1) Time Complexity
2) Space Complexity
3) Big O
4) TLE → Time Limit Exceeded

} Next Class TC-2

Today's class → **# iterations**

Basic Maths

Q1 → sum of first N natural numbers ? $\frac{N*(N+1)}{2}$

$S = 1 + 2 + 3 \; ... \; + N$

$S = \underline{N + (N-1) + (N-2) + ... , 1}$

$2S = N*(N+1) \Rightarrow \boxed{S = \frac{N*(N+1)}{2}}$

Q → Numbers in range of [3 8] → {3, 4, 5, 6, 7, 8} $\underline{6}$

[L R] (L R] (L R)

L, L+1, ... R L+1, L+2 ... R L+1, L+2 ... (R-1)

[L R] → $\underline{R - L + 1}$ [3 8] → $8 - 3 + 1 = \underline{6}$

Q → # times N is divided by 2 to reach 1 → $\underline{\log_2 (N)}$

Arithmetic Progression

Eg → $\overset{a}{4}, 7, 10, 13, 16 \; ---$

$\underset{d}{3} \quad 3 \quad 3$ $16-13 = \underline{3}$

$a \quad (a+d) \quad (a+2d) \quad .... \quad a + (n-1)d$

Nth Term

$$s = a + (a+d) + (a+2d) + \ldots \quad a+(n-2)d + a+(n-1)d$$

$$s = a+(n-1)d + (a+(n-2)d) + a+(n-3)d + \ldots \quad a+d + a$$

$$2s = 2a+(n-1)d + (2a+(n-1)d) + \ldots \quad (2a+(n-1)d)$$ N times

$$2s = n * (2a+(n-1)d) \Rightarrow \boxed{s = \frac{n}{2}(2a+(n-1)d)}$$

Geometric Progression

$$\overset{a}{2}, 6, 18, 54 \ldots$$

$$\frac{6}{2} = 3 \qquad \frac{18}{6} = 3 \qquad \frac{54}{18} = \overset{r}{3}$$

$$a \quad a*r \quad a*r^2 \quad a*r^3 \; \ldots\ldots \; a*r^{n-1}$$

Nth term

$$rS = ar + ar^2 + ar^3 + \ldots\ldots ar^{n-1} + ar^n$$

$$-S = a + ar + ar^2 + \ldots \quad ar^{n-2} + ar^{n-1}$$

$$rS - S = ar^n - a$$

$$\Rightarrow S(r-1) = a(r^n - 1)$$

$$\Rightarrow \boxed{S = \frac{a(r^n-1)}{(r-1)}} \longrightarrow \boxed{\frac{a(1-r^n)}{(1-r)}}$$

$$3S = 6 + 18 + 54 + 162 \qquad a=2 \quad r=3 \quad n=4$$

$$S = 2 + 6 + 18 + 54$$

$$3S - S = 162 - 2 \Rightarrow 2S = 160 \Rightarrow S = \underline{80}$$ ✓

$$(6+18+54+162) - (2+6+18+54)$$

Q → What is $\log_a a^x \Rightarrow$ $\qquad \log_y \boxed{x} = 3 \Rightarrow y^3 = x$

$$\log_a \boxed{a^x} = \underline{x}$$

Number of Iterations

Q7 →

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

$i \rightarrow [1 \quad N] \Rightarrow$ **N iterations**

$TC = O(N)$

Q8 →

```
void solve (N, M) {
    for (int i = 1 ; i <= N ; i++) {
        if (i % 2 == 0)
            print (i)
    }
    for (int i = 1 ; i <= M ; i++) {
        if (i % 2 == 0)
            print (i)
    }
}
```

N

+

M

#iterations = $(N + M)$

$TC = O(N + M)$ or $O(\max(N, M))$

$7/2 = 3$ $\quad 8/2 = 4$ $\quad 9/2 = 4$ $\quad \frac{6+1}{2} = \frac{7}{2} = 3$

Q9 →

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

$TC = O(N)$

#iterations → $\frac{(N+1)}{2}$

int division

$i = 1 \rightarrow 3 \rightarrow 5 \rightarrow 7 \ldots.$

$\frac{5}{2} = 2$

$N = 4 \quad i = 1 \rightarrow 3 \rightarrow \cancel{5}$ #iterations = 2

$N = 5 \quad i = 1 \rightarrow 3 \rightarrow 5 \rightarrow \cancel{7}$ #iterations = 3 ✓

$N = 6 \quad i = 1 \rightarrow 3 \rightarrow 5 \rightarrow \cancel{7}$ #iterations = 3

Q10 →

```
int s=0;
for (int i=0; i <= 100; i++) {
        s = s + i + i*i;
}
return s;
```

$i \longrightarrow [0 \quad 100] \longrightarrow 100-0+1 = \underline{101}$ iterations

$TC = \underline{O(1)}$

Q11 →

```
int s=0;
for (int i=1; i*i <= N; i++) {
        s=s+i;
}
return s;
```

$i = 1 \longrightarrow i^2 = N$

$\Rightarrow i = \sqrt{N}$

$i \rightarrow [\underset{L}{1} \quad \underset{R}{\sqrt{N}}] \longrightarrow \underset{(R-L+1)}{\sqrt{N}-1+1} = \underline{\sqrt{N}}$ iterations

$TC = \underline{O(\sqrt{N})}$

Q12 →

```
int i = N;  // N>0
while (i > 1) {
      i /= 2;
}
```

$i = N \longrightarrow \frac{N}{2} \longrightarrow \frac{N}{4} \longrightarrow \frac{N}{8} \dots 1$

$\Rightarrow \underline{\underline{\log_2(N)}}$ iterations

optional

$TC = O(\log_2(N))$

OR

$\underline{O(\log(N))}$

$$\boxed{\log_b a = \frac{\log a}{\log b}}$$

$\log_4 64 = \frac{\log_2 64}{\log_2 4} \longrightarrow \frac{6}{2} = \underline{3}$

$\log_4 64 \longrightarrow \underline{3}$

$\log_2(N) = \frac{\log(N)}{\log(2)}$

Q13 →

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

$0 * 2 = \underline{0}$ $i = 0 \longrightarrow 0 \longrightarrow 0 \longrightarrow 0 \ldots \sim$

$\#$ iterations $= \underline{\infty}$

N=5 $i = 1 \to 2 \to 4 \to x$ $\log_2(5) = 2\ldots \to \underline{3}$

N=10 $i = 1 \to 2 \to 4 \to 8 \to x$ $\log_2(10) = 3\ldots \to \underline{4}$

N=8 $i = 1 \to 2 \to 4 \to X$ $\log_2(8) = \underline{3}$

Q14 →

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

$i = 1 \longrightarrow 2 \longrightarrow 4 \longrightarrow 8 \ldots\ldots \quad 2^K = N$

$\Rightarrow K = \log_2(N)$

$8 \to 4 \to 2 \to 1$

$1 \to 2 \to 4 \to 8$

$i \longrightarrow [1 \quad \log_2(N)] \longrightarrow \underline{\underline{\lceil \log_2(N) \rceil}}$ iterations

$TC = \underline{O(\log_2 N)}$

Q15 →

```
for (int i = 1; i <= 10 ; i++) {
    for (int j = 1; j <= N; j++) {
            print(i+j);
    }
}
```

i	j	#iterations
1	[1 N]	N
2	[1 N]	N
3	[1 N]	N
⋮		
10	[1 N]	N

$\underline{10 * N}$

$TC = \underline{O(N)}$

10:20 PM

Q16 →

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

i	j	#iterations
1	[1 N]	N
2	[1 N]	N
⋮		
N	[1 N]	N

$\underline{N * N}$

$TC = O(N^2)$

Q17 →

```
for (int i =0; i < N ; i++) {
    for (int j=0; j <= i; j++) {
        print(i+j);
    }
}
```

i	j	#iterations
0	[0 0]	1
1	[0 1]	2
2	[0 2]	3
⋮		
(N-1)	[0 N-1]	N

$1+2+3+...+N$

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

$TC = O(N^2)$

Q18 →

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

i	j	#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	$(\log_2(N)+1)$
		$N(\log_2(N)+1)$

$N = 8$ 1 → 2 → 4 → 8 $\log_2(8) = 3$

$N = 6$ 1 → 2 → 4 $\log_2(6) = 2..$ → 2

$TC = O(N \log_2(N))$

Q19 →

```
for (int i = 1 ; i <= 2^N ; i++) {
        print(i);
}
```

$i \rightarrow [1 \quad 2^N] \rightarrow 2^N$ iterations

$TC = O(2^N)$

Q20 →

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

i	j	#iterations
1	$[1 \quad 2^1]$	2^1
2	$[1 \quad 2^2]$	2^2
3	$[1 \quad 2^3]$	2^3
⋮		
N	$[1 \quad 2^N]$	2^N

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

$a = 2 \qquad r = 2 \qquad n = N$

$$\frac{2(2^N-1)}{(2-1)} = \underline{\underline{2(2^N-1)}}$$

$TC = \underline{O(2^N)}$

Q21 →

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

i	j	#iterations
N	$[1 \quad N]$	N
N/2	$[1 \quad N/2]$	N/2
N/4	$[1 \quad N/4]$	N/4
N/8	$[1 \quad N/8]$	N/8
⋮		
1	$[1 \quad 1]$	1

$N + N/2 + N/4 + \dots 1$

$a = N \qquad r = 1/2 \qquad n = \log_2(N)$

$$\frac{N*(1-(1/2)^{\log_2(N)})}{(1-1/2)} = \frac{N}{1/2}\left(1 - \frac{1}{\underbrace{2^{\log_2(N)}}_{2^x}}\right)$$

$\boxed{\log_2(N) = x} \Rightarrow 2^x = N$

$2^{\log_2(N)} = \underline{N}$

$$\Rightarrow 2N\left(1-\frac{1}{N}\right)$$

$$\Rightarrow \frac{2\cancel{N}(N-1)}{\cancel{N}} = \underline{\underline{2(N-1)}} \text{ #iterations}$$

$TC = \underline{O(N)}$

$$\boxed{x^{\log_x(N)} = N}$$

$2^{\log_2(8)} \rightarrow \underline{8}$

$\downarrow$

2^3

$\log_2(8) = 3 \Rightarrow 2^3 = 8$

For N=16, find →

$\log_2(N) = 4$

$\sqrt{N} = 4$

$N * \log_2(N) = 16 * 4 = 64$

$N^2 = 16 * 16 = 256$

$2^N = 65536$

$N! = 2 * 10^{13}$

$(1*2*3*..*N)$

$1*2*3*..*15*16$

$\log_2(N) < \sqrt{N} < N < N\log(N) < N^2 < 2^N < N!$ ←

Big O → $f(N) = 2N^2 - 3N + 10$ → quadratic $O(N^2)$

↳ rate of growth of function wrt input

1) $f(N) = N^3 + N\log(N) \longrightarrow O(N^3)$

2) $f(N) = 2^N + N^3 + N! \longrightarrow O(N!)$

3) $f(N) = 2N\log_2(N) + 3N + 100 \longrightarrow O(N\log_2(N))$

4) $f(N) = 4N\log(N) + 3N\sqrt{N} + 1000 \longrightarrow O(N\sqrt{N})$

TC = $\underline{O(\#\text{iterations})}$

$\log_a \boxed{a^x} = t$

$a^t = a^x$

$\Rightarrow \underline{t = x}$

$\boxed{\log_x y = 3 \Rightarrow x^3 = y}$