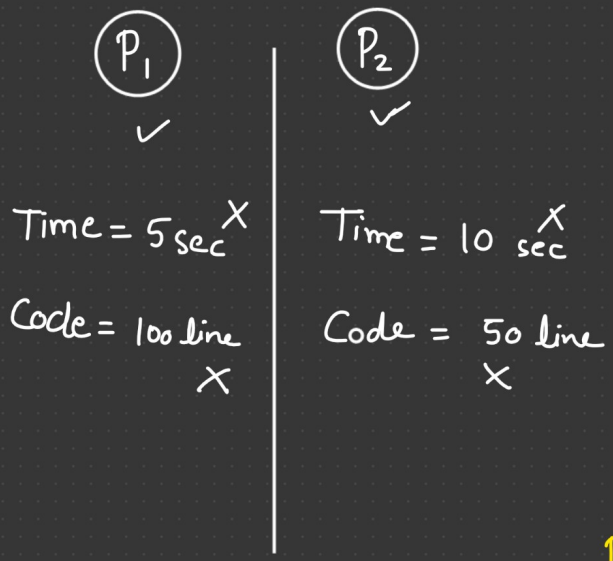
Time Complexity & Space Complexity



Rate of Growth

Type of eqn:-

1) Linear eqn:- $y = mx + c \Rightarrow ax + by + c = 0$
2) Quadratic eqn:- $y = ax^2 + bx + c$
3) Cubic eqn:- $ax^3 + bx^2 + cx + d = y$
4) Bi-quadratic eqn:- $ax^4 + bx^3 + cx^2 + dx + e = y$
5) Logarithmic eqn:- $a \log x + b$
6) Expo. eqn:- $a \cdot e^{bx}$
7) Polynomial eqn:- $a_n x^n + a_{n-1} x^{n-1} + \cdots - a_0 = y$

$f(x) = 3x + 5$

$f(1) = 3(1) + 5 = 8$

$f(2) = 3(2) + 5 = 11$

$\vdots$

$f(10) = 3(10) + 5 = 35$

$\vdots$

$f(1000) = 3(1000) + 5 = 3005$

$\vdots$

$f(10000) = 3(10000) + 5 = 30,005 \approx 30,000$

$f(x) = ax + b \approx (x)$

5

$f(x) = ax + b$

↓

for higher value of x.
if x is very very large
then we can ignore
a & b.

$f(x) = ax^2 + bx + c$

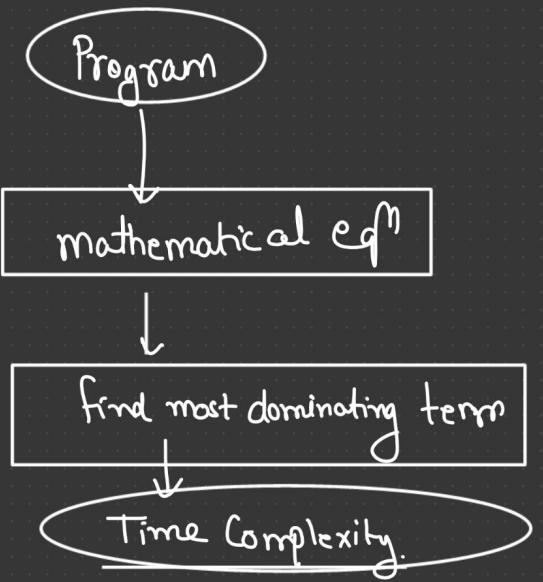
$f(x) = 3x^2 + 2x + 5$

$f(1) = 3 + 2 + 5 = 10$

$\vdots$

$f(10000) = 3x^2 + 2x + 5 \approx (x^2)$

↓



$\log \log x < \log x$

$2+2=4$ → fast → constant time → 1

$3^{\sin 36^\circ} + 2 \times \cos 72^\circ + 5 = ?$

$1 < \log x < x < x \log x < x^2 < x^3 < x^4 < e^x$ → for large value of x.

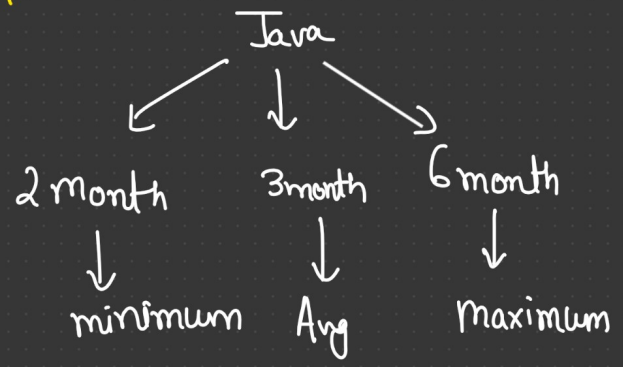
$1 < \log 10 < 10 < 10 . \log 10 < 100 < 1000 < 10000 < e^{10}$

Rate of Growth

1) Best case:- Ω → omega (lower)

2) Average Case:- Θ → Theta (Tight)

3) Worst Case:- O → Big-O (Upper)



1, 2, 3, 4, 5, 6

Time Complexity & its Name

1) Constant ⟶ $O(1)$, $\Omega(1)$, $\Theta(1)$
2) logarithmic ⟶ $\log n$
3) Linear ⟶ n
4) Quadratic ⟶ n^2
5) Cubic ⟶ n^3
6) exponential ⟶ e^n, 2^n, 3^n

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

$O(n) \longrightarrow O(\log n)$

1) Constant Time :-

```
main()
{
    int x, y, z;      ⟶  3 sec
    z = x+y;          ⟶  2 sec
    print(z);         ⟶  1 sec
}                        6 sec
```

independent of input size

y = 6 → constant time

O(1)

```
main()
{
  int a, b, c, d, e, f;          ⟶  6 sec
    a = b+c+d;                   ⟶  3 sec
    f = 2*a - 3*b + c-d/e;       ⟶  10sec
    print(a, b, c, d);           ⟶  2sec
}                                   21 sec → constant time
```

↳ O(1)

2) Linear Time:-

This program depends on input size.

```
main()
{
    int a,b,c,d; int n;        ----> 5 sec      O(1)
    a = b+c;                   ----> 2 sec
    print(a);                  ----> 1 sec

    for(int i=0; i<n; i++)     ----> 2n+10 sec  O(n)
    {
        a++;                   ----> n sec
    }
    print(a);                  ----> 1 sec
}
```

$\max\{O(1), O(n)\} = O(n)$

$3n+20 \rightarrow$ Linear $\rightarrow O(n)$

```
main()
{
    int a,b,c;                 ----> O(1)
    a = b+c;

    for( int=0; i<n/2; i++)    ----> n/2 + 1 ≈ O(n)
    {
        a++;
    }

    for( int j=0; j<n; j++)    ----> O(n)
    {
        b++;
    }
    print(a,b);                ----> O(1)
}
```

$\max\{1, n, n, 1\} = O(n)$

$O(1) + O(n) + O(n) + O(1) = 2n+2 \rightarrow O(n)$

Both loops are independent

```
for( i = 0;  i < n/3 ; i++)
 {
   ==
 }

 for ( j = 0 ;  j < n ; j++)
 {
   ==
 }

 for( k = n ;  k >= 0 ;  k = k-2)
 {
   ==
 }
```

$\left.\begin{matrix} \\ \\ \end{matrix}\right\} \frac{n}{3}$

$+$

$\left.\begin{matrix} \\ \\ \end{matrix}\right\} n$

$+$

$\left.\begin{matrix} \\ \\ \end{matrix}\right\} \frac{n}{2}$

$\left.\begin{matrix} \\ \\ \\ \\ \\ \\ \end{matrix}\right\} \frac{n}{3} + \frac{n}{2} + n$

$= \frac{11n}{6} \approx \underline{\underline{O(n)}}$

3) Quadratic time:-

```
main()
{
    int a, b, c;
    a = b + c;
    print(a);
    for(i=0; i<n; i++)
    {
        for(j=0; j<m; j++)
        {
            a++;
        }
    }
}
```

`a = b + c; print(a);` → O(1)

`for(i=0; i<n; i++)` → n

`for(j=0; j<m; j++)` → n×m

`a++;` → n×m

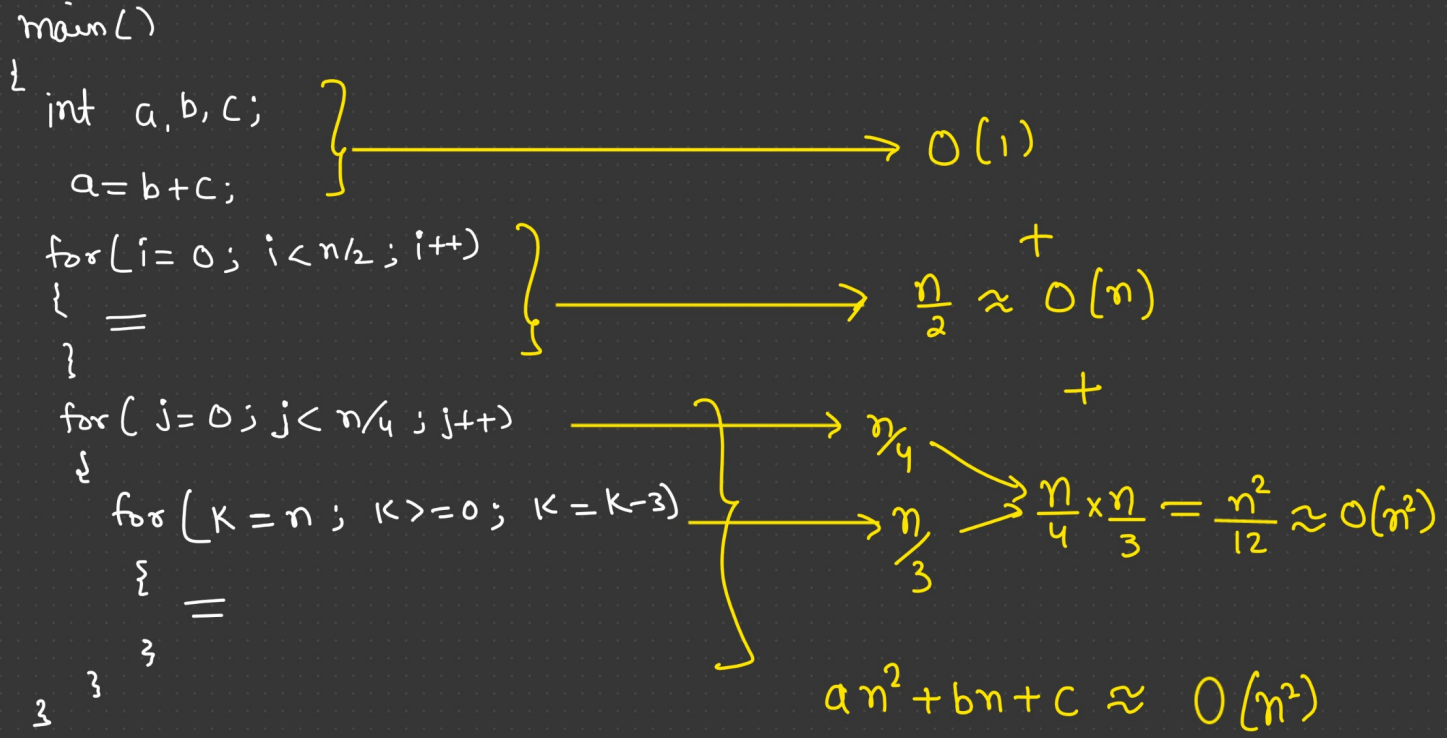
inner loop depends on outer loop

+

i=0	(0 − m) − m
i=1	(0 − m) − m
i=2	(0 − m) − m
⋮	⋮
i=n	(0 − m) − m
	n×m

$O(n \times m) \approx O(n^2)$

$\Rightarrow \underline{\underline{O(n^2)}}$


main()
{
int a, b, c;
a = b + c;
for(i = 0; i < n/2; i++)
{
=
}
for(j = 0; j < n/4 ; j++)
{
for(K = n ; K >= 0; K = K-3)
{
=
}
}
}
O(1)
+
n/2 ≈ O(n)
+
n/4
n/3
n/4 × n/3 = n²/12 ≈ O(n²)
an² + bn + c ≈ O(n²)

4) Cubic :-

```
for ( i= 0; i<n; i++)
{
   ==
}
```
$\longrightarrow O(n)$

$+$

```
for( a= 1;  a < n/3 ; a++)
{
   for ( b=0;  b< n/4 ; b++)
   {
      for ( c=n ;  c >=0;  c = c-4)
      {
      }
   }
}
```
$\longrightarrow n/3$, $\longrightarrow n/4$, $\longrightarrow n/4$

$\frac{n}{3} \times \frac{n}{4} \times \frac{n}{4} \simeq O(n^3)$

$O(n) + O(n^3) = \underline{O(n^3)}$

$2^{10} = 1024$

1, 2, 4, 8, 16, 32, 64, ...

2^0 2^1 2^2 2^3 2^4 2^5 2^6

$i = i+2$ $\longrightarrow$ linear growth $\rightarrow 2n$

```
for( int i=1 ;  i <= n ;  i = i*2)
{
   ==
}
```
$\longrightarrow$ K times $\longrightarrow O(K)$

$\rightarrow O(\log n)$

$2^K = n$

$\log(2^K) = \log n \Rightarrow K \log 2 = \log n$

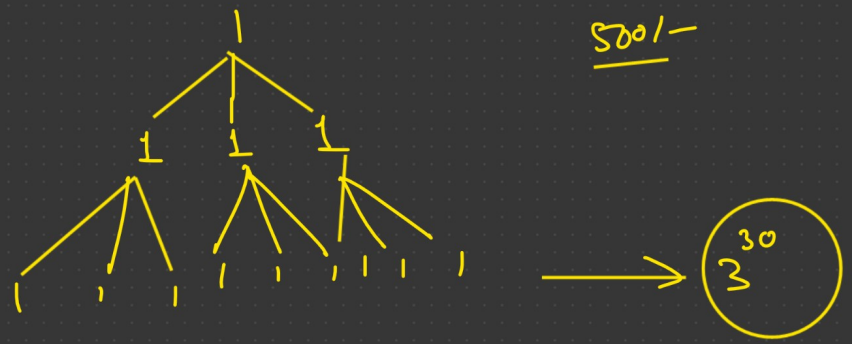
$K = \log n / \log 2 = \log_2 n$

$1+2+2+2+2 \longrightarrow$ Linear growth

2 rupees each day $\longrightarrow$ 30 days $\Rightarrow$ 60/-

$1, 2, 4, 8, \ldots\ldots, 2^9, 2^{10}, \text{----}, 2^{29} \Rightarrow \dfrac{1\cdot(2^{30}-1)}{2-1} \approx 2^{30}$

$2^0, 2^1, \ldots\ldots 2^{29} = \dfrac{a(r^n-1)}{r-1} =$



500/-

```
for (i=1 ; i<n ; i = i*3)
{
   ==
}
```

$\longrightarrow$ K times

$$3^k = n$$

$$\log 3^k = \log n$$

$$k \log 3 = \log n$$

$$k = \frac{\log n}{\log 3} = \log_3 n \approx O(\log n)$$

```
for( i=0;  i<n ; i++)
{
  for( j=1 ; j<n ; j=j*2)
  {
    ==
  }
}
```

$\longrightarrow$ n

$\longrightarrow$ $\log n$

$n \log n$

```
for( i=0 ; i<n ; i=i+2)
{
  for (j=1 ; j<n ; j=j*4)
  {
    for ( k=n ; k>=1 ; k=k/2)
    {
      =
    }
  }
}
```

i → n

j → $\log_4 n$

k → $\log_2 n$

$n \times \log_4 n \times \log_2 n = n(\log n)^2$

$n, \frac{n}{2}, \frac{n}{4}, \frac{n}{8}, \dots 1$

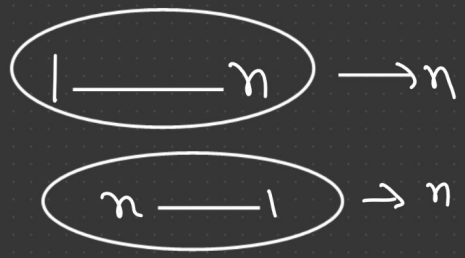
$\frac{n}{2^0}, \frac{n}{2^1}, \frac{n}{2^2}, \dots \frac{n}{2^k} = 1$

$\Rightarrow n = 2^k$

$\log n = \log 2^k$

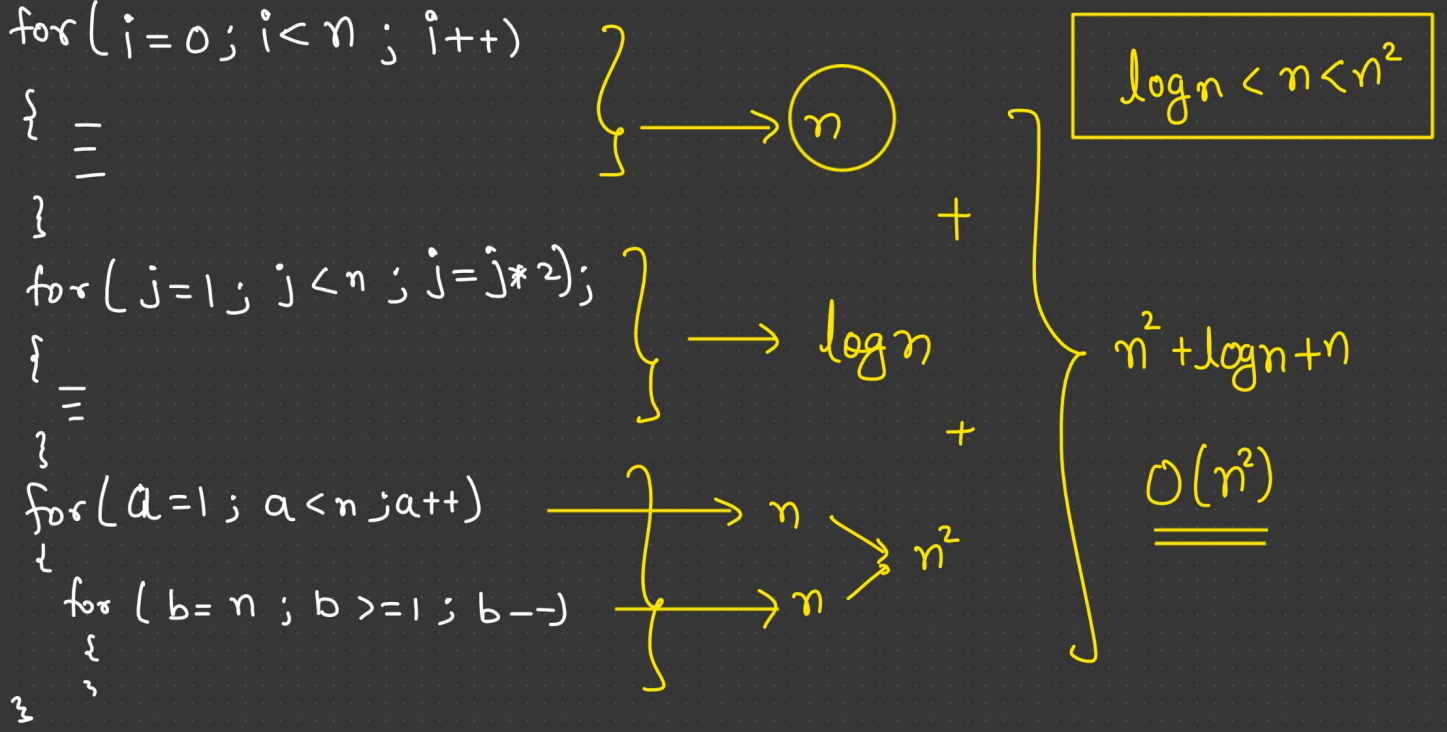
$\log n = k \log 2$

$k = \log_2 n$



$\to n$

$\to n$


for(i=0; i<n ; i++)
{
=
=
}
for(j=1; j<n ; j=j*2);
{
=
=
}
for(a=1; a<n ;a++)
{
for (b= n ; b >=1 ; b--)
{
}
}
n
logn
n
n
n²
+
+
logn < n < n²
n² + logn + n
O(n²)