**CODE:**

SELECTION SORT:

#include<stdio.h>

#include<conio.h>

void selection(int a[],int n)

{

int i,j,min,temp;

for(i=0;i<n-1;i++)

{

min=i;

for(j=i+1;j<n;j++)

{

if(a[min]>a[j])

min=j;

}

if(min!=i)

{

temp=a[i];

a[i]=a[min];

a[min]=temp;

}

}

}

int main()

{

int n,i;

printf("\n Enter the Number of Elements: ");

scanf("%d",&n);

int a[n];

printf("\n Enter %d Elements:\n ",n);

for(i=0;i<n;i++)

{

scanf("%d",&a[i]);

}

selection(a,n);

printf("\n The Sorted array by selection sort are: ");

for(i=0;i<n;i++)

{

printf("%d ",a[i]);

}

return 0;

}

OUTPUT: 

INSERTION SORT:

#include <stdio.h>

#include <conio.h>

void insertion\_sort(int arr[], int n);

void main()

{

int i, n;

printf("\n Enter the number of elements in the array: ");

scanf("%d", &n);

int arr[n];

printf("\n Enter the elements of the array: ");

for(i=0;i<n;i++)

{

scanf("%d", &arr[i]);

}

insertion\_sort(arr, n);

printf("\n The sorted array is: \n");

for(i=0;i<n;i++)

printf(" %d\t", arr[i]);

getch();

}

void insertion\_sort(int arr[], int n)

{

int i, j, temp;

for(i=1;i<n;i++)

{

temp = arr[i];

j = i-1;

while((temp < arr[j]) && (j>=0))

{

arr[j+1] = arr[j];

j--;

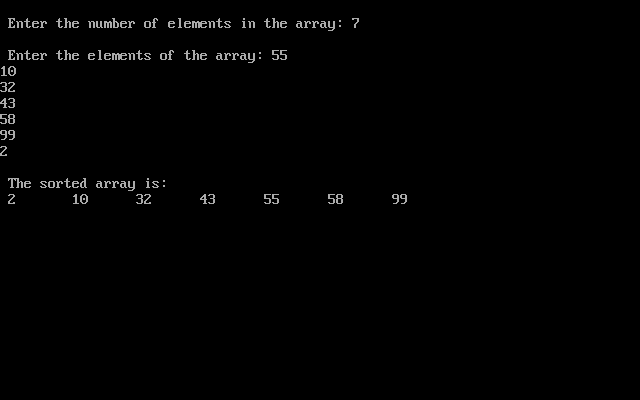
}

arr[j+1] = temp;

}

}

**OUTPUT:**

****

CONCLUSION:

By performing above practical we can conclude that insertion sort is much more stable than selection sort. Insertion sort is a live sorting technique where the arriving elements are immediately sorted in the list whereas selection sort cannot work well with immediate data. For selection sort the best case for time complexity is Ω(n^2) and the average case is θ(n^2) and the worst case is O(n^2) .For insertion sort the best case for time complexity is Ω(n) and the average case is θ(n^2) and the worst case is O(n^2).Space complexity of selection sort is 1 and for space complexity insertion sort is also 1