Nikhil Aswal _ Sec-F_ 17_ 2016877

Tutorial - 3

Ans.1.

```
for ( i = 0 to n)
{
        if ( arr [i] == value )

}
```

Ans. 2. Iterative

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

                }
              arr[j+1] = x ;

        }

}
```

Recursive

```
    void  Insertion_Sort ( int arr[ ], int n)
    {
            if( n<=1)
                return ;
          Insertion_Sort (arr, n-1);
          int  last =  arr[n-1];
          int   j=n-2;
```

```
while (j>=0 && arr[j] > last)
{
        arr [j+1] = arr[j];
        j--;
}
arr[j+1]= last ;
}
```

Insertion sort is called 'Online sort' because it does not need to know anything about what value it will sort and information is requsted while algorithm running.

Other Sorting algorithm:

- Bubble Sort
- Quick Sort
- Merge Sort
- Selection Sort
- Heap Sort

Ans. 3.

Sorting Algorithm	Best	Worst	Average
Selection Sort	$O(n^2)$	$O(n^2)$	$O(n^2)$
Bubble Sort	$O(n)$	$O(n^2)$	$O(n^2)$
Insertion Sort	$O(n)$	$O(n^2)$	$O(n^2)$
Heap Sort	$O(n\log n)$	$O(n\log n)$	$O(n\log n)$
Quick Sort	$O(n\log n)$	$O(n^2)$	$O(n\log n)$
Merge Sort	$O(n\log n)$	$O(n\log n)$	$O(n\log n)$

Ans. 4.

INPLACE SORTING	STABLE SORTING	ONLINE SORTI
Bubble Sort	Merge Sort	Insertion Sort
Selection Sort	Bubble Sort	
Insertion Sort	Insertion Sort	
Quick Sort	Count Sort	
Heap Sort.		

Ans. 5. Iterative:

```
int  It_Search (int arr[], int l, int r, int key)
{
    while(l<=r)
    {
        int m= ((l+r)/2);
        if (arr[m] = = key)
            return m;
        else if ( key < arr[m])
            r=m-1;

        else
            l=m+1;
    }
    return -1;
}
```

Recursive

```
int be_Search( int arr[], int l, int r, int key)
{
    while (l<=n)
    {
        int m= ((l+r)/2);
        if ( key == arr[m])
            return m;
    else if ( key < arr[m])
        return b-search ( arr, l, mid-1, key);
    else
        return b-search ( arr, mid+1, r, key);
  }
  return -1;
}
```

Time complexity:

Linear Search - $O(n)$

Binary Search - $O(\log n)$

Ans. 6. $T(n) = T(n/2)+1$ —— (I)

$T(n/2) = T(n/4)+1$ —— (II)

$T(n/4) = T(n/4)+1$ —— (III)

$$T(n) = T(n/2)+1$$
$$= T(n/4)+1+1$$
$$= T(n/8)+1+1+1$$
$$\vdots$$
$$= T\left(\frac{n}{2^n}\right)+1 \text{ (k times)}$$

Let $2^k = n$

$k = \log n$

$T(n) = T(n/n) + \log n$

$T(n) = T(1) + \log n$

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

Ans 7.

```
for( i=0; i<n; i++)
{
    for (int j=0; j<n; j++)
    {
        if ( a[i] + a[j] = = k)
            printf(" %d %d", i,j);

    }

}
```

Ans.8. Quick Sort is fastest general pupose Sort, In most practical Situation quicksort is the method of choice as stalutity is important and space is available, merge sort might be best.

Ans.9. A pair (A[i], A[j]) is said to be enversion of

$a[i] > a[j]$

$i < j$

total no. of inversions is given array arr 31 using merge sort.

Ans 10. Worst case $O(n^2)$ - The worst case is occur when the pivot element is an extreme element This happen when input array is Sorted or re verse sorted and either first or both element is selected as pivot.

Best Case $O(n\log n)$ - The best case occurs when we will Select pivot element as a mean element.