# Analysis of Algorithm

## Algorithm

- Design
- Domain knowledge
- Language
- Hardware, OS
- Analysis

## Program

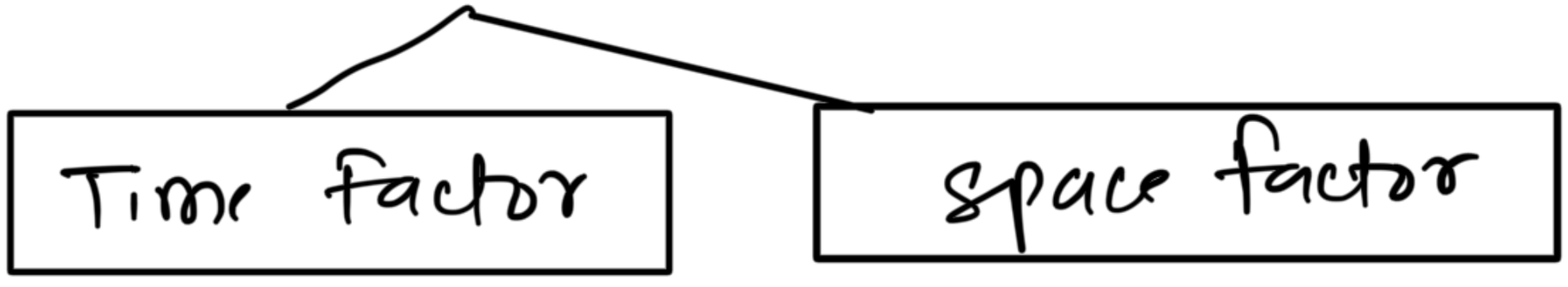
- Implementation
- Programmer
- Programming language
- H/W & OS
- Testing

## Priori Analysis

## Posterious Analysis

| | |
|---|---|
| - Algorithm | - Program |
| - Independent P/f | - Dependent P/f |
| - Independent H/w | - Dependent H/w |
| - Time & Space | - Time |

## Algorithm Complexity




measured by counting the number of key operations & ignoring other factors

measured by counting the maximum memory needed by the algorithm

| | |
|---|---|
| Comparison in the sorting algorithms | memory space required by the algorithm |

## Asymptotic Notations (Symbols)

Asymptotic analysis of an algorithm refers to defining the mathematical boundations of its run-time performance.

Notations

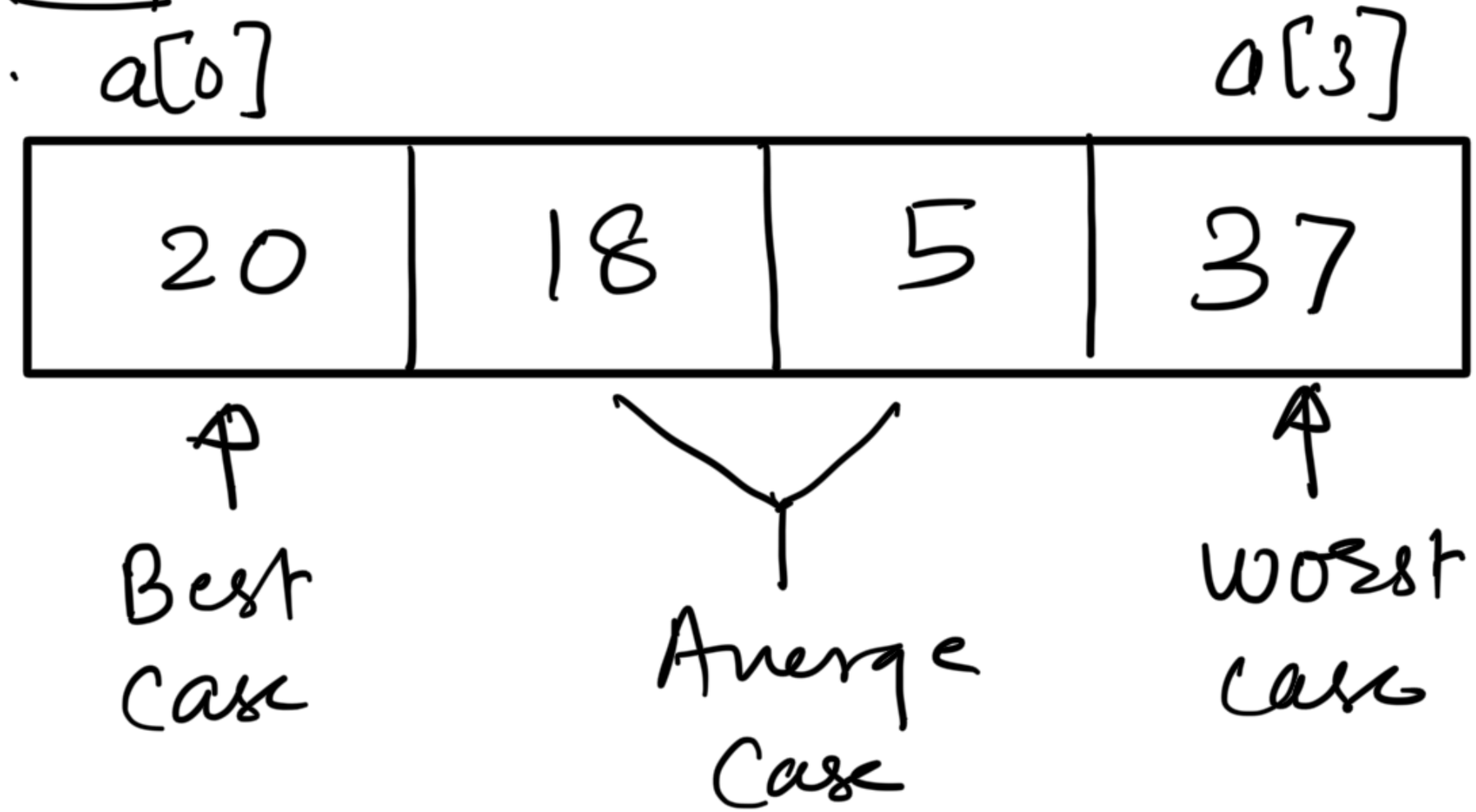
1. Best case – Minimum time required for program execution
   Ω (ohm)

2. Average Case – Average time required for program execution
   Θ (theta)

3. Worst case – Maximum time required for program execution
   O (Big O notation)

(Big-O-Notation)

Eg.




---

Ex: Swapping of 2 number

Swap (a, b)

Time

Space

{

temp = a; ⟶ 1s  a → 1

a = b; ⟶ 1  b → 1

b = temp; ⟶ 1  temp → 1

}

f(n) = 3 → O(1)

s(n) = 3 (min) → O(1)

x = 5*a + 6*b ⟶ 1

x = 5*a + 6*b

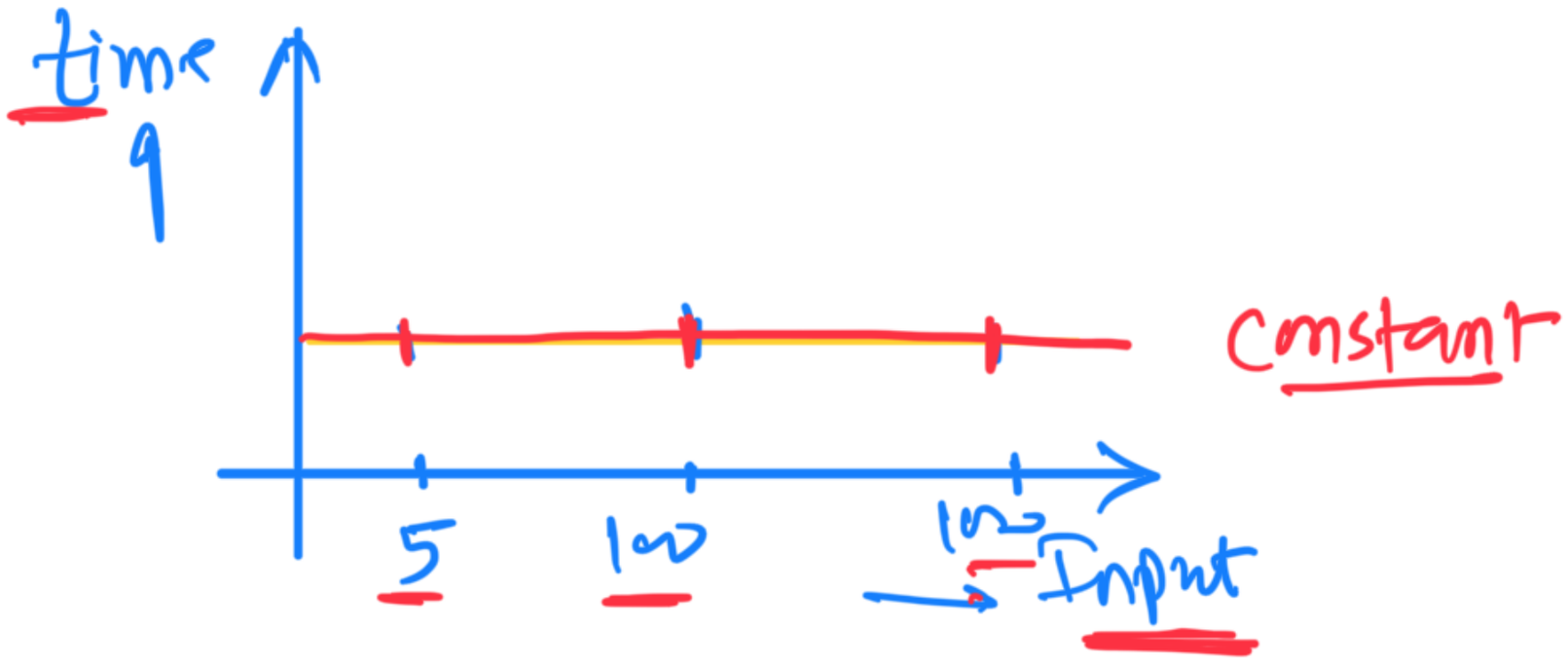
x = 5*a + 6*b

x = 5*a + 6*b

1 = Constant

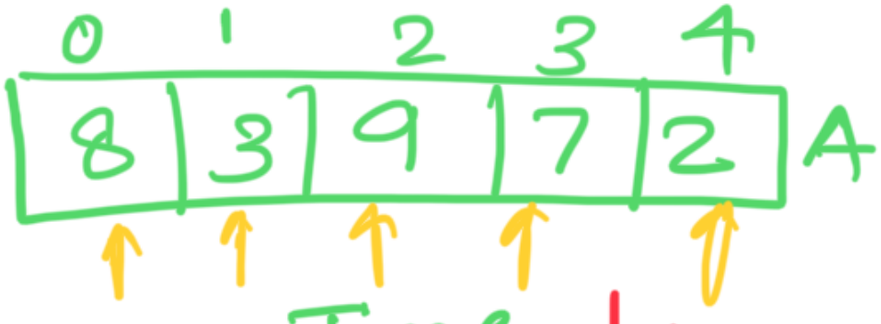
$f(n) = 4 \longrightarrow \boxed{O(1)} \rightarrow$ Notation

**Constant Time Complexity**




---

Ex: $n = 5$, $i = 0, 1, 2, 3, 4, 5$




```
sum(A, n)
{
    for(i=0;
```

Time | Space

```
2  for(i=0; i<n; i++)
   {
3      s = s+A[i];
   }
4  return s;
}
```

| | |
|---|---|
| 2 | $s+1$ $(n+1)$ |
| 3 | $5$ $n$ |
| 4 | $1$ |

$A \to n$
$n \to 1$
$S \to 1$
$i \to 1$

$f(n) = 2n+3$

$s(n) = n+3$

$O(n)$

Linear
Time Complexity

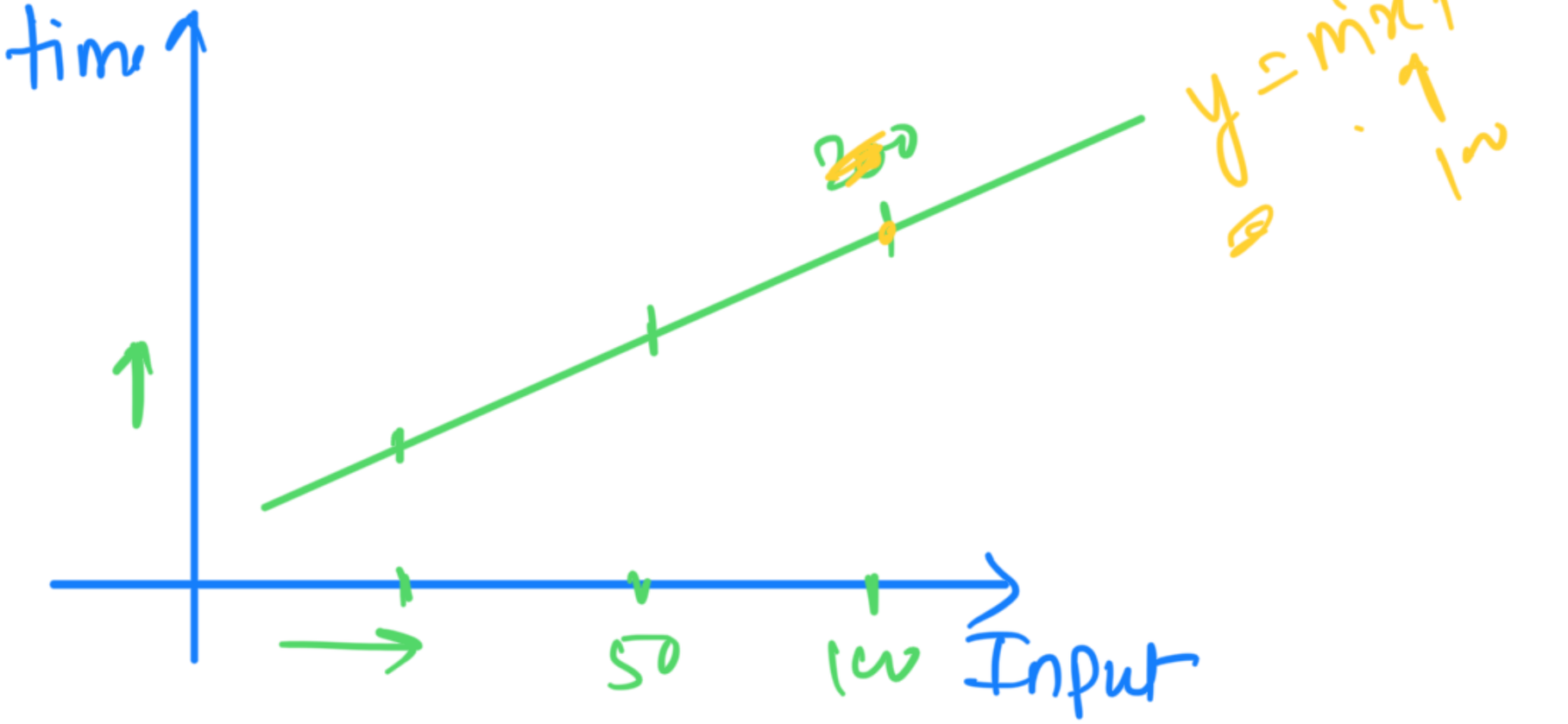
$O(n)$

Sum $\Rightarrow$ $1 + (n+1) + n + 1$

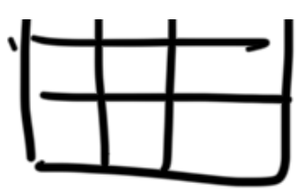
$= 2n + 3$

Coeff. Const

$= O(n) \longrightarrow$ Linear




Ex:



n×n
3×3

```
Add(A, B, n)
{
  for(i=0; i<n; i++)
  {
    for(j=0; j<n; j++)
    {
      C[i,j] = A[i,j] + B[i,j]
    }
  }
}
```

n+1

n

n+1

n

$$f(n) = (n+1) + n[(n+1)+n]$$

$= 2n^2 + 2n + 1$

$\frac{n+1}{n}$

$n+1$

$n \times (n+1)$

$n \times n$

$f(n) = 2n^2 + 2n + 1$

$O(n^2)$ $O(n)$ $O(1)$

quadratic Time Complexity $\leftarrow O(n^2)$

Spark

$A \rightarrow n^2$

$B \rightarrow n^2$

$C \rightarrow n^2$

$n \rightarrow 1$

$m \rightarrow 1$

$k \rightarrow 1$

$S(n) = 3n^2 + 3$

$O(n^2)$ $O(1)$

$O(n^2)$

for ( ) $\rightarrow (n+1)$

for