1. **Find common elements between two arrays**

```java
import java.util.Arrays;
import java.util.HashSet;
import java.util.Set;

public class CommonElementsSet {
    public static void main(String[] args) {
        Integer[] array1 = {4, 2, 3, 1, 6};
        Integer[] array2 = {6, 7, 9, 8, 4};

        Set<Integer> set1 = new HashSet<>(Arrays.asList(array1));
        Set<Integer> set2 = new HashSet<>(Arrays.asList(array2));

        set1.retainAll(set2); // set1 now contains only common elements
        System.out.println("Common Elements: " + set1);
    }
}
```

```
Common Elements: [4, 6]
```

2. **Find first and last element of Arraylist**

```java
import java.util.ArrayList;

public class ArrayListElements {
    public static void main(String[] args) {
        ArrayList<String> fruits = new ArrayList<>();
        fruits.add("Apple");
        fruits.add("Banana");
        fruits.add("Cherry");

        if (!fruits.isEmpty()) {
            String firstFruit = fruits.get(0);
            String lastFruit = fruits.get(fruits.size() - 1);

            System.out.println("First fruit: " + firstFruit);
            System.out.println("Last fruit: " + lastFruit);
        } else {
            System.out.println("The ArrayList is empty.");
        }
    }
}
```

```
First element: Apple
Last element: Elderberry
```

3. Sort an array without using in-built method

```java
import java.util.Arrays; // Import the Arrays class

public class SortArrayBuiltIn {
    public static void main(String[] args) {
        int[] numbers = {5, 2, 8, 1, 9, 3}; // Declare and initialize an integer arr

        System.out.println("Original array: " + Arrays.toString(numbers));

        Arrays.sort(numbers); // Sort the array in ascending order

        System.out.println("Sorted array (ascending): " + Arrays.toString(numbers));
    }
}
```

```
Original array: [5, 2, 8, 1, 9, 3]
Sorted array (ascending): [1, 2, 3, 5, 8, 9]
```

4. Remove duplicates from an Array

```java
public class Test {
    public static void main(String[] args) {
        int[] array = { 5, 2, 9, 1, 6, 2, 5 };
        int[] uniqueArray = removeDuplicates(array);
        System.out.println("Array with duplicates removed:");
        for (int num : uniqueArray)
        {
            System.out.print(num + " ");
        }
    }

    public static int[] removeDuplicates(int[] array)
    {
        Set<Integer> set = new HashSet<>();
        for (int num : array) {
            set.add(num);
        }
        int[] result = new int[set.size()];
        int i = 0;
        for (int num : set) {
            result[i++] = num;
        }
        return result;
    }
}
```

Console × Problems Debug Shell

<terminated> Test [Java Application] C:\Users\Sunil\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x8

```
Array with duplicates removed:
1 2 5 6 9
```

5. Remove duplicates from an ArrayList

```java
public class Test {
    public static void main(String[] args) {
        ArrayList<String> originalList = new ArrayList<>();
        originalList.add("Apple");
        originalList.add("Banana");
        originalList.add("Apple");
        originalList.add("Orange");
        originalList.add("Banana");

        System.out.println("Original List: " + originalList);

        // Using HashSet (order not guaranteed)
        HashSet<String> uniqueElementsSet = new HashSet<>(originalList);
        ArrayList<String> listWithoutDuplicates = new ArrayList<>(uniqueElementsSet);
        System.out.println("List without duplicates (HashSet): " + listWithoutDuplicates);

        // Using LinkedHashSet (preserves insertion order)
        LinkedHashSet<String> uniqueElementsLinkedSet = new LinkedHashSet<>(originalList);
        ArrayList<String> listWithoutDuplicatesOrdered = new ArrayList<>(uniqueElementsLinkedSet);
        System.out.println("List without duplicates (LinkedHashSet, ordered): " + listWithoutDuplicatesOrdered);
    }
}
```

Console × Problems Debug Shell

```
<terminated> Test [Java Application] C:\Users\Sunil\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.7.v20250502-0916\jre\bin\javaw.exe (18-Oct-2025, 5:36:31 pr
Original List: [Apple, Banana, Apple, Orange, Banana]
List without duplicates (HashSet): [Apple, Orange, Banana]
List without duplicates (LinkedHashSet, ordered): [Apple, Banana, Orange]
```

6. Find the missing number in an Array

```java
public class Test {
    public static void main(String[] args) {
        int[] arr = { 1, 2, 4, 5, 6 };
        int n = arr.length + 1; // n is the total count of numbers in the complete sequence
        int expectedSum = n * (n + 1) / 2;
        int actualSum = 0;

        for (int num : arr) {
            actualSum += num;
        }
        int l = expectedSum - actualSum;
        System.out.println("Missing number in arr1: " + l); // Output: 3

    }
}
```

Console × Problems Debug Shell

```
<terminated> Test [Java Application] C:\Users\Sunil\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.7.v20250502-0
Missing number in arr1: 3
```

7. Find the largest and smallest element in an Array

```java
public class ArrayMinMax {

    public static void main(String[] args) {
        int[] arr = {12, 13, 1, 10, 34, 10};

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

        // Iterate through the array starting from the second element
        for (int i = 1; i < arr.length; i++) {
            if (arr[i] < smallest) {
                smallest = arr[i]; // Update smallest if a smaller element is fou
            }
            if (arr[i] > largest) {
                largest = arr[i]; // Update largest if a larger element is found
            }
        }

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

```
Smallest element: 1
Largest element: 34
```

8. Search element in an Array

```java
package test;

import java.util.ArrayList;

public class Test {
    public static void main(String[] args) {
        int[] array = {5, 2, 9, 1, 6, 3};
        int target = 6;
        for (int i = 0; i < array.length; i++) {
            if (array[i] == target) {
                if (i != -1) {
                    System.out.println("Element " + target + " found at index: " + i);
                    } else {
                    System.out.println("Element " + target + " not found in the array.");
                    }           }
            }

    }
}

```

Console × Problems Debug Shell

```
<terminated> Test [Java Application] C:\Users\Sunil\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.7.v20250502-0916\jre\bin
Element 6 found at index: 4
```

9. Array consists of integers and special characters, sum only integers

```
 9
10  public class Test {
11      public static void main(String[] args) {
12          String[] array = { "5", "2", "9", "a", "1", "6", "#", "3" };
13          int sum = 0;
14          for (String element : array) {
15              try {
16                  int num = Integer.parseInt(element);
17                  sum += num;
18              } catch (NumberFormatException e) {
19                  // Ignore non-integer elements
20              }
21          }
22          System.out.println("Sum of integers in the array: " + sum);
23      }
24  }
25
```

Console × Problems Debug Shell

```
<terminated> Test [Java Application] C:\Users\Sunil\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21
Sum of integers in the array: 26
```

10. Find Minimum and Maximum from an Array

```
10  public class Test {
11      public static void main(String[] args) {
12          int[] arr = { 10, 34, 67, 55, 89, 76, 14 };
13
14          // Initialize min and max with the first element of the array
15          int min = arr[0];
16          int max = arr[0];
17
18          // Iterate through the array starting from the second element (index 1)
19          for (int i = 1; i < arr.length; i++) {
20              if (arr[i] < min) {
21                  min = arr[i]; // Update min if current element is smaller
22              }
23              if (arr[i] > max) {
24                  max = arr[i]; // Update max if current element is larger
25              }
26          }
27
28          System.out.println("Minimum value in the array is: " + min);
29          System.out.println("Maximum value in the array is: " + max);
30      }
31  }
```

Console × Problems Debug Shell

```
<terminated> Test [Java Application] C:\Users\Sunil\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.7.v202
Minimum value in the array is: 10
Maximum value in the array is: 89
```

11. Java program to count Odd and Even number from given array

```java
public class Test {
    public static void main(String[] args) {
        // Declare and initialize an array of integers
        int[] numbers = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };

        // Initialize counters for even and odd numbers
        int evenCount = 0;
        int oddCount = 0;

        System.out.println("Original Array: " + Arrays.toString(numbers));

        // Iterate through the array to count even and odd numbers
        for (int number : numbers) { // Enhanced for loop for cleaner iteration
            if (number % 2 == 0) {
                // If the number is divisible by 2, it's even
                evenCount++;
            } else {
                // Otherwise, it's odd
                oddCount++;
            }
        }
        // Print the counts of even and odd numbers
        System.out.println("Number of even elements in the array: " + evenCount);
        System.out.println("Number of odd elements in the array: " + oddCount);
    }
}
```

Console × Problems Debug Shell

```
<terminated> Test [Java Application] C:\Users\Sunil\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.7
Original Array: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Number of even elements in the array: 5
Number of odd elements in the array: 5
```

12. Java program – input array was given [1,1,2,2,3,4,5,5,6,6]

```java
public class Test {
    public static void main(String[] args) {
        int[] array = { 1, 1, 2, 2, 3, 4, 5, 5, 6, 6 };
        List<Integer> result = findNonRepeatedElements(array);
        System.out.println("Non-repeated elements: " + result);
    }

    public static List<Integer> findNonRepeatedElements(int[] array) {
        // Step 1: Count occurrences of each element using a HashMap
        Map<Integer, Integer> countMap = new HashMap<>();
        for (int num : array) {
            countMap.put(num, countMap.getOrDefault(num, 0) + 1);
        }
        // Step 2: Identify elements with count equal to 1 (non repeated)
        List<Integer> nonRepeatedElements = new ArrayList<>();
        for (Map.Entry<Integer, Integer> entry : countMap.entrySet()) {
            if (entry.getValue() == 1) {
                nonRepeatedElements.add(entry.getKey());
            }
        }
        return nonRepeatedElements;
    }
}
```

Console × Problems Debug Shell

```
<terminated> Test [Java Application] C:\Users\Sunil\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.7.v20250502-0916\jre\bin\javaw.exe
Non-repeated elements: [3, 4]
```