[Video - 11]

Time Complexity

↳ It is the amount of time taken by an algorithm to run, based on the length of the input.

↳ as a function of length of the input

- Why
  - ↳ for making better Program.
  - ↳ Comparision of Algorithm.

- Big O notation
  - ↳ Upper Bound
  - ↳ Jada se Jada itna time lagayega.
- Theta Θ
  - ↳ for Avg-Case Complexity
  - ↳ Avg Time
- Omega Ω
  - ↳ Lower Bound
  - ↳ Kam se kam itna Time.

↳ Worst Case

- Constant time: → O(1) for(i=0; i<10)
  ↳ Same Time

- Linear time → O(n) for(i=0; i<n)
  ↳ Depend on input [n].
  ↳ ↓n ↓time
  ↳ ↑n ↑time

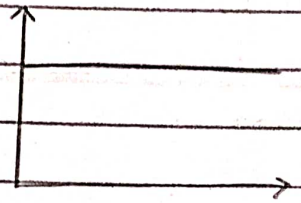
- Logarithmic time → O(log n) → Binary Search

- Quadratic time → $O(n^2)$
  for(1 → n)
  ↳ for(1 → n)

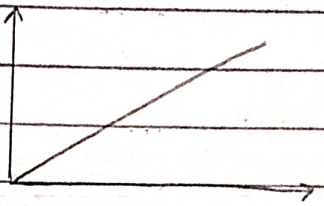
- Cubic time → $O(n^3)$ for(1 → n)
  ↳ for(1 → n)
  ↳ for(1 → n)

Graph

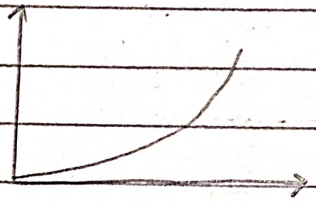
- $O(1)$



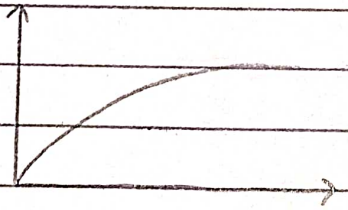
- $O(n)$



- $O(n^2)$



- $O(\log n)$



Complexity

Highest ↑

$O(N!)$
$O(2^n)$
$O(N^3)$
$O(N^2)$
$O(N \log N)$
$O(N)$
$O(\log N)$
$O(1)$

↓ Least

Big O → Upper Bond

[Ignore numeric constant and low power Degree

$f(n) \rightarrow 2n^2 + 3n \rightarrow O(n^2)$

$f(n) \rightarrow 4n^4 + 3n^3 \rightarrow O(n^4)$

$f(n) \rightarrow N^2 + \log N \rightarrow O(N^2)$

$f(n) \rightarrow 12001 \rightarrow O(1)$

$f(n) \rightarrow 3n^3 + 2n^2 + 5 \rightarrow O(n^3)$

$f(n) \rightarrow \frac{n^3}{300} \rightarrow O(n^3)$

$\rightarrow 5n^2 + \log n \rightarrow O(n^2)$

$\rightarrow \frac{n}{4} \rightarrow O(n)$

$\rightarrow \frac{n+4}{4} \rightarrow O(n)$

Time Complexity

- for ( ) → O(n)
  {
  }
  for ( ) → O(m)
  {
  }

  → O(n + m)

- for ( ) → O(n)
  {
  for ( ) → O(m)
  {
  }
  }

  → O(n x m)

  Nested = multiply.

- for (0 — N)
  {
  for (0 — N)
  {
  }
  }

  → O(N x N)
  → O(N²)

  for (0 — N) → O(N)
  {
  }

  O(N²) + O(N)

10^8 Operation Rule → Most of the modern machine can perform 10^8 Operation / Second.

< [10 .... 11]	$O(N!)$, $O(N^6)$
[15 .... 18]	$O(2^N * N^2)$
100	$O(n^4)$
400	$O(n^3)$
2000	$O(n^2 * \log n)$
10^4	$O(n^2)$
10^6	$O(n \log n)$
10^8	$O(n)$, $O(\log N)$

Space Complexity

↳ It is the amount of space taken by an algorithm to run.

- int arr[5] = {1, 2, 3, 4, 5}
  ↳ It is constant / fixed.

  SC → O(1)

- int n;
  cin >> n;
  vector <int> v(n);
  ↳ length → N

  SC → O(N)