Date
Page No.

24/09/23

-: Lacture-011 :-

: Time complexity & Space complexity :

Time complexity :-

- It is an amount of time taken by an algorithm to run.
- As a function of length of the input

Why need?

- for making better program
- comparision of Algorithm

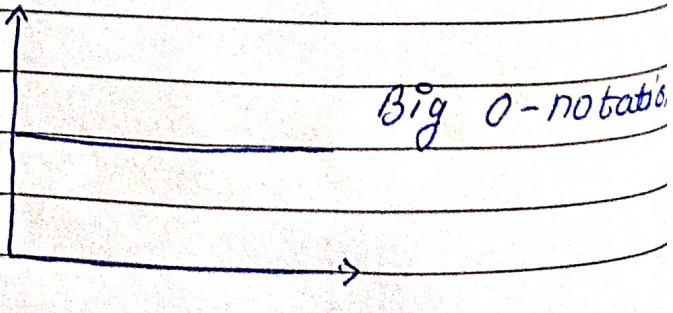
Notation

→ Big O Notation → Theta θ → Omega Ω

- Big O Notation ↓ upper bound
- Theta θ ↓ for avg case complexity
- Omega Ω ↓ lower case complexity

→ Constant time → O(1)
→ Linear time → O(n)
→ Logarithmic time → O(n³)
→ Quadratic time → O(n²)
→ Cubic time → O(n³).

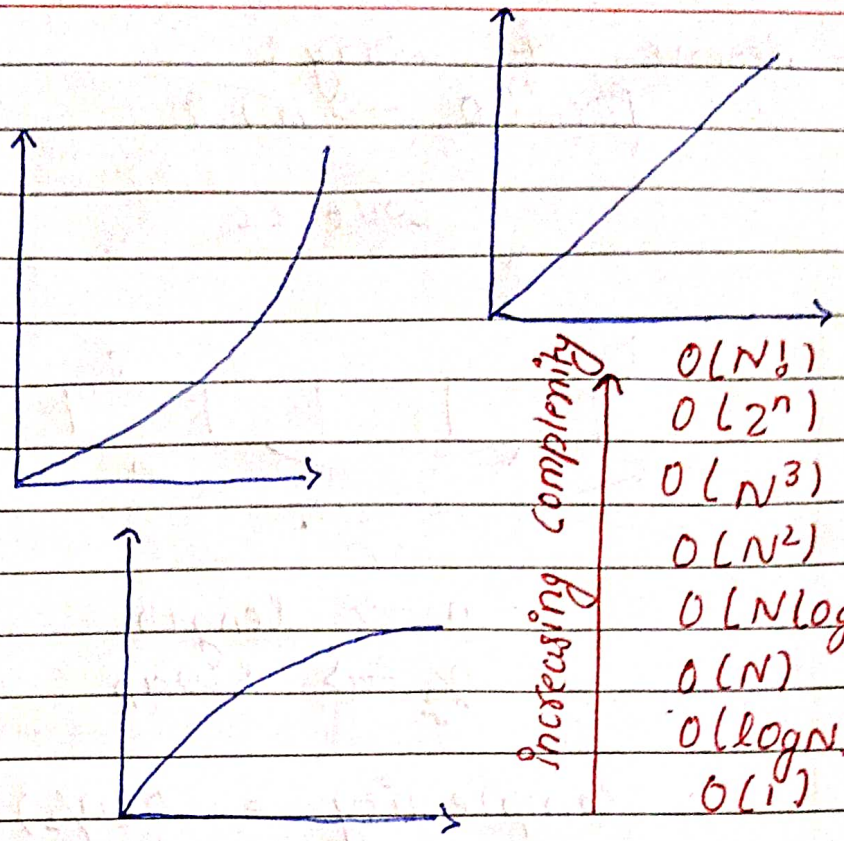
= → O(1)



→ O(n)

→ $O(n^2)$

→ O(log n)



= Question:-

$f(n) \longrightarrow 2n^2 + 3n \longrightarrow O(n^2)$

$f(n) \longrightarrow 4(n^4) + 3n^3 \longrightarrow O(n^4)$

$f(n) \longrightarrow N^2 + \log N \longrightarrow O(N^2)$

$f(n) \longrightarrow 1200 \longrightarrow O(1)$

$f(n) \longrightarrow 5n^2 + \log n \longrightarrow O(n^2)$

$f(n) \longrightarrow 3n^2 + 2n^2 + 5 \longrightarrow O(n^3)$

$f(n) \longrightarrow \frac{n^3}{5} \longrightarrow \longrightarrow O(n^3)$

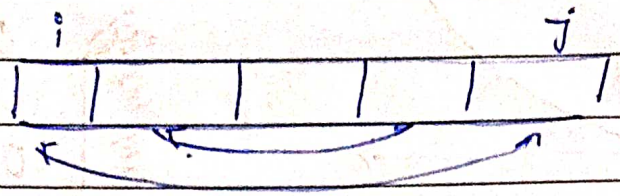
$f(n) \longrightarrow \frac{n+4}{4} \longrightarrow \longrightarrow O(n)$

= Reseve of array :

```
for (0 -> n) {
        cout << " " ;
}
```

n iteration

O(n)



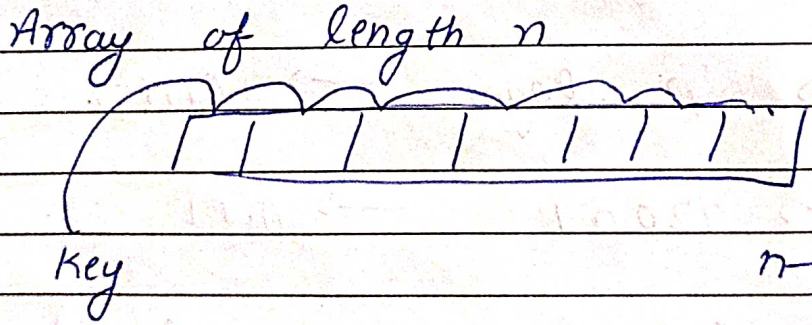
n → length

n/2 → swap

complexity = O(n/2)

= O(n)

= Linear Search :-



n → comparison

complexity = O(n)

Q what is time & space complexity of following loop.

```
int a = 0, b = 0 ;
for (i=0 ; i < N ; i++ {
        a = a + rand() ;
}
```

```
for (j = 0;  j < m ; j++){
        b = b + rand();
    }
```

Nested loop = ×
Seperate loop = +

time complexity = O(N) + O(m)

Q-2 What is time complexity:

```
int a = 0, b = 0;
for(i = 0; i < N; i++){
    for( j = 0 , j < N ; j++ ){
        a = a + j;
    }
}
for(k = 0 ; k < N ; k++ ) {
    b = b + k;
}
```

time complexity = O(N) * O(N) + O(N)

= O(N²) + O(N)

= O(N²)

Space complexity:-

→ How much memory take.

```
function ( )
{
    int arr[5] = { 1, 2, 3, 4, 5 }
}
```

fixed.

O(1)

3

```
int n;
cin >> n;
vector<int> v(n);
```

Space = O(n)

n length

```
for(0 — n)
{
    vector<int> v(n)
    for(0 — n)
    {
    }
}
```

O(n)