Arrays & Some Important Question of Array

✅ What is an Array in Java?

An **array** in Java is a **container object** that holds a **fixed number of values** of a **single type**. Each item in an array is called an **element**, and each element is accessed by its **index**.

🔢 Features of Arrays

-> **Fixed size** (declared at creation).

-> **Indexed** (starts from 0).

-> **Homogeneous** (all elements must be of the same type).

-> **Stored in contiguous memory**.

-> **Supports both primitive and object types**.

✅ Why Do We Need Arrays?

1. **To Store Multiple Values**
   -> Arrays allow us to **store multiple values in one variable** instead of creating separate variables for each value.

2. **Organized Data Handling**

   -> Arrays give you an easy way to **group and organize related data** (e.g., scores, marks, names).

3. **Indexed Access**

   -> Makes data retrieval and processing much easier and faster.**852**

4. **Efficient Memory Usage**

   -> Arrays are **contiguous in memory**, making them memory-efficient and improving performance during iteration.

5. **Loop-Friendly**

   You can use loops to:

-> Traverse

-> Search

-> Sort

-> Modify elements

🛠️ **Array Declaration ,Initialization & Accessing**

1. **Array Declaration :**

   -> In Java, **declaring an array** means **telling the compiler** that you're creating a variable that will hold **a collection of values** of a specific data type (like `int`, `String`, etc.).

   -> Declaration **does not allocate memory** — it only defines the **type and structure** of the array variable.

   Syntax of Array Declaration: datatype[] arrayName;

Declaration Type	Example	Description
Integer Array	int[] numbers;	Declares an array to hold integers
String Array	String[] names;	Declares an array to hold strings
Double Array	double[] prices;	Declares an array to hold decimal numbers
Char Array	char[] vowels;	Declares an array to hold characters

2. **Creating an Array :**

-> Creating an array means **allocating memory** for it so you can store values.

-> Syntax: dataType[] arrayName = new dataType[size];

3. **Array Initialization :**

1. Static Initialization (with values):

->You provide values directly at the time of creation.

-> Syntax: dataType[] arrayName = {value1, value2, value3, ...};

2.Dynamic Initialization (with new keyword):

-> You first create the array with a specific size, and then assign values later.

-> Syntax: dataType[] arrayName = new dataType[size];

Note: the **new** keyword is used to create objects in memory**.**

4. Accessing Elements of an Array

1.Using Indexing:

->You access elements in an array using indexes, which start from 0.

-> Syntax: arrayName[index]

2. Access Using Loop:

-> Can use loops to access all elements efficiently.

📌 **Array Properties:**

-> length: returns number of elements.

-> Example: int size = arr.length;

🔢 **Array Index:**

-> An **array index** is the **position number** used to access individual elements in an array.

🔍 **Why Does Array Index Start at 0**

1. Memory Address Calculation :

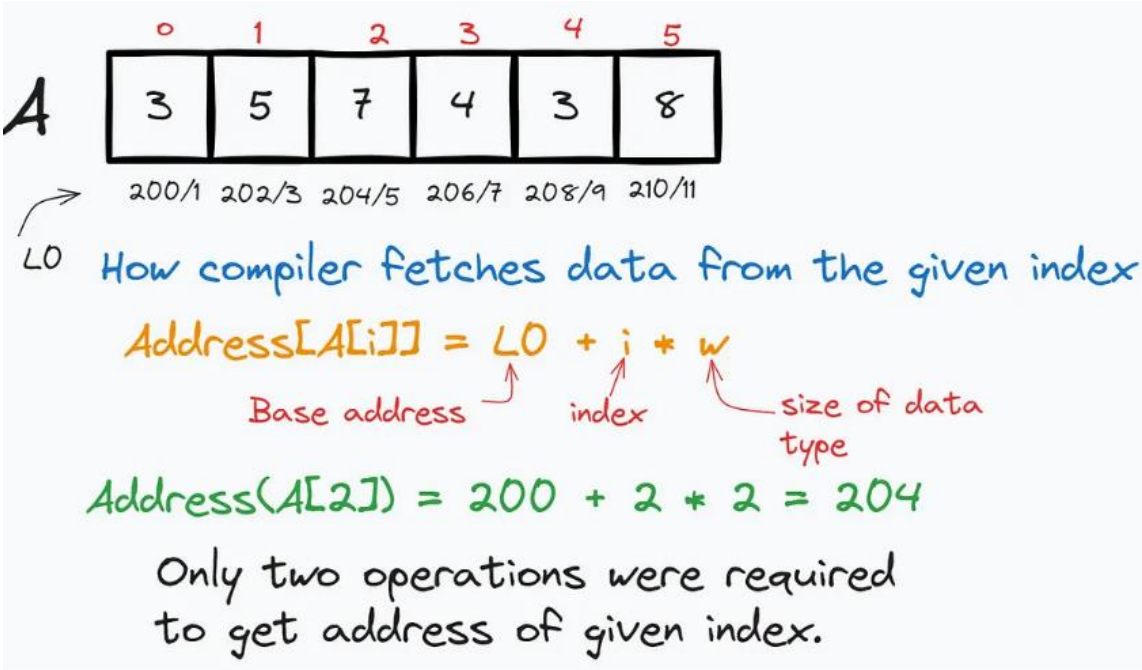
If:

-> base_address is the memory address of the first element

-> i is the index

-> size_of_element is the size of one array element

-> address_of_element[i] = base_address + (i × size_of_element)



✅ **Types of Arrays in Java**

Java supports two main types of arrays:

1. Single-Dimensional Array (1D Array)

-> A **linear collection** of elements (like a list).

-> All elements are of the **same data type**.

->Example:

Day5OneDimensionalArray.java > Day5OneDimensionalArray > main(String[])

```java
public class Day5OneDimensionalArray
Run | Debug
{       public static void main(String[] args) {
            // Declare and initialize a 1D array
            int[] numbers = {10, 20, 30, 40, 50};

            // Print all elements using a loop
            System.out.println(x:"Elements of the array:");
            for (int i = 0; i < numbers.length; i++) {
                System.out.println("Element at index " + i + ": " + numbers[i]);
            }
            /*
            Elements of the array:
            Element at index 0: 10
            Element at index 1: 20
            Element at index 2: 30
            Element at index 3: 40
            Element at index 4: 50
            */
        }
}
```

2. Multi-Dimensional Array

->Arrays that contain **other arrays as elements**.

->The most common is the **2D array**, like a **matrix**.

a) Two-Dimensional Array (2D Array)

->An array of arrays (like rows and columns).

->Example:

Day5TwoDimensionalArray.java > Day5TwoDimensionalArray > main(String[])

```java
public class Day5TwoDimensionalArray
 {
    Run | Debug
    public static void main(String[] args) {
        // Declare and initialize a 2D array
        int[][] arr = {
            {1, 2},
            {3, 4}
        };

        // Print specific elements of the 2D array
        System.out.println(arr[0][0]); // Output: 1
        System.out.println(arr[1][1]); // Output: 4
    }
}
```

b) Three-Dimensional Array (3D Array)

-> An array of 2D arrays.

3. Jagged Array (Array of Arrays with Different Lengths)

- A **2D array** where each row can have a **different number of columns**.

✅ How to Take Array Input from User in Java

We use the **Scanner class** to read input from the keyboard.

🔷 1. Import Scanner Class

```
import java.util.Scanner;
```

-> This allows you to use the Scanner class.

🔷 2. Create Scanner Object

```
Scanner sc = new Scanner(System.in);
```

->This object is used to read input from the user.

🔷 3. Ask for Array Size

```
System.out.print("Enter size of array: ");
int size = sc.nextInt();
```

-> This reads how many elements the array will have.

🔷 4. Declare and Create Array

```
int[] arr = new int[size];
```

-> Creates an integer array of user-defined size.

🔷 5. Take Input in Array Using Loop

```
System.out.println("Enter " + size + " elements:");
        for (int i = 0; i < size; i++) {
                arr[i] = sc.nextInt();
                }
```

-> Fills the array with values from the user.

🔷 6. Display the Array

```
System.out.println("You entered:");
        for (int i = 0; i < size; i++) {
          System.out.println(arr[i]);
        }
```

-> Example:

Day5UserArray.java

Day5UserArray.java > Day5UserArray > main(String[])

```java
import java.util.Scanner;
public class Day5UserArray
{
    Run | Debug
    public static void main(String[] args) {
        // This program reads the size and elements of an array from user input,
        // then prints the elements of the array.
        Scanner sc = new Scanner(System.in);                // Step 1

        System.out.print(s:"Enter size of array: ");          // Step 2
        int size = sc.nextInt();

        int[] arr = new int[size];                          // Step 3

        System.out.println("Enter " + size + " elements:");  // Step 4
        for (int i = 0; i < size; i++) {
            arr[i] = sc.nextInt();                          // Step 5
        }

        System.out.println(x:"You entered:");                 // Step 6
        for (int i = 0; i < size; i++) {
            System.out.println(arr[i]);
        }

        sc.close();
    }
}
```

✅ Common Built-in Functions for Arrays (via Arrays class):

-> Java provides **utility methods** for arrays in the **java.util.Arrays** class.

-> You need to import: import java.util.Arrays;

1. Arrays.sort(array)

-> Sorts the array in ascending order.

2. Arrays.toString(array)

-> Returns a string representation of the array.

3. Arrays.copyOf(array, newLength)

-> Copies original array to a **new array** with specified length.

4. Arrays.equals(arr1, arr2)

-> Compares two arrays. Returns true if all elements are the same.

5. Arrays.fill(array, value)

-> Fills the entire array with a specific value.

6. Arrays.binarySearch(array, value)

-> Performs **binary search** (on a **sorted** array). Returns the **index** or negative if not found.

Important Coding Question On Array

1. Write a Java program to sort an array of integers in descending order without using built-in sorting functions.

Day5Q1Array.java

Day5Q1Array.java > Day5Q1Array > main(String[])

```java
public class Day5Q1Array
{
    Run | Debug
    public static void main(String[] args) {
        int[] arr = {5, 2, 9, 1, 7};

        for (int i = 0; i < arr.length - 1; i++) {
            for (int j = i + 1; j < arr.length; j++) {
                // Swap if next element is greater
                if (arr[j] > arr[i]) {
                    int temp = arr[i];
                    arr[i] = arr[j];
                    arr[j] = temp;
                }
            }
        }
        // Print sorted array
        System.out.println(x:"Array in Descending Order:");
        for (int num : arr) {
            System.out.print(num + " ");
        }
    }
}
```

Output:

Array in Descending Order:

9 7 5 2 1

2. Write a Java program to find the sum of all even and odd numbers separately in an integer array.

Day5Q1Array.java Day5Q2Array.java ×

Day5Q2Array.java > ...

```java
public class Day5Q2Array {

    Run | Debug
    public static void main(String[] args) {
        int[] numbers = {5, 2, 7, 8, 4, 3, 9};

        int evenSum = 0;
        int oddSum = 0;

        // Loop through the array
        for (int i = 0; i < numbers.length; i++) {
            if (numbers[i] % 2 == 0) {
                evenSum += numbers[i];  // Add to even sum
            } else {
                oddSum += numbers[i];   // Add to odd sum
            }
        }

        // Print results
        System.out.println("Sum of Even Numbers: " + evenSum);
        System.out.println("Sum of Odd Numbers: " + oddSum);
    }
}
```

Output:

Sum of Even Numbers: 14

Sum of Odd Numbers: 24

3. Write a Java program to find the sum of digits of each element in an array.

Day5Q3Array.java > ...

```java
public class Day5Q3Array
{
    Run | Debug
    public static void main(String[] args) {
        int[] numbers = {123, 45, 9, 100, 78};

        System.out.println(x:"Sum of digits for each array element:");

        for (int i = 0; i < numbers.length; i++) {
            int num = numbers[i];
            int sum = 0;

            // Calculate sum of digits
            while (num > 0) {
                sum += num % 10; // get last digit
                num = num / 10;  // remove last digit
            }

            System.out.println("Number: " + numbers[i] + " → Sum of digits: " + sum);
        }
    }
}
```

Output:

Sum of digits for each array element:
Number: 123 → Sum of digits: 6
Number: 45 → Sum of digits: 9
Number: 9 → Sum of digits: 9
Number: 100 → Sum of digits: 1
Number: 78 → Sum of digits: 15

4. Write a Java program to check if an array contains only unique values.

Day5Q4Array.java > Day5Q4Array > main(String[])

```java
public class Day5Q4Array
{

    Run | Debug
    public static void main(String[] args) {
        int[] arr = {10, 20, 30, 40, 50}; // Try changing values to include duplicates
        boolean isUnique = true;
        // Compare each element with every other element
        for (int i = 0; i < arr.length; i++) {
            for (int j = i + 1; j < arr.length; j++) {
                if (arr[i] == arr[j]) {
                    isUnique = false;
                    break;
                }
            }
            if (!isUnique) break;
        }
        // Print result
        if (isUnique) {
            System.out.println(x:"Array contains only unique values.");
        } else {
            System.out.println(x:"Array contains duplicate values.");
        }
    }
}
```

Output:

Array contains only unique values.

5. Write a Java program to copy elements from one array to another manually.

C:\Users\ANKITA\OneDrive\Desktop\javalinkedin\Day5Q4Array.java

```java
public class Day5Q5Array
{
    Run | Debug
    public static void main(String[] args) {
        int[] original = {10, 20, 30, 40, 50};  // Original array

        // Create another array with same length
        int[] copy = new int[original.length];

        // Copy elements manually
        for (int i = 0; i < original.length; i++) {
            copy[i] = original[i];
        }

        // Display copied array
        System.out.println(x:"Copied Array:");
        for (int i = 0; i < copy.length; i++) {
            System.out.print(copy[i] + " ");
        }
    }
}
```

Output:

Copied Array:

10 20 30 40 50

6. Write a Java program to find the index of an array element.

Day5Q6Array.java > ...

```java
public class Day5Q6Array

    Run | Debug
    public static void main(String[] args) {
        int[] arr = {10, 20, 30, 40, 50};
        int target = 30;  // Element to find

        int index = -1;  // Default index if not found

        // Loop through the array to find the target element
        for (int i = 0; i < arr.length; i++) {
            if (arr[i] == target) {
                index = i;
                break;  // Stop searching once found
            }
        }

        // Display result
        if (index != -1) {
            System.out.println("Element " + target + " found at index: " + index);
        } else {
            System.out.println("Element " + target + " not found in the array.");
        }
    }
}
```

Output:

Element 30 found at index: 2

7. Write a Java program to remove every second element from an array.

Day5Q7Array.java > Day5Q7Array

```java
public class Day5Q7Array {

    Run | Debug
    public static void main(String[] args) {
        int[] original = {10, 20, 30, 40, 50, 60, 70};

        // Calculate size of new array (half of original, rounded up)
        int newSize = (original.length + 1) / 2;
        int[] result = new int[newSize];

        int index = 0;
        // Copy elements at even indices only
        for (int i = 0; i < original.length; i += 2) {
            result[index] = original[i];
            index++;
        }

        // Print result
        System.out.println(x:"Array after removing every second element:");
        for (int i = 0; i < result.length; i++) {
            System.out.print(result[i] + " ");
        }
    }
}
```

Output:

Array after removing every second element:

10 30 50 70

8. Write a Java program to insert an element (specific position) into an array.

Day5Q8Array.java

Day5Q8Array.java > ...

```java
public class Day5Q8Array {

    Run | Debug
    public static void main(String[] args) {
        int[] original = {10, 20, 30, 40, 50};
        int element = 25;   // Element to insert
        int position = 2;   // Position to insert (0-based index)
        // Check if position is valid
        if (position < 0 || position > original.length) {
            System.out.println(x:"Invalid position!");
            return;
        }
        // Create new array with one extra size
        int[] newArray = new int[original.length + 1];
        // Copy elements before the position
        for (int i = 0; i < position; i++) {
            newArray[i] = original[i];
        }
        // Insert new element at the position
        newArray[position] = element;
        // Copy the rest of the elements after position
        for (int i = position; i < original.length; i++) {
            newArray[i + 1] = original[i];
        }
        // Print new array
        System.out.println(x:"Array after insertion:");
        for (int i = 0; i < newArray.length; i++) {
            System.out.print(newArray[i] + " ");
        }
    }
}
```

Output:

Array after insertion:

10 20 25 30 40 50

9. Write a Java program to find the maximum and minimum value of an array.

Day5Q9Array.java > Day5Q9Array

```java
public class Day5Q9Array
{
    Run | Debug
    public static void main(String[] args) {
        int[] arr = {15, 22, 8, 19, 31, 4};

        // Initialize max and min with first element
        int max = arr[0];
        int min = arr[0];

        // Loop through the array
        for (int i = 1; i < arr.length; i++) {
            if (arr[i] > max) {
                max = arr[i];  // Update max
            }
            if (arr[i] < min) {
                min = arr[i];  // Update min
            }
        }

        // Print results
        System.out.println("Maximum value: " + max);
        System.out.println("Minimum value: " + min);
    }
}
```

Output:

Maximum value: 31

Minimum value: 4

10. Write a Java program to find the second highest in an array.

Day5Q9Array.java Day5Q10Array.java ×

Day5Q10Array.java > ...

```java
public class Day5Q10Array
{

    Run | Debug
    public static void main(String[] args) {
        int[] arr = {15, 22, 8, 19, 31, 4};

        int highest = Integer.MIN_VALUE;
        int secondHighest = Integer.MIN_VALUE;

        for (int num : arr) {
            if (num > highest) {
                secondHighest = highest; // update second highest
                highest = num;           // update highest
            } else if (num > secondHighest && num < highest) {
                secondHighest = num;     // update second highest if in between
            }
        }

        if (secondHighest == Integer.MIN_VALUE) {
            System.out.println(x:"No second highest value found (array may have all equal elements).");
        } else {
            System.out.println("Second highest value is: " + secondHighest);
        }
    }
}
```

Output:

Second highest value is: 22

11. Write a Java program to find the largest and smallest element in an array without using sorting.

Day5Q11Array.java > Day5Q11Array

```java
public class Day5Q11Array
{
    Run | Debug
    public static void main(String[] args) {
        int[] arr = {25, 11, 7, 75, 56, 22};

        // Initialize largest and smallest with first element
        int largest = arr[0];
        int smallest = arr[0];

        // Traverse the array
        for (int i = 1; i < arr.length; i++) {
            if (arr[i] > largest) {
                largest = arr[i];  // Update largest
            }
            if (arr[i] < smallest) {
                smallest = arr[i]; // Update smallest
            }
        }

        System.out.println("Largest element: " + largest);
        System.out.println("Smallest element: " + smallest);
    }
}
```

Output:

Largest element: 75

Smallest element: 7

12.Write a Java program to reverse an array of integer values.

Day5Q11Array.java Day5Q12Array.java

Day5Q12Array.java > Day5Q12Array

```java
public class Day5Q12Array
{
    Run | Debug
    public static void main(String[] args) {
        int[] arr = {10, 20, 30, 40, 50};

        int start = 0;
        int end = arr.length - 1;

        // Swap elements from start and end until they meet
        while (start < end) {
            int temp = arr[start];
            arr[start] = arr[end];
            arr[end] = temp;

            start++;
            end--;
        }

        // Print reversed array
        System.out.println(x:"Reversed array:");
        for (int i : arr) {
            System.out.print(i + " ");
        }
    }
}
```

Output:

Reversed array:

50 40 30 20 10

13. Write a Java program to find duplicate values in an array of integer values.

Day5Q13Array.java

Day5Q13Array.java > ...

```java
public class Day5Q13Array
    Run | Debug
    public static void main(String[] args) {
        int[] arr = {10, 20, 30, 20, 40, 10, 50, 30};
        System.out.println(x:"Duplicate values in the array:");
        // Boolean array to mark duplicates (optional)
        boolean foundDuplicate = false;
        for (int i = 0; i < arr.length; i++) {
            // Check if this element has appeared before
            boolean isDuplicate = false;
            for (int j = 0; j < i; j++) {
                if (arr[i] == arr[j]) {
                    isDuplicate = true;
                    break;
                }
            }
            // If not found before, check duplicates ahead
            if (!isDuplicate) {
                for (int k = i + 1; k < arr.length; k++) {
                    if (arr[i] == arr[k]) {
                        System.out.println(arr[i]);
                        foundDuplicate = true;
                        break;
                    }
                }
            }
        }
        if (!foundDuplicate) {
            System.out.println(x:"No duplicates found.");
        }
    }
}
```

Output:

Duplicate values in the array:

10

20

30

14. Write a Java program to remove all duplicate values from an integer array.

Day5Q14Array.java

Day5Q14Array.java > Day5Q14Array

```java
public class Day5Q14Array{
    Run | Debug
    public static void main(String[] args) {
        int[] arr = {10, 20, 30, 20, 10, 40, 50, 30};
        int n = arr.length;
        int[] temp = new int[n];
        int newIndex = 0;
        // Loop to find unique elements
        for (int i = 0; i < n; i++) {
            boolean isDuplicate = false;
            // Check if arr[i] is already in temp array
            for (int j = 0; j < newIndex; j++) {
                if (arr[i] == temp[j]) {
                    isDuplicate = true;
                    break;
                }
            }
            // If not duplicate, add to temp
            if (!isDuplicate) {
                temp[newIndex] = arr[i];
                newIndex++;
            }
        }   // Print result
        System.out.println(x:"Array after removing duplicates:");
        for (int i = 0; i < newIndex; i++) {
            System.out.print(temp[i] + " ");
        }

}
}
```

Output:

Array after removing duplicates:

10 20 30 40 50

15. Write a Java program to find the most frequently occurring number in an array.

Day5Q14Array.java Day5Q15Array.java ×

Day5Q15Array.java > ...

```java
public class Day5Q15Array
{
    Run | Debug
    public static void main(String[] args) {
        int[] arr = {1, 3, 2, 3, 4, 3, 5, 1, 2};

        int maxCount = 0;
        int mostFrequent = arr[0];

        for (int i = 0; i < arr.length; i++) {
            int count = 0;

            // Count occurrences of arr[i]
            for (int j = 0; j < arr.length; j++) {
                if (arr[i] == arr[j]) {
                    count++;
                }
            }

            // Update if current count is greater than maxCount
            if (count > maxCount) {
                maxCount = count;
                mostFrequent = arr[i];
            }
        }

        System.out.println("Most frequent number is: " + mostFrequent);
        System.out.println("It appears " + maxCount + " times.");
    }
}
```

Output:

Most frequent number is: 3

It appears 3 times.