Insertion Sort – In this technique, the basic idea is to place an unsorted element into it’s correct position in a growing list of sorted data.

For e.g. while playing cards, if the player picks up a new card (single) from the pack of cards, then he places/inserts that card at proper position in between the other cards held in his hand. Similarly in insertion sort, when a new element is to be inserted in the list, it’s position has to be found out & the elements from this position have to be shifted by single position to the right to make a place for the new element.

0 1 2 3 4 5

For eg to sort elements 40 20 60 30 10 50

Step Sorted List Unsorted List

1 Empty 40 20 60 30 10 50

2 40 20 60 30 10 50

3 20 40 60 30 10 50

4 20 40 60 30 10 50

5 20 30 40 60 10 50

6 10 20 30 40 60 50

7 10 20 30 40 50 60 Empty

// Here we have 2 partitions (logical) in the same array

// 1. Sorted 2. Unsorted

// Sorting procedure will begin when there is atleast single

// element in Sorted list

// var i – To refer to elements from unsorted list

// var j – To refer to the elements from sorted list

i < noe

i < 6

for (i=1;i<=5;i++)  
{

key = nos[i]; // make the element of unsorted list

// safe

for (j=i-1;j >= 0;j--) // compare elements of sorted

// list R to L

{

if (key < nos[j])

nos[j+1] = nos[j];

else

break; // terminate the loop

}

nos[j+1] = key;

}

Case Study-1, to move 30 in sorted list

j j j i

0 1 2 3 4 5

20 ~~40~~ ~~60~~ ~~30~~ 10 50

30 40 60

key

30

Case Study-2, move 10 in sorted list

-1 0 1 2 3 4 5

j j j j j i

~~20~~ 10 ~~30~~ 20 ~~40~~ 30  ~~60~~ 40 ~~10~~ 60 50

key

10

Case Study – 3, move 50 in sorted list