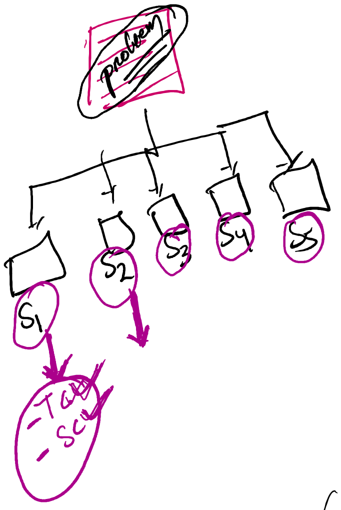
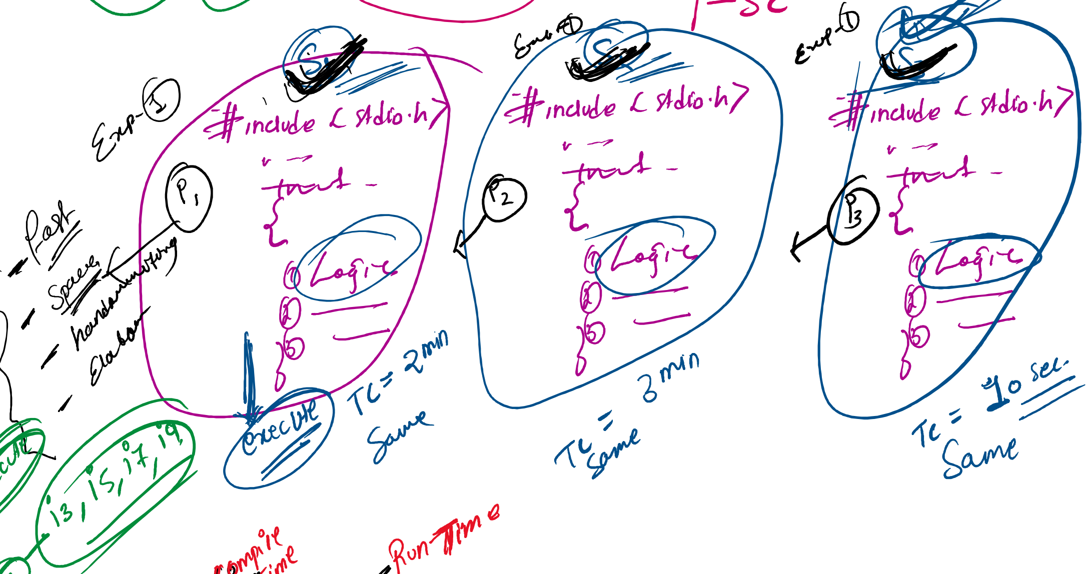
Analysis of Algorithms

- Time Complexity
- Space Complexity

⇒ If the problem Contains more than 1 Solution best one will be decided by analysis based on 2 - Factors

- TC
- SC

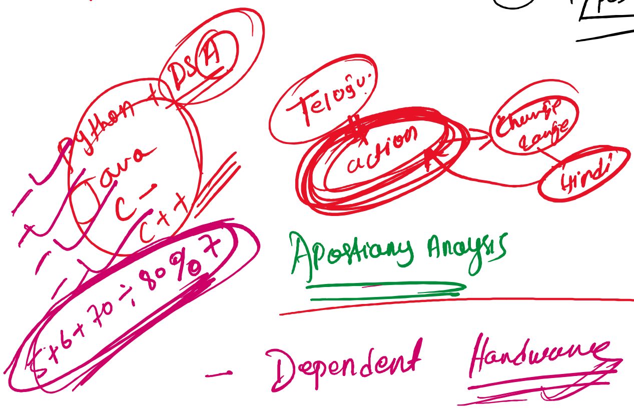


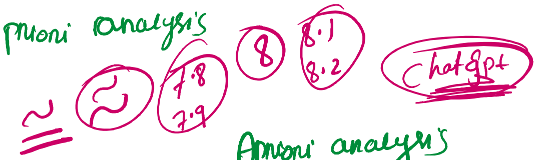


$$TC(P) = \overset{\text{Compile time}}{\underset{\text{Software}}{C(P)}} + \overset{\text{Run-Time}}{\underset{\text{H/w} \to \text{Processor}}{R(P)}}$$

Types of Analysis :

- Apostriory analysis
- Apriori analysis





Apostriory Analysis	Apriori analysis
- Dependent Hardware	- Approximately
- Exact	- Time Complexity = Constant
- Time Complexity changed	

Rs

- Time complexity

L changed

Ex: i_3 ✓ i_7 ✓ i_5 i_9

Apriori Algorithms:

It is a determinations of order of magnitude of a statement.

No. of time a statement will execute by processor.

y = 10
z = 5

y z

Ex: 1

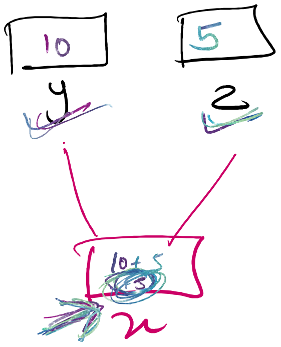
```
main()
{
    x = y+z;  -> 1 ✓
}
```

Time complexity = O(1)

i++ = 2

i = ~~1~~ 2 3 n n+1

n+1



Ex: 2

```
main()
{
    x = y+z;          1
    for( i=1; i<n; i++)
    {
        x = y+z;      -> n.times
    }
}
```

Time Complexity =

n+1 = O(n)

update

n+1

i = n
i++ = i+1
= n+1

2 3 n n+1

```
for ( i=1; i<n; i++)
{
    x = y+z;     n times
}
```

1 2 3 n

3

i

Team A

Team B

[illegible]

(n) '

[illegible]

min

Longest loop

[illegible]

Ex:

```
main()
{
    x=y+z;                 1
    for (i=1 to n)
    {
        x=y+z;             n
    }
    for (i=1 to n)
    {
        for (j=1 to n)
        {
            x=y+z;         n×n
        }
    }
}
```

n   6   4 - 2

Time complexity =

```
for (i=1; i≤n; i++)          1  2  3
{
    for (j=1; j≤n; j++)      3
    {
        x=y+z;  ←  2
    }
}
```

$$TC = O(1 + n + n^2)$$

$$= O(n^2)$$

Note

Time complexity means finding the Longest loop. if no loop in the program then Constant time Complexity (1)

Que

```
main()
{
    while (n≥1)              n=64
    {
        n = n/2;             log_2 n     n/2
    }                                    n/2^2
```

n≥1

32   64   2   16   32/2

64≥1 → n

32≥1 → n/2

16≥1 → n/

$L_3 \; L_3$

$\frac{n}{2^k} = 1$

$n = 2^k$

$k = \boxed{\log_2 n}$

$n/2^2$

$n/2^3$

$n/2^4$

$n/2^5$

$n/2^6$

16	$\cancel{32/2}$	$32 \geq 1 \rightarrow n/2$
8	$\cancel{16/2}$	$16 \geq 1 \rightarrow n/2^2$
	$\cancel{8/4}$	$8 \geq 1 \Rightarrow n/2^3$
2	$\cancel{4/2}$	$4 \geq 1 \Rightarrow n/2^4$
1	$\cancel{2/2}$	$2 \geq 1 = n/2^5$
		$1 \geq 1 = n/2^6$

Q what is the difference b/w ln & log ?

ln → e   log → 10

* If n = n/2 changed to n = n/3 then,

$\frac{n}{3^k} = 1$

$n = 3^k \Rightarrow \boxed{k = \log_3 n}$

* If while loop, $n = n/2;$ $n = n/3;$ → $n = n/6$

$n = n/6^k \Rightarrow \boxed{k = \log_6 n}$