Tutorial-3

Q1.
```
int lsearch (int *a[], int n, int key).
{
    for (i=0 to n).
        if a[i] = key
            return i.
    Retur -1.
}
```

Q2.
```
void insertionsort (int a[], int n)
{
    for (i=1 to n)
        int key = a[i]
        int j = i-1
        while ( j>=0 && a[j]>key).
            a[j+1] = a[j]
            j--;
        a[j+1] = key.
}
```

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

Q3. Avg. case complexities of sorting algo's:-
Bubble = $O(n^2)$, Insertion = $O(n^2)$, Selection = $O(n^2)$,
Merge = $O(n \log n)$, Quick = $O(n \log n)$, Heap = $O(n \log n)$.

Q4.

	Stable	Inplace
Bubble	✓	✓
Selection	x	✓
Insertion	✓	✓
Merge	✓	x
Quick	x	x
Heap	x	✓

Q5.
```
Int l=0, h= size-1
while ( l<=h)
{ int mid= (l+ (h-l))/2
  if key = a[mid]
        return mid
  else if key < a[mid].
        h = mid-1;
  else
        l= mid+1;
}
return -1
```

T.C = O(log n).
Space complexity = O(1).

Linear Search
T.C = O(n)
S.C = O(1).

Q6. Recurrence relation of binary search =
$T(n) = T(n/2)+1$.

Q8. Quick sort is best sorting algo in practical use as it follows the locality of reference & also its best case time complexity is O(nlogn).

Q9. 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 45.

No. of Inversion: 4+7+7+4+4+3+2
= 31.

Q10 Quick sort will give:—

a] Best case T.C ⇒ where array is totally unsorted.

b] Worst case T.C ⇒ When array is sorted or reverse sorted

Q11. Reccurrence Relation of:

Merge Sort

Best
Worst } $2T(n/2)+\Theta(n)$

Quick sort

$T(n)=T(k)+T(n-k-1)+\Theta(n)$.

$T(n)=T(n-1)+\Theta(n)$

Similarity: Both are of type divide & conquer.

Difference: Worst case T.C of Merge sort is $O(n \log n)$ whereas of Quick sort is $O(n^2)$.

Q13. Optimized Bubble Sort

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

Q14. 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 type of sort in which whole sorting takes place in main memory of computer.