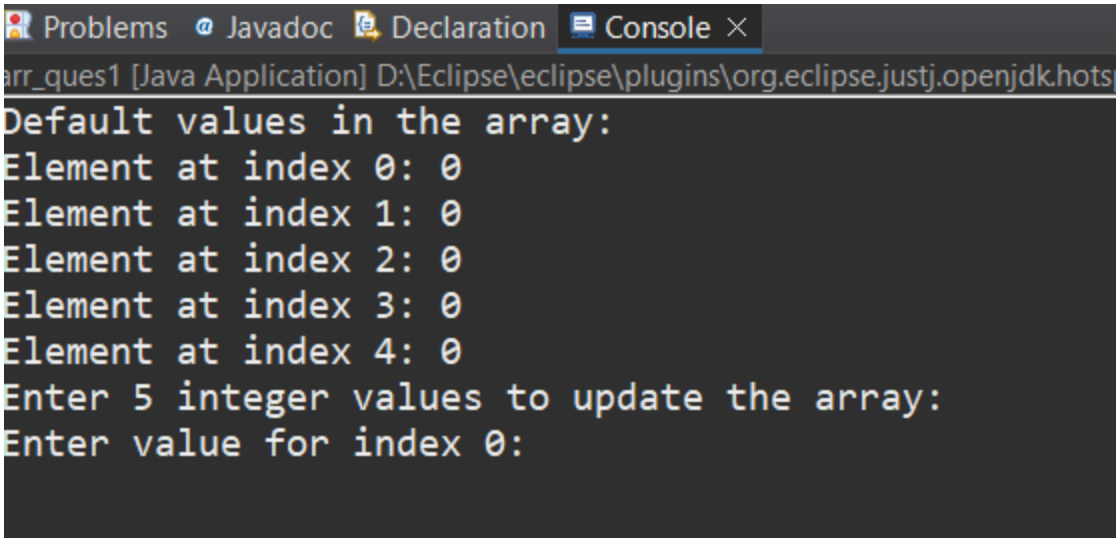
1. Declare a single-dimensional array of 5 integers inside the `main` method. Define a method named `acceptRecord` to get input from the terminal into the array and another method named `printRecord` to print the state of the array to the terminal.

```java
package com.org.ques1;
import java.util.Scanner;
public class arr_ques1 {

	public static void main(String[] args) {
    // Step 1: Declare a single-dimensional array of 5 integers
    int[] numbers = new int[5];

    // Step 2: Traverse the array to print the default values
    System.out.println("Default values in the array:");
    for (int i = 0; i < numbers.length; i++) {
        System.out.println("Element at index " + i + ": " + numbers[i]);
    }

    // Step 3: Accept records from the user
    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter 5 integer values to update the array:");
    for (int i = 0; i < numbers.length; i++) {
        System.out.print("Enter value for index " + i + ": ");
        numbers[i] = scanner.nextInt();
    }

    // Step 4: Print the updated values of the array
    System.out.println("Updated values in the array:");
    for (int i = 0; i < numbers.length; i++) {
        System.out.println("Element at index " + i + ": " + numbers[i]);
    }

    // Close the scanner to prevent resource leaks
    scanner.close();
  }
}
```



Declare a single-dimensional array of 5 integers inside the `main` method. Define a method named `acceptRecord` to get input from the terminal into the array and another method named `printRecord` to print the state of the array to the terminal.

```
package ques2;
import java.util.Scanner;
public class arr_ques2 {

        public static void main(String[] args) {
    // Step 1: Declare a single-dimensional array of 5 integers
    int[] numbers = new int[5];

    // Step 2: Call the acceptRecord method to get input from the user
    acceptRecord(numbers);

    // Step 3: Call the printRecord method to print the state of the array
    printRecord(numbers);
  }
  // Method to get input from the user and update the array
  public static void acceptRecord(int[] array) {
    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter " + array.length + " integer values to update the array:");
    for (int i = 0; i < array.length; i++) {
      System.out.print("Enter value for index " + i + ": ");
      array[i] = scanner.nextInt();
    }
```

```
  }
  // Method to print the state of the array
  public static void printRecord(int[] array) {
     System.out.println("Array contents:");
     for (int i = 0; i < array.length; i++) {
        System.out.println("Element at index " + i + ": " + array[i]);
     }
  }
}
```

```
<terminated> arr_ques2 [Java Application] D:\Eclipse\eclipse\plugins\org.ec
Enter 5 integer values to update the array:
Enter value for index 0: 3
Enter value for index 1: 3
Enter value for index 2: 32
Enter value for index 3: 5
Enter value for index 4: 4
Array contents:
Element at index 0: 3
Element at index 1: 3
Element at index 2: 32
Element at index 3: 5
Element at index 4: 4
```

Write a program to find the maximum and minimum values in a single-dimensional array of integers.

```
package ques3;
public class ques3 {

        public static void main(String[] args) {
     // Step 1: Declare and initialize a single-dimensional array of integers
     int[] numbers = {3, 5, 7, 2, 8, -1, 4};
     // Step 2: Find and print the maximum and minimum values
     findMinMax(numbers);
  }
```

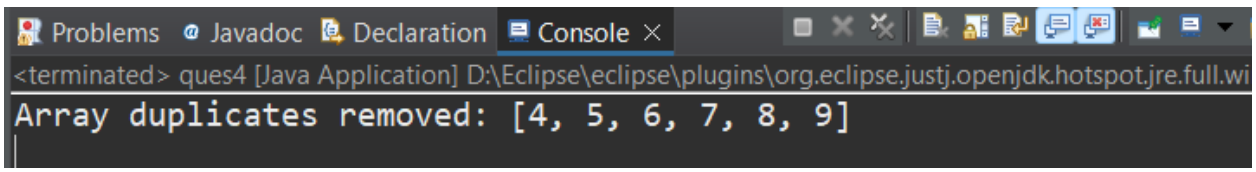
```
    // Method to find and print the maximum and minimum values in the array
    public static void findMinMax(int[] array) {
        // Check if the array is empty
        if (array.length == 0) {
            System.out.println("The array is empty.");
            return;
        }
        // Initialize min and max with the first element of the array
        int min = array[0];
        int max = array[0];
        // Traverse the array to find the min and max values
        for (int i = 1; i < array.length; i++) {
            if (array[i] < min) {
                min = array[i];
            }
            if (array[i] > max) {
                max = array[i];
            }
        }
        // Print the results
        System.out.println("Minimum value: " + min);
        System.out.println("Maximum value: " + max);
    }
}
```

```
<terminated> ques3 [Java Application] D:\Eclipse
Minimum value: -1
Maximum value: 8
```

Write a program to remove duplicate elements from a single-dimensional array of integers.

```
package ques4;
import java.util.Arrays;
import java.util.LinkedHashSet;
import java.util.Set;
```

```java
public class ques4 {

	public static void main(String[] args) {
		// Step 1: Declare and initialize a single-dimensional array of integers
		int[] numbers = {4, 5, 6, 4, 7, 8, 7, 9, 5};
		// Step 2: Remove duplicates
		int[] uniqueNumbers = removeDuplicates(numbers);
		// Step 3: Print the array with duplicates removed
		System.out.println("Array duplicates removed: " +
Arrays.toString(uniqueNumbers));
	}
	// Method to remove duplicates from the array
	public static int[] removeDuplicates(int[] array) {
		// Use a LinkedHashSet to maintain insertion order and remove duplicates
		Set<Integer> uniqueSet = new LinkedHashSet<>();

		// Add elements from the array to the set
		for (int num : array) {
			uniqueSet.add(num);
		}
		// Convert the set back to an array
		int[] uniqueArray = new int[uniqueSet.size()];
		int index = 0;
		for (int num : uniqueSet) {
			uniqueArray[index++] = num;
		}
		return uniqueArray;
	}
}
```



Write a program to find the intersection of two single-dimensional arrays.

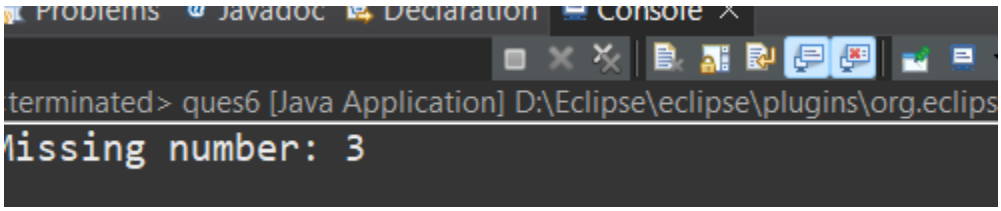
```java
import java.util.HashSet;
import java.util.Set;
```

```java
import java.util.Arrays;
public class ques5 {
        public static int[] findIntersection(int[] arr1, int[] arr2) {
    // Create a set to store the elements of the first array
    Set<Integer> set1 = new HashSet<>();
    for (int num : arr1) {
       set1.add(num);
    }
    // Create a set to store the intersection result
    Set<Integer> intersection = new HashSet<>();
    for (int num : arr2) {
       if (set1.contains(num)) {
          intersection.add(num);
       }
    }
    // Convert the result set to an array
    int[] result = new int[intersection.size()];
    int index = 0;
    for (int num : intersection) {
       result[index++] = num;
    }
    return result;
  }
  public static void main(String[] args) {
    int[] arr1 = {1, 2, 3, 4, 5};
    int[] arr2 = {4, 5, 6, 7, 8};
    int[] result = findIntersection(arr1, arr2);
    System.out.println("Intersection: " + Arrays.toString(result));
  }
}
```

```
<terminated> ques5 [Java Application] D:\Eclipse\eclipse\plugins\org.eclipse
Intersection: [4, 5]
```

Write a program to find the missing number in an array of integers ranging from 1 to N.

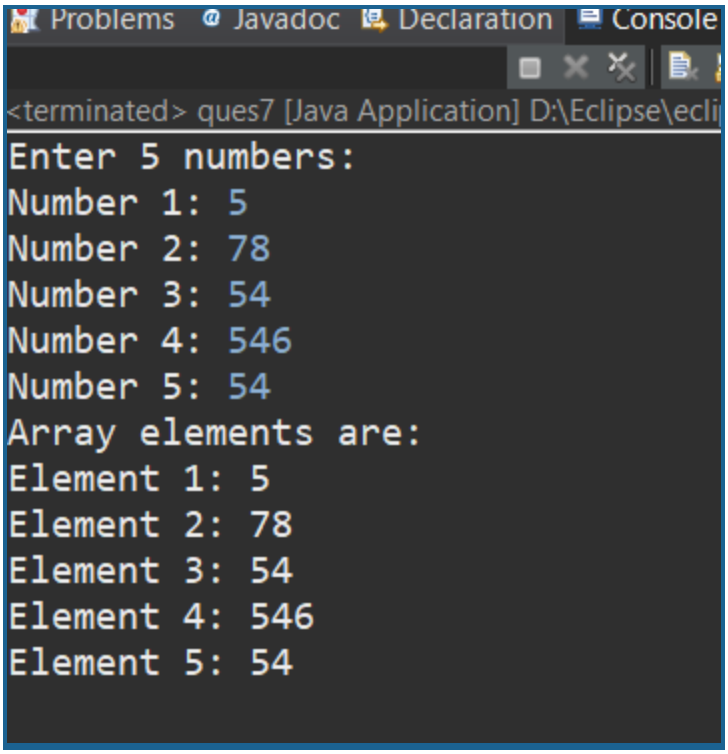
```
public class ques6 {
        public static int findMissingNumber(int[] arr, int N) {
    // Calculate the expected sum of numbers from 1 to N
    int expectedSum = N * (N + 1) / 2;

    // Calculate the actual sum of the array elements
    int actualSum = 0;
    for (int num : arr) {
       actualSum += num;
    }

    // The missing number is the difference between the expected sum and the actual sum
    return expectedSum - actualSum;
  }
  public static void main(String[] args) {
    int[] arr = {1, 2, 4, 5, 6};
    int N = 6; // The maximum number expected

    int missingNumber = findMissingNumber(arr, N);
    System.out.println("Missing number: " + missingNumber);
  }
}
```



Declare a single-dimensional array as a field inside a class and instantiate it inside the class constructor. Define methods named `acceptRecord` and `printRecord` within the class and test their functionality.

```
import java.util.Scanner;
public class ques7 {
private int[] array;

  public ques7(int size) {
```

```
        array = new int[size];
    }

    public void acceptRecord() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter " + array.length + " numbers:");
        for (int i = 0; i < array.length; i++) {
            System.out.print("Number " + (i + 1) + ": ");
            array[i] = scanner.nextInt();
        }
    }

    public void printRecord() {
        System.out.println("Array elements are:");
        for (int i = 0; i < array.length; i++) {
            System.out.println("Element " + (i + 1) + ": " + array[i]);
        }
    }

    public static void main(String[] args) {
            ques7 handler = new ques7(5);

handler.acceptRecord();
        handler.printRecord();
    }
}
```



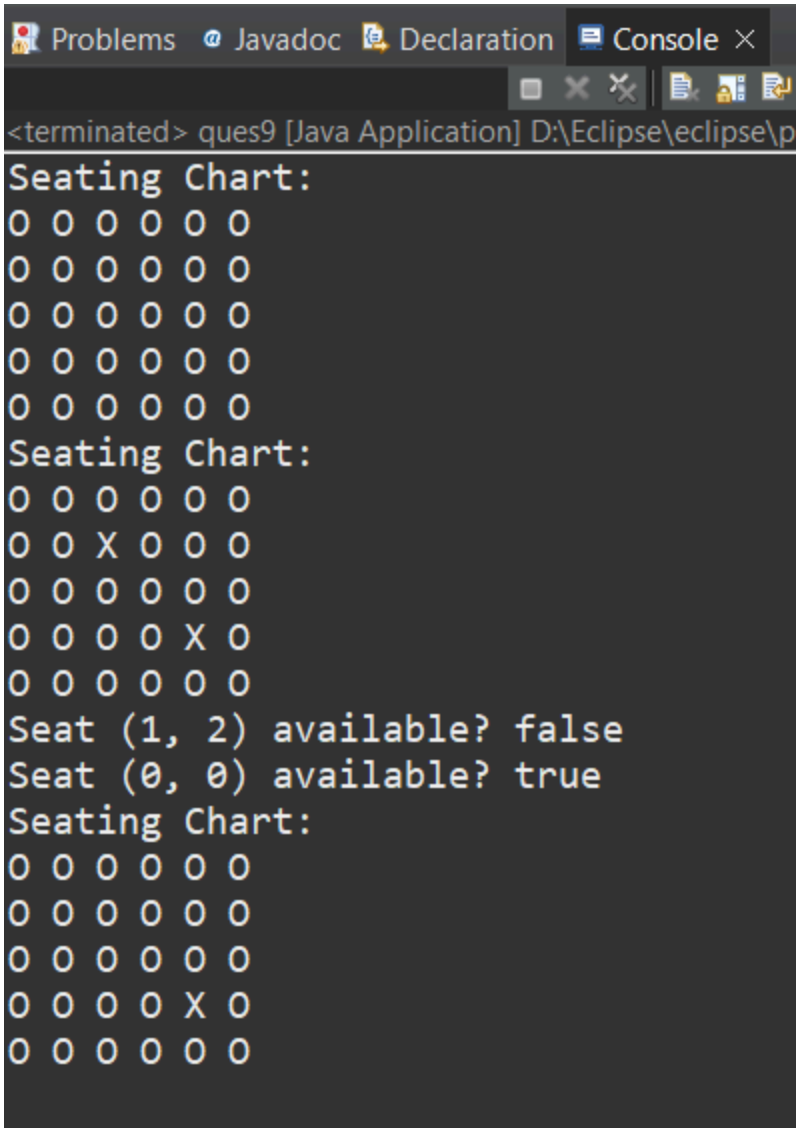
You need to implement a system to manage airplane seat assignments. The airplane has seats arranged in rows and columns. Implement functionalities to:

- **Initialize the seating arrangement with a given number of rows and columns.**
- **Book a seat to mark it as occupied.**
- **Cancel a booking to mark a seat as available.**
- **Check seat availability to determine if a specific seat is available.**
- **Display the current seating chart.**

```
public class ques9 {

        private char[][] seats;
  private final char AVAILABLE = 'O';
  private final char OCCUPIED = 'X';

  public ques9(int rows, int columns) {
     seats = new char[rows][columns];
     // Initialize all seats as available
     for (int i = 0; i < rows; i++) {
        for (int j = 0; j < columns; j++) {
           seats[i][j] = AVAILABLE;
```

```
        }
    }
}
// Method to book a seat
public boolean bookSeat(int row, int column) {
    if (isValidSeat(row, column)) {
        if (seats[row][column] == AVAILABLE) {
            seats[row][column] = OCCUPIED;
            return true;
        }
    }
    return false;
}
// Method to cancel a booking
public boolean cancelBooking(int row, int column) {
    if (isValidSeat(row, column)) {
        if (seats[row][column] == OCCUPIED) {
            seats[row][column] = AVAILABLE;
            return true;
        }
    }
    return false;
}
// Method to check if a seat is available
public boolean isSeatAvailable(int row, int column) {
    if (isValidSeat(row, column)) {
        return seats[row][column] == AVAILABLE;
    }
    return false;
}
// Method to display the seating chart
public void displaySeatingChart() {
    System.out.println("Seating Chart:");
    for (int i = 0; i < seats.length; i++) {
        for (int j = 0; j < seats[i].length; j++) {
            System.out.print(seats[i][j] + " ");
        }
        System.out.println();
    }
}
```

```
  // Helper method to validate seat coordinates
  private boolean isValidSeat(int row, int column) {
      return row >= 0 && row < seats.length && column >= 0 && column <
seats[row].length;
  }
  public static void main(String[] args) {
          ques9 airplane = new ques9(5, 6); // 5 rows and 6 columns
      // Display initial seating chart
      airplane.displaySeatingChart();
      // Book a few seats
      airplane.bookSeat(1, 2);
      airplane.bookSeat(3, 4);
      // Display seating chart after booking
      airplane.displaySeatingChart();
      // Check seat availability
      System.out.println("Seat (1, 2) available? " + airplane.isSeatAvailable(1, 2));
      System.out.println("Seat (0, 0) available? " + airplane.isSeatAvailable(0, 0));
      // Cancel a booking
      airplane.cancelBooking(1, 2);
      // Display seating chart after canceling
      airplane.displaySeatingChart();
  }
}
```


Problems
Javadoc
Declaration
Console
<terminated> ques9 [Java Application] D:\Eclipse\eclipse\p
Seating Chart:
O O O O O O
O O O O O O
O O O O O O
O O O O O O
O O O O O O
Seating Chart:
O O O O O O
O O X O O O
O O O O O O
O O O O X O
O O O O O O
Seat (1, 2) available? false
Seat (0, 0) available? true
Seating Chart:
O O O O O O
O O O O O O
O O O O O O
O O O O X O
O O O O O O