Tutorial-3

Ques1 → write linear search pseudocode to search an element in a sorted array with minimum comparision

```
int linearSearch (arr[], key, lb, ub)
{
    mid = lb+ub/2;
    if (key
    For(i: lb to ub)
    {if (arr[i] == key)
        {
          return arr[i];
        }
     else
          return 0;
    }
}
```

Best case: if the element is at 1st position in the array, then its no of comparion are 1.

average case:- if element is at middle of array

worst case: if the element is at last of the array, then it is having n-1 comparison

Ques2 → write a pseudo code for iterative and recursive insertion sort. Insertion sort is called Online sorting. why?

it is also known as online sort, since it can sort a list as it receive it. In all other algorithm we need all elements to be provided to the sorting algorithm before applying it. But on insertion sorting allow us to start with partial set of elements sort it, - if even additional one element is inserted during sorting process other algo doesn't respond easily but insert sort do

Iterative method

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

recursive method

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

Ques- complexity of all sorting algorithm that has been searched?

Linear search $O(\log n)$
Binary search $O(\log n)$
Selection sort $O(n^2)$
Bubble sort $O(n^2)$
insertion sort $O(n^2)$
Merge sort $O(n \log n)$
quick sort $O(n \log n)$
heap sort $O(n \log n)$
Count sort $O(n, k)$

Ques - Divide all the sorting algorithm into inplace / stable / online sorting.

In-Place Algorithm	Stable Algo	Online Algo
Bubble sort	Merge sort	Insertion sort
Selection sort	Count sort	
Heap sort	Insertion sort	
Insertion sort	Bubble sort	

Ques - Write recursive / iterative pseudo code for binary search. what is the Time and space complexity of Linear and Binary search.

Iterative code

```
int Binary (int arr[], int lb, int ub)
{ if (lb == ub)
  {
   arr[lb] = key)
    return arr[lb]
  }
 else
    if (lb ≤ ub)
   { mid = (lb+ub)/2
    if (arr[mid] == key)
       reutrun arr[mid]
    else if (arr[mid] > key)
        return lb = mid-1
    else
        return lb = mid+1
  }
}
```

Recursive Code

```
int Binary (int arr[], int lb, int ub)
  {
   while (lb < ub)
     { mid = (lb+ub)/2
    if (arr[mid] == key)
       return arr[mid]
    else if (key > arr[mid])
     Binary(int arr[], mid+, ub)
    else
     Binary(arr[], lb, mid-1)
   }
   return 0;
}
```

Time complexity of Binary: $O(\log n)$
Time complexity of Linear: $O(n)$

Ques 6 → write recurrence relation for binary search algorithm.

$T(n) = T(n/2) + 1$ —① $T(1) = 1$

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

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

⋮

$T(n) = T\left(\frac{n}{2^k}\right) + k$

$\frac{n}{2^k} = 1$

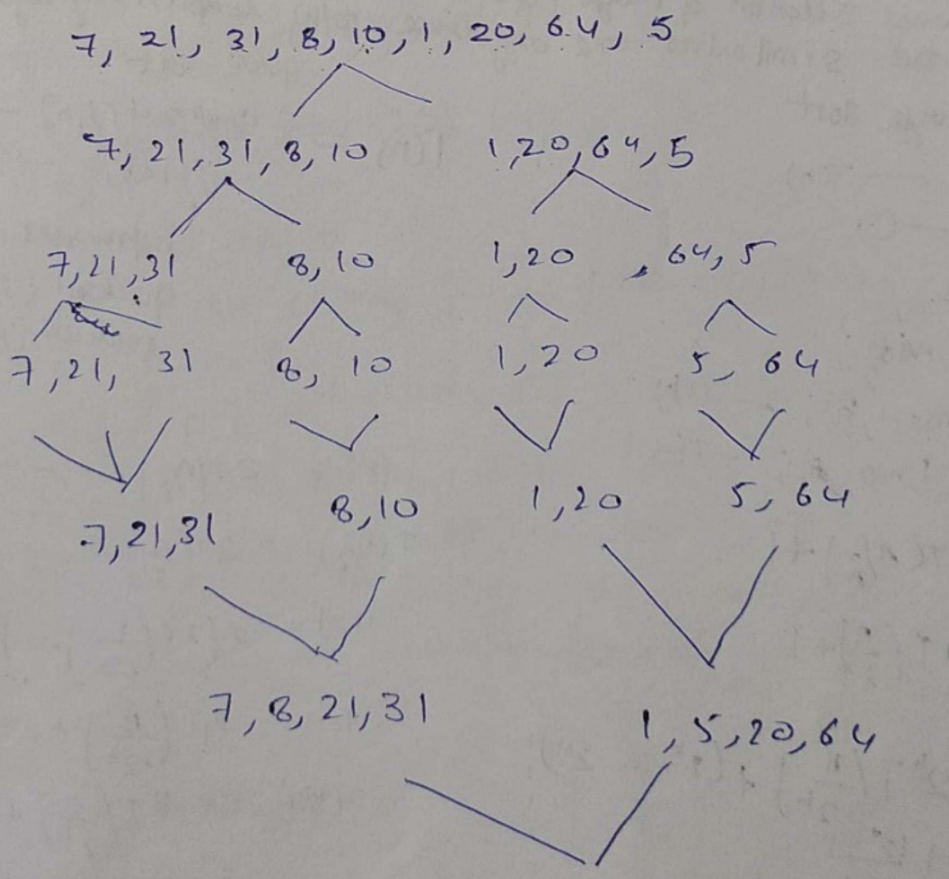
$n = 2^k$ taking log both side

$\log n = k \log_2^2$

$\boxed{k = \log n}$

Ques 7 → what do you meant by No. of inversion in an array! count the number of inversion in Array arr[] = {7, 21, 31, 8, 10, 1, 20, 6, 4, 5} using merge sort.

The number of inversion of an array. indicate the total changes that are required in the array to convert it into sorted form



Ques – In which case of quicksort, it will gave best and worst case time complexity?

Best case complexity :- In quick sort, the best case occurs, when the pivot element is in the middle of the array. The best - case time complexity of quicksort is O(n*logn)

Worst case Time complexity :- it occure when the pivot element is either greatest or smallest element, suppose ;if the pivot element is always the last element of array, the worst case would occur when the given array is sorted already in ascending or descending order. The worst case time complexity is O(n2

Ques - Selection sort is not stable by default but can you write a version of stable selection sort

A sorting algorithm is said to be stable if two object with equal or same key appear in the same order in sort- output as in input selection sort work by finding the minimum element and then inserting it is its correct position by swapping with element which is in the position of minimum element.

4A 5 3 2 4B 1

1 4A 5 3 2 4B 4A

Ques → write Recurrence Relation of Merge / Quick sort in best and worst case? what are the similarities and difference b/w complexity of algorithm & why.

Merge Sort

```
Mergesort (l,h)        — T(n)
{ if(l<h)              — (1)
  { mid=(l+h)/2;
    Mergesort (l,mid);
    Mergesort (mid+1,h); — T(n/2)
    Mergesort (l,mid);   — T(n/2)
```

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

$T(n) = 4T\left(\frac{n}{2^2}\right)+1$

$\vdots$

$T(n) = 2^k T\left(\frac{n}{2^k}\right) + (2^k + \dots 2^0)$

$\log n = k \quad +1 \quad \frac{(2^k-1)}{2-1}$

$O(n(\log n))$ → Best Case

$O(n \log n)$ worst case

quick sort

```
T(n) —  quicksort(l,h) —
        { if(l<h)      — (1)
          { j=partition(l,h);
            quicksort(l,j);  — T(n/2)
            quicksort(j+1,h); — T(n/2)
```

$T(n) = 2T(n/2)+1$ — (1)

$T(n/2) = 2T\left(\frac{n}{2^2}\right)+1$

$T(n) = 2\left(2T\left(\frac{n}{2^2}\right)+1\right)+1$

$T(n) \Rightarrow 2^2 T\left(\frac{n}{2^2}\right)+2+1$

$T(n/4) = 8T\left(\frac{n}{2^3}\right)+4+2+1$

$T(n) \Rightarrow 2^k T\left(\frac{n}{2^k}\right) + (2^k + \dots + 2^0)$

Best case $\frac{n}{2^k} = 1$

$O(n \log n)$

ques- your computer has a RAM of 2 GB and you are given an array of 4 GB of sorting which algorithm to use for this purpose and why? Also explain the concept of External & Internal sorting

To process 4GB of data we need to divide the large dataset into smaller subsets of sizeless than 2GB as the ram of computer is 2GB, so data will be processed is breaked into smaller bits.
Hence External Sorting is used for the purpose.

if the data sorting process take place entirely within the Random-Access Memory (RAM) of the computer. it's called internal sorting. The is possible whenever the size of the dataset to be sorted is small enough to be held in RAM

For sorting larger dataset, it may be necessary to hold only a smaller chunk of data in memory at a time, since it won't all fit in the RAM. The rest of the data is normally held on some larger like harddisk. called external sorting

ques - Bubble sort scans whole array even when array is sort
Can you modify the bubble sort so that it scan the whole array once it is sorted?

```
void bubble (int *arr, int n)
{
  for (int i=0; i<n; i++)
  {
    bool flag = false;
    for (array j=0; j<n-i-1; j++)
    {
      if array[j] > array[j+1])
      {
        flag = true;
        int temp = array[j+1];
        array[j+1] = array[j];
        array[j] = temp;
      }
    }
    If (flag != 0)
    {
      return;
    }
```