

→ Factors and Divisors

- Brute Force Approach to find Divisors, Divisors count and Divisors Sum.
- Time Complexity $O(n)$

```

VOID BRUTE DIVISOR (INT n)
{
    INT CNT = 0, SUM = 0;
    FOR (INT i = 1; i <= n; i++)
    {
        IF (n % i == 0)
        {
            CNT << i << " | n";
            CNT++;
            SUM += i;
        }
    }
    CNT << CNT << " " << SUM << " | n";
}

```

→ But we can observe a pattern in divisors:

For $n = 24$,

$$1 \longrightarrow 24/1 = 24$$

$$2 \longrightarrow 24/2 = 12$$

$$3 \longrightarrow 24/3 = 8$$

$$4 \longrightarrow 24/4 = 6$$

→ We can see we can get those lower part of divisors from the upper one as well.

→ So, we just need to iterate our loop till $\text{sqrt}(n)$ (\sqrt{n}).

6

8

12

24

→ OPTIMIZED APPROACH

→ Square Root Method to find divisors, divisors count and divisors sum.

→ It is faster than Brute Force as we have to iterate our loop till $\text{sqrt}(n)$ only.

→ Time Complexity : $O(\sqrt{n})$

```
VOID sqrtDIVISOR (INT n)
{
```

```
    INT CN = 0, SUM = 0;
```

```
    FOR (INT i = 1; i * i ≤ n; i++)
```

→ $(i * i \leq n)$ is same as $(i \leq \text{sqrt}(n))$

```
{
```

```
    IF (n % i == 0)
```



```
{
    cout << i << " " << n/i << "\n";
    cnt++;
    sum += i;
}
```

→ Check in case if we don't consider a number twice.

→ eg: In case of $n = 36$

1	36
2	18
3	12
4	9
6	6

→ We don't want to consider this 6 twice.

```
if (n/i != i)
```

```
{
```

```
    cnt++;
```

```
    sum += n/i;
```

```
}
```

```
}
```

```
}
```

```
cout << cnt << " " << sum << "\n";
```

```
}
```

→ There is even more faster ~~way~~ approach to find divisor count and divisor sum using Prime factorization.

Let x be a no., representing its Prime Factors like this:

$$x = p_1^{m_1} \times p_2^{m_2} \times p_3^{m_3}$$

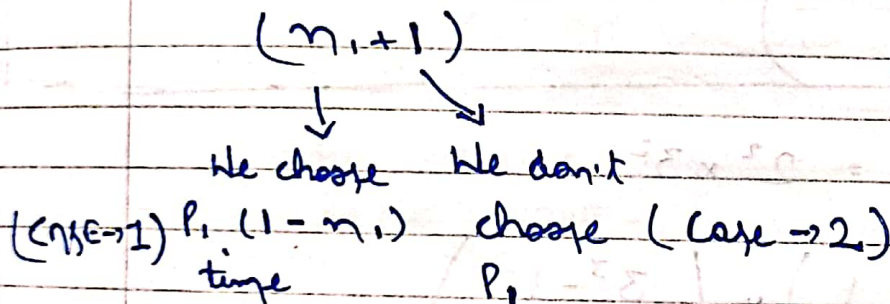
Suppose $x = 36$

$$\therefore 36 = 2^2 \times 3^2$$
$$= \underbrace{2 \times 2 \times 3 \times 3}$$

The subsets of these will give divisors

$$\therefore \text{For } x = p_1^{m_1} \times p_2^{m_2} \times p_3^{m_3}$$

In how many way we can select P_1 , P_2 and P_3



Similarly :

$$(n_1+1)(n_2+1)(n_3+1) = \text{no. of divisors}$$

Proof: if $x = 36 = 2^2 \times 3^2$

$$n_1 = 2 \quad n_2 = 2$$

$$(2+1)(2+1) \Rightarrow \underline{9 \text{ divisors}}$$

→ Now, For sum of divisors:

$$\times \frac{(1 + p_1 + p_1^2 + p_1^3 + \dots + p_1^{n_1})}{(1 + p_2 + p_2^2 + p_2^3 + \dots + p_2^{n_2})}$$

Proof : $36 = 2^2 \times 3^2$
 $\Rightarrow (1 + 2 + 4) \times (1 + 3 + 9) \leftarrow \text{For } 2^2$
 $\Rightarrow 7 \times 13 \Rightarrow \textcircled{91}$

\rightarrow Now general formula for sum:

$$(1 + p_1 + p_1^2 + p_1^3 + \dots + p_1^{n_1})$$

\hookrightarrow We can see this is G.P.

$$\therefore \text{Sum of divisors} = \left(\frac{p_1^{n_1+1} - 1}{p_1 - 1} \right) \times \left(\frac{p_2^{n_2+1} - 1}{p_2 - 1} \right) \times \left(\frac{p_3^{n_3+1} - 1}{p_3 - 1} \right)$$

Proof : $36 = 2^2 \times 3^2$

$$\Rightarrow \left(\frac{2^3 - 1}{2 - 1} \right) \times \left(\frac{3^3 - 1}{3 - 1} \right)$$

$$\Rightarrow 7 \times \frac{26}{2} = 13$$

$$\Rightarrow \textcircled{91}$$