- **Time Complexity (T.C)** :- It refers to the amount of time an algorithm takes to run as a function of input size (n).

1. Best Case
2. Average Case
3. **Worst Case** ⇒ It is represented using Θ notation.

eg:- { 20, 50, 30, 6, 4 } search = 100

$O(n)$

① In worst case scenario, it had to explore all the elements and still the element was not found.

So, let's say that if the size of the set is n, then all n elements had to be checked whether it's equal to 100 or not.

- Time Taken :-

Best case < Avg case < **Worst case**

- $\theta\left(\frac{n+1}{2}\right) = \theta\left(\frac{n}{2} + \frac{1}{2}\right)$

$= \theta\left(\frac{n}{2} + c\right)$ where $c = \frac{1}{2}$

$\frac{n}{2} = n \cdot \frac{1}{2}$

$= c \cdot \theta\left(\frac{n}{2}\right) = \frac{c}{2} \cdot \theta(n)$

①

0	1	2	3	4	5
10	72	89	76	90	5

arr =

n = 6

target = 160

→ for (int i = 0; i < n; i++)

{ if (arr(i) == target)

return i;

O(1) = constant

// for loop

T.C = O(n)

0

10	20	30	40

Res = 10+20+30
+40 = 100

```
int sum=0;
for (int i=0; i< n; i++)
{→sum= sum+ arr[i];   (1)
}
return sum;
```

O(1) = Constant

① int c = 0;

c++;

int sum = sum + 10;

return sum;

T.C = $\Theta(1)$

① Nested Loop :-

```
for (int i = 0; i < m; i++)
{
    for (int j = 0; j < n; j++)
    {
        // Some statements  → O(k)
    }
}
```

T.C = $\Theta(m * n)$

i = 0, j = n times
i = 1, j = n times
i = m, j = n times

① $\theta(m * n) = \theta(500 \times n)$

10^{10} 10^{10}

$= 500\ \theta(n)$

$\theta(10^{20})$

① $\theta(c \cdot n) = c \cdot \theta(n)$

constant

① $\theta(n+n) = \theta(2 \cdot n)$

$= 2 \cdot \theta(n) = \theta(n)$

② $\boxed{\theta(m+n) = \theta(\max(m,n))}$

V. IMP

① `for (int i = 0; i < m; i++)`

```
{
    // some statement
}
```

```
for (int i = 0; i < n; i++)
{
    // some statement
}
```

```
if (m > n)
⇒ O(m)
if (n > m) ⇒ O(n)
```

int count = 0;

① for (int i = 0; i < m; i++)

{

for (int j = 0; j < n; j++)

{ for (int k = 0; k < q; k++)

{

count = count + 10 ;

}

}

T.C = $\theta(m * n * q)$

}

- **Space Complexity** :- It refers to the amount of extra memory or storage space required by an algorithm to complete it's execution.

① eg:- void func(int arr1(m), int arr2(n))

aux. array

int arr(k), int arr2(n)

Input array

aux. array

{

for (int i = 0 ; i<k ; i++)

{ count = count + 10v;

}

} S.C = $\Theta(m+n)$

④ void func2 (int arr1[n] , int arr2[m])

{

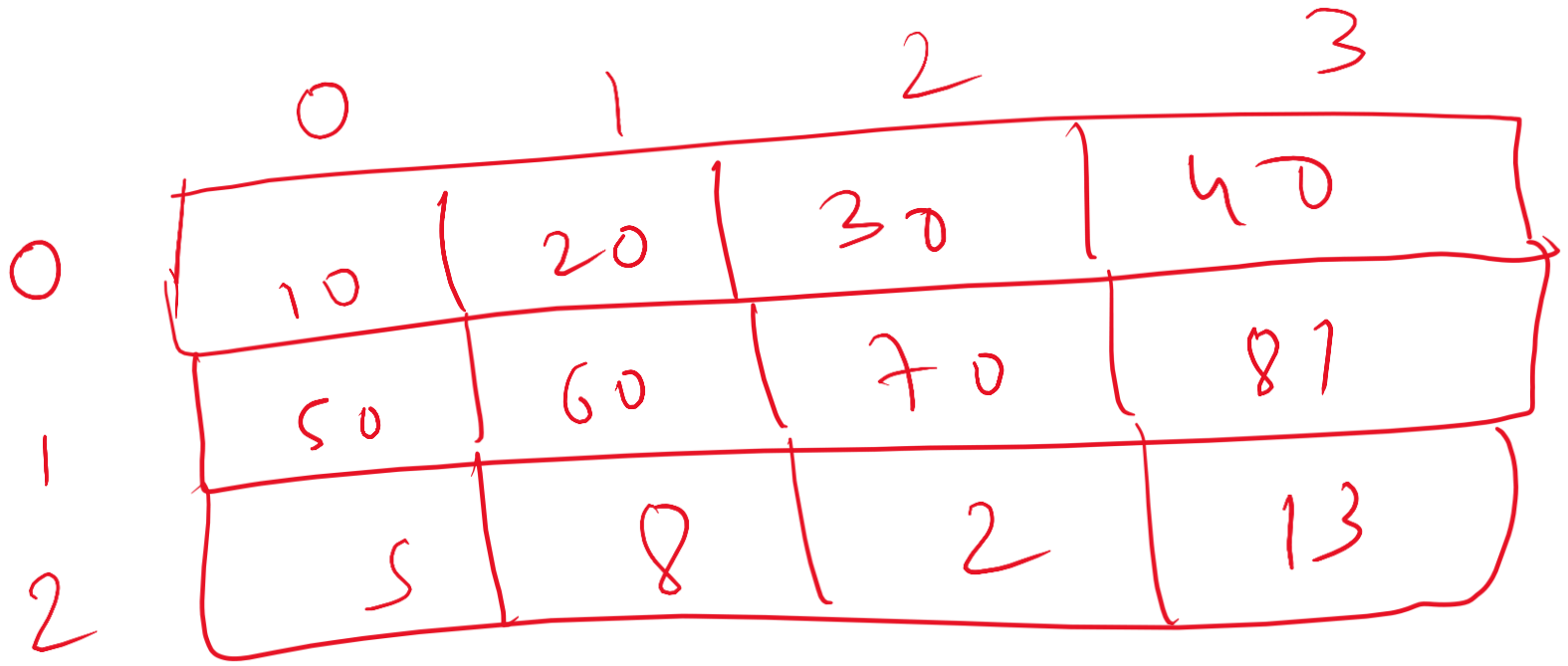
int vis1[n][m];

int vis2[q][q];

}

⑤ S.C = $\Theta(nm + qq)$

$= \Theta(\max(nm, qq))$



3 x 4

$\theta(m * n)$

m

n