Name ⇒ Jatin Aggarwal
Course ⇒ B-Tech (IT)
Roll no. ⇒ 1912078

Page No.
Date

D.S Assignment

Que 1

Let the array be :

arr[] = {1, 2, 3, 4}
which is already sorted

Insertion sort Algo :

```
for (int 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;
}
```

Complexity Analysis :-

	0	1	2	3	4
i		1	2	3	4
j			2	3	4
key			2	3	4
Comp			1	1	1
Movement			1	1	1

$T(n) = 2+2+2-n = 3(n)$

time complexity $= O(n)$

The complexity of the following algorithm has been reduced by

① Binary Search:
By using a sorted array & using binary search can reduced the complexity to $(O(n))$

② By using linked list:
By using linked list, the complexity of insertion becomes const & the overall complexity becomes $O(n)$

Que 2

Quick sort algorithm:

```
Quick_sort (*arr, p, q)
{
  if (p < q)
  { r = Partition (arr, p, q)
    Quick_sord (arr, p, r-1)
    Quick_sort (arr, r+1, q)
  }
Partion (*arr, p, q)
{
  n = arr[q]
```

```
i = P-1
for (j = P to q-1)
    if (arr[j] < x)
        swap(arr[j], arr[j+1])
    arr[i+1] = x
    return (i+1);
}
```

The complexity of best case of quick sort is $T(n) = O(n \log n)$

The complexity of worst case of quick sort is $T(n) = O(n^2)$

Bubble sort

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

The complexity of best case of bubble sort is

$$T(n) = O(n^2)$$

The complexity of worst case of bubble sort is

$$T(n) = O(n^2)$$

Complexity Comparison

	Merge sort	Quick sort	Insertion	Bubble
worst case time	$T(n) = O(n \log n)$	$T(n) = O(n^2)$	$T(n) = O(n^2)$	$T(n) = O(n^2)$
best case time	$T(n) = O(n \log n)$	$O(n \log n)$	$T(n) = O(n)$	$T(n) = O(n^2)$