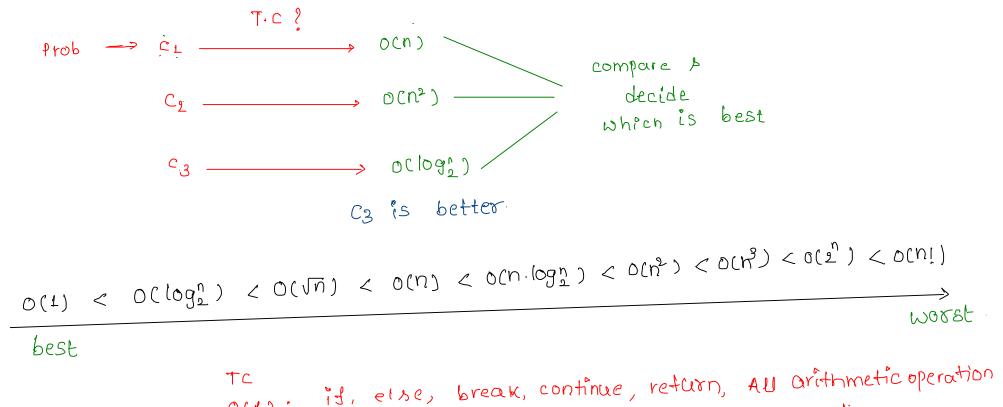
Time Complexity - 2



id, else, break, continue, return, All arithmetic operation 0(1): ets. point ~

Note:- when we are finding the T.C of any program we can take approximate values, exact values are not required

1 Template1 !-

for (i=1; i<=n; i=i+1)
$$\rightarrow$$
 n times \rightarrow 0(n)

{

print("*"); \rightarrow 0(1)
}

constant time

of times loop runs.

```
for(i=1;i<=n;i=i+a) \rightarrow \frac{h}{a} times { print("*"); }
```

for(i=1;i<=n;i=i+3)
$$\rightarrow \frac{n}{3}$$
 times o(n)
{

print("*");
}

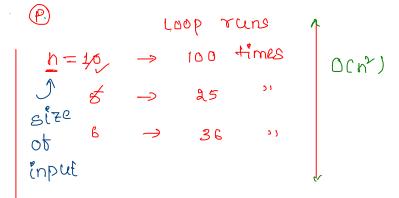
```
for(i=1;i<=n;i=i+n) \rightarrow \frac{\gamma}{\beta} = 1
                                                \Rightarrow 0(1)
          print("*");
 template.
\checkmark for(i=1;i<=n;i=i+a)
                                           times.
          print("*")
                     Q = L 
                                              a=n/
                                               0(1)
                        O(n)
```

whenever we find the T.C always see how many times loop runs

based on that decide O()

Noti: Remember

Template2



P
$$N=1$$
44 \rightarrow 12 times 1
49 \rightarrow 7 \rightarrow 0 (\sqrt{n})
625 \rightarrow 85 \rightarrow

P if
$$N = 2^{10} \rightarrow 10$$
 times

 $n = 3^{5} \rightarrow 5$
 $n = 2^{10} \rightarrow 100$
 $n = 2 \rightarrow 100$

in T.C, we focus on how many times loop runs

```
for(i=1;i\leq n;i=i*2)
            print("*")
* Remember:-
             O(\log_2^n)
 for(i=n;i>=1;i=i/2) \rightarrow o(\log_2^n)
      print("*")

h → n/2 → n/4 → . . . 1
                                                               O(\log_2)
```

$$\frac{\sqrt{k} \text{ times}}{\sqrt{k-1}} > n \quad (Stop)$$

Coop runs

1c-1

;i<=<u>n</u>

$$\frac{2^{K-1} > n}{2^{K-1}} = n$$

$$\frac{2^{K-1}}{2^{K-1}} = n$$

$$\frac{2^{K-1}}{2$$

```
for(i=1;i<n;i=i*3) → logn {
    print("*") }
```

```
for(i=1;i<=n;i=i+a) \Rightarrow \frac{\eta}{\alpha} time {

print("*")
}
```

```
for(i=1;i<=n;i=i*a) \rightarrow \log_a^n times

{

print("*")
}
```

(Un) o∟ for(i=1;i<=5;i++) →5 /

On)

$$\begin{array}{c}
| Un-roll(\log Process)| \\
| Un-roll(\log Process)| \\
| J=L I J ... J .$$

 C_2

```
for (i=1; i<=n;i++)

{

for (j=1; j<=n;j++) \rightarrow

{

c=c+1
```

```
dependency
```

c=c+1

```
or for(i=1; i<=n;i++)
{
    for(j=1;j<=i;j++)-
{
```

number of times inner loop runs depends on outer loop variable

```
Son) No-Dep

if for (i=1;i<=n;i++) \longrightarrow n times

{

if for (j=1;j<=n/4;j++) \longrightarrow n times

{

if for (k=1;k<=n;k++) \longrightarrow n times

{

print("*") \longrightarrow O(1)

}
}
```

```
for(i=1;i<=n;i++) \rightarrow n

{

for(j=1;j<=n/4;j++) \rightarrow \frac{n}{4}

{

for(k=1;k<=n;k++) {

print("*")

break; \leftarrow
}
}
```

$$n * \frac{n}{4} + 1 = \frac{n^2}{4}$$
$$= 0 c n^2$$

```
No-dep
an)
                                                                    n \cdot n \cdot 2 = 2 \cdot n^{\gamma} \Rightarrow o(n^{\gamma})
       •for(i=1; i<=n; i++) ✓ 🖔
               for(j=1; j<=n; j++) \sim n

{

for(k=n/2; k<=n; k=k+n/2)
{

    c=c+1 \leftarrow
}

2 fines ( cosns+)
```

```
i=1 \longrightarrow 2 \longrightarrow 4 \longrightarrow 8 \longrightarrow \frac{h}{2} \longrightarrow \frac{h}{2
```

```
@) No-dep
```

```
ifor(i=1; i<=n; i++) \rightarrow n times

{

ifor(i=1; i<=n; i++) \rightarrow n times

{

ifor(j=1; j<n; j=2*j) \rightarrow log<sub>2</sub> times

{

c=c+1

}
```

```
> Un-rollerg
for(i=1; i <= n; i++)
    for(j=1; j <= i; j++) \rightarrow i times
            c=c+1
            print(*)
```

```
of times
IL Runs ( i times)
                                     for(j=1; j<=<u>i</u>; j++) \rightarrow \frac{i}{1} = i times
                                              c=c+1
                                              print(*)
L+2+3+\cdots+n=\frac{n(n+1)}{9} \Rightarrow O(n^2)
                                    for(i=1;i<=n;i=i+a) \rightarrow \frac{\eta}{a} times
                                             print("*")
```

process

On) dep - un rolling process

```
for(i=1; i<=n; i++)
         for (j=1; j < =n; j=j+i) \rightarrow \frac{n}{i} times
                                                                                c=c+1
               c=c+1
                                  # of times IL Runs ( times)
                                                    = n + \frac{n}{2} + \frac{n}{3} + \frac{n}{4} + \cdots + 1
                                                     = n \left[ \frac{1 + 1/2 + 1/3 + 1/4 + \cdots + 1/n}{4} \right]
                                                     = 0(nlogn) /
```

```
B) dep. -> un volling.
for(i=1;i<=n;i=i+1)
        for (j=i;j<=n;j=j+1) \rightarrow n-i times

{

//O(1)

i=x \rightarrow (n-i)
}
                        \underline{\underline{n}}-\underline{L} + \underline{\underline{n}}-2 + \underline{\underline{n}}-3 + \underline{\underline{n}}-4+ · · · + \underline{\underline{L}} + \underline{\underline{0}}

\begin{array}{c}
n+n+n+\dots + n + (-1-2\dots -n) \\
- & \end{array}
```

```
for(j=\underline{i};j<=\underline{n};j=j+1) \frac{n}{1}
        //O(1)
  for(i=1;i<=n;i=i+a) \rightarrow \frac{n}{a} times -1
           o(1)
      for(i=10;i<=n;i=i+a) \rightarrow \frac{\eta}{\alpha} times -10
               o(1)
```

```
for(i=1;i<=n;i++) \sim n times
{

for(j=1;j<=sqrt(n);j++) \sim \sqrt{n} times
{
                                                               0(n. \n)
            // O(1)
      for(k=1;k<=n;k=k*2) ~> logg times
```

```
L<sub>1</sub> for(i=1;i<=n;i++) \sim \sim n times
{

L<sub>2</sub> for(j=1;j<=sqrt(n);j++) \sim \sim \sim n times

{
                    // O(1)
                                                                                        n[in + logn]
         _3 for(k=1; k<=n; k=k*2) ~>> logg +imel
                                                                                               mocx = ?
                                                                                                                      Ans
                   //O(1)
```

```
for(i=1;i<=n;i++)
{
    for(j=i;j<=n;j++)
    {
        O(1)
    }
}</pre>
```

```
function fun(n)
     for(i=1;i<=n;i++)
           p=0
           for(j=n; j>1; j=j/2)
                 ++p
           for(k=1; k<p; k=k*2)
                ++q
```