Tutorial 3

Name: Simran

Class Roll No: 57

Section: B

Uni Roll No: 2014888

Ans 1 pseudofunction for linear search :

```
int Linears (int *arr[], int n, int key)
{
    for (int i = 0; i< n; i++){
        if (arr[i] = = key)
            return i;
    }
    return -1;
}
```

Ans 2 pseudo code for Insertion Sort :

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

* Insertion sort is called online sort as if an element comes in an array it is automatically inserted at its correct position.

Ans 3 Average case complexities of Sorting Algos:

* Bubble = $O(n^2)$
* Insertion = $O(n^2)$
* Selection = $O(n^2)$
* Merge = $O(n \log n)$
* Quick = $O(n \log n)$
* Heap = $O(n \log n)$

Ans 4

	Stable (appears in same order)	Inplace (O(1))
Bubble	✓	✓
Selection	✗	✓
Insertion	✓	✓
Merge	✓	✗
Quick	✗	✗
Heap	✗	✓

Ans 5 pseudocode for Binary Search

```
int start = 0
int end = size - 1
while (start <= end)
{
    int mid= [start + (end-start)]/2;
    if (key == arr[mid]
        return mid;
    else if (key < arr[mid])
        end = mid-1;
    else
        start = mid+1;
}
return -1
```

T.C = $O(\log n)$

Space Complexity = $O(1)$

Linear Search

T.C. = $O(n)$

S.C. = $O(1)$

Ans 6 Recurrence relⁿ of Binary Search:

$T(n) = T(n/2) + 1$

Ans 8 Quick Sort is the best sorting algo in practical use as it follows the locality of reference & also its best case time complexity is $O(n\log n)$.

Ans 9 No. of inversions: It tells us how far is the array is from being sorted.

if $a[i] > a[j]$ & $i < j$

⇒ 7 21 31 8 10 1 20 6 4 5

no. of inversions: 4 + 7 + 7 + 4 + 4 + 3 + 2

= 31

Ans 10 Quick Sort will give:

- Best case complexity: When array is totally unsorted
- Worst complexity: when array is sorted or reverse sorted

Ans 11 Recurrence rel^n of :

	Merge Sort	Quick Sort
Best	$2T(n/2) + \theta(n)$	$T(n) = T(k) + T(n-k-1) + \theta(n)$
Worst		$T(n) = T(n-1) + \theta(n)$

Similarity : Both are of type divide & conquer

Differences : Worst case complexity of Merge Sort is $O(n \log n)$ whereas of Quick Sort is $O(n^2)$.

Ans 13 Optimised Bubble Sort :

```
for (int i=0; i<n; i++)
{
    swap= false;
    for (j=0; j<n-i-1; j++)
    {
        if (arr[j] <arr[j+1])
        {
            swap(arr[j], arr[j+1]);
            swap = true;
        }
    }
}
```

Ans 14 In such case, Merge Sort would be efficient as it is an External Sorting algorithm i.e. data is divided into chunks & then sorted using Merge Sort.

⇒ Sorted data is dumped into files.

- Internal Sorting: It is a type of sort in which whole sorting takes place in main memory of computer.