Logarithmic time algorithm

```
i=n;
while (i>0){
    body of the loop;
 i = i/2;
}
```

Complexity??

16 25

i*i<=16

```
for (i =1; i*i<n; i++){
    body of the loop;
}
```

$i^2 <= n$

$i <= \sqrt{n}$

Question Complexity is $O(\sqrt{n})$ $O(n^2)$

for (i=1; i <= n*n; i++)

n = 16

$1 <= 16$ ✓

$2*2 \leq 16$

$3*3 \leq 16$ ✓

$4*4 \leq 16$

$5*5 \leq 16$ — false

1. Constant time
2. Linear time
3. Quadratic time
4. Logarithmic time
5. Square Root
   - I - part
   - II - part

for Loop / while Loop

Problem Solving

n^2 is $\Theta(n^2)$

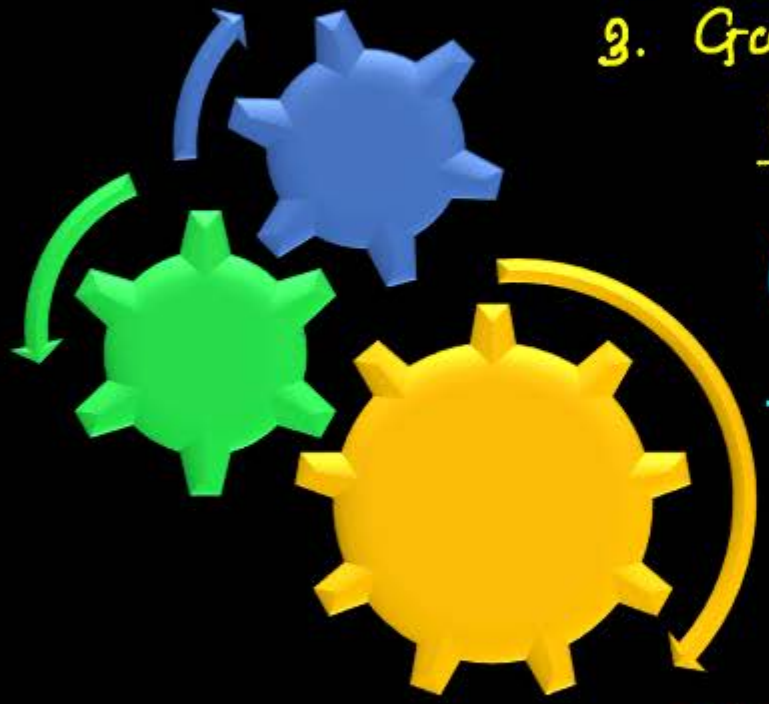
n^2 is $O(n^2)$ n^2 is $\Omega(n^2)$

1. Text book
2. Work book
3. Gate Question Bank

Work sheet - 1 – Asymptotic Notation

n^2+n+1 – $O(n^2)$ yes

n^2+n+1 – $\Theta(n^2)$



Core Idea

Asymptotic Notation

1. Confusing ∴ Comparing Mathematical function a, b, a>b, a==b, a<b

n^2 is $O(n^3)$

function $f(x) \leq g(x)$ 'O'

$f(x) \geq g(x)$ – Ω

$f(x) == g(x)$ – Θ

Algorithmic Complexity

for and while loop

program/function/fragment

Complexity??

Complexity : Counting No. of iteration

```
for (i =1; i<n*n; i++){
    body of the loop;
}
```

Square Root Algorithm

Time Complexity can be expressed as function of "input size" by using (RAM and step count)

1 Linear time

Square Root time algorithm

Squar Root time Algorithm $1+2+3+\cdots+4 \leq n$

```
S = 0; i=1;
while (S<=n){
 S =S+i;
 i = i+1;
}
```

$1^2+2^2+3^2+\cdots+k^2 \leq n$

$1^3+2^3+3^3+\cdots+k^3 \leq n$

for (S=0, i=1, S<=n; S = S+i, i++)

Bound

$<= n$ — $S = 0+1$, $i = 2$

$1 <= n$ — $S = 0+1+2$, $i = 3$

$0+1+2 \leq n$ — $S = 0+1+2+3$. $i = 4$

S. accumulation of values

$0+1+2+3 \leq n$ — $S = 0+1+2+3+4$, $i = 5$

but k^{th} iteration

(Last iteration) $0+1+2+3+\cdots+k \leq n$

$\frac{k(k+1)}{2} \leq n$

$\approx k^2 \leq n$

$k \leq \sqrt{n}$

Question

```
#include<stdio.h>
int main ( ){
int x,sum;
for ( x = 0, sum = 0; x <= 500; x+=10 )
      sum += x ;
printf("%d",sum);
return 0;
}
```

what is the value

0 + 10 + 20 + 30 + - - - + 500

10 (1 + 2 + 3 + - - + 50) → Arithmetic Series

$$\frac{10 \times 50 \times 51}{2} = 12750$$

Output of the C-program is 12750

Answer

When The first term and the last term is given

- $S = \frac{n}{2}\ (a_1 + a_n)$, where a_1 is the first term and a_n the last one.
- $S = \frac{n}{2}\left(2a_1 + d(n-1)\right)$
- The series is 0 , 10 , 20 , 30, …. 500, number of terms 51

Question

Consider the following code

```
int P = 0;
for(i= 1; i < 2n; i++) {
        for(j = 1, j<=n; j++){
                if(j < i) P = P+1;
        }
}
printf( "%d", P);
```

n = 4

i	j	P
1	1 →	0
2	2	1
3	3	2
4	4	3
5 →		4
6		4
7		4

6+12 = 18

What is the output printed by the above code in terms of n?

$3 \times 16 - 12$

$48 - 12 = \frac{36}{2} = 18$

(A) $\frac{4n^2 - n}{2}$

(C) $\frac{n^2 - 4n}{2}$

(B) $\frac{3n^2 - 3n}{2}$ ✓

(D) $\frac{n^2 - 3n}{2}$

(B)

Question

Consider the following code

```
int P = 0;
for(i= 1; i < 2n; i++) {
        for(j = 1, j<=n; j++){
                if(j < i) P = P+1;
        }
}
printf( "%d", P);
```

What is the output printed by the above code in terms of n?

(A) $\frac{4n^2 - n}{2}$

(C) $\frac{n^2 - 4n}{2}$

(B) $\frac{3n^2 - 3n}{2}$

(D) $\frac{n^2 - 3n}{2}$

Summation form

UB − LB + 1

2n − 1 − n + 1 = (n − 1)

i	j	—
i = 1	j = 1	0
i = 2	j = 2	1
i = 3	j = 3	2
⋮	⋮	
i = n	j = n	n − 1
i = n + 1	—	n
i = n + 2	—	
⋮		
i = 2n − 1	—	n

$0 + 1 + 2 + 3 + \cdots + (n-1) + \underbrace{n + n + \cdots + n}_{n-1}$

$\frac{n(n-1)}{2} + n(n-1)$

$\frac{n(n-1)}{2}\left[\frac{1}{2} + 1\right] = \frac{3(n^2 - n)}{2}$

$= \frac{3n^2 - 3n}{2}$

Level-1 Question 6 :Work Book

Practice time

06. Which choice gives the best Θ - notation for the amount of time used by the code segment below?

```
for (i = 1; i <= n; i++)
    for (j = 1; j <= n; j = 2 * j)
        x = x + 1;
```

(a) Θ(log n)
(b) Θ(n)
(c) Θ(n log n)
(d) Θ(n²)

(c)

n

log n

j = j++
j = j+2

j = 1
j = 2
j = 2²
j = 2³

j = 2^k <= n

take the log₂

log₂ 2^k = log₂ n

k = log₂ n

Level-1 Question 8 :Work Book

08. Which choice gives the best Θ-notation for the amount of time used by the code segment below?

```
for (i = 1; i<=n; i++)
    for (j = 1; j<=n; j = j +i)
        x = x +1;
```

(a) $\Theta(\log n)$ (b) $\Theta(n)$

(c) $\Theta(n \log n)$ (d) $\Theta(n^2)$

Hold will come back to this question today we will complete

Level-1 Question 7 :Work Book

07 Which choice gives the best Θ - notation for the amount of time used by the code segment below?

```
for (i = 1; i <= n; i++)
    for (j = n/3; j <= 2n; j+ = n/3)
        x = x + 1;
```

(a) Θ(log n) (b) Θ(n)

(c) Θ(n log n) (d) Θ(n²)

Constant No. of time

$\approx 2n - n/3 \approx 5n/3$

$2n - n/3 + 1 = 5n/2 + 1$

$= \frac{5n}{3} \times \frac{3}{n} = 5$

$j = 1$

$j = 1 + n/2 \le n$

$j = 1 + n/2 + n/2 \quad n+1 \le n$

Suppose $n/1 = n$

```
for (i = 1; i <= n; i++)
    for (j = 1; j <= n; j = j + n/2)
```

Inner for Loop?

$n/n/2 \quad n \times 2 = 2n$

No. of times Inner for loop is execule

$j = j + 10$

$n/10$

Level-2 Question 1 :Work Book

07. Which choice gives the best Θ - notation for the amount of time used by the code segment below?

```
for (i = 1; i <= n; i++)
    for (j = n/3; j <= 2n; j+ = n/3)
        x = x + 1;
```

(a) Θ(log n)
(b) Θ(n)
(c) Θ(n log n)
(d) Θ(n^2)

n/3 ← Linear

5n/3 / n/3 = 5

for (i=1; i<=n; i++) — ≈ n

for (i=1, i<=n; i=i+2) — n/2

1 3 5 7 — · · — n

for (i=i i<=n i = i+10) — n/10

for (i=1 i i<=n i = i+n/2)

n/n/2 = 2

Level-1 Question 9 :Work Book

D is is Answer

09. What is the time complexity of the following program?

```
void Fun (int n)
 {
     int i;
     for (i = 1; i *i ≤ n; i++)
     count ++;
 }
```

(a) $\Theta(n \log n)$ (b) $\Theta(n^2)$

(c) $\Theta(\log \log n)$ (d) $\Theta(\sqrt{n})$

Level-2 Question 1 :Work Book

01. Which choice gives the Θ - notation for the amount of time used by the code segment below?

```
for (i = n/2; i <= n; i++)
    for (j = 1; j <= n; j+ = n/2)
        for (k = 1; k <= n; k * =2)
        x = x + 1;
```

(a) Θ(n log n)
(b) Θ(n)
(c) Θ(n log²n)
(d) Θ(n² log n)

$n - n/2 + 1$

$n/2 + 1$ ← n ← Linear time

Constant

$\log_2 n$

n x Constant x $\log_2 n$

Level-2 Question 2 :Work Book

02. Which choice gives the best Θ-notation for the amount of time used by the code segment below?

```
for (i = n/2; i <= n; i++)
    for (j = 1; j <= n; j *= 2)
        for (k = 1; k <= n; k *= 2)
            x = x + 1;
```

(a) Θ(n log n)
(b) Θ(n)
(c) Θ(n log² n)
(d) Θ(n² log n)

$n/2 * \log n * \log_2 n$

$n/2 \ (\log n)^2$

$n/2 \log^2 n$

$\Theta(n \log^2 n)$

Level-2 Question 3 :Work Book

03. Which choice gives the best Θ-notation for the amount of time used by the code segment below?

```
for (i = 1; i <= n/3; i++)
    for (j = 1;j <= n; j += 4)
        x = x +1;
```

(a) Θ(log n)
(b) Θ(n)
(c) Θ(n log n)
(d) Θ(n²)

[D]

n/3 - 1 + 1 — n/3

n/4

n/3 * n/4 = n²/12

θ (n²)

Level-2 Question 4 :Work Book

04. Which choice gives the best Θ-notation for the amount of time used by the code segment below?

```
for (i =1; i < = n; i++)
{
  for (j = 1; j < = n; j++)
    x =x +1;
  for (j = 1; j < = n; j =2*j)
    x = x +1;
}
```

(a) Θ(log n)
(b) Θ(n)
(c) Θ(n log n)
(d) Θ(n²) ✓

Not Nested
Independent
max (n, log n)
n

Question Related
Asymptotic complexity
from work book

n(n + log n)

n² + n log n = Θ(n²)

① drop the constant

GATE 2007 Question 45

Consider the following C code segment:

```
int IsPrime(n){
        int i,n;
        for(i=2;i<=sqrt(n);i++)
        {if(n%i == 0){
            printf("Not Prime\n"); return 0;}
        }
        return 1;
    }
```

Let $T(n)$ denote the number of times the *for* loop is executed by the program on input n. Which of the following is TRUE?

(A) $T(n)=O(\sqrt{n})$ and $T(n)=\Omega(\sqrt{n})$

(B) $T(n)=O(\sqrt{n})$ and $T(n)=\Omega(1)$

(C) $T(n)=O(n)$ and $T(n)=\Omega(\sqrt{n})$

(D) None of the above

terminate before

upper bound $\sqrt{n}$

prim No : 17, 19

$O(n^2)$ $O(n)$ $\Theta(n^2)$ $\Theta(n)$

Practice Question - 9

Big 'O' - Notation

upper bound

maximum time taken by algorithm

Big Ω - Lower bound

How early can algorithm can terminate

$O(n)$ - $\Omega(1)$

Lower Bound

Constant No. time

$O(\sqrt{n})$

$\Omega(1)$

Linear Search

GATE 2007 Question 45

Let $T(n)$ denote the number of times the *for* loop is executed by the program on input n. Which of the following is TRUE?

(A) $T(n)=O(\sqrt{n})$ and $T(n)=\Omega(\sqrt{n})$

(B) $T(n)=O(\sqrt{n})$ and $T(n)=\Omega(1)$ (B)

(C) $T(n)=O(n)$ and $T(n)=\Omega(\sqrt{n})$

(D) None of the above

Lower Bound : Mininimum programming steps taken by the Algorithm.

GATE 2013, Question Number 42

Consider the following function

```
int unknown(int n){
      int i, j, k = 0;
      for (i = n/2; i <=n;i++)
            for (j=2; j<= n;j=j*2;
                  k=k+n/2;
      return (k);
}
```

The return value of the function is

(A) $\Theta(n^2)$ (B) $\Theta(n^2 \log n)$ (C) $\Theta(n^3)$ (D) $\Theta(n^3 \log n)$

GATE 2013, Question Number 42

Consider the following function:

```
int unknown(int n){
int i, j, k=0;
for (i=n/2; i<=n; i++)
    for (j=2; j<=n; j=j*2)
        k = k + n/2;
return (k);
}
```

The return value of the function is

a) $\Theta(n^2)$

b) $\Theta(n^2 \log n)$

c) $\Theta(n^3)$

d) $\Theta(n^3 \log n)$

(13) $\Theta(n^2 \log n)$ — Returned value

— Complexity. $n \log n$

$n/2 + n/2 + n/2 + n/2 + n/$

$n/2$

$\approx \log_2 n$

what value of k is returned by Algorithm

$n/2 + n/2 + n/2 + \cdots$ ~

$k = 0 + n/2 + n/2 + n/2 + \cdots + n/2$ (for Loop $\log_2 n$ times) $= n/2 \log n$

$\approx$ Approximated outer for loop

$\downarrow$

$\approx n/2 \left(\frac{n}{2} * (\log_2 n) \right)$

$\approx \frac{n^2}{4} \log_2 n$ $\Theta(n^2 \log n)$

GATE 2013, Question Number 42

The return value of the function is

a) $\Theta(n^2)$

b) $\Theta(n^2 \log n)$

c) $\Theta(n^3)$

d) $\Theta(n^3 \log n)$

GATE 2018, Question Number 32, 1-Mark

Consider the following C code. Assume that unsigned long int type length is 64 bits.

```
unsigned long int fun(unsigned long int n){
  unsigned long int i,   j = 0,   sum = 0;
  for(i = n;  i > 1;   i = i/2)
  for( ;j > 1;   j = j/2)
        sum++;
  return(sum);
}
```

The value returned when we call fun with the input 2^{40} is

(A) 4 (B) 5 (C) 6 (D) 40

Develope - this queshon
* Memorizahon *

j = 40

$40 > 1$	—	1
$20 > 1$	—	2
$10 > 1$	—	3
$5 > 1$	—	4
$2 > 1$	—	5
$1 > 1$		

formula

Something - $\log_2 n$

$2^{40} > 1$	—	1
$2^{39} > 1$	—	2
$2^{38} > 1$	—	3
$2^{1} > 1$	—	40
$1 > 1$	—	

$\lfloor \log \log_2 n \rfloor$

O Θ

Bound Θ

GATE 2007 Question 15

Q. 12 Text Book / work sheet

Consider the following segment of C-code:

```
int j, n;
    j = 1;
    while (j <=n)
        j = j*2;
```

Count

No. of comparison in the execution of the loop.

The number of comparisons made in the execution of the loop for any $n > 0$ is:

(A) $\lceil \log_2 n \rceil + 1$ (B) n

(C) $\lceil \log_2 n \rceil$ (D) $\lfloor \log_2 n \rfloor + 1$

Ceiling function $\lceil 5/2 \rceil = \lceil 2.5 \rceil = 3$

Floor function $\lfloor 5/2 \rfloor = \lfloor 2.5 \rfloor = 2$

GATE 2007 Question 15

Consider the following segment of C-code:

```
int j, n;
j = 1;
while (j <=n)
    j = j*2;
```

The number of comparisons made in the execution of the loop for any $n > 0$ is:

(A) $\lceil \log_2 n \rceil + 1$ (B) n

(C) $\lceil \log_2 n \rceil$ (D) $\lfloor \log_2 n \rfloor + 1$

2020/21

$\log_2 n$

Tests Not count exit comparison

3 C

n=2 — 1 + 1 + 1 ← exit (1<=2) (2<=2) (4<=2) iteration

n=3 — 1 + 1 + 1 ← exit (1<=3) (2<=3) (4<=3) (Not counting exit comparison)

n=4 — 1 + 1 + 1 + 1 (1<=4) (2<=4) (4<=4) (8<=4

n=7 — 1 + 1 + 1 + 1 (1<=7) (2<=7) (4<=7) (8<=

$\log_2 n$

Has to be rejected

GATE 2015 Set-I, Question Number 54

Consider the following C function.

```
int fun1 (int n) {
  int i, j, k, p, q = 0;
  for (i = 1; i < n; ++i)
  {
    p = 0;
    for (j = n; j > 1; j = j/2)
      ++p;
    for (k = 1; k < p; k = k * 2)
      ++q;
  }
  return q;
}
```

Which one of the following most closely approximates the return value of the function fun1?

a) n^3

b) $n(\log n)^2$

c) $n\log n$

d) $n\log(\log n)$

Linear time

n

Not Nested

$\log_2 n$

$p = \log_2 n$

$\log_2 p$

$\log_2 \log_2 n$

approximation

No Need to be mathematically accurate.

* Complexity : $\Theta(n \log_2 n)$
* Returned value

$n(\log_2 n \pm \log_2 \log_2 n) = \Theta(n \log_2 n)$

$n * (\log_2 \log_2 n)$ = q is incremented $n \log \log_2 n$

$n \log^2 n$

GATE 2015 Set-I, Question Number 54

Which one of the following most closely approximates the return value of the function fun1?

a) n^3

b) $n(\log n)^2$

c) $n \log n$

d) $n \log(\log n)$

GATE 2006, Question Number 15

Consider the following C-program fragment in which i, j and n are integer variables.

```
for (i = n,j = 0;i>0;i/= 2,j+=i);
```

Let $val(j)$ denote the value stored in the variable j after termination of the *for* loop. Which one of the following is TRUE?

(A) $val(j)=\theta(\log n)$

(B) $val(j)=\theta\left(\sqrt{n}\right)$

(C) $val(j)=\theta(n)$

(D) $val(j)=\theta(n\log n)$

GATE 2006, Question Number 15

Consider the following C-program fragment in which i, j and n are integer variables.

```
for(i=n,j=0; i>0; i/=2, j+=i );
```

Let `val(j)` denote the value stored in the variable j after termination of the for loop. Which one of the following is true?

Analysis

8 Question C-program question No. 4

n/2

1 H.W

Work book

90% Gate question + Text book

$n = 2^k$

GATE 2006, Question Number 15

a) val(j)=Θ(logn)

b) val(j)=Θ(√n)

c) val(j)=Θ(n)

d) val(j)=Θ(nlogn)

2^K

What is time complexity of fun ()?

```
int fun (int n)
    {
        int count = 0;
        for (int i = n; i > 0; i /=2)
            for (int j = 0; j < i; j++)
                count + = 1;
        return count;
    }
```

Problem Solving

work book Level - I
question - 8

GATE 2017

Consider the following C function

```
int fun (int n) {
  int i, j;
  for (i = 1; i <= n; i++) {
    for (j = 1 ; j < n ; j+=i) {
      printf ("%d %d , i, j ) ;
    }
  }
}
```

Linear time

$n/2$

Time complexity of fun in terms of θ notation is

(A) $\theta(n\sqrt{n})$

(B) $\theta(n^2)$

(C) $\theta(n\log n)$

(D) $\theta(n^2 \log n)$

$i=1$ — $j=1; j<n, j=j+1$ — n

$i=2$ — $j=1\ j<n, j=j+2$ — $n/2$

$i=3$ — $j=1\ j<n, j=j+3$ — $n/3$

$i=4$

$\vdots$

$i=n$ $\quad j=1\ j<n\ \ j=j+n$ — $n/n = 1$

① — for (j=1, j < i, j++)

for (j=n; j < i, j++) ←

$T_n = \frac{n}{1} + \frac{n}{2} + \frac{n}{3} + \cdots + \frac{n}{n}$

$= n\left(\frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}\right)$

$\approx n \cdot \log n = \theta(n\log n)$

GATE 2017

Time complexity of fun in terms of θ notation is

(A) $\theta (n\sqrt{n})$

(B) $\theta(n^2)$

(C) $\theta (n\log n)$

(D) $\theta(n^2 \log n)$

for loop

Beyond this - clear understanding

then - complexity question

Returned value : 1 more question

Homwork