What is DSA?

DSA stands for **Data Structures and Algorithms**.
It is the foundation of computer science and programming that helps us solve problems efficiently.

- **Data Structures** → Ways of organizing and storing data (like arrays, linked lists, stacks, queues, trees, graphs).

- **Algorithms** → Step-by-step instructions or procedures to solve a problem (like searching, sorting, traversing).

- Together, DSA helps programmers write code that is not only correct but also **fast** and **memory-efficient**.

Definition of DSA

DSA is the study of how data can be stored, arranged, and processed using efficient methods (data structures), and how problems can be solved using logical steps (algorithms).

What is Time Complexity?

Time Complexity measures **how much time** an algorithm takes to run depending on the size of the input.

- It shows how the execution time grows as the input grows.

- Expressed using **Big O notation** (e.g., O(1), O(n), O(log n), O(n^2)).

Example:

- Searching in a sorted list with Binary Search → **O(log n)** (fast).

- Searching in an unsorted list with Linear Search → **O(n)** (slower).

What is Space Complexity?

Space Complexity measures **how much memory** an algorithm uses while running.

- Includes memory for variables, data structures, and recursion stack.
- Also expressed in Big O notation.

Example:

- Binary Search uses **O(1)** extra space (very little memory).
- Recursive algorithms may use more memory because of the **function call stack**.