Understanding Arrays

An **array** is like a collection of variables stored together under one name. Instead of creating separate variables for each piece of data, arrays let us store multiple values in a single structure, making programs more organized and efficient.

■ Why Do We Need Arrays?

Imagine you want to store the marks of 50 students. Without arrays, you would need 50 separate variables! Arrays allow us to group similar data under one name and access them easily using an index (a position number).

■ Example

Storing 5 numbers without and with arrays:

Without Arrays	With Arrays
mark1 = 80 mark2 = 75 mark3 = 90 mark4 = 85 mark5 = 70	marks = [80, 75, 90, 85, 70]

■ Accessing Elements

We use **indexes** (position numbers) to access array elements. Indexes usually start from **0**. For example: - marks[0] → 80 - marks[2] → 90 - marks[4] → 70

■ Types of Arrays

1. **One-Dimensional Array** → A simple list of elements. Example: [10, 20, 30] 2. **Two-Dimensional Array** → An array of arrays, like a table (rows & columns). Example: [[1,2,3], [4,5,6], [7,8,9]] 3. **Multi-Dimensional Array** → Arrays with more than two dimensions (used in advanced cases).

■ Real-Life Analogy

Think of an **array** as a **row of mailboxes**: - The mailbox row is the array. - Each mailbox is a position (index). - The letters inside are the values stored. You can quickly find a letter by knowing the mailbox number (index).