Time Complexity - 1

- → Time & Space complexity
- → Asymptotic Analysis
- → Big O notation
- → TLE - Time limit Exceeded

} Not Today (next class)

Today: How to calculate no. of iterations ?

Quiz 1 : Sum of first N natural no. ?

$(N*(N+1))/2$

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

if use formula = O(1)

Quiz 2 : How many numbers are there in range [3,10] ?

[ ] → closed bracket / inclusive

( ) → open bracket / exclusive

[3,10] → 3, 4, 5, 6, 7, 8, 9, 10 → 8 numbers

[3,8) → 3, 4, 5, 6, 7, ~~8~~ → 5 numbers

(3,8) → 4, 5, 6, 7 → 4 numbers

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

$[a, b) \rightarrow b - a$

$(a, b) \rightarrow b - a - 1$

$(a, b] \rightarrow b - a$

what if $a > b$?

↳ invalid input

$(a, a) = 0$

$[3, 3] = 3$ → 1

$[-7, -4] = -7, -6, -5, -4$

$-4 - (-7) + 1$

$= 4$

Quiz 3: How many times do we need to divide N by 2 to reduce it to 1 ?

if $N = 10$

$10 \xrightarrow{/2} 5 \xrightarrow{/2} 2 \xrightarrow{/2} 1 \Rightarrow$ 3 times

$N \xrightarrow{/2} N/2 \xrightarrow{/2} N/4 \xrightarrow{/2} N/8 \rightarrow \dots \rightarrow 1$

assume no. of times need to divide = K

$N \xrightarrow{1^{st}} N/2 \xrightarrow{2^{nd}} N/2^2 \xrightarrow{3^{rd}} N/2^3 \dots \xrightarrow{K^{th}} N/2^K$

after K times, N becomes 1

$$\frac{N}{2^K} = 1 \Rightarrow N = 2^K$$

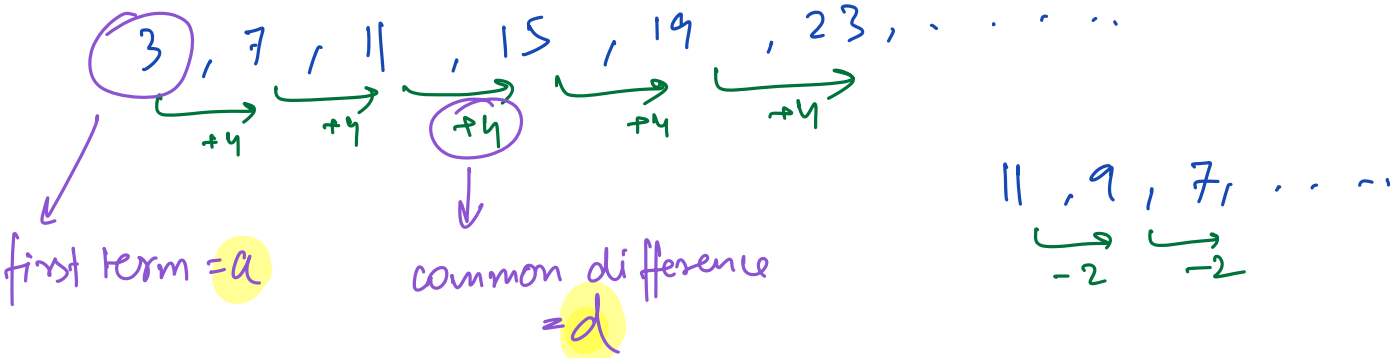
$$\log_2 N = \log_2 2^K$$

$$\log_2 N = K$$

$$K = \log_2 N$$

$$\log_a a^c = c$$

Arithmetic Progressions (AP)



N^{th} term of the AP ?

a , $a+d$, $a+2d$, ...

1st 2nd 3rd

$a + (n-1)d$

N^{th}

Sum of first N terms of an AP ?

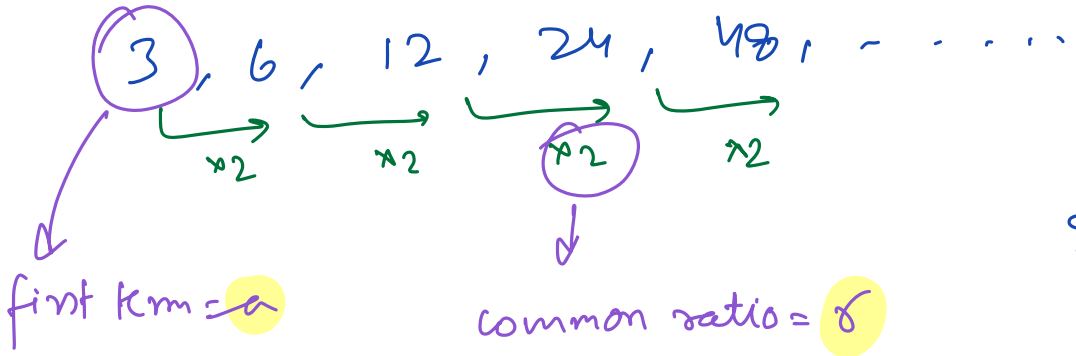
$$\frac{n}{2}\left[2a + (n-1)d\right]$$

$$\frac{n}{2}\left[a + \frac{a + (n-1)d}{\text{last term}}\right]$$

N^{th} term (last term) = l

$\frac{n}{2}(a+l)$ → n^{th} term
→ first term

Geometric Progression (GP)



8 4 2 1
→ 1/2 → 1/2 → 1/2

N^{th} term of a GP ?

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

1st 2nd 3rd n^{th}

Sum of first N terms of a GP ?

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

$$a\left(\frac{r^n - 1}{r-1}\right) \quad r \neq 0, 1$$

Quiz 4

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

i = 1, 2, 3, ... , N

i = [1, N]

count = N − 1 + 1 = N

O(N)

```
func(N, M) {
    for (i=1; i<=N; ++i) {        // N iterations
        if (i%2 == 0)
            print(i)
    }
    for (i=1; i<=M; ++i) {        // M iterations
        if (i%2 == 1)
            print(i)
    }
}
```

N/2 + M/2

N + M/2

N/2 + M

N + M

N × M

O(N+M)

Quiz 5:

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

i = 0, 1, 2, ... , 100

= [0, 100]

count = 100 - 0 + 1 = 101 $O(1)$

Quiz 6

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

$i*i <= N$

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

$i = 1, 2, 3, \ldots, \sqrt{N}$

$= [1, \sqrt{N}]$

count = $\sqrt{N} - 1 + 1 = \sqrt{N}$ $O(\sqrt{N})$

Quiz 7

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

$i = N, N/2, N/4, \ldots$

count = no. of times taken to divide N by 2 to reach 1.

$O(\log N)$

count = $\log_2 N$

Quiz 8

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

$i=0 \xrightarrow{*2} 0 \xrightarrow{*2} 0 \cdots \cdots \to 0$

Infinite

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

}
```

$i=1 \xrightarrow{*2} 2 \xrightarrow{*2} 4 \longrightarrow \cdots \cdots \to N$

iteration = $\log_2 N$

Quiz 9

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

i	j	iteration
1	[1,N]	N
2	[1,N]	N
3	.	.
.	.	.
.	.	.
.	.	.
10	[1,N]	N

~~1N~~ = 10N

$O(N)$

instead of $i<=10$

if $i<=N$

$\Rightarrow N^2$

Quiz 10

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

i	j : [0, i]	iterations
0	[0,0]	0-0+1 = 1
1	[0,1]	2
2	[0,2]	3
⋮		⋮
N-1	[0,N-1]	N

~~N~~

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

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

Quiz 11

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

i	j	iterations
1	[1,N] 1,2,4,8,16,...	~~N~~ $\log_2 N$
2	1,2,4,...	$\log_2 N$
3		1
⋮		⋮
N		$\log_2 N$

$\Rightarrow$ $N * \log_2 N$ $O(N \log N)$

Quiz 12

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

  }
}
```

i	j : [1, 2^i]	iteration
1	[1, 2^1]	$2^1 - 1 + 1 = 2^1$
2	[1, 2^2]	$2^2 - 1 + 1 = 2^2$
3	[1, 2^3]	2^3
...		...
N	[1, 2^N]	2^N

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

GP : $a = 2$ $r = 2$ term = N

$$\text{sum} = a\left(\frac{r^n - 1}{r - 1}\right) = 2\left(\frac{2^n - 1}{2 - 1}\right) = 2(2^N - 1)$$

$O(2^N)$

Some basics

$\log_2 N$ < $\text{sqrt}(N)$

$N = 8$ $\log_2 N = 3$ $\text{sqrt}(N) = 2$

$N = 2^{10}$ $\log_2 N = 10$ $\text{sqrt}(N) = 2^5 = 32$

$N = 2^{20}$ $\log_2 N = 20$ $\text{sqrt}(N) = 2^{10} = 1024$

$1 < \log_2 N$

$N < N \log_2 N$

$N > \text{sqrt}(N)$

$N^2 < 2^N$

$$1 < \log_2 N < \sqrt{N} < N < N \log_2 N < N\sqrt{N} < N^2 < N^3 < 2^N < N!$$

How to write Big O ?

~~What~~ ?

~~Why~~ ?

1. Calculate iterations based on input.
2. Neglect lower order terms.
3. Neglect constant coefficient term

$N^2 + N \Rightarrow O(N^2)$

$N^4 + N^3 + 2N^2 \Rightarrow O(N^4)$

$10N^2 + 2N\log N + 5 \Rightarrow O(N^2)$

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