1) $\log n$

$\log_b n \Rightarrow k$

$b^k \Rightarrow n$

$\log_3 81 = 4$

$\underline{3^4 = 81}$

eg $\log_2 32 \Rightarrow 5$

$\underline{2^5 \Rightarrow 32}$

eg $\log_2 4 \Rightarrow 2$

eg $\log_2 2^{10} \Rightarrow 10$

$\log_2 n \Rightarrow$ number of times we need to divide n by 2 so that it reaches 1.

32
16
8
4
2
1

OR

number of time we need to multiply 1 by 2 so that it reaches n.

Numbers in Range $[3, 10]$

$\Rightarrow$ 3, 4, 5, 6, 7, 8, 9, 10

$\Rightarrow$ $[a, b] \Rightarrow b - a + 1$

3) Arithmetic Progression (A.P)

$\Rightarrow$ Sequence of number where successive terms differ by the same margin.

Eg 1, 4, 7, 10, 13, 16 Difference $\Rightarrow$ Common Diff (d)

Generalize: $a, a+d, a+2d \ldots\ldots$

nth term. $\Rightarrow a + (n-1)d$: n starts 1.

Sum of 1st n terms : $\frac{n}{2}\left[2a + (n-1)d\right]$

4) Geometric progression.

→ Sequence of number, where the next term can be found by multiplying the previous term by a fixed number.

↳ Common Ratio.

Eg: 2 4 8 16 32

Generalize: $a, ar, ar^2, ar^3 \ldots ar^{n-1}$

Sum of 1st n terms → $\dfrac{a(r^n-1)}{r-1}$

$\Downarrow$

$r<1$

$[1, 1/2, 1/4, 1/8, 1/16]$

$\dfrac{a(-1)}{r-1}$

→ $\dfrac{a}{1-r}$

1 2 4, 8, 16 ...... n

Q1

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

$\Rightarrow [1, N] \Rightarrow N - 1 + 1$

$\Rightarrow N$ times.

$\Rightarrow \boxed{O(N)}$

Q2

```
for (i = 0 ; i <= 100 ; i++) {
}
```

$[0, 100] = 100 - 0 + 1$

$\Rightarrow 101$

$\Rightarrow \boxed{O(1)}$

Q3

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

$[0, n]$ $\quad i = 0, 2, 4, 6\ 8, 10$

$\simeq n/2$

$\Rightarrow 1/2 \times n$

$\Rightarrow \boxed{O(n)}$

Q4

```
for (i=1, i*i <= N ; i++) {
}
```

$[1, \sqrt{N}]$

$i^2 = N$

$i = \sqrt{N}$

$= \sqrt{N} - 1 + 1$

$\Rightarrow \sqrt{N} = N^{1/2}$

$= O(\sqrt{N})$

Q5

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

$i = N, N/2, N/4, N/8, N/16 \ldots\ldots 1$

$\Rightarrow \boxed{\log_2 N} \Rightarrow O(\log n)$

32, 16, 8, 4, 2, 1

Q6

```
for (i=0 ; i<=N ; i=i*2) {
}
```

$i = 0, 0, 0, 0, 0, 0$

$\Rightarrow$ infinite

Q7

```
for (i=1 ; i<=N ; i=i×2) {
}
```

$i = 1, 2, 4, 8, 16 \ldots\ldots N$

$\Rightarrow \boxed{\log_2 N}$

$\Rightarrow O(\log N)$

Nested loops

Q1 for (i = 1 ; i ≤ 10 ; i++) {

for (j = 1 ; j ≤ N ; j++) {

}

}

i	j	No of iteration
1	[1, N]	N
2	[1, N]	N
3	[1, N]	N
⋮		
10	[1, N]	N

10N

⇒ O(N)

Q2 for (i = 1 ; i ≤ N ; i++) {

for (j = 1 ; j ≤ N ; j++) {

}

}

i	j	No of iteration
1	[1, N]	N
2	[1, N]	N
3	[1, N]	N
⋮		
N	[1, N]	N

N^2

$\Rightarrow O(N^2)$

Q_3 for (int i=1 ; i $\leq$ N ; i++) {

for (int j=1 ; j $\leq$ N ; j=j×2) {

$\equiv$

}

}

i	j	No of iteration
1	logn	logn
2	logn	logn
3	logn	logn
⋮		
N	logn	logn

$N \log N$

$\Rightarrow O(N \log N)$

$\underline{\underline{Q_3}}$ for (i = 1 ; i ≤ N ; i++) {

for (j = 1 ; j ≤ (2^i) ; j++) {

—

}

}

i	j	No of iteration
1	$[1, 2^1)$	2^1
2	$[1, 2^2]$	2^2
3	$[1, 2^3]$	2^3
⋮		
N	$[1, 2^N]$	2^N

$\frac{a(r^n - 1)}{r - 1}$

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

$a = 2$ $\quad S_n \Rightarrow \frac{2(2^n - 1)}{2 - 1}$

$r = 2$

$n = N$

$\boxed{2 \times 2^n - 2 = O(2^n)}$ $\Rightarrow 2(2^n - 1)$

How to calculate Big O from the number of iterations.

1) Ignore lower order terms.
2) Ignore Constants

Q No of iterations $\Rightarrow 4n^2 + 3n + 2$

1^{st} Step: Ignore lower order terms

$$\Rightarrow 4n^2$$

2^{n} Step: Ignore Constants.

$$4n^2 = Cn^2$$

$$\Rightarrow n^2$$

$$O(n^2)$$

Q_2 $\quad 3N\sqrt{N} + 4\log N + 3N\log N$

$2^{32} \times 32 = 2^{32} \times 2^5 = 2^{37}$

N	$N\sqrt{N}$	$N\log N$
2^{32}	2^{48}	2^{37}
2^{64}	2^{96}	2^{70}

$N = 2^{32}$ $\quad N\sqrt{N} \Rightarrow 2^{32} \times \sqrt{2^{32}}$

$2^{32} \times 2^{16}$

$2^{32+16} = 2^{48}$

$\log_2 2^{32} \Rightarrow 32$

Step 1 : Ignore lower order terms

$= 3N\sqrt{N}$

Step 2 : Ignore Constants

$= N\sqrt{N}$

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