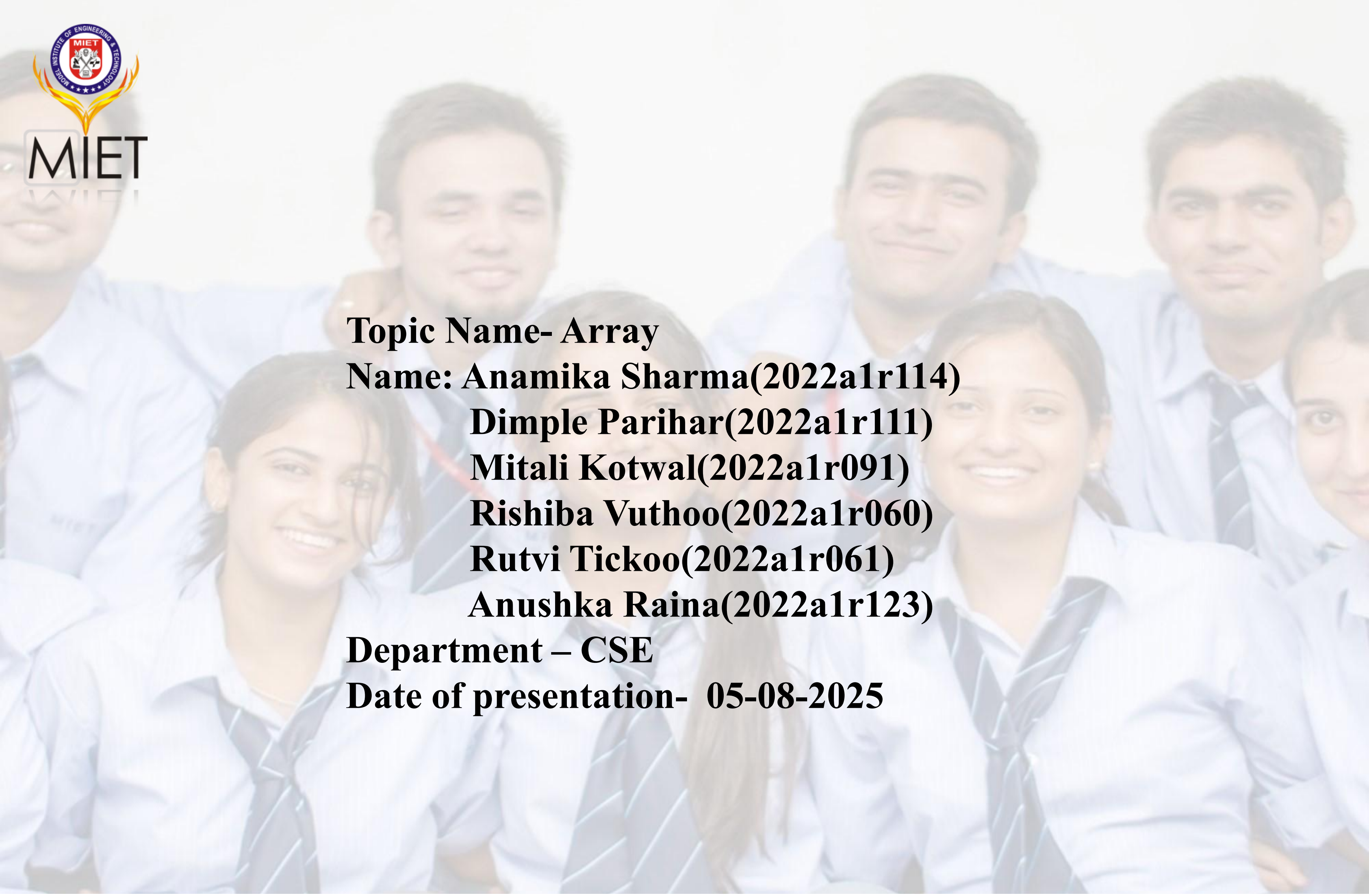


Topic Name- Array

Name: Anamika Sharma(2022a1r114)
Dimple Parihar(2022a1r111)
Mitali Kotwal(2022a1r091)
Rishiba Vuthoo(2022a1r060)
Rutvi Tickoo(2022a1r061)
Anushka Raina(2022a1r123)

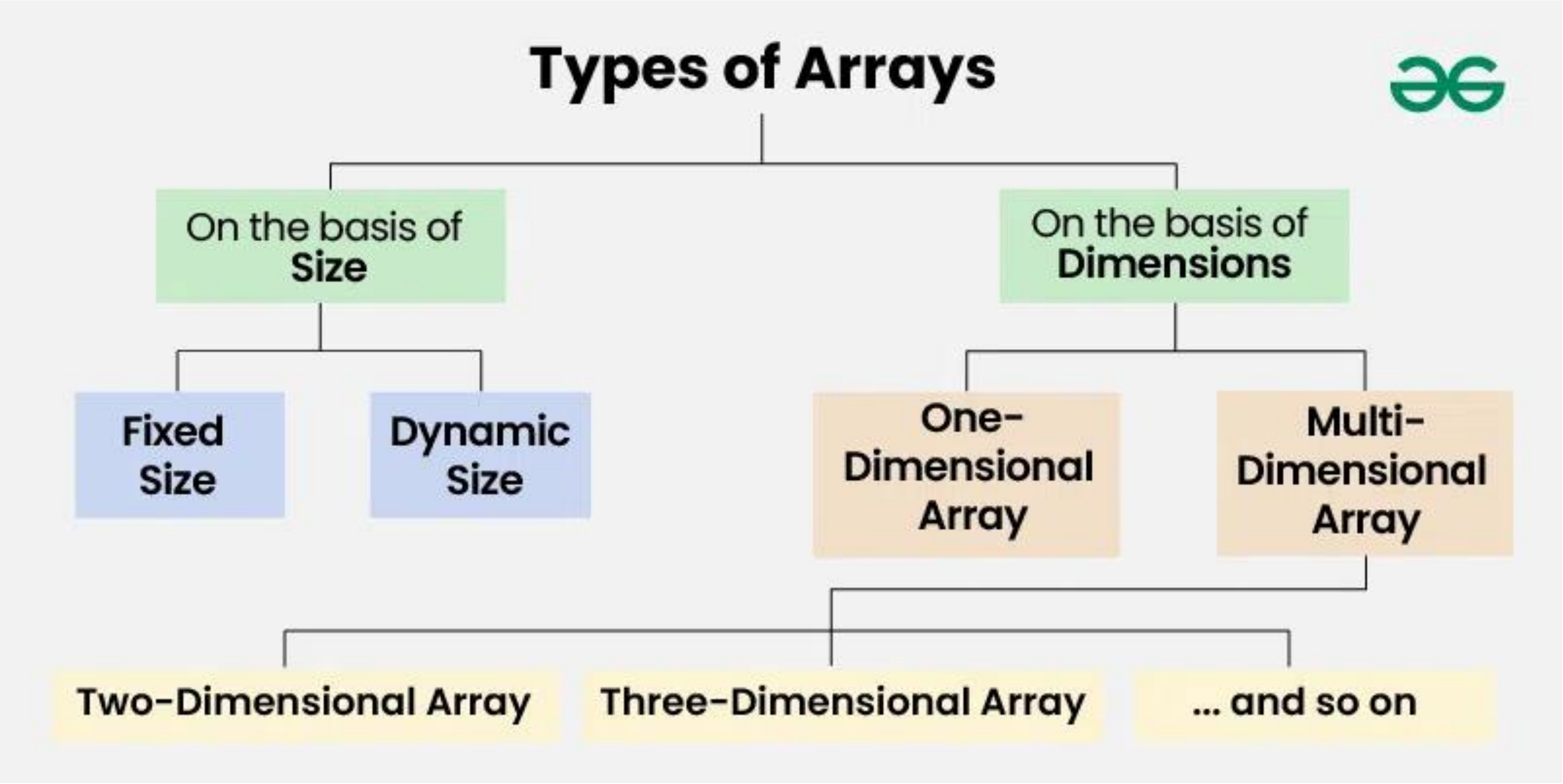
Department – CSE

Date of presentation- 05-08-2025

Array

- Arrays are fundamental data structures in programming.
- It helps to store collections of elements. They enable efficient data manipulation and retrieval, playing a crucial role in algorithms and software development.
- An array is a collection of items stored at contiguous memory locations. It allows for efficient access and modification of elements through indexing.

Types of Array



Types

- □ **One-Dimensional Array (1D):** A linear array that stores elements in a single row using a single index.

  Example: int arr[5];

- □ **Two-Dimensional Array (2D):** An array of arrays that stores data in a matrix form (rows and columns).

  Example: int matrix[3][3];

- □ **Multi-Dimensional Array:** Arrays with more than two dimensions, used for complex data representation.

  Example: int tensor[2][3][4];

Characteristics of Arrays

- **Fixed Size:** Arrays have a fixed size upon initialization, meaning the number of elements cannot change dynamically.
- **Homogeneous Elements:** All elements in an array are of the same data type, which ensures consistency and enables easier manipulation of the data.
- **Random Access:** Arrays allow random access to elements, meaning any element can be accessed

Operations on Arrays

- **Traversal:** Traversing an array involves accessing each element, typically using loops, allowing for processing or displaying array contents.
- **Insertion:** Inserting elements can be complex due to the fixed size, often requiring the creation of a new array if the current one is full.
- **Deletion:** Similar to insertion, deleting elements involves shifting elements to fill gaps, which can be inefficient in terms of time complexity.
- **Searching:** Searching for an element can be done using linear or binary search algorithms, depending on whether the array is sorted, affecting performance.

Advantages and Disadvantages

Advantages

- Simple and Easy to Understand: The code uses a basic for loop and swap() function — beginner-friendly and readable.
- In-Place Reversal (No Extra Space): It reverses the array **in-place**, meaning it does not use extra memory (space complexity = O(1)).

Disadvantages

- Fixed size: This works only for arrays declared with a fixed size at compile time.
- Time complexity: In large array or real time systems , frequent shifting becomes slow and inefficient.

Thank You