Today's Content

TC-2
- → Big O - O( )
- → TLE - (Time Limit Exceed Error)
- → Why TLE?

TC-1 → { How to Calculate iterations } → 15/17

↳ pen / paper

Basic Maths Revision

Quiz 1 : Sum of Natural Numbers :

$$S_N = 1 + 2 + 3 + \ldots N = \frac{(N)(N+1)}{2}$$ → { gauss story }

Quiz 2 : [3 10] → { 3 4 5 6 7 8 9 10 } ⇒ 8 elements

[4 8] = { 4 5 6 7 8 } ⇒ 5 elements

Number of elements from [a b] included

Quiz 3 : [a b] = b − a + 1 → #imp

GP basic : → [When we divide any 2 consecutive elements ratio have to same]

↳ Geometric Progression

S_1 = 3 6 12 24 48 96 ....

ratio = $6/3 = \frac{12}{6} = \frac{24}{12} = \frac{48}{24} = \frac{96}{48} = \ldots\ 2$

S_2 = 2 6 18 54 162 .....

ratio = $6/2 = 18/6 = 54/18 = \frac{162}{54} = \ldots\ 3$

// given G.P

First term = a

Common ratio = r

1^{st}	2^{nd}	3^{rd}	4^{th}	...	$N-2^{nd}$	$N-1^{th}$	N^{th}
a	ar^1	ar^2	ar^3	...	ar^{N-3}	ar^{N-2}	ar^{N-1}

// Given

First term = a ; S_N = sum of N Terms in G.P

Common ratio = r

$$S_N = a + ar + ar^2 + ar^3 + \dots ar^{N-3} + ar^{N-2} + ar^{N-1}$$

#Terms = N

$$S_N = a + ar + ar^2 + ar^3 + \dots ar^{N-3} + ar^{N-2} + ar^{N-1}$$

Multiply with r on both sides

$$rS_N = \cancel{ar} + \cancel{ar^2} + \cancel{ar^3} + \cancel{ar^4} + \cancel{\dots} \cancel{ar^{N-2}} + \cancel{ar^{N-1}} + ar^N$$

$-$

$$S_N = a + \cancel{ar} + \cancel{ar^2} + \cancel{ar^3} + \dots \cancel{ar^{N-3}} + \cancel{ar^{N-2}} + \cancel{ar^{N-1}}$$

$$rS_N - S_N = ar^N - a$$

$$S_N[r-1] = a[r^N - 1]$$

$$\boxed{S_N = \frac{a[r^N-1]}{r-1}}$$

// Sum of N terms in G.P

Ex:

$a=5$, $r=2$, $N=5$

Series = 5 10 20 40 80 = 155

$$S_5 = \frac{5 * [2^5 - 1]}{2-1} \Rightarrow \frac{5*31}{1} \Rightarrow 155$$

// **log basics**:

$\log_b a = c$ # $\underline{b^c = a}$ → To what power we need to raise b to get a

$\log_2 8 = 3 \rightarrow 2^3 = 8$

$\log_2 2^5 = 5 \rightarrow 2^5 = 2^5$

$\log_3 81 = 4 \rightarrow 3^4 = 81$

$$\boxed{\underline{N} = 2^k \longrightarrow k = \log_2 N}$$

$\log_2 N \Rightarrow \log_2 2^k \Rightarrow k$

Q1

```
void func(int N){
    int s = 0;
    i = 0 ; i <= 100; i++){
        s = s + i
    }
}
```

a b → b - a + 1

$i : [0 \quad 100] \rightarrow 100 - 0 + 1 \rightarrow 101$

101 iterations

Q2)

```
void func(int N){
    int s = 0;
    i = 35; i <= 87; i = i + 1)
        s = s + i
    }
}
```

a b b - a + 1

$i = [35 \quad 87] = 87 - 35 + 1$

$= 53$

→ #53 iterations

Q3)

```
void func(int N){
    int s = 0;
    i = 1 ; i <= N ; i++){
        s = s + i }
    }
}
```

a b b - a + 1

$i = [1 \quad N] = N - 1 + 1 \rightarrow N$

N iterations

Q4) void func(int N, int M){

```
int s = 0;
i = 1; i <= N; i++){
    if( i % 2 == 0){
        s = s + i
    }
}
i = 1; i <= M; i++){
    if( i % 2 == 1){
        s = s + i
    }
}
}
```

N times

M times

Total iterations

(N + M) iterations

Q6) void func(N){

int s = 0

i = 1 ; i*i <= N; i++){

s = s+i

}

}

$i*i <= N \Rightarrow i^2 <= N \Rightarrow i <= \sqrt{N}$

a b $\Rightarrow$ b-a+1

$i : [1, \sqrt{N}] \Rightarrow \sqrt{N} - 1 + 1 \Rightarrow \sqrt{N}$

#iterations = $\sqrt{N}$ iterations

Q7) void fun(int N){

```
int i = N
while( i > 1){
    i = i/2
}
}
```

$i = N$

iterations	i value after each iteration
1	$i = i/2$, $N/2$: $N/2^1$
2	$i = i/2$, $N/4$: $N/2^2$
3	$i = i/2$ $N/8$: $N/2^3$
4	$i = i/2$ $N/16$: $N/2^4$
5	$i = i/2$ $N/32$: $N/2^5$

After k iterations, $\left[i = N/2^k \right]$

N	i
10	$10 \to 5 \to 2 \to$ ($i == 1 > 1$ break)
19	$19 \to 9 \to 4 \to 2 \to$ ($i == 1 > 1$ break)

Obs1: When our i reaches 1 breaks

Ass: Above code breaks after y iterations?

$i = 1$ && $i = N/2^y$ } $a = b$ & $a = c \Rightarrow b = c$

$N/2^y = 1$

TODO:

$N = 2^y$, $y = \log_2^N$

// After $\log_2^N$ iterations code breaks

$i = N \to N/2 \to N/4 \to N/8 \to N/16 \to$ --- ($i == 1 > 1$ break) iterations $\log_2^N$

```
void fun(int N){
    int i = N
    while(i > 0){
        i = i/2
    }
}
```

→ TODO & try it out

→ Number of iterations ?

→ 0 > 0 → {No}

8Q)

```
void fun(int N){
    int s = 0
    i = 0; i <= N; i = i*2){
        s = s + i
    }
}
```

$i = 0$

iterations	i value after each iterat
1	$i = i*2$, $i = 0$
2	$i = i*2$, $i = 0$
3	$i = i*2$, $i = 0$
4	$i = i*2$, $i = 0$
5	$i = i*2$, $i = 0$
⋮	

∞ loop

8:22am → 8:32am

9Q) → {Near approximation}

```
void fun(int N){
    int s = 0
    i=1 ; i<=N ; i = i*2){
        s = s+i
    }
}
```

⇒ $\log_2^N + 1$ iterations

$i = 1$

iterations	i value after each iteration
1	$i = i*2$ $\;i = 2 \to 2^1$
2	$i = i*2$ $\;i = 4 \to 2^2$
3	$i = i*2$ $\;i = 8 \to 2^3$
4	$i = i*2$ $\;i = 16 \to 2^4$
5	$i = i*2$ $\;i = 32 \to 2^5$

After k^{th} iteration = $i = 2^k$

N	i		iterations
16	$1 \xrightarrow{} 2 \to 4 \to 8 \to 16 \to 32 < 16$ {break}	+1 +1 +1 +1 +1	5
25	$1 \to 2 \to 4 \to 8 \to 16 \to 32 < 25$ {break}	+1 +1 +1 +1 +1	5
14	$1 \to 2 \to 4 \to 8 \to 16 < 14$ {break}	+1 +1 +1 +1	4

$\log_2^{25} = 4+1 = 5$

$i > N$

Obs1: If $i > N$ loop breaks

Ass: Say after k iteration loop breaks

$i = 2^k$, $i > N$ ⇒ [i value greater than $\log_2^N$]

$2^k > N$ $\quad k > \log_2^N$ } ⇒ $\boxed{k = \log_2^N + 1}$ → after $\log_2^N + 1$ $i > N$ breaks

10Q)

```
void func( ){
    i = 1; i <= 4; i = i+1){
        j = 1; j <= 3; j = j+1){
            print ("Hello World") ?
        }
    }
}
```

→ Table:

i	j:	Total iterations
1	j: [1 3]	3 +
2	j: [1 3]	3 +
3	j: [1 3]	3 +
4	j: [1 3]	3
5	Break	Total = 12 iterations

11Q)

```
void func(N){
    i = 1; i <= 10; i = i+1){
        j = 1; j <= N; j = j+1){
            print()
        }
    }
}
```

i	j	Total iterations
1	[1 N]	N +
2	[1 N]	N +
3	[1 N]	N
⋮		⋮ +
10	[1 N]	N
11	{break}	Total itera = 10N

12) void func(N){

```
for(i = 1; i <= N; i = i+1){
    for(j = 1; j <= N; j = j+1){
        print()
    }
}
}
```

i	j	Total iterations
1	[1 N]	N
2	[1 N]	+ N
3	[1 N]	+ N
:		+
N	[1 N]	N
N+1	break	

Total iterations = N^2

13) void func(N){

```
for(i = 1; i <= N; i = i+1){
    for(j = 1; j <= i; j = j+1){
        print()
    }
}
}
```

i	j : [1 i]	Total iterations
1	[1 1]	1
2	[1 2]	+ 2
3	[1 3]	+ 3
4	[1 4]	+ 4 +
:		
N	[1 N]	N
N+1	break	

Total iterations

$$\boxed{S = 1 + 2 + 3 + 4 + \cdots N}$$

Sum of first N Natural Numbers

$$S = \frac{(N)(N+1)}{2} = \frac{N^2 + N}{2}$$

14) void func(N){

```
void func(N){
    i = 1; i <= N; i = i+1){
        j = 1; j <= N; j = j*2){
            print()
        }
    }
}
```

i	j : [1 N]	Total iterations
1	[1 N]	$\log_2^N + 1$
2	[1 N]	$\log_2^N + 1$
3	[1 N]	$\log_2^N + 1$
⋮		
N	[1 N]	$\log_2^N + 1$
N+1	{Break}	

Total iterations

$N * [\log_2^N + 1]$

Q15 void func(N){

```
void func(N){
    i = 1; i <= 2^N; i = i+1){
        print()
    }
}
```

→ a b ⇒ b − a + 1

$i \Rightarrow [1 \quad 2^N] \Rightarrow 2^N - 1 + 1 \Rightarrow 2^N$

2^N iterations

Q16

```
void func(N){
    i=1; i<=N; i=i+1){
        j=1; j<=2^i; j=j+1){
            print()
        }
    }
}
```

$\rightarrow 2[2^N - 1]$

i	j : [1 2^i]	Total iterations
1	[1 2^1]	2^1
2	[1 2^2]	2^2
3	[1 2^3]	2^3
4	[1 2^4]	2^4
:		
N	[1 2^N]	2^N
N+1	{Break}	

Total iterations =

$2^1 + 2^2 + 2^3 + \dots 2^N$

G.P

$$S_N = 2^1 + 2^2 + 2^3 + 2^4 + \dots 2^N$$

$a = 2$

$r = 2$

#terms = N

$$S_N \text{ in gp} = \frac{a * \{r^N - 1\}}{r-1} \rightarrow \frac{2 * \{2^N - 1\}}{2-1} \rightarrow 2[2^N - 1]$$

→ Sum of N Natural Numbers
Sum of N Terms in G.P
Logarithmic basics

→ {Learn}

Q17) void func(N){ → TODO

```
void func(N){
    i = N; i > 0; i = i/2){
        j = 1; j <= i; j++){
            print(i+j)
        }
    }
}
```

Comparism functions : { large N values }

$\log(N) < \mathrm{sqrt}(N) < N < N\log N < N\sqrt{N} < N^2 < N^3 < 2^N$

N^2 N^3 → higher order term

$N\log N$ N

→ higher order term

$N\sqrt{N}$ N^2 → higher order term

Big (O) → What?

→ Why?

→ Where do we use it?

→ Tuesday session

→ How to do we calculate Big(O) for any code

→ Calculate iterations ✓

→ Only take higher order term ✓

→ Neglect constant coe-fficients ✓

Ex1: Iterations $\Rightarrow$ $3N^2 + 5N + 10^4$ $\Rightarrow$ $O(N^2)$

Ex2: Iterations $\Rightarrow$ $5N^2 + 10N^3 + 6N\log N + 100$ $\Rightarrow$ $O(N^3)$

Ex3: Iterations $\Rightarrow$ $4N^2 + 3N + 10^6$ $\Rightarrow$ $O(N^2)$

Ex4: Iterations $\Rightarrow$ $4N + 3N\log N + 10^6$ $\Rightarrow$ $O(N \log N)$

Ex5 Iterations $\Rightarrow$ 10^3 $\Rightarrow$ Constant Iterations $\Rightarrow$ $O(1)$

$O(1)$ $\Rightarrow$ Means constant iterations

```
func(N) {
    i = 0; i <= 9; i++) {
        print(i)
    }
}
```

a b $\Rightarrow$ b - a + 1

$i : [0, 9] \Rightarrow 9 - 0 + 1 \Rightarrow$ 10 iterations

$\Rightarrow$ 10 iterations, constant $\Rightarrow$ $O(1)$

TODO: For all Questions we discussed calculate $O(\ )$