the sorted array. If the current element is greater, then it leaves the element in its place and moves on to the next element else it finds its correct position in the sorted array and moves it to that position. This is done by shifting all the elements, which are larger than the current element, in the sorted array to one position ahead

**Implementation**

```
void insertion_sort ( int A[ ] , int n)
{
      for( int i = 0 ;i < n ; i++ ) {
    /*storing current element whose left side is checked for its
              correct position .*/

       int temp = A[ i ];
       int j = i;

        /* check whether the adjacent element in left side is greater or
             less than the current element. */

           while(  j > 0  && temp < A[ j -1]) {

            // moving the left side element to one position forward.
                 A[ j ] = A[ j-1];
                 j= j - 1;

            }
          // moving current element to its  correct position.
            A[ j ] = temp;
      }
}
```

Take array $A[] = [7, 4, 5, 2]$.




Since $7$ is the first element has no other element to be compared with, it remains at its position. Now when on moving towards $4$, $7$ is the largest element in the sorted list and greater than $4$. So, move $4$ to its correct position i.e. before $7$. Similarly with $5$, as $7$ (largest element in the sorted list) is greater than $5$, we will move $5$ to its correct position. Finally for $2$, all the elements on the left side of $2$ (sorted list) are moved one position forward as all are greater than $2$ and then $2$ is placed in the first position. Finally, the given array will result in a sorted array.

**Time Complexity:**

In worst case,each element is compared with all the other elements in the sorted array. For $N$ elements, there will be $N^2$ comparison Therefore, the time complexity is $O(N^2)$

*Contributed by: Anand Jaisingh*

Did you find this tutorial helpful? YES NO



**TEST YOUR UNDERSTANDING**

## Insertion Sort

You have been given an *A* array consisting of *N* integers. All the elements in this array are guaranteed to be unique. For each position *i* in the array *A* you need to find the position $A[i]$ should be present in, if the array was a sorted array. You need to find this for each *i* and print the resulting solution.

**Input Format**:

The first line contains a single integer *N* denoting the size of array *A*. The next line contains *N* space separated integers denoting the elements of array *A*.

**Output Format**:

Print *N* space separated integers on a single line , where the *i*th integer denotes the position of $A[i]$ if this array were sorted.

**Constraints**:

$1 \leq N \leq 100$

$1 \leq A[i] \leq 100$

SAMPLE INPUT

```
5
1 2 3 4 5
```

SAMPLE OUTPUT

```
1 2 3 4 5
```



Enter your code or Upload your code as file. Save Language

```
1  Loading...
```

1·1

*Press Ctrl/Command+Spacebar for autocomplete suggestions (accuracy dependent on connection stability).*

View all comments

7

LIVE EVENTS

+1-650-461-4192

contact@hackerearth.com



?