06/09/2020

**Experiment No:3**

**INSERTION SORT**

**AIM:**

To perform insertion sort in an array and to arrange the elements of the

array in ascending order.

**DATA STRUCTURE USED:**

Array is the data structure used.

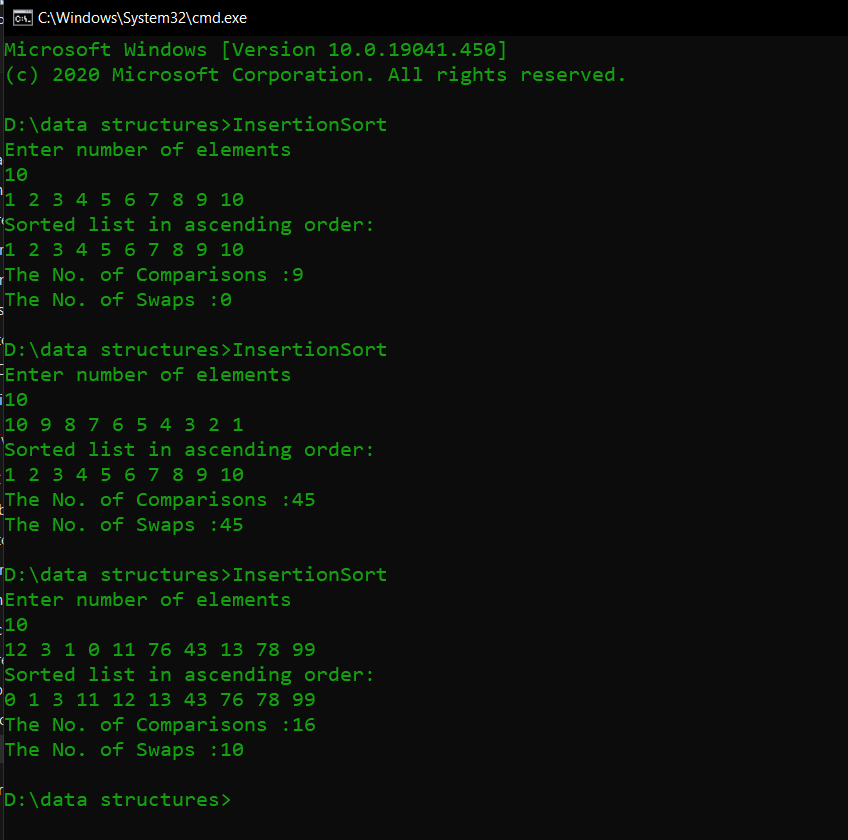
**ALGORITHM:**

1. Read the elements of the array arr[size]
2. for i=1 to size
3. x = arr[i]
4. for j= i-1 to 0
5. if arr[j]>x
6. arr[j+1]=x
7. else
8. break
9. end if
10. endfor
11. arr[i+1]=x
12. endfor

**PROGRAM:**

#include <stdio.h>  
void main(){  
 int n, arr[100], i, j, x,c=0,s=0, flag = 0;  
 printf("Enter number of elements\n");  
 scanf("%d", &n);  
  
 for (i = 0; i < n; i++){  
 scanf("%d", &arr[i]);  
 }  
  
  
 for (i = 1 ; i < n ; i++){  
 x = arr[i];  
 for (j = i - 1 ; j >= 0; j--){  
 c++;  
 if (arr[j] > x) {  
 s++;  
 arr[j+1] = arr[j];  
 flag = 1;  
 }  
 else  
 break;  
  
 }  
  
 if (flag==1)  
 arr[j+1] = x;  
 }  
  
 printf("Sorted list in ascending order:\n");  
  
 for (i = 0; i < n ; i++) {  
 printf("%d ", arr[i]);  
 }  
 printf("\nThe No. of Comparisons :%d\n", c);  
 printf("The No. of Swaps :%d\n", s);  
}

**OUTPUT:**



**RESULT:**

Insertion sort was performed in the array and the array elements were arranged in

ascending order. Also, the number of comparisons and swaps performed were found out. Number of comparisons performed was found to be (n-1) for best case and n(n-1)/2 for worst case where n is the number of array elements.

Time complexity:

Best case – O(n)

Average case – O(n2)

Worst case – O(n2)