Date

Expt. No. 01 Page No.

Experiment - 1

☆ Aim :

a. WAP to implement Bubble Sort Algorithm and analyze its complexity.

b. WAP to implement Selection Sort Algorithm and analyze its complexity

c. WAP to implement Insertion Sort Algorithm and analyze its complexity.

A) Bubble Sort

```
int main () {
            int n= 5, arr[5];
            for (int i=0; i<n; i++)
            cin >> arr[i];

            for (int j=n-1 ; j>=0 ; j--) {
                 for (int k=0; k<j; k++)
             {
                  if ( arr[k] > arr[k+1] ) {
                     int temp= arr[k];
                     arr[k] = arr[k+1];
                     arr[k+1] = temp;     } } }
```

Teacher's Signature

```
  for (int i=0; i<n; i++)
       cout<< arr[i] <<" ";
 return 0;
}
```

Complexity:

In bubble sort we have 2 loops and the outer loop is running n times whereas the inner loop runs from 0 to n-1

There are n-1 comparisons in pass 1
n-2 comparisons in pass 2 and in the nth pass (n-n)

so $(n-1) + (n-2) + (n-3) \ldots\ldots (n-n)$

sum of (n-1) numbers $= \frac{(n-1)[(n-1)+1]}{2} = \frac{n(n-1)}{2} = \frac{n^2-n}{2}$

worst case scenario= $O(n^2)$

B) Selection Sort

```
int main () {
          int n=5, arr[5], min=0;
          for( int i=0; i<n; i++)
          cin >> arr [i];
          for ( int i=0; i<n; i++)
```

Date

Expt. No. Page No.

```
min = i;
for (int j=i; j<n; j++)
if ( arr[j] < arr[min])
    min = j;
}
    int temp = arr[min];
        arr[min] = arr[i];
        arr[i] = temp;
}
```

Complexity

In selection sort we have two loops the outer loops runs from 0 to n-1 which n times and the inner loop works from j (outer loop val) to n.

n-1 comparisons in pass 1 to find smallest element
n-2 " " pass 2 to find the next smallest

total: $(n-1) + (n-2) + (n-3) \dots + 3 + 2 + 1$

$$= \frac{(n-1)((n-1)+1)}{2} = \frac{n(n-1)}{2}$$

Worst case complexity = $O(n^2)$

Teacher's Signature

Date

Expt. No.

Page No.

c) Insertion sort

```
void insertionsort (int arr[], int n]
{
    int i, key, j;
    for (i=1; i<n; i++)
{
      key = arr[i];
      j= i-1;
    while (j>=0 && arr[j]> key)
{
       arr [j+1] = arr[j];
         j= j-1;
}
    arr [j+1] = key;
}
}

int main ()
{
    int arr [5], n=5;
    for (int i=0; i<n; i++)
        cin>> arr [i];
    insertionsort (arr, n);

    for (int i=0; i<n; i++)
    cout<< arr [i] <<" "; return 0;
}
```

Teacher's Signature

Date

Expt. No.   Page No.

Complexity

For insertion sort is better when the list is partially sorted but we calculate the worst case scenario for time complexity

In the first pass there is one comparison
In " second " " " two "

In the $n-1^{th}$ " " " $(n-1)$ comparison.

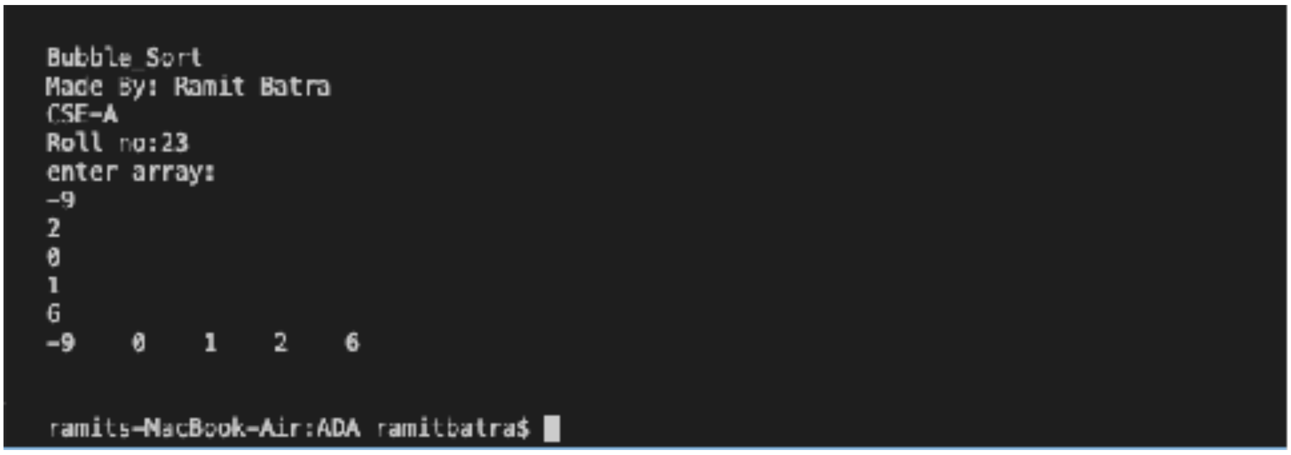
total comparison: $0+1+2+\dots(n-1) = \frac{(n-1)((n-1)+1)}{2}$

Worst case complexity $= \Theta(n^2)$

Teacher's Signature

Bubble Sort

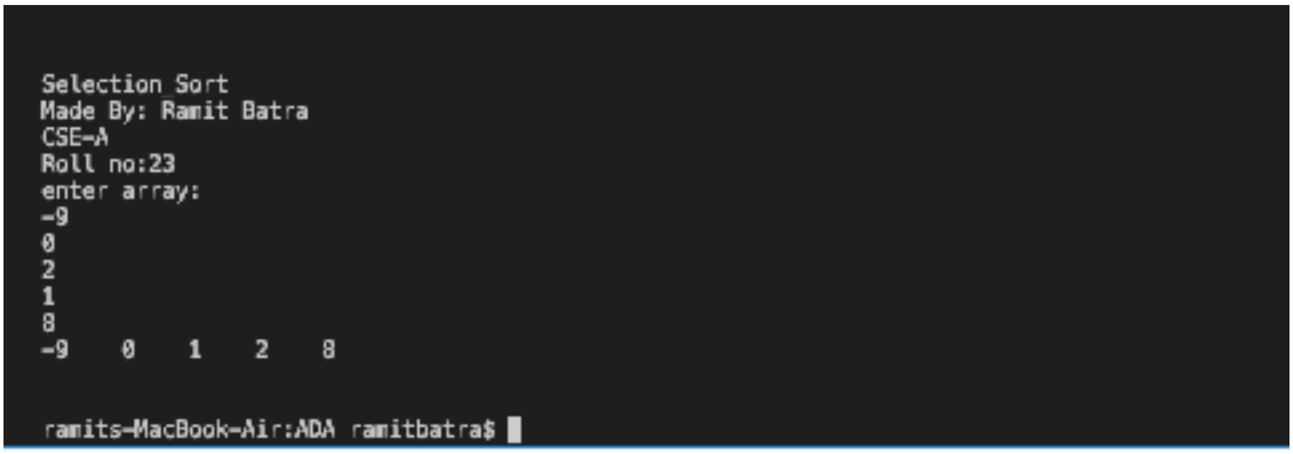
```
#include <stdio.h>
#include<iostream>
using namespace std;
int main()
{  int n=5;
    int arr[5];
    cout<<endl<<endl<<endl;
    cout<<"Bubble_Sort\nMade By: Ramit Batra\nCSE-A\nRoll
no:23\nenter array:\n";

    for(int i=0;i<n;i++)
    cin>>arr[i];


for(int j=n-1;j>=0;j--){
     for(int k=0;k<j;k++)
    {
    if(arr[k]>arr[k+1]){

    int temp=arr[k];
    arr[k]=arr[k+1];

    arr[k+1]=temp;}}
}
for(int i=0;i<n;i++)
    cout<<arr[i]<<"    ";
cout<<endl<<endl<<endl;

    return 0;
}
```



```
Bubble_Sort
Made By: Ramit Batra
CSE-A
Roll no:23
enter array:
-9
2
0
1
6
-9    0    1    2    6


ramits-MacBook-Air:ADA ramitbatra$ 
```

Selection Sort

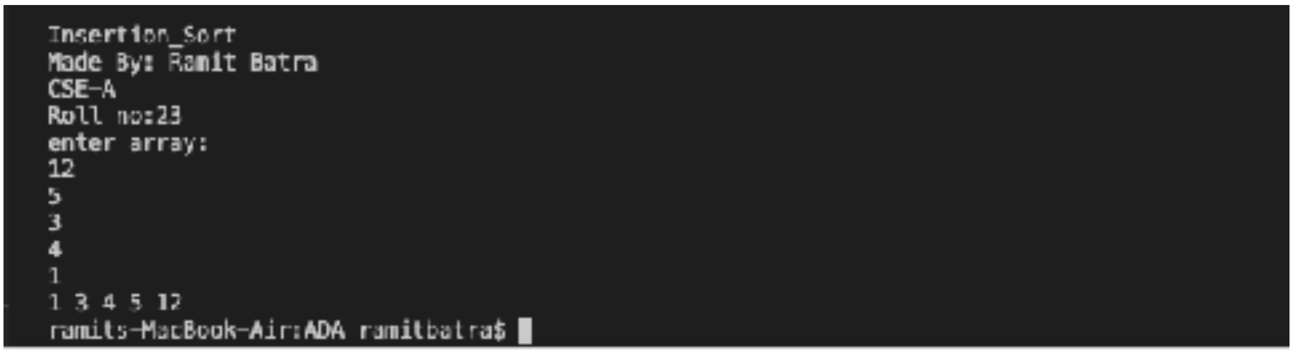
```
#include <stdio.h>
#include<iostream>
using namespace std;
int main()
{
    int n=5,arr[5],min=0;
    cout<<endl<<endl<<endl;
cout<<"Selection_Sort\nMade By: Ramit Batra\nCSE-A\nRoll
no:23\nenter array:\n";
    for(int i=0;i<n;i++)
    cin>>arr[i];
    for(int i=0;i<n;i++){
min=i;
for(int j=i;j<n;j++){
if(arr[j]<arr[min])
min=j;
    }
    int temp=arr[min];
    arr[min]=arr[i];
    arr[i]=temp;
    }
    for(int i=0;i<n;i++)
    cout<<arr[i]<<"    ";
cout<<endl<<endl<<endl;
    return 0;
}
```



```
Selection Sort
Made By: Ramit Batra
CSE-A
Roll no:23
enter array:
-9
0
2
1
8
-9    0    1    2    8


ramits-MacBook-Air:ADA ramitbatra$ 
```

Insertion Sort

```
#include <stdio.h>
#include<iostream>
using namespace std;
void insertionSort(int arr[], int n)
{
    int i, key, j;
    for (i = 1; i < n; i++)
    {
        key = arr[i];
        j = i - 1;

        /* Move elements of arr[0..i-1], that are
        greater than key, to one position ahead
        of their current position */
        while (j >= 0 && arr[j] > key)
        {
            arr[j + 1] = arr[j];
            j = j - 1;
        }
        arr[j + 1] = key;
    }
}

// A utility function to print an array of size n
void printArray(int arr[], int n)
{
    int i;
    for (i = 0; i < n; i++)
        cout << arr[i] << " ";
    cout << endl;
}

/* Driver code */
int main()
{
    int arr[5] ;
    int n = 5;
 cout<<endl<<endl<<endl;
cout<<"Insertion_Sort\nMade By: Ramit Batra\nCSE-A\nRoll
no:23\nenter array:\n";
for(int i=0;i<n;i++)
    cin>>arr[i];
    insertionSort(arr, n);
    printArray(arr, n);

    return 0;
}
```



```
Insertion_Sort
Made By: Ramit Batra
CSE-A
Roll no:23
enter array:
12
5
3
4
1
1 3 4 5 12
ramits-MacBook-Air:ADA ramitbatra$
```