Time and Space Complexity

(1) Set of code ⇒ # iterations } Time & Space

(2) Minor rule

steps / Bytes

Examples of Time Complexity

fun(int, N)

(1)
```
for (int i=1; i≤N; i++)
{
    (sum = sum+i)
}
```
} negligible

change in i value

1, 2, 3, 4 ..... N

= [1, N]

= N − ✓ + ✓ = N

TC = O(N)

fun(N)

(2)
```
sum=0;
for (i=1; i≤N; i=i+2)
{
    sum = sum+i;
}
```

# of times the line is processed

steps

iteration

change in i value

1, 3, 5, 7 .... N

[1, N] only odd numbers

N = 4

1, 2, 3, 4

if N is even : $\left(\frac{N}{2}\right)$

$\frac{4}{2} = \frac{N}{2}$

if N is odd : (N+1)

N = 5

$\left(\frac{N+1}{2}\right)$ 1, 2, 3, 4, 5

$\frac{N+1}{2} = \frac{5+1}{2} = \frac{6}{2} = 3$

$N/2 = N$ $\left(\frac{N+1}{2}\right) = \frac{N}{2} + \frac{1}{2} = N$

$TC = O(N)$

fun(N)

3

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

change in i value

i value	1,	2,	3,	4	.....	k
condition	1^2,	2^2,	3^2,	4^2	.....	k^2

$k^2 \le N$

$k^2 = N$

$\boxed{k = \sqrt{N}}$

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

fun(N)

4

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

$\log_2 N$

change in i value

$$N, \frac{N}{2}, \frac{N}{2\times 2}, \frac{N}{2^3}, \frac{N}{2^4} \cdots\cdots \frac{N}{2^k}$$

$$\left(\frac{N}{2^2}\right)$$

$$\frac{N}{2^k} \leq 1$$

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

$$N = 2^k$$

$$\log_2 N = \log_2 2^k$$

$$\boxed{\log_2 N = k}$$

$$\log_a a^p = p$$

$$\log_2 4 = \log_2 2^2 = 2$$

$$\left.\begin{array}{l} 4/2 = 2 \\ 2/2 = 1 \end{array}\right\} 2$$

$$N \cdots \frac{N}{2} \cdots\cdots \frac{N}{2^2} \cdots\cdots 1$$

$$\boxed{\log_2 N}$$

$$N \cdots \frac{N}{3} \cdots \frac{N}{3^2} \cdots$$

$$\log_3 N$$

$$\log_2 N = O(\log N)$$

$\text{fun}(N)$ $(0 \cdots\cdots 100)$

5 for $(i = 0;\ i \leq N;\ i = i * 2)$

{

}

}

change in i value } infinite loop

$0, \underset{(0)}{0*2}, \underset{(0)}{0*2} \ldots = 0$

$TC = O(1)$

---

⑥ for (int i=1; i ≤ N; i=i*2)

{

}

change in i value							
i	1	1*2(2)	2*2(4)	(4*2)=8	....2^4	2^5	....2^k
condi	1	2	4	8	2^4	2^5	..2^k

2^3

$$2^k > N$$

$$2^k = N$$

$$\log_2 2^k = \log_2 N$$

$$\boxed{k = \log_2 N}$$

$N \quad \frac{N}{2} \quad \frac{N}{4} \quad \ldots \quad 1$

$1 \quad 1*2 \quad 1*2^2 \quad 1*2^4 \quad \ldots \quad N$ } $\log_2 N$

$TC = O(\log N)$

(6)

```
for(i=1; i≤10; i++)
{
    for(j=1; j≤N; j++)
    {
        ...
    }
}
```

$[1, N]$
$= N - 1 + 1 = N$

i	j
1	N
2	N
3	N
⋮	
10	N

$(i) * (j)$

$= 10 * N$

$= \underline{\underline{10N}}$

$\cancel{10}N = N$

$TC = O(N)$

$[0, N) = N - 0 = N$

(8)

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

$[0, N)$
$= N - 0 = \underline{\underline{N}}$

i	j
0	N
1	N

$(i) * (j)$

$N * N$

$= N^2$

N

⋮	
N-1	N

$[0, N-1] = (N-1) - 0 + \cancel{1} = N$

$N^2 = TC = O(N^2)$

---

(9)

```
for( i=1; i≤N; i++)
{
    for( j=1; j≤i; j++)
    {
        print()
    }
}
```

i	j
1	1
2	2
3	3
⋮	
N	N

iteration =

$1 + 2 + 3 + \dots N$

$2-1=1 \quad 3-2=1$

$S = \frac{K}{2}(2a + (K-1)d)$

$= K = [1, N] = N-1+1 = N$

$a = 1$

$d = 1$

$= \frac{N}{2}[2(1) + (N-1)1]$

$= \frac{N}{2}[2 + N - 1]$

$$= \frac{N}{2}[N+1]$$

$$\# \text{ iteration} = \frac{N^2}{2} + \frac{N}{2}$$

$$\frac{N^2}{2} + \frac{N}{2} = N^2$$

$$TC = O(N^2)$$

AP

Gauss

$a+0d$ $\quad a+1d \quad a+(2d)$ $\quad N \rightarrow$

$S = a \quad a+d \quad a+2d \quad \cdots \quad a+(n-1)d$

$S = a+(n-1)d \quad a+(n-2)d \quad \cdots \quad a$

$2S = 2a+(n-1)d + 2a+(n-1)d \quad \cdots \quad 2a+(n-1)d$

AP

$2S = (2a+(n-1)d)\,n$

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

(10)

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

1	$\log_2 N$
2	$\log_2 N$
⋮	
N	$\log_2 N$

$N \log_2 N$

$TC = O(N \log N)$

(11)

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

i	j
1	$[1, 2^1] = 2^1$
2	$[1, 2^2] = 2^2$
3	$[1, 2^3] = 2^3$
⋮	
N	$[1, 2^N] = 2^N$

$\underbrace{2^1 + 2^2}_{\times 2} \underbrace{+ 2^3}_{\times 2} + \cdots 2^N$

$S_K = \frac{a(r^K - 1)}{r - 1}$

$a = 2$, $r = 2$

$K = N$

$S_K = \frac{2(2^N - 1)}{2 - 1} = 2(2^N - 1)$

iteration $= 2^{N+1} - 2$

$2^{N+1} - 2$ $\quad O(2^{N+1})$

$= O(2^N)$

---

(3) for(i=1; i<100000 ; i++)

{

}

iteration = 1K

TC = $O(1)$

Time Complexity

a) No. of iteration

b) lower order terms, neglect

c) constants, neglect

} Most high order term

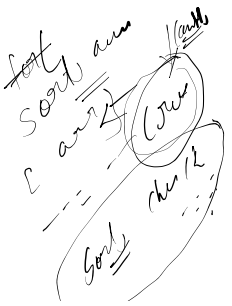
i) Maximum time a code can take iteration

O (big-O notation)

ii) Ω (omega)

best time can (minimum can)

iii) Θ (Avg time)



TC cases

$O(1)$

$O(\log N)$

$O(\sqrt{N})$

$O(N)$

$O(N \log N)$

$O(N\sqrt{N})$

$O(N^2)$

$O(2^N)$

lower to higher order TC

Space Complexity

```
fun ( int arr[ ])
{
  temp arr[ ]
  ans arr[ ]
  int sum
  int prev
```

$= N*4$

$\leftarrow N*4$

$\leftarrow N*4$

$\leftarrow 4$

$\leftarrow 4$

$4N + 4 + 4$

$= 4N$

$= N$

Space com = $O(N)$