Examples

Find out time complexity for the following pseudo code using O-notation.

```
for (i = 0 ; i < n ; i++)
{
        for (j = n ; j > 0 ; j--)
        {
                if (i < j)
                        c = c + 1;
        }
}
```

Sol The time complexity is computed step by step by considering the no. of execution of each statement.

code	Time complexity
`for(i=0; i<n ; i++)`	$\longrightarrow O(n)$
`{`	
`for(j= n ; j>0 ; j--)`	$\longrightarrow O(n*n)$
`{`	
`if(i<j)`	$\longrightarrow O(n*n)$
`c=c+1;`	$\longrightarrow O(n*n)$
`}`	
`}`	
Total	$\longrightarrow 3O(n^2)+O($

Thus considering the order of magnitude and neglecting the constants, we get the time complexity as $O(n^2)$.

Find big oh (O) notation for the following:

1. f(n) = 6993,
2. $f(n) = 6n^v + 135$.

Sol

① $f(n) = O(g(n))$ where

$$f(n) \leq c * g(n)$$

If $c > 1$ and $g(n) = 6993$.

Then $f(n) = O(n)$.

② As $f(n) = 6n^v + 135$

$$f(n) = O(n^v)$$

where $c < 6$ and $n <= 2$

This is because with these values.

$$f(n) <= c * g(n)$$

Is $3\log n + \log\log n$ is $O(\log_{10} n)$?

Is $3\log n + \log\log n$ is $\Omega(\log_{10} n)$?

Sol: We assume $n = 1000 = 2^{10}$ (approximately)

Consider RHS

$$3\log_2 n = 3\log_2 1000$$

$$= 3\log_2 2^{10}$$

$$= 3 \times 10(\log_2 2)$$

$$\approx 30 \cdot \log_2 2 \quad (\because \log_2 2 = 1$$

$$\approx 30$$

$$\log\log n = \log_2(\log_2 1000) = \log_2(10) = 3$$

$$\therefore 3\log n + \log\log n = 30 + 3 = 33$$

Consider LHS

$$\log_{10} n = \log_{10} 1000 = \log_{10} 10^3$$

$$= 3$$

$\therefore$ $3\log n + \log\log n > \log_{10} n$

$\therefore$ $3\log n + \log\log n \in \Omega(\log_{10} n)$

Check the correctness for the following equality

$5n^3 + 2n = O(n^3)$

Sol For big oh notation $f(n) \leq c * g(n)$ is true where $f(n)$ and $g(n)$ are functions and c is constant.

Here $f(n) = 5n^3 + 2n$

and $g(n) = n^3$

Also assume $n=1$, $c=1$, then,

$L.H.S = f(n)$

$= 5n^3 + 2n$

$= 5(1)^3 + 2(1) = \underline{7}$

$R.H.S = c * g(n)$

$= 1 * (1)^3$

$= \underline{1}$

Here $f(n) \leq c * g(n)$ is not true with $c=1$ and $n=1$.

Now if we assume $c=7$ and $n=1$ then,

$L.H.S = 5n^3 + 2n$

$= 5(1)^3 + 2(1) = 7$

$R.H.S = c * n^3$

$= 7 * 1 = \underline{7}$

for $c = 7$ and $n = 2$

$$L.H.S = 5n^3 + 2n$$

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

$$= 5 \times 8 + 4$$

$$= 40 + 4$$

$$= 44.$$

$$\therefore R.H.S = c * g(n)$$

$$= 7 * (2)^3$$

$$= 7 * 8$$

$$= 56$$

Here $f(n) < c * g(n)$ is true.

Thus $5n^3 + 2n = O(n^3)$ for $n \geq 0$ and $c =$

Find upper bound of function $f(n) = \log(n^v) + n^v$

Sol In general the highest bound is the upper bound in case of addition.

$$f(n) = \log(n^v) + n^v \log n.$$

$$= 2\log(n) + n^v \log n.$$

$$\therefore f(n) = O(n^v \log n)$$

Find Omega (Ω) notation of function

$f(n) = 2n^2 + 6n + \log n + 6n.$

Sol

Let

$f(n) = 2n^2 + 6n + \log n + 6n.$

Assume $g(n) = 14n$

If $n = 1$

$f(n) = 2 + 6 + 0 + 6$

$g(n) = 14 \times 1 = 14$

If $n = 2$

$f(2) = 2(2)^2 + 6(2) + \log 2 + 6(2)$

$= 8 + 12 + 1 + 12$

$= 33$

$g(2) = 14 \times 2$

$= 28$ if $f(n) > g(n)$

If $n = 3$

$f(3) = 2(3)^2 + 6(3) + \log 3 + 6(3)$

$= 18 + 18 + 1.58 + 18$

$= 55.58$

$g(n) = 14 \times 3$

$= 42$ Again $f(n) > g(n)$

Thus for $n \geq 1$

$f(n) = 2n^2 + 6n + \log n + 6n \in \Omega(n)$