Arrays:

1. Basics of Arrays

Definition:

An array is a linear data structure in computer science that stores a collection of elements in contiguous memory locations. Each element in an array is of the same data type and can be accessed using an index. Arrays are widely used for organizing and manipulating data due to their simplicity and efficiency.

Linear Data Structures are a type of data structure in computer science where data elements are arranged sequentially or linearly. Each element has a previous and next adjacent, except for the first and last elements.

Key Characteristics:

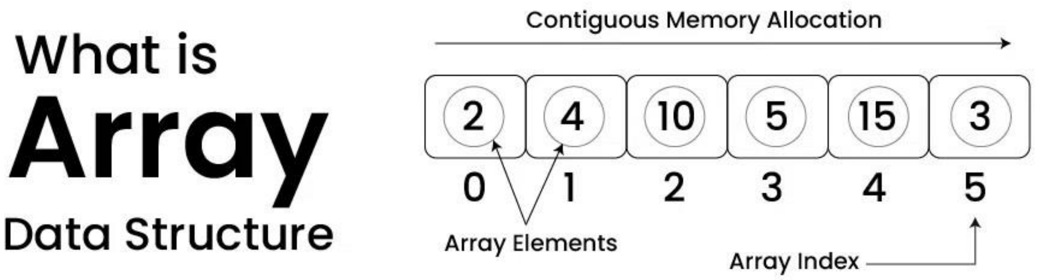
- **Fixed Size**: The size of an array is defined at creation and cannot be changed.

- **Homogeneous**: Elements of an array are of the same type (e.g., all integers or all strings).

- **Indexed Access**: Elements can be accessed using their index, starting from 0.

Syntax (Python):

```
arr = [1, 2, 3, 4, 5]
```



Types of Arrays

There are two main types of arrays:

- **One-dimensional arrays**: These store a single row of elements.
- **Multidimensional arrays**: These store multiple rows of elements, such as matrices.

Advantages:

- Fast access to elements using indices (O(1) time complexity).

- Simple and easy to implement.

- Efficient for storing and processing multiple values of the same type.

Limitations:

- Fixed size (in static arrays), leading to potential memory wastage.

- Insertion and deletion are expensive (O(n) complexity for worst cases)

2. Operations on Arrays

1. Traversal

Accessing and processing each element in the array.

2. Insertion

Adding a new element into the array at a specific position.

Time Complexity:

- Worst Case: O(n) (when inserting at the beginning or middle).
- Best Case: O(1) (when inserting at the end).

3. Deletion

Removing an element from a specific position.

- **Time Complexity**: O(n) (due to shifting elements).

4. Searching

Finding the position of a given element.

Applications of Arrays

Arrays are used in various applications, including:

- Storing data for processing
- Implementing data structures such as stacks and queues
- Representing data in tables and matrices
- Creating dynamic data structures such as linked lists and trees

3. Practice Problems

1. Find the Largest and Smallest Element in an Array
2. Reverse an Array
3. Find Missing Number in an Array
4. Move Zeros to End
5. Rotate Array by K Steps