Date : 10th - 09 - 2020
Morning Session : 9am – 11.00 PM
By ~ Rohan Kumar

Topics: Time Complexity

Data Structure:

- **Data structure is a particular way of storing and organizing data in a computer so that it can be used efficiently.**
- **A data structure is a special formula for organizing and storing data.**
- **General Data Structure types include arrays, files, linked lists, stacks, queues, Trees, graphs and so on**

Algorithm:

- **An algorithm is Ihe step-by-step instructions to solve a given problem**

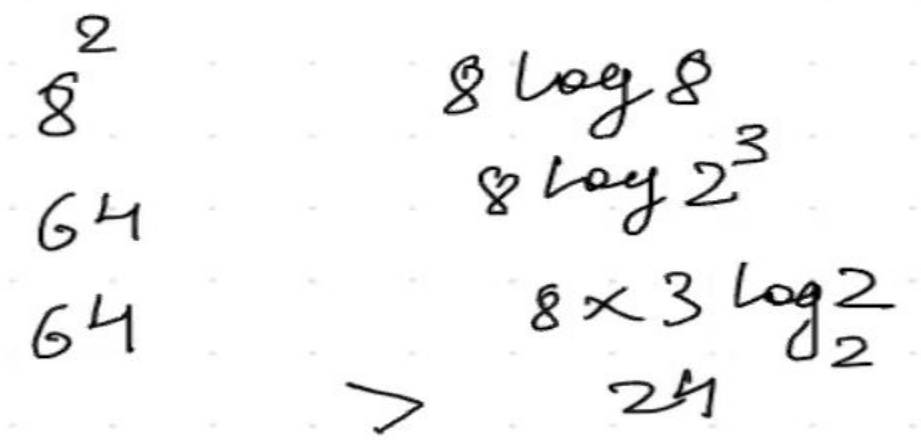
Algorithms contain some loops, Statement, Calculations.

How we can find that which is better algorithm is which algorithm

When we talk about algo we talk about better algorithm,

how can we do that ?? using time complexity we can do that.

Ex: n^2 , n(logn) which algo takes best??



Time Complexity : how much time it consumes to execute the problem.

Ex: for i in range(n):

print (i) (its running n time we can say time complexity of o(n) bcoz it is run for n times)

How many times will I be executed?? It will execute o(n)

We do not care if it is executing n+1, n+n, n+n+n times

Our order is nothing is Big O

Order tell you time complexity, tell you degree of polynomial

O = degree of polynomial (o())

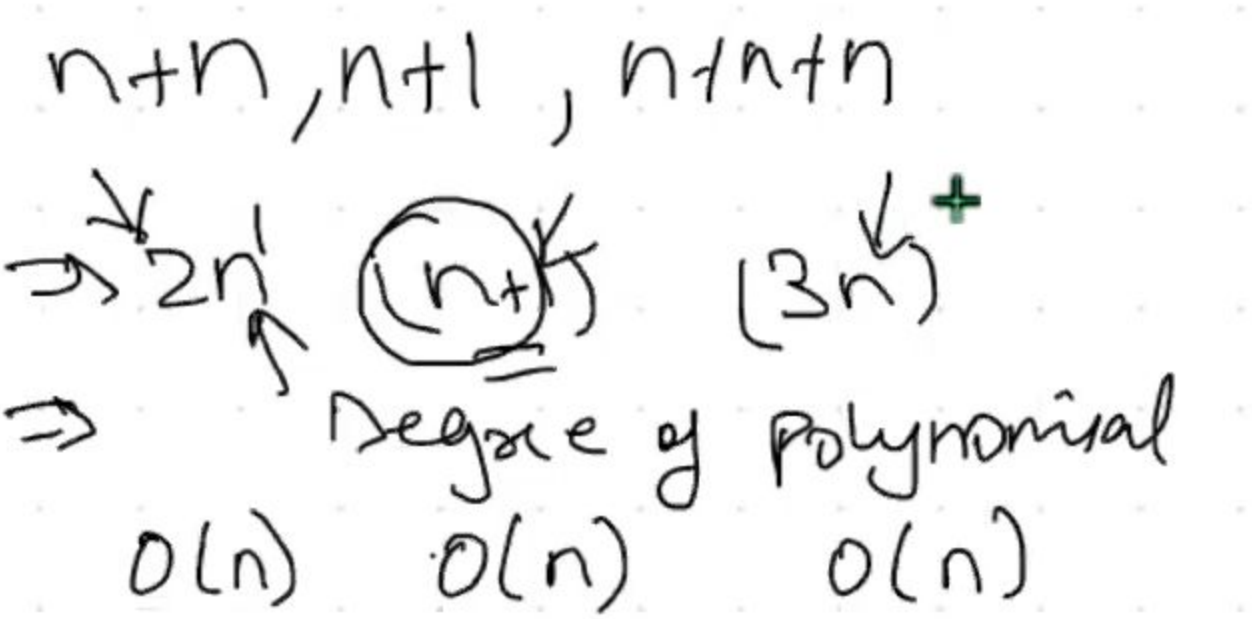
O(n) degree of n = 1

- **We can't use constants in time complexity**

Ex: for (i=0 ; i<n ; i++):

print(i)

- **This loop will go till n times we can say time complexity o(n).**



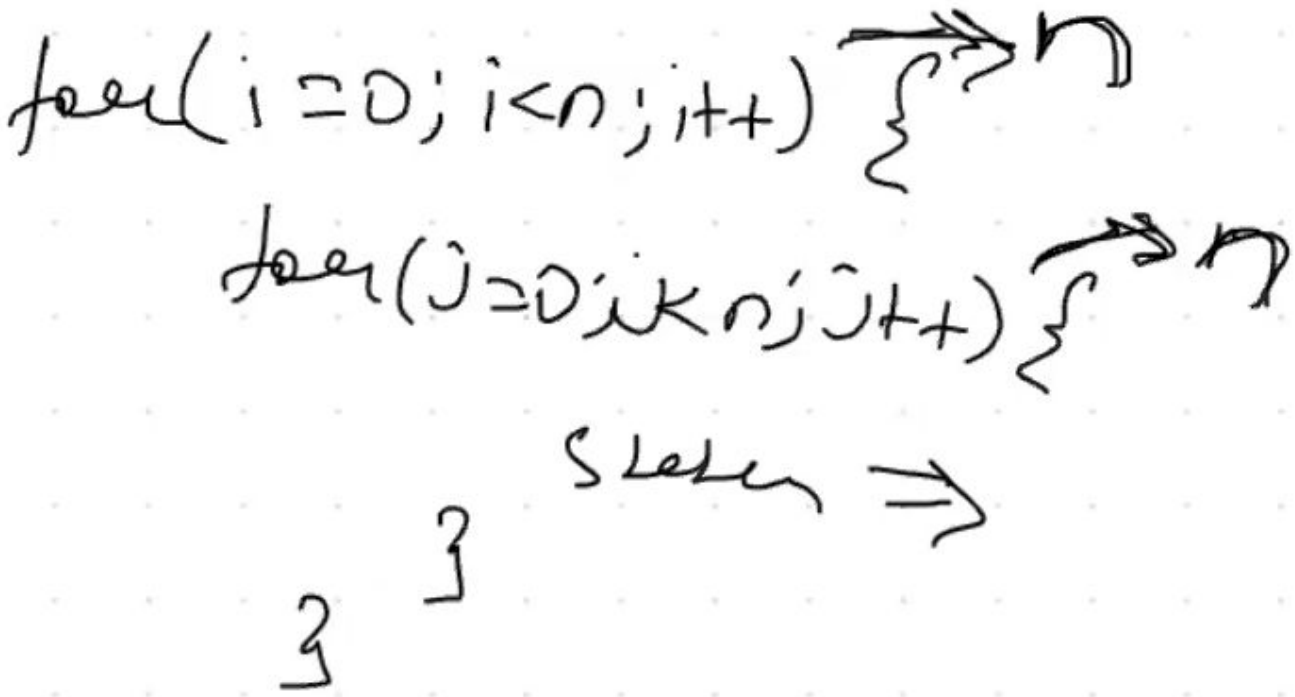
When we represent Time complexity make one thing clear always display the degree of polynomial

n^2 + n + 1 = o(n^2)

```
for (i = 0 ; i<n ; i = i+2) {
        Statement
        }
```

It will execute o(n/2) times we can consider the degree of polynomial so it will run for o(n) times.

Question:



Solution:

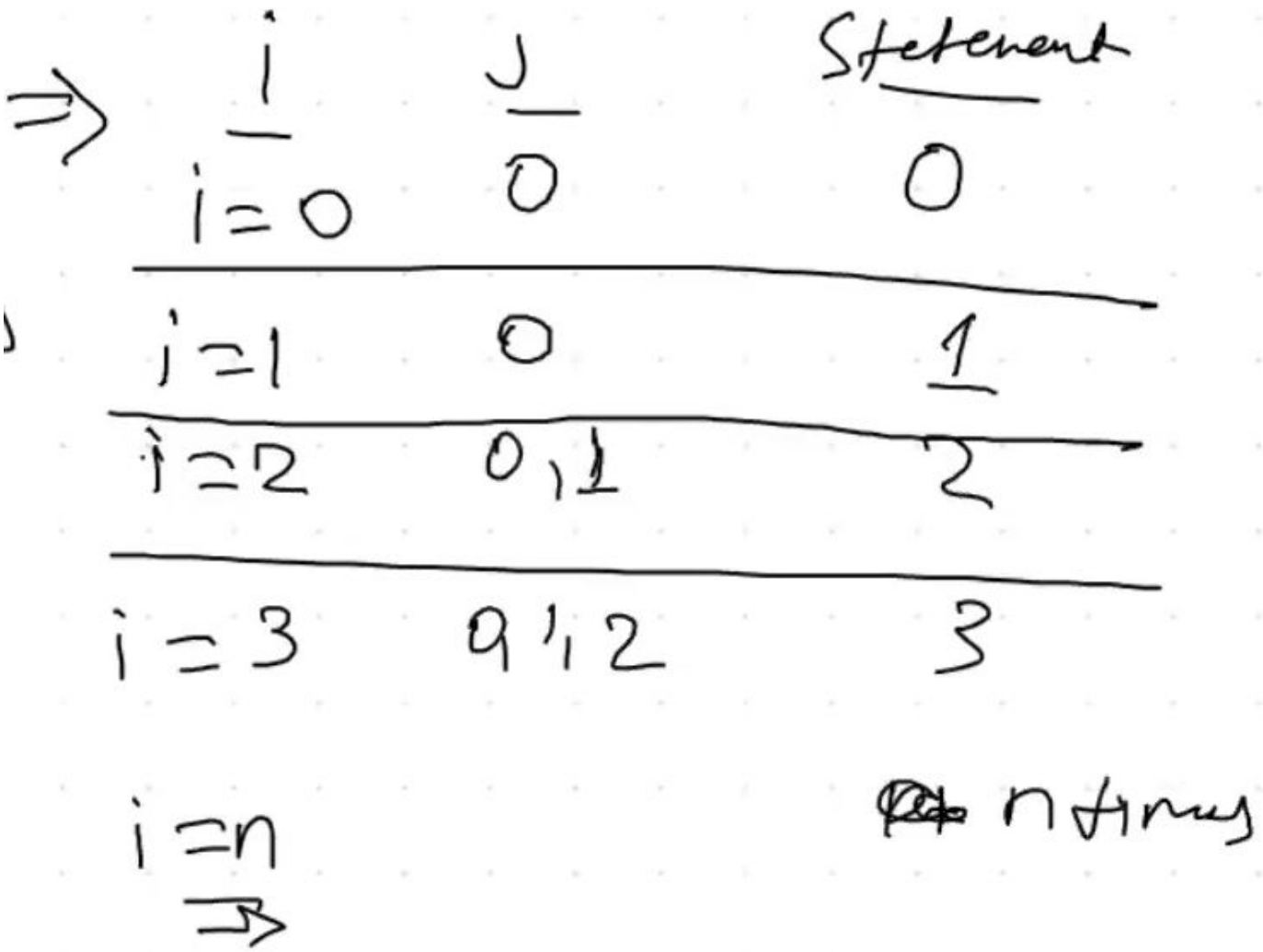
j loop running for n times and i loop running for n times so we can say

n X n = n^2 times since the degree of polynomial is n^2 we can say oder is n^2

Question:

```
for(i=0; i<n; i++) {
    for(j=0; j<i; j++) {
        Statement
    }
}
```

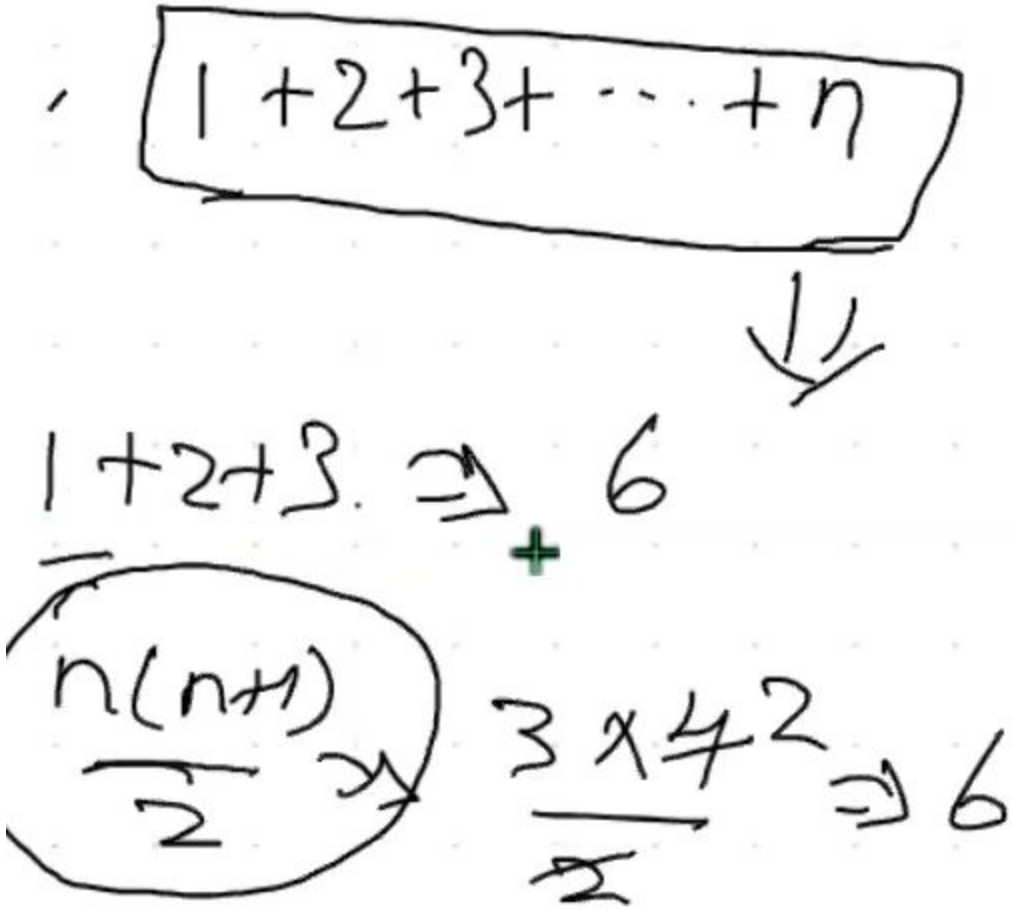
Solution:



Total number of times this statement will executed

1+2+3+--------+n = n(n+1)/2 = > n^2+n/2 => o(n^2)

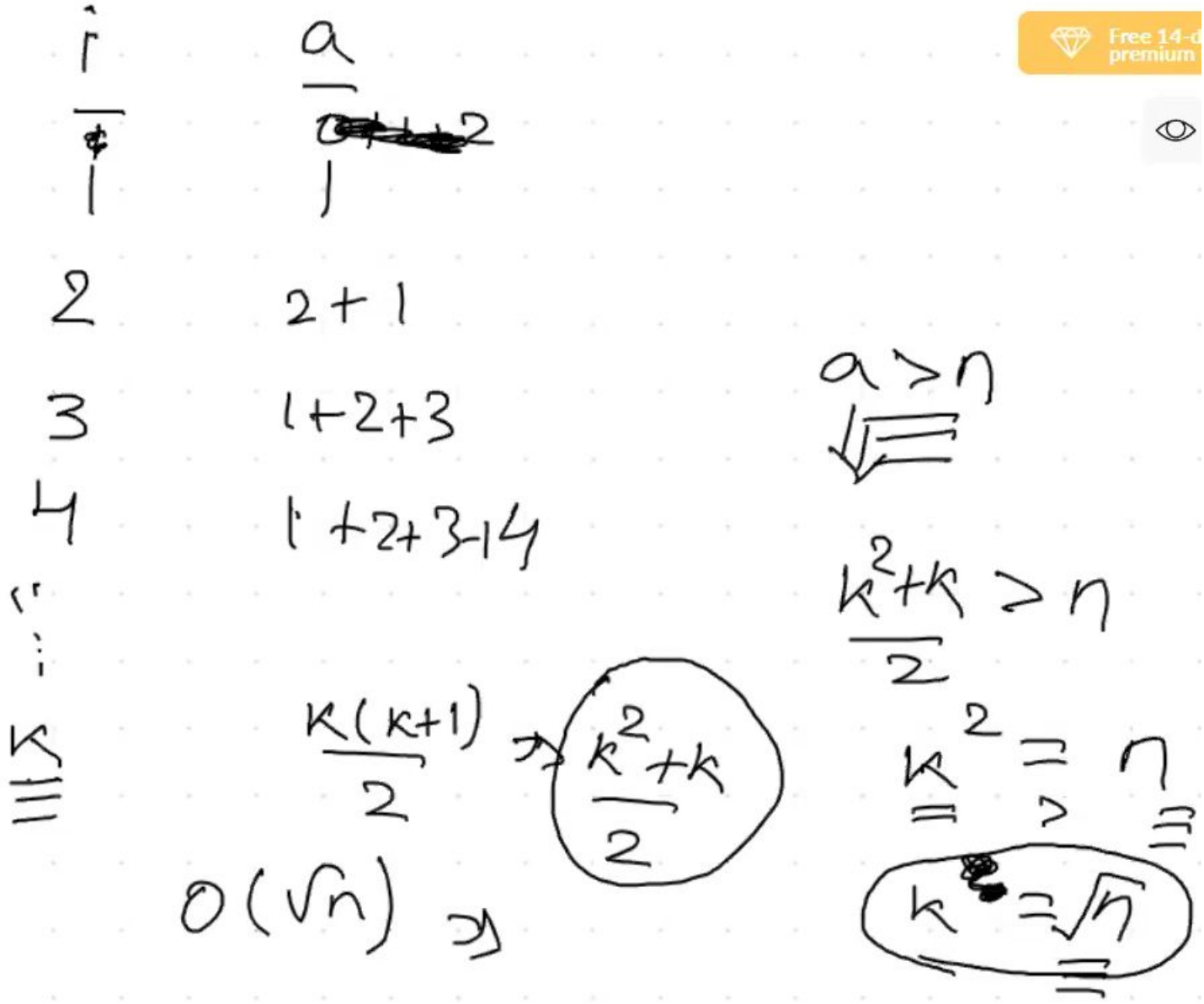
Formula:



Question:

```
a=0
for(i=1 ; a <=n ; i++) {
        a = a + i
}
```

Solution:



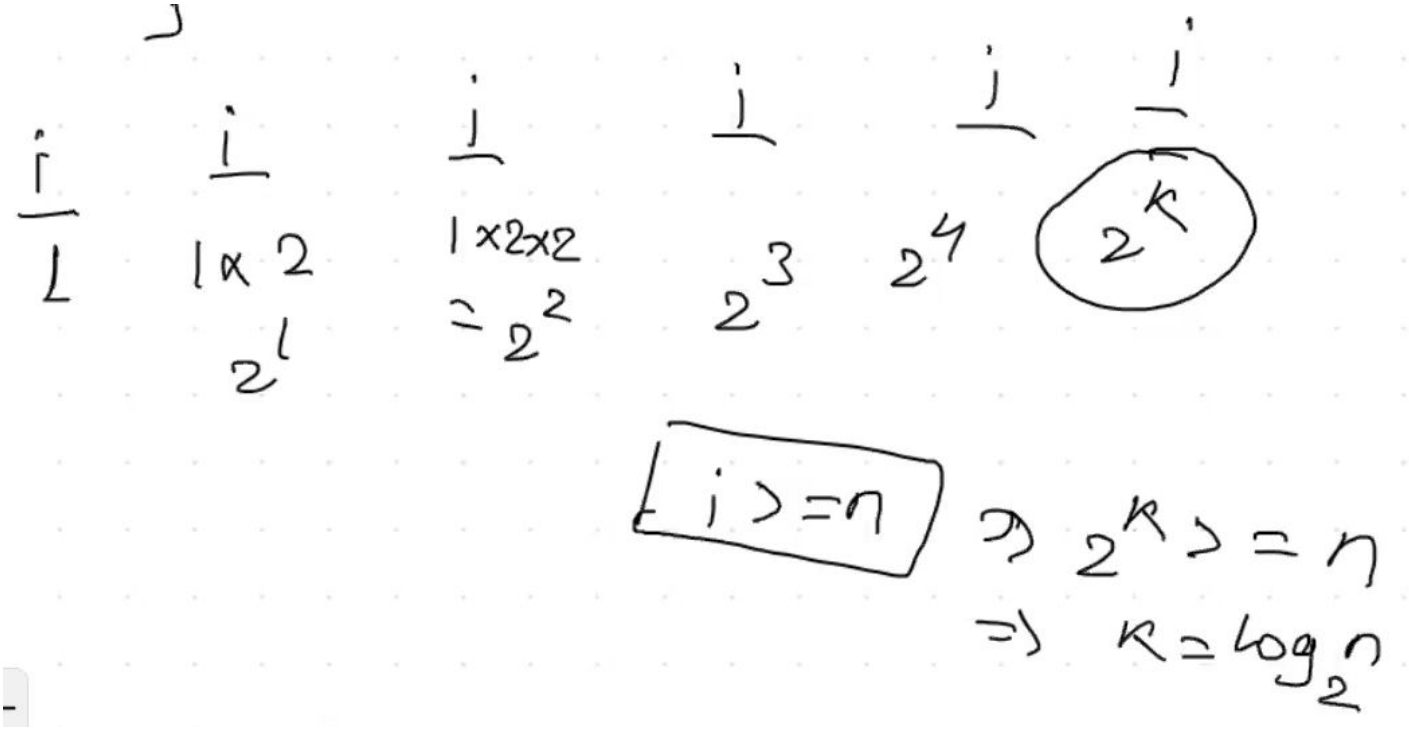
Question:

```
for (i=1 ; i<n; i = i*2){

    Statement

}
```

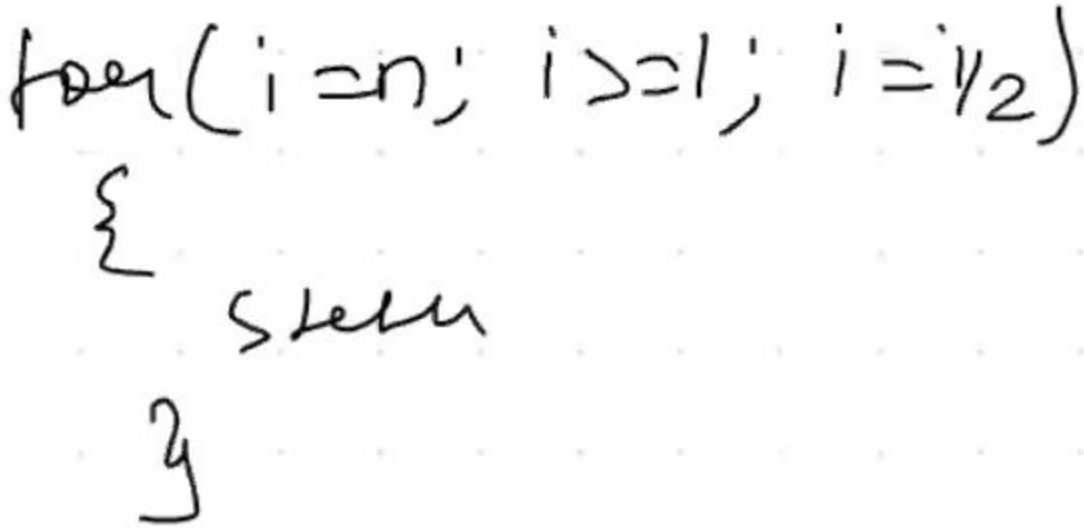
Solution:



The order is o(logn)

One thing is very important here whenever i is increasing in exponential order the time complexity is logn.

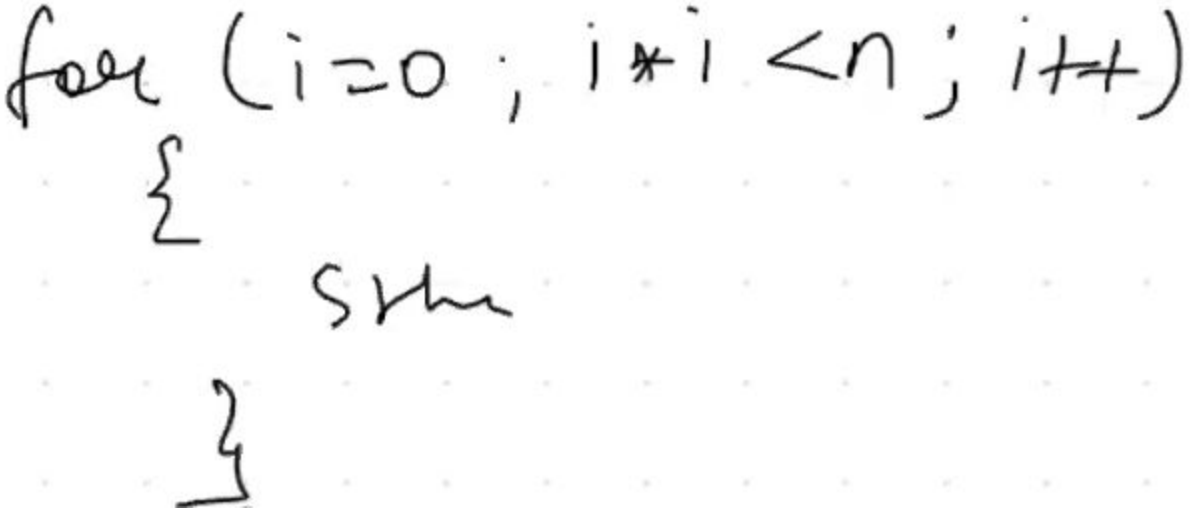
Question:



Solution: I = n , i = n/2, i = n/2^2, i = n/2^3 ----- n/2^k (i become i<1)

n/2^k <1 = > n < 2^k => 2^k > k = > k > logn => k = log n => o(log n)

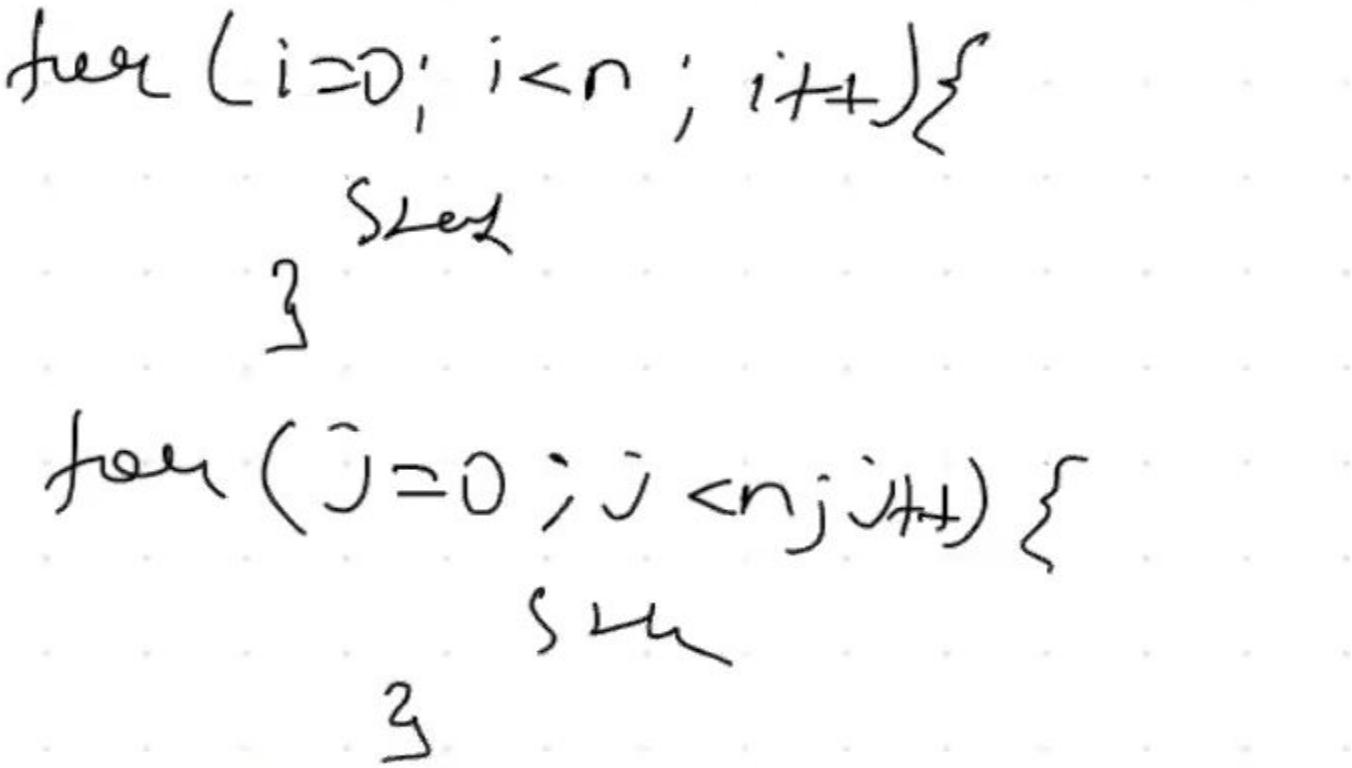
When i value is increasing or decreasing exponentially we can say order will be o(log n)
Question:



Solution :

i*i >= n
i^2 >= n
i^2 =n
I = square root n
o(square root of n)

Question:

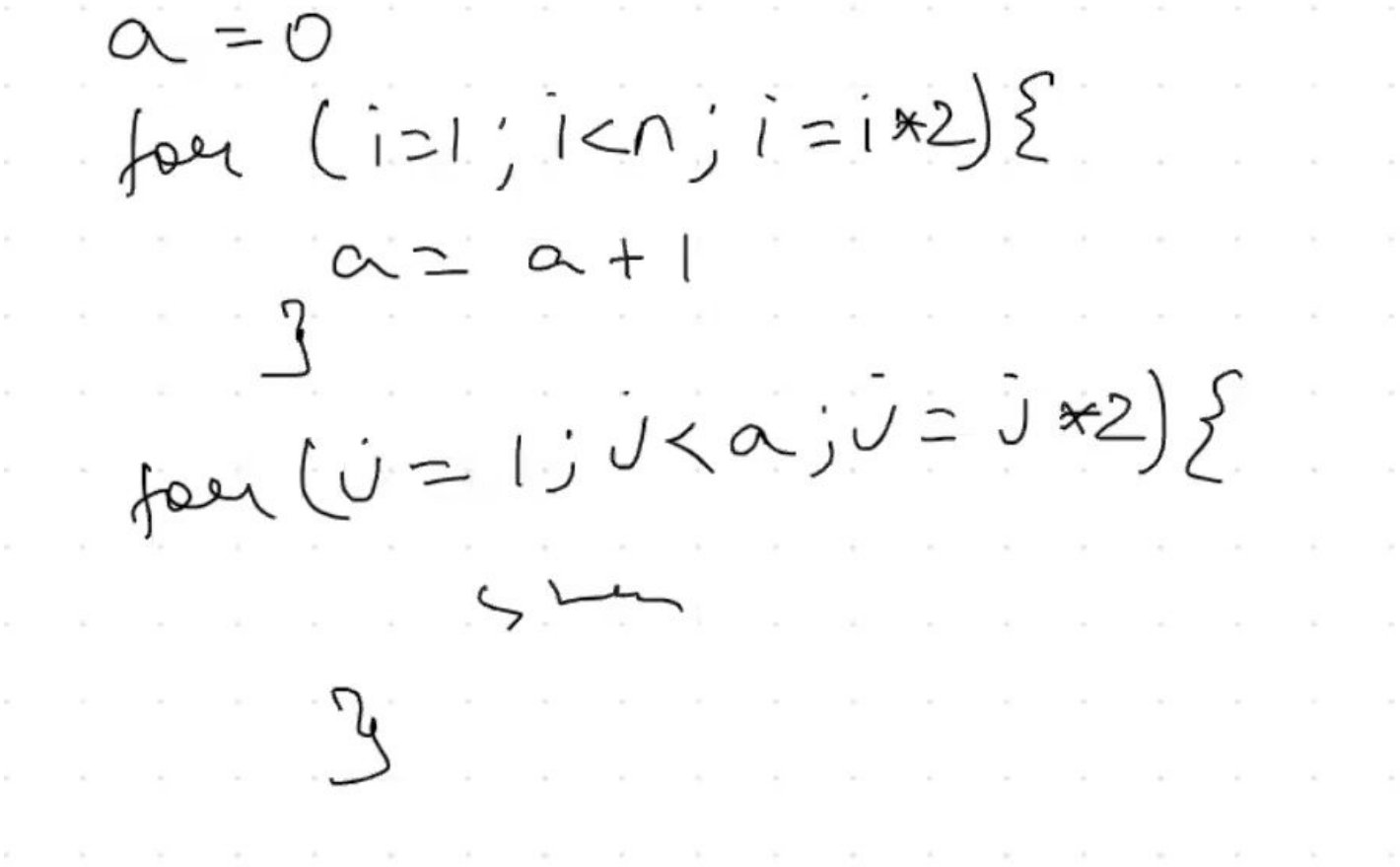


Solution: (bcoz i for loop not under in j for loop)
I run for n times and j run n times

n+n = 2n
o(n)

- **If i for loop under j for loop we can say n X n = n^2**

Question:



Solution:
I for log n times.
j for log a times
a value is incrementing every time so it will run log n times

The time complexity is o (log log n)

Question:

```
for( i=0; i<n; i++)
  {
    for(j=0; j<n; j=j*2)
      {
        Sum
      }
  }
```

Solution :
I for run n times
J for run log n times
The statement will run o (n log n)

for(i=0; i<n; i++) ⇒ O(n)

for(i=0; i<n; i=i+2) ⇒ O(n)

for (i=n; i>=1 ; i--) ⇒ O(n)

for (i=1; i<n ; i=i*2) ⇒ O($\log_2 n$)

for (i=1; i<n; i = i*3) ⇒ O($\log_3 n$)

for (i=n; i>1 ; i=i/2) ⇒ O(log n)

MCQ 1 :

```
for(i=n; i>=1; i--) {
    for(j=n; j>=1; j--){
        for(k=n; k>=1; k--) {
            Statement
        }
    }
}
```

Time Complexity

Attempted - 39 (125.81%) EASY

☑ n**3	69.23%
☐ n**2	12.82%
☐ log n	7.69%
☐ n	10.26%

MCQ 2 :

```
for(i=0; i<n; i++) {
        SLen
}
for(j=0; j<m; j++) {
        SLen
}
for (k=0; k<0; k++){
        SLen
}
```

Time Complexity

Attempted - 40 (129.03%) EASY

- [ ] n*m*o 25%
- [x] n+m+o 75%

MCQ 3:

```
i = 0
while (i<n)
    a = a+i
    i = i*2
```

Time Complexity — Attempted - 39 (125.81%) — EASY

☑ log n	74.36%
☐ n**2	10.26%
☐ n	15.38%

MCQ 4:

```
for (i=0; i<n; i=i*3){
    a = a+10
}
```