Answer the following Question

1. Define array. List all the Important properties of array.

An **array** is a data structure in programming that stores a fixed-size sequence of elements of the same type. It is a collection of items stored at contiguous memory locations, and the elements in an array are indexed, meaning they can be accessed using their index number.

Important Properties of Arrays:

1. **Fixed Size**:
   - Arrays have a fixed size, meaning the number of elements an array can hold is determined when the array is created. This size cannot be changed dynamically.
2. **Homogeneous Elements**:
   - All the elements stored in an array must be of the same data type (e.g., all integers, all strings).
3. **Indexing**:
   - Arrays use zero-based indexing, which means the first element is at index 0, the second element is at index 1, and so on. The last element is at n-1 where n is the size of the array.
4. **Contiguous Memory Allocation**:
   - Array elements are stored in contiguous memory locations, which allows for efficient access using their index.
5. **Random Access**:
   - Arrays allow direct access to elements using their index. This means you can retrieve or modify an element in constant time O(1) by using the index.
6. **Static or Dynamic Nature**:
   - In some languages, arrays are static, meaning their size is fixed at compile time (e.g., C, Java). In other languages, arrays can be dynamic (e.g., JavaScript, Python) and grow or shrink at runtime.
7. **Multidimensional Arrays**:

   - Arrays can be single-dimensional (like a list) or multi-dimensional (like a matrix). For example, a 2D array can be thought of as an array of arrays.

8. **Default Values**:
   - In many programming languages, arrays are initialized with default values. For example, in Java, an array of integers is initialized to 0, and an array of objects is initialized to null.
9. **Low-level Representation**:
   - Arrays are often implemented at a low level, making them a more efficient data structure in terms of memory and speed when compared to other structures like linked lists.

2. List two different ways to define the array.

1. Using Declaration with Size

You can define an array by specifying its type and size. The size of the array determines how many elements it can hold.

```
// Syntax

dataType[] arrayName = new dataType[arraySize];



// Example

int[] numbers = new int[5];  // Array of integers with size 5
```

In this case, the array numbers can hold 5 integer elements, and the values will be initialized to 0 by default.

2. Using Declaration with Initialization

You can define and initialize the array at the same time by directly specifying the elements.

```
// Syntax
dataType[] arrayName = {element1, element2, element3, ...};


// Example
int[] numbers = {10, 20, 30, 40, 50};  // Array with 5 integers
```

3. Define 2D array. Write the declaration of a 2D array.

A 2D array (two-dimensional array) in Java is an array of arrays. It can be thought of as a matrix with rows and columns. Each element in a 2D array is accessed using two indices: one for the row and one for the column.

Declaration of a 2D Array in Java

1. Declaration with Size

You can declare a 2D array by specifying its type and size for rows and columns.

```
// Syntax
dataType[][] arrayName = new dataType[rows][columns];


// Example
int[][] matrix = new int[3][4];  // 2D array with 3 rows and 4 columns
```

In this example, matrix is a 2D array of integers with 3 rows and 4 columns. All elements are initialized to 0 by default.

2. Declaration with Initialization

You can also declare and initialize a 2D array at the same time.

```
// Syntax

dataType[][] arrayName = {

    {element1, element2, element3, ...},

    {element1, element2, element3, ...},

    ...

};

// Example

int[][] matrix = {

    {1, 2, 3, 4},

    {5, 6, 7, 8},

    {9, 10, 11, 12}

};  // 2D array initialized with 3 rows and 4 columns
```