DSA - Experiment 2

Name: Divyesh Khunt **Sapid:**60009210116 **Batch: A/3**

Aim: To implement and analyze Insertion and Selection sort.

Selection Sort

Theory:
The selection sort algorithm sorts an array by repeatedly finding the minimum element (considering ascending order) from unsorted part and putting it at the beginning.

Time Complexity: the time complexity of Selection Sort is O(N2) as there are two nested loops

• one loop to select an element of Array one by one = O(N)

• Another loop to compare that element with ever other Array element

O(N)

therefore overall complexity O(N*N) =$O(N^2)$

CODE:

```c
int main()
{

int a[100], n, i, j, min, temp;
printf("Enter value of n: ");
scanf("%d", &n);
printf("Enter array: ");
for(i=0; i<n; i++)
{
scanf("%d", &a[i]);
}
for(i=0; i<=n-2; i++)
{
min=i;
for(j=i+1; j<=n-1; j++)
{ if(a[j]<a[min]) min=j;

}
temp=a[i]; a[i]=a[min]; a[min]=temp;
} printf("The sorted array is: ");
for (i=0; i<n; i++) {

printf("%d\n", a[i]);
}
}
```

OUTPUTS:

```
Enter value of n: 5
Enter array: 23
45
1
45
7
The sorted array is: 1
7
23
45
45

--------------------------------
Process exited after 5.355 seconds with return value 0
Press any key to continue . . .
```

Insertion sort

Theory:

Insertion sort is a simple sorting algorithm that works similar to the way you sort playing cards in your hands. The array is virtually split into a sorted and an unsorted part. Values from the unsorted part are picked and placed at the correct position in the sorted part.

Time Complexity :

Since there is a while loop enclosed by the for loop the time complexity adds up to **$O(N^2)=O(N*N)$**

CODE:

```c
#include<stdio.h>
#include<conio.h>
int main()
 {
int a[100], i, n, j, temp;
 printf("Enter value of n: ");
 scanf("%d", &n);
printf("Enter the array: ");
for(i=0; i<n; i++) {
scanf("%d", &a[i]);
}
for(i=1; i<=n-1; i++)
 {
 temp=a[i]; j=i-1;
 while(j>=0 && a[j]>temp)
    {
    a[j+1]=a[j];
    j=j-1;
    } a[j+1]=temp;

  }
 printf("The sorted array is: ");
 for(i=0; i<n; i++)
{
printf("%d \n", a[i]);
}
}
```

OUTPUTS:

```
Enter value of n: 5
Enter the array: 53
1
43
87
6
The sorted array is: 1
6
43
53
87

--------------------------------
Process exited after 13.57 seconds with return value 0
Press any key to continue . . .
```

Conclusion: Thus insertion and selection sort were implemented.