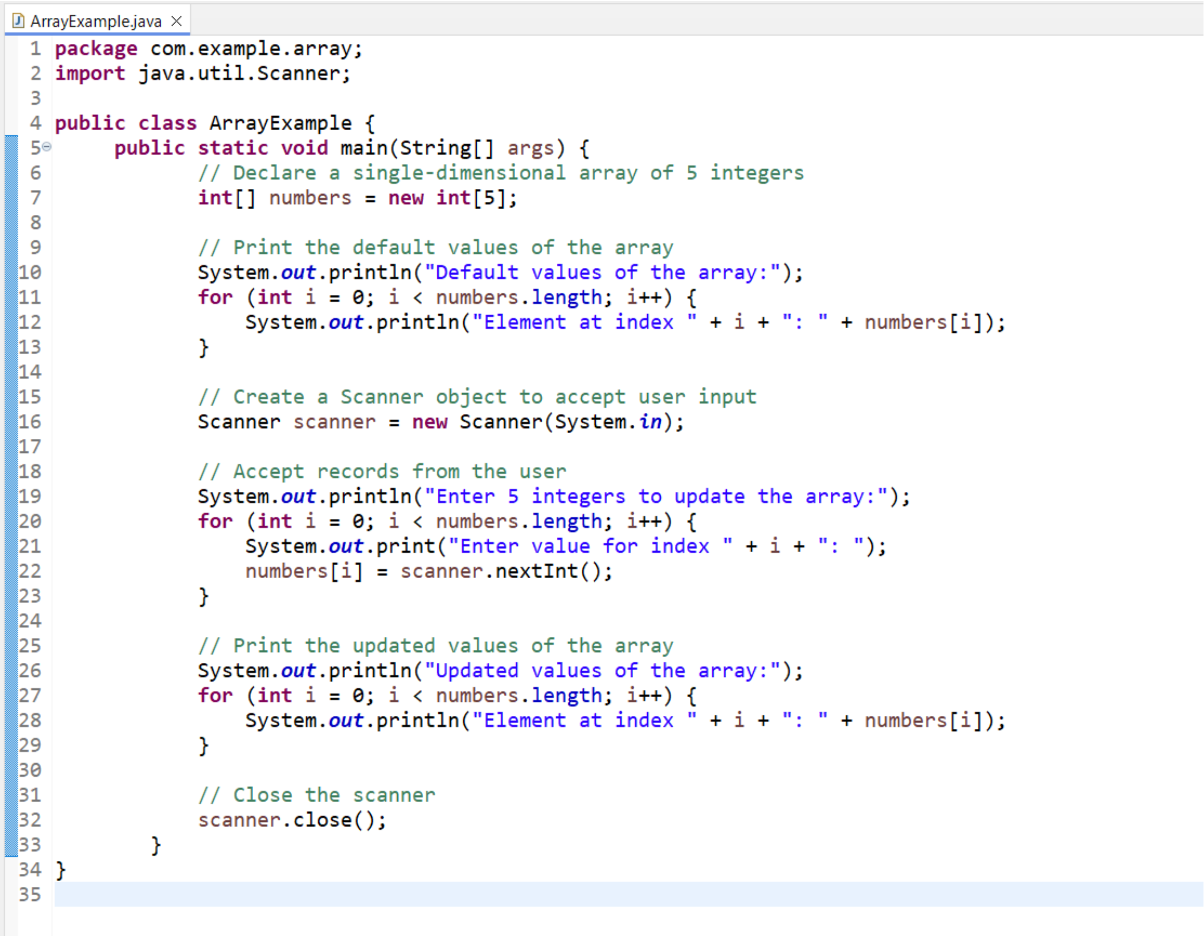
ASSIGNMENT NO.7

1. Declare a single-dimensional array of 5 integers inside the `main` method. Traverse the array to print the default values. Then accept records from the user and print the updated values of the array.

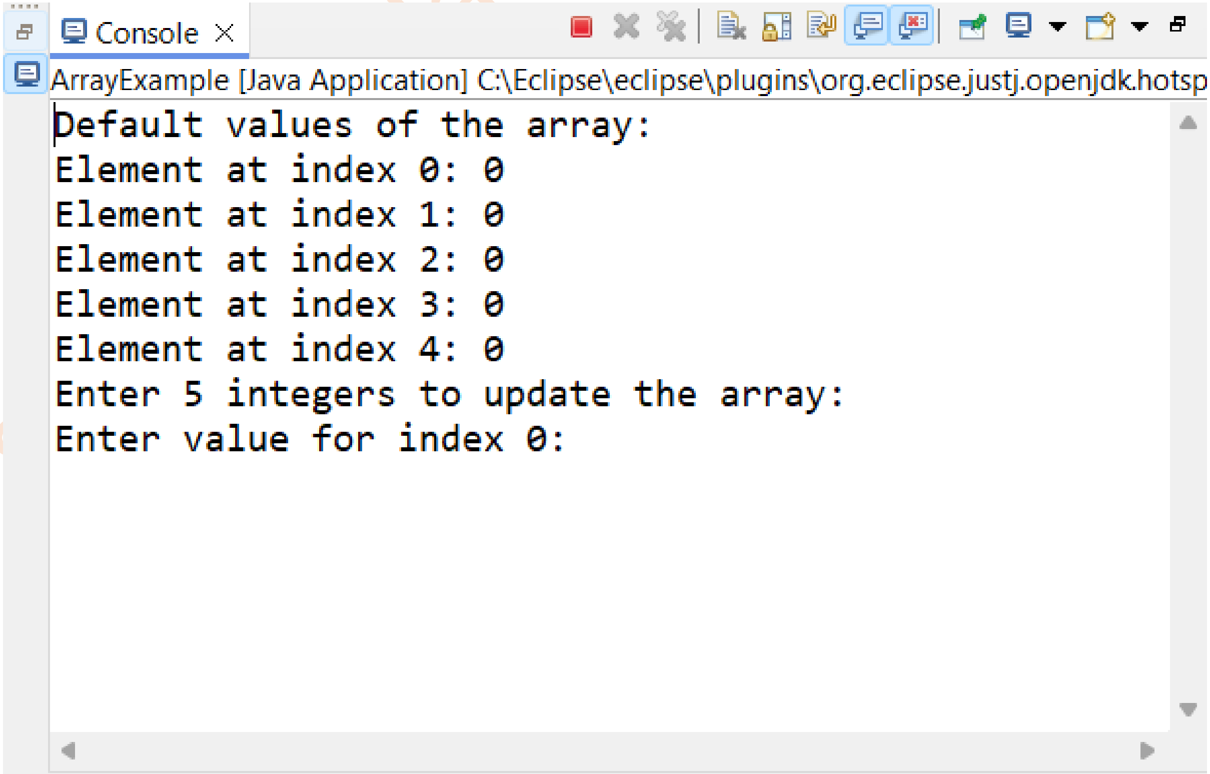
Ans:

```
package com.example.array
public static void main(String[] args) {
        // Declare a single-dimensional array of 5 integers
        int[] numbers = new int[5];

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

        // Create a Scanner object to accept user input
        Scanner scanner = new Scanner(System.in);

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

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

        // Close the scanner
        scanner.close();
```

}



```java
package com.example.array;
import java.util.Scanner;

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

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

            // Create a Scanner object to accept user input
            Scanner scanner = new Scanner(System.in);

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

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

            // Close the scanner
            scanner.close();
        }
}
```



ArrayExample [Java Application] C:\Eclipse\eclipse\plugins\org.eclipse.justj.openjdk.hotsp

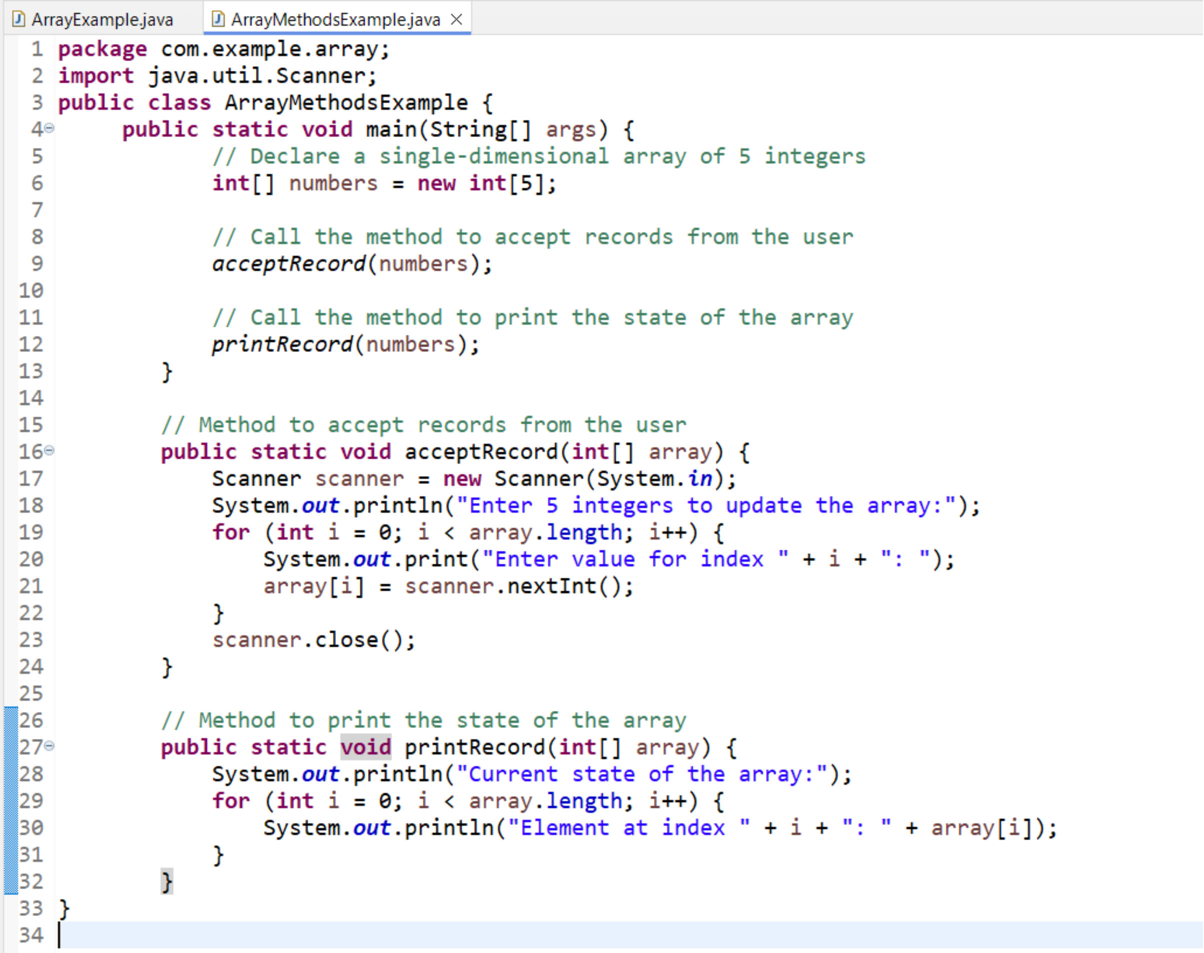
```
Default values of the array:
Element at index 0: 0
Element at index 1: 0
Element at index 2: 0
Element at index 3: 0
Element at index 4: 0
Enter 5 integers to update the array:
Enter value for index 0:
```

2. 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.

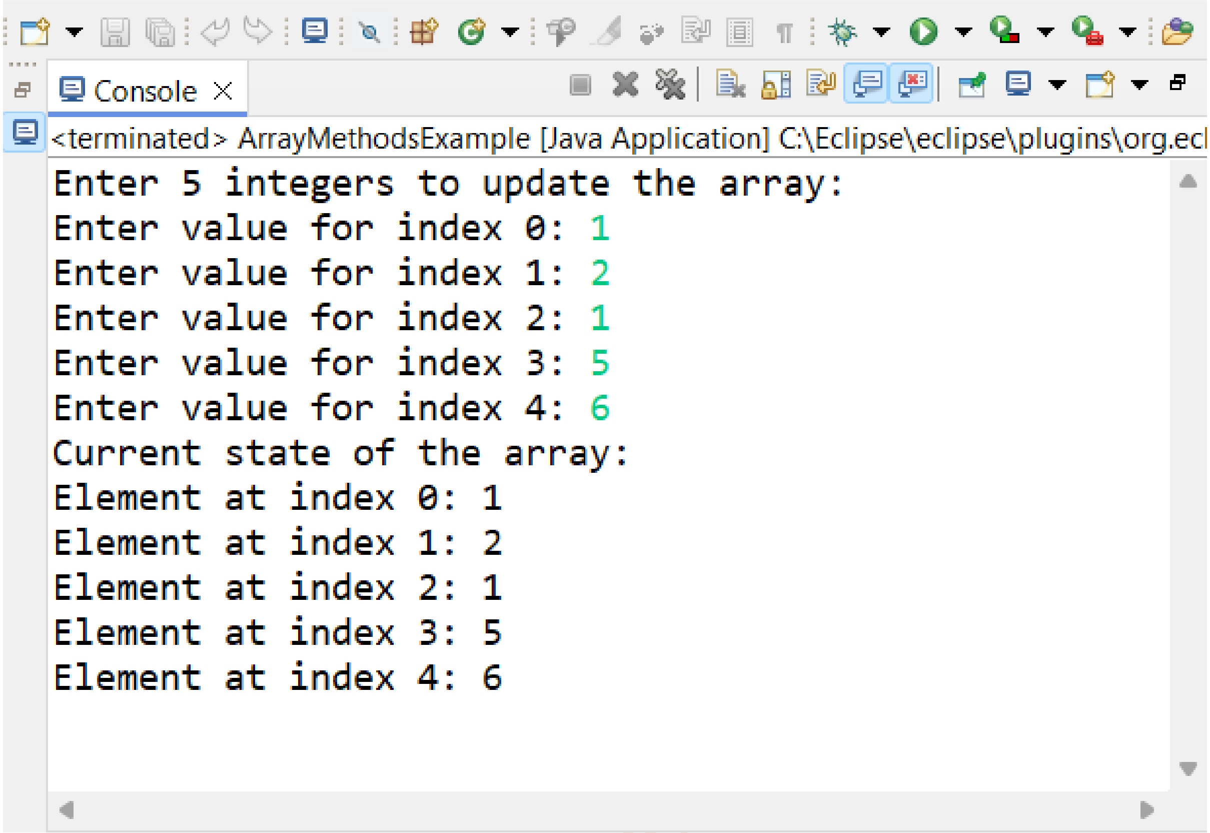
Ans:

```
import java.util.Scanner;

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

        // Call the method to accept records from the user
        acceptRecord(numbers);

        // Call the method to print the state of the array
        printRecord(numbers);
    }

    // Method to accept records from the user
    public static void acceptRecord(int[] array) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter 5 integers to update the array:");
        for (int i = 0; i < array.length; i++) {
            System.out.print("Enter value for index " + i + ": ");
            array[i] = scanner.nextInt();
        }
        scanner.close();
    }

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

}



```java
package com.example.array;
import java.util.Scanner;
public class ArrayMethodsExample {
     public static void main(String[] args) {
            // Declare a single-dimensional array of 5 integers
            int[] numbers = new int[5];

            // Call the method to accept records from the user
            acceptRecord(numbers);

            // Call the method to print the state of the array
            printRecord(numbers);
        }

        // Method to accept records from the user
        public static void acceptRecord(int[] array) {
            Scanner scanner = new Scanner(System.in);
            System.out.println("Enter 5 integers to update the array:");
            for (int i = 0; i < array.length; i++) {
                System.out.print("Enter value for index " + i + ": ");
                array[i] = scanner.nextInt();
            }
            scanner.close();
        }

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

```



```
<terminated> ArrayMethodsExample [Java Application] C:\Eclipse\eclipse\plugins\org.ecl
Enter 5 integers to update the array:
Enter value for index 0: 1
Enter value for index 1: 2
Enter value for index 2: 1
Enter value for index 3: 5
Enter value for index 4: 6
Current state of the array:
Element at index 0: 1
Element at index 1: 2
Element at index 2: 1
Element at index 3: 5
Element at index 4: 6
```

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

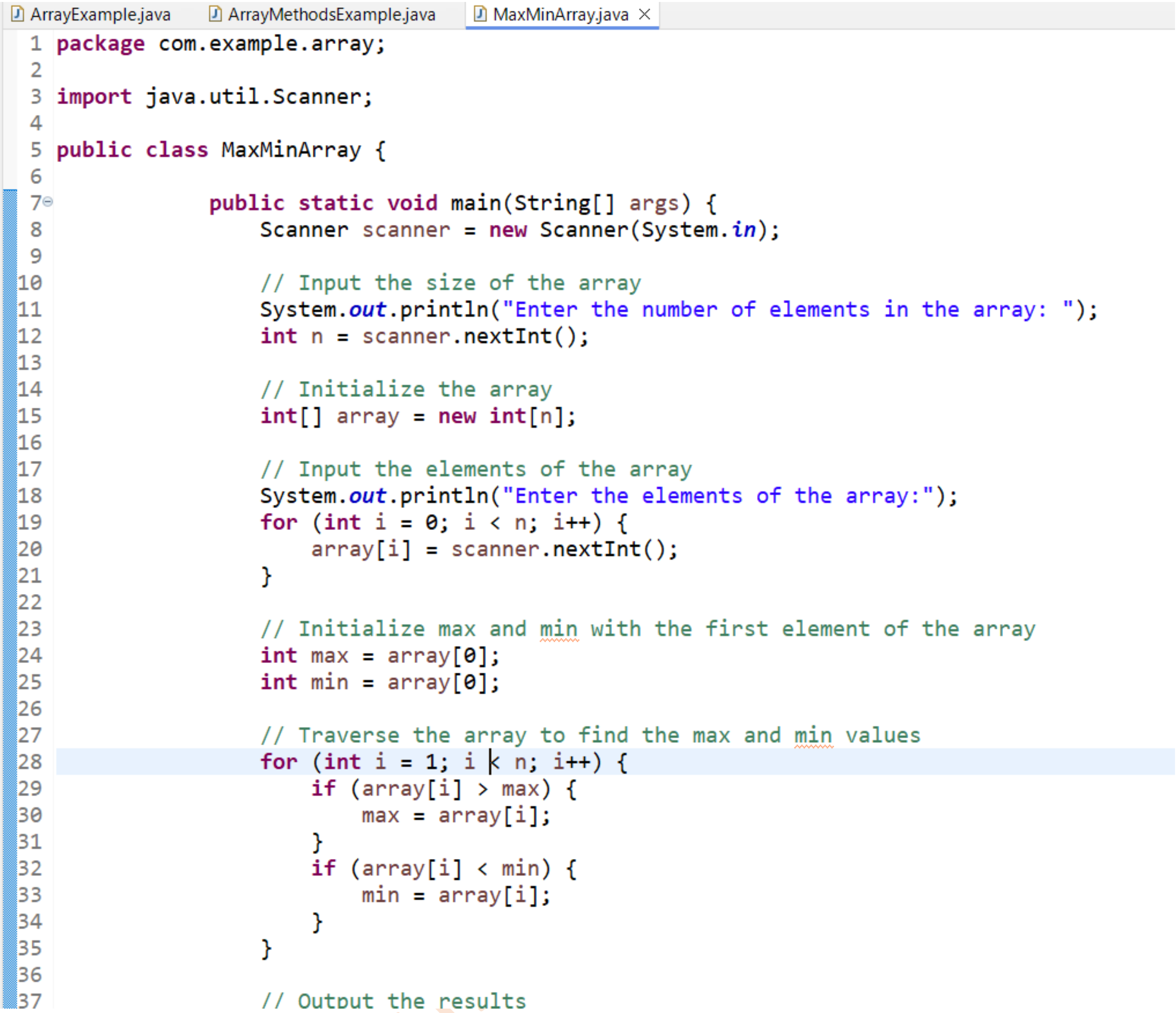
Ans:

```
package com.example.array;
 import java.util.Scanner;
 public class MaxMinArray {

              public static void main(String[] args) {
                 Scanner scanner = new Scanner(System.in);

                 // Input the size of the array
                 System.out.println("Enter the number of elements in the array: ");
                 int n = scanner.nextInt();

                 // Initialize the array
                 int[] array = new int[n];

                 // Input the elements of the array
                 System.out.println("Enter the elements of the array:");
                 for (int i = 0; i < n; i++) {
                    array[i] = scanner.nextInt();
                 }

                 // Initialize max and min with the first element of the array
```

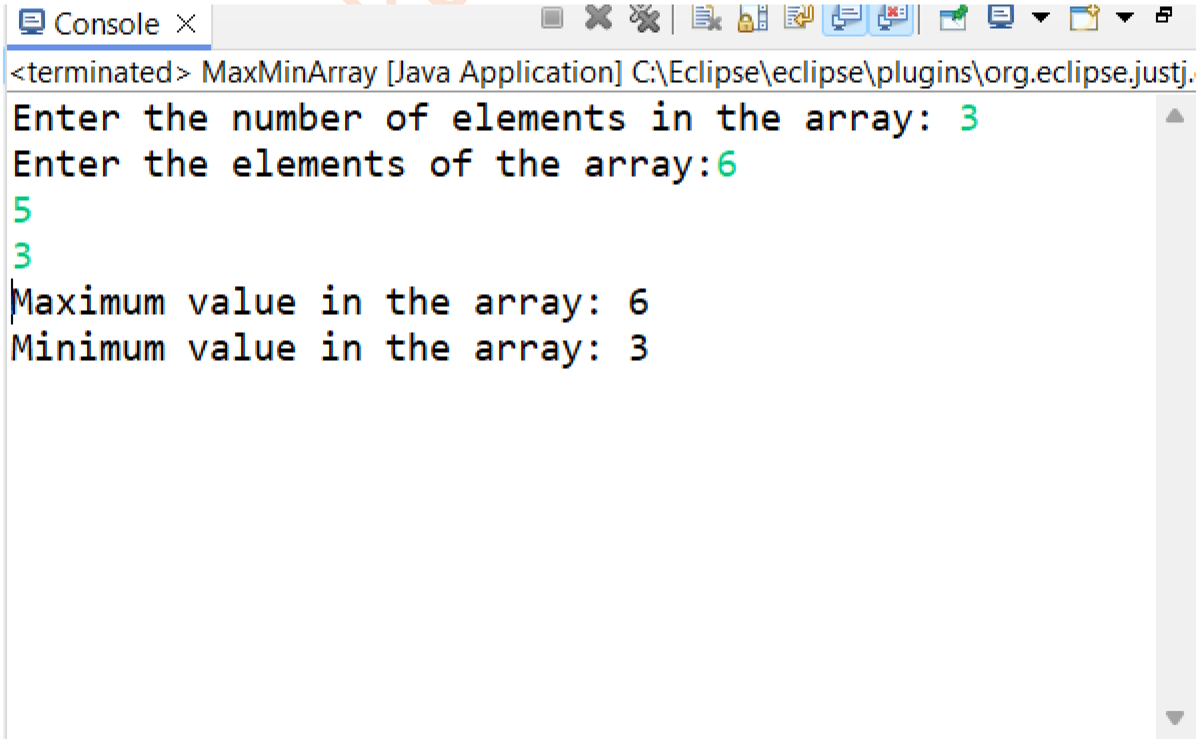
```
        int max = array[0];
        int min = array[0];

        // Traverse the array to find the max and min values
        for (int i = 1; i < n; i++) {
            if (array[i] > max) {
                max = array[i];
            }
            if (array[i] < min) {
                min = array[i];
            }
        }

        // Output the results
        System.out.println("Maximum value in the array: " + max);
        System.out.println("Minimum value in the array: " + min);

        scanner.close();

    }
```

ASSIGNMENT NO.7

}



```java
package com.example.array;

import java.util.Scanner;

public class MaxMinArray {

            public static void main(String[] args) {
                Scanner scanner = new Scanner(System.in);

                // Input the size of the array
                System.out.println("Enter the number of elements in the array: ");
                int n = scanner.nextInt();

                // Initialize the array
                int[] array = new int[n];

                // Input the elements of the array
                System.out.println("Enter the elements of the array:");
                for (int i = 0; i < n; i++) {
                    array[i] = scanner.nextInt();
                }

                // Initialize max and min with the first element of the array
                int max = array[0];
                int min = array[0];

                // Traverse the array to find the max and min values
                for (int i = 1; i < n; i++) {
                    if (array[i] > max) {
                        max = array[i];
                    }
                    if (array[i] < min) {
                        min = array[i];
                    }
                }

                // Output the results
```



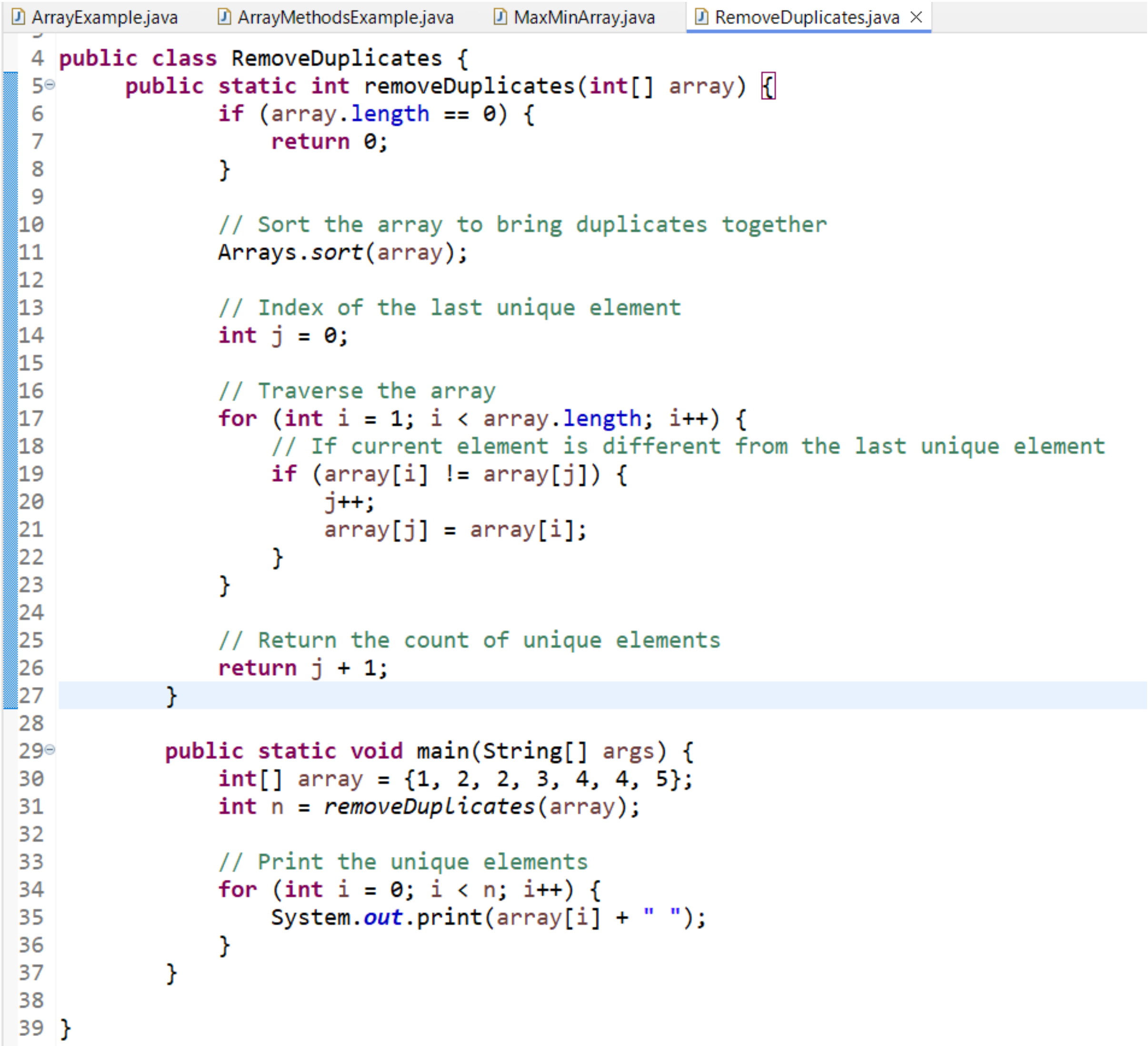
```
<terminated> MaxMinArray [Java Application] C:\Eclipse\eclipse\plugins\org.eclipse.justj.
Enter the number of elements in the array: 3
Enter the elements of the array:6
5
3
Maximum value in the array: 6
Minimum value in the array: 3
```

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

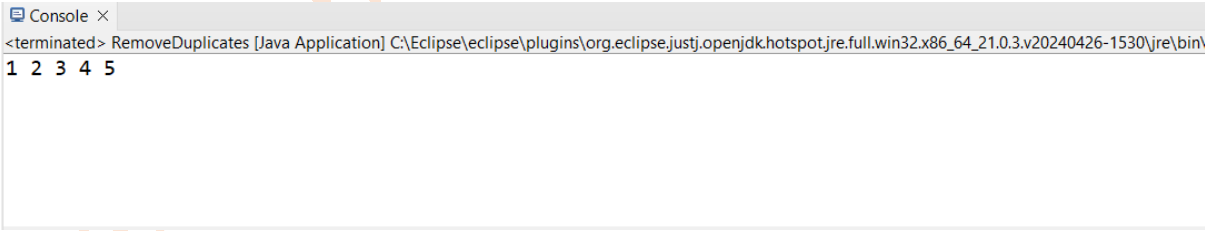
Ans:

```
package com.example.array;
import java.util.Arrays;

public class RemoveDuplicates {
    public static int removeDuplicates(int[] array) {
        if (array.length == 0) {
          return 0;
        }

        // Sort the array to bring duplicates together
        Arrays.sort(array);

        // Index of the last unique element
        int j = 0;

        // Traverse the array
        for (int i = 1; i < array.length; i++) {
          // If current element is different from the last unique element
          if (array[i] != array[j]) {
            j++;
            array[j] = array[i];
          }
        }

        // Return the count of unique elements
        return j + 1;
      }

      public static void main(String[] args) {
        int[] array = {1, 2, 2, 3, 4, 4, 5};
        int n = removeDuplicates(array);

        // Print the unique elements
        for (int i = 0; i < n; i++) {
          System.out.print(array[i] + " ");
        }
      }
```

}



```java
public class RemoveDuplicates {
    public static int removeDuplicates(int[] array) {
        if (array.length == 0) {
            return 0;
        }

        // Sort the array to bring duplicates together
        Arrays.sort(array);

        // Index of the last unique element
        int j = 0;

        // Traverse the array
        for (int i = 1; i < array.length; i++) {
            // If current element is different from the last unique element
            if (array[i] != array[j]) {
                j++;
                array[j] = array[i];
            }
        }

        // Return the count of unique elements
        return j + 1;
    }

    public static void main(String[] args) {
        int[] array = {1, 2, 2, 3, 4, 4, 5};
        int n = removeDuplicates(array);

        // Print the unique elements
        for (int i = 0; i < n; i++) {
            System.out.print(array[i] + " ");
        }
    }

}
```

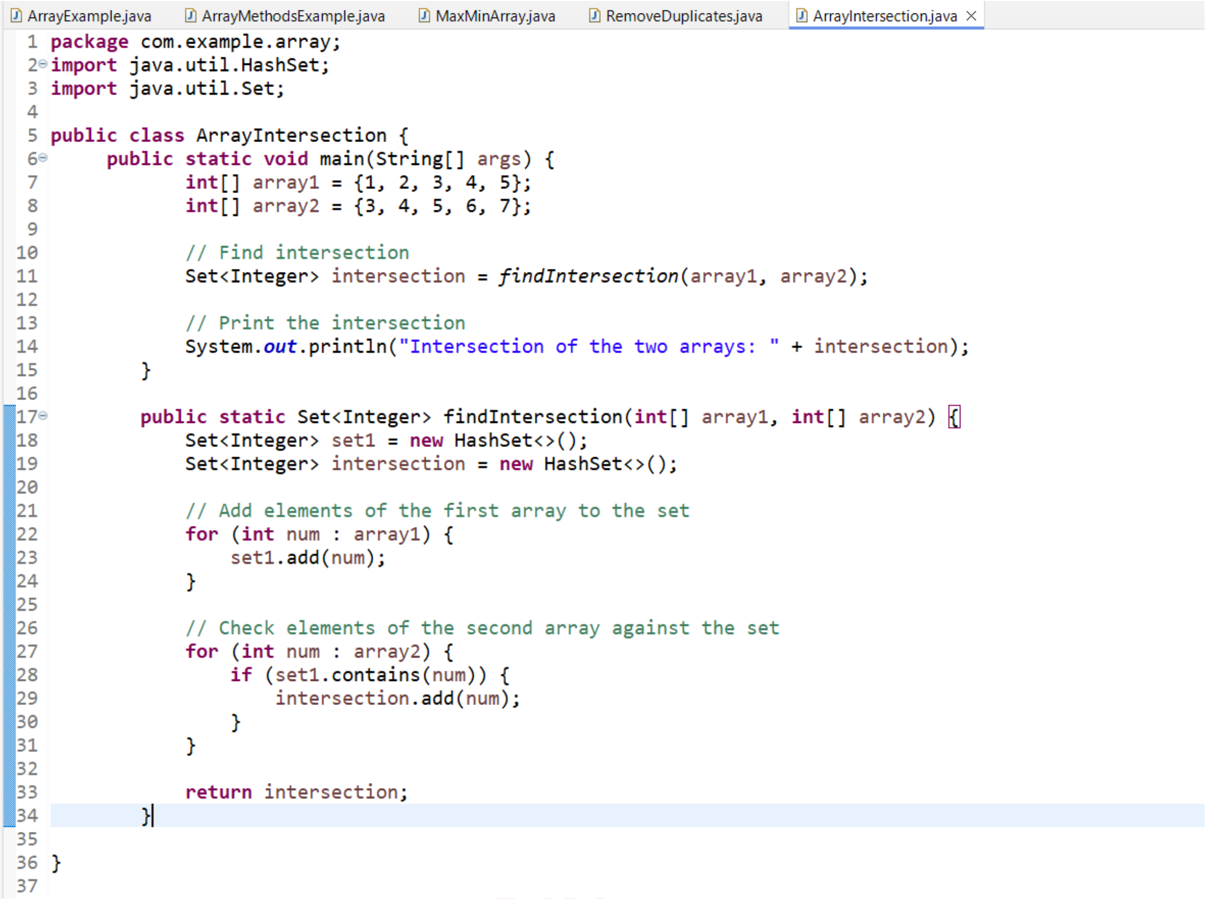


```
<terminated> RemoveDuplicates [Java Application] C:\Eclipse\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.3.v20240426-1530\jre\bin\
1 2 3 4 5
```

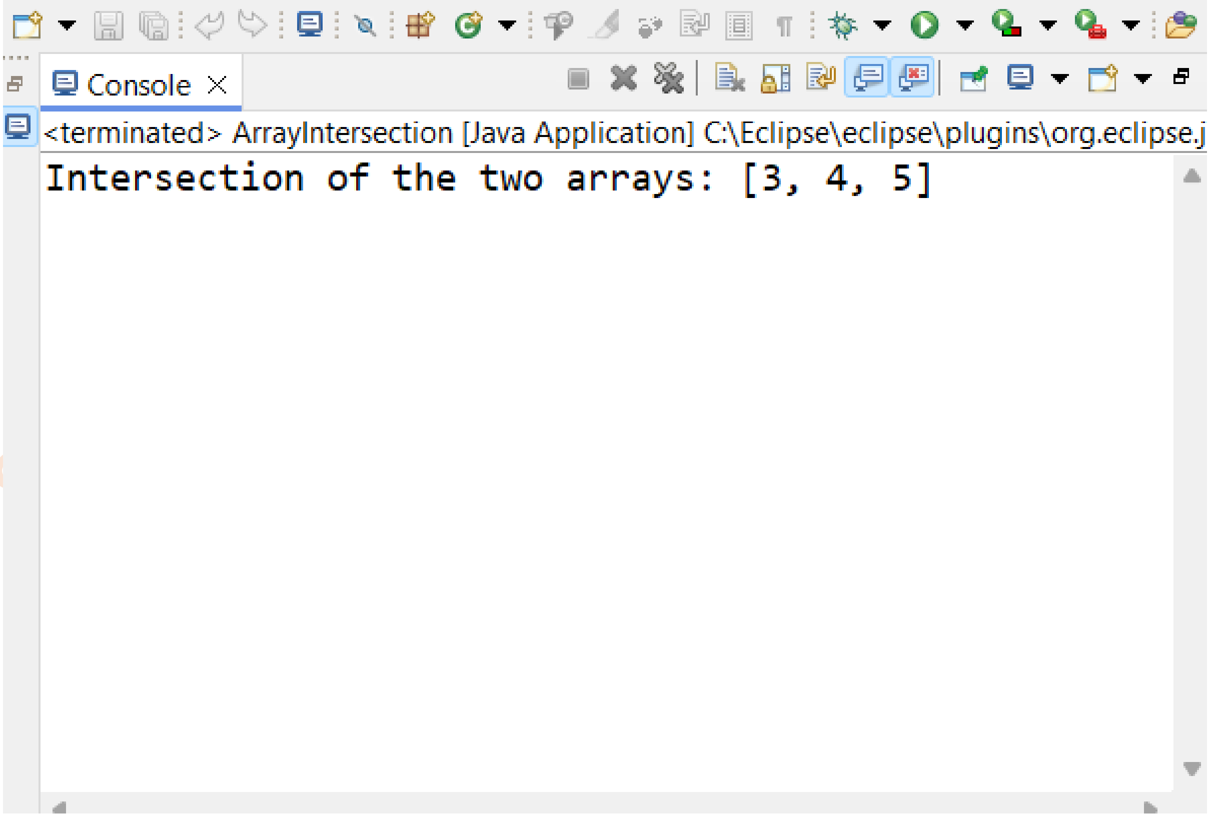
5. Write a program to find the intersection of two single-dimensional arrays.
   Ans:

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

public class ArrayIntersection {
   public static void main(String[] args) {
    int[] array1 = {1, 2, 3, 4, 5};
    int[] array2 = {3, 4, 5, 6, 7};

    // Find intersection
    Set<Integer> intersection = findIntersection(array1, array2);

    // Print the intersection
    System.out.println("Intersection of the two arrays: " + intersection);
}

public static Set<Integer> findIntersection(int[] array1, int[] array2) {
    Set<Integer> set1 = new HashSet<>();
    Set<Integer> intersection = new HashSet<>();

    // Add elements of the first array to the set
    for (int num : array1) {
        set1.add(num);
    }

    // Check elements of the second array against the set
    for (int num : array2) {
        if (set1.contains(num)) {
            intersection.add(num);
        }
    }

    return intersection;
}
```

}



```java
package com.example.array;
import java.util.HashSet;
import java.util.Set;

public class ArrayIntersection {
    public static void main(String[] args) {
        int[] array1 = {1, 2, 3, 4, 5};
        int[] array2 = {3, 4, 5, 6, 7};

        // Find intersection
        Set<Integer> intersection = findIntersection(array1, array2);

        // Print the intersection
        System.out.println("Intersection of the two arrays: " + intersection);
    }

    public static Set<Integer> findIntersection(int[] array1, int[] array2) {
        Set<Integer> set1 = new HashSet<>();
        Set<Integer> intersection = new HashSet<>();

        // Add elements of the first array to the set
        for (int num : array1) {
            set1.add(num);
        }

        // Check elements of the second array against the set
        for (int num : array2) {
            if (set1.contains(num)) {
                intersection.add(num);
            }
        }

        return intersection;
    }

}
```

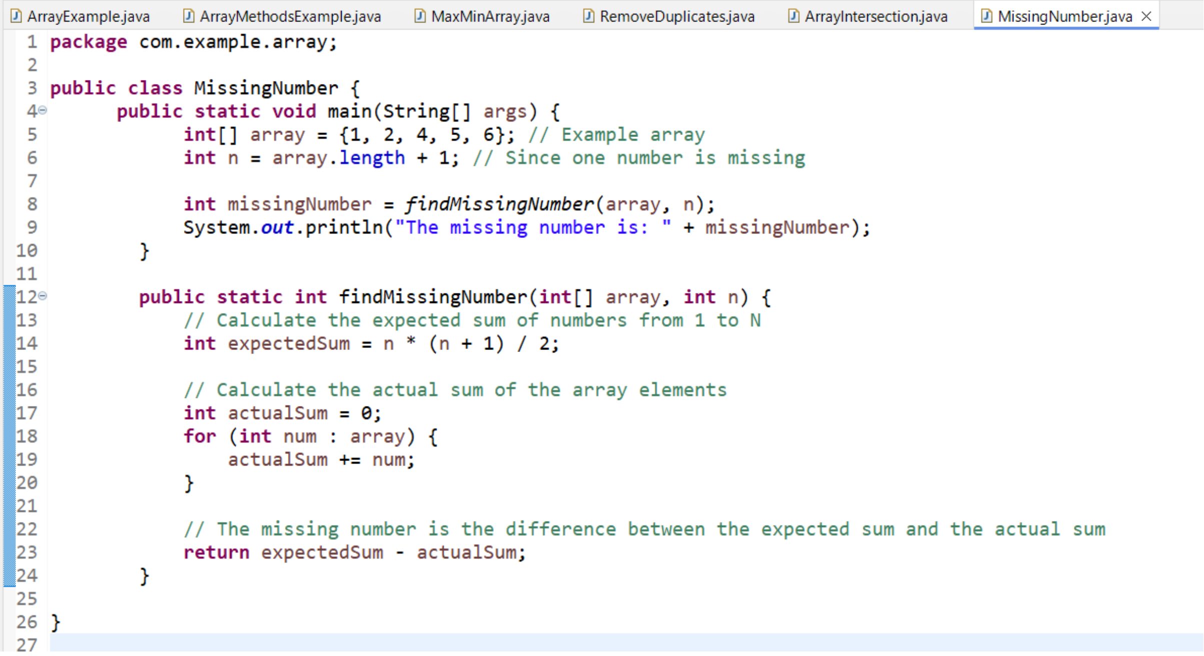


```
<terminated> ArrayIntersection [Java Application] C:\Eclipse\eclipse\plugins\org.eclipse.j
Intersection of the two arrays: [3, 4, 5]
```

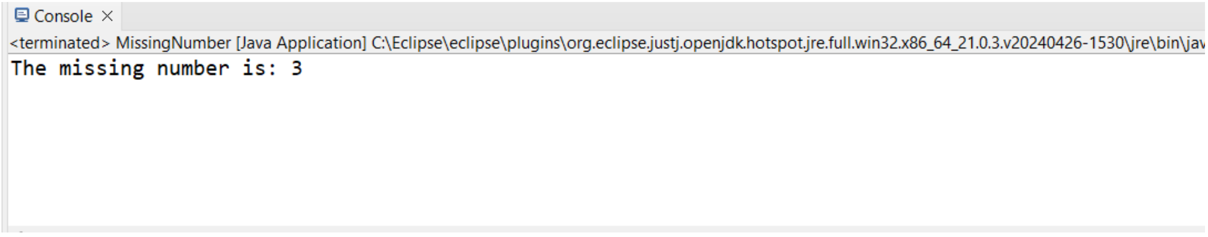
6. Write a program to find the missing number in an array of integers ranging from 1 to N.

Ans:

```
public class MissingNumber {
    public static void main(String[] args) {
        int[] array = {1, 2, 4, 5, 6}; // Example array
        int n = array.length + 1; // Since one number is missing

        int missingNumber = findMissingNumber(array, n);
        System.out.println("The missing number is: " + missingNumber);
    }

    public static int findMissingNumber(int[] array, 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 : array) {
            actualSum += num;
        }

        // The missing number is the difference between the expected sum and the actual sum
        return expectedSum - actualSum;
    }
}
```



```
package com.example.array;

public class MissingNumber {
	public static void main(String[] args) {
		int[] array = {1, 2, 4, 5, 6}; // Example array
		int n = array.length + 1; // Since one number is missing

		int missingNumber = findMissingNumber(array, n);
		System.out.println("The missing number is: " + missingNumber);
	}

	public static int findMissingNumber(int[] array, 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 : array) {
		    actualSum += num;
		}

		// The missing number is the difference between the expected sum and the actual sum
		return expectedSum - actualSum;
	}

}
```



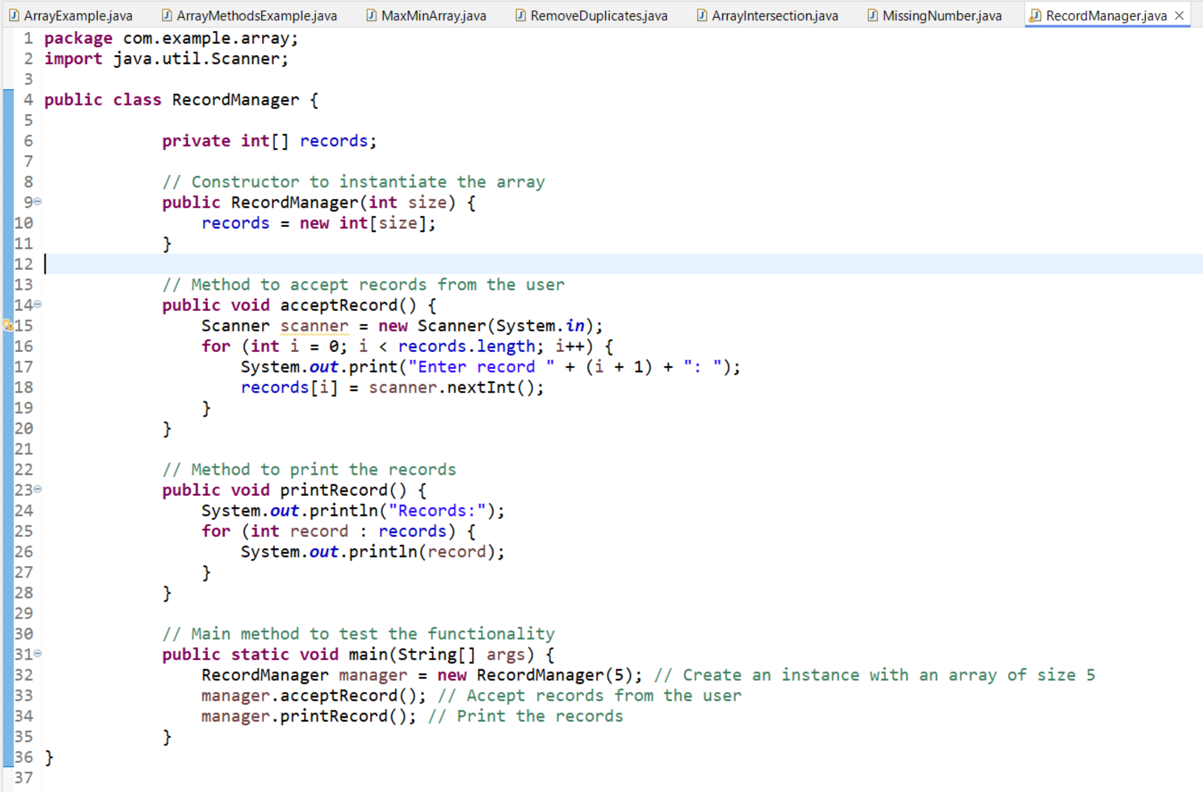
```
<terminated> MissingNumber [Java Application] C:\Eclipse\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.3.v20240426-1530\jre\bin\jav
The missing number is: 3
```

7. 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.

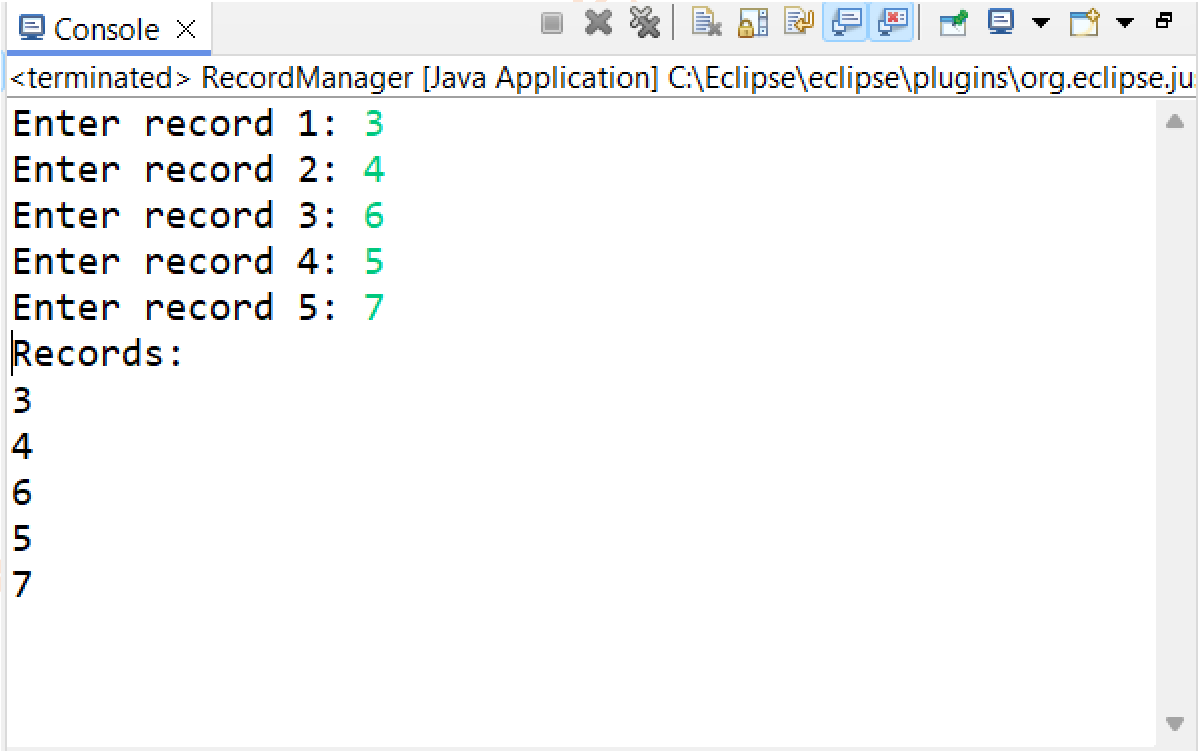
Ans:

```
Package com.example.array;
import java.util.Scanner;

public class RecordManager {
   private int[] records;

   // Constructor to instantiate the array
   public RecordManager(int size) {
      records = new int[size];
   }

   // Method to accept records from the user
   public void acceptRecord() {
      Scanner scanner = new Scanner(System.in);
      for (int i = 0; i < records.length; i++) {
         System.out.print("Enter record " + (i + 1) + ": ");
         records[i] = scanner.nextInt();
      }
   }

   // Method to print the records
   public void printRecord() {
      System.out.println("Records:");
      for (int record : records) {
         System.out.println(record);
      }
   }

   // Main method to test the functionality
   public static void main(String[] args) {
      RecordManager manager = new RecordManager(5); // Create an instance with an
array of size 5
      manager.acceptRecord(); // Accept records from the user
      manager.printRecord(); // Print the records
   }
```

}



```java
package com.example.array;
import java.util.Scanner;

public class RecordManager {

            private int[] records;

            // Constructor to instantiate the array
            public RecordManager(int size) {
                records = new int[size];
            }

            // Method to accept records from the user
            public void acceptRecord() {
                Scanner scanner = new Scanner(System.in);
                for (int i = 0; i < records.length; i++) {
                    System.out.print("Enter record " + (i + 1) + ": ");
                    records[i] = scanner.nextInt();
                }
            }

            // Method to print the records
            public void printRecord() {
                System.out.println("Records:");
                for (int record : records) {
                    System.out.println(record);
                }
            }

            // Main method to test the functionality
            public static void main(String[] args) {
                RecordManager manager = new RecordManager(5); // Create an instance with an array of size 5
                manager.acceptRecord(); // Accept records from the user
                manager.printRecord(); // Print the records
            }
}
```



```
<terminated> RecordManager [Java Application] C:\Eclipse\eclipse\plugins\org.eclipse.ju
Enter record 1: 3
Enter record 2: 4
Enter record 3: 6
Enter record 4: 5
Enter record 5: 7
Records:
3
4
6
5
7
```

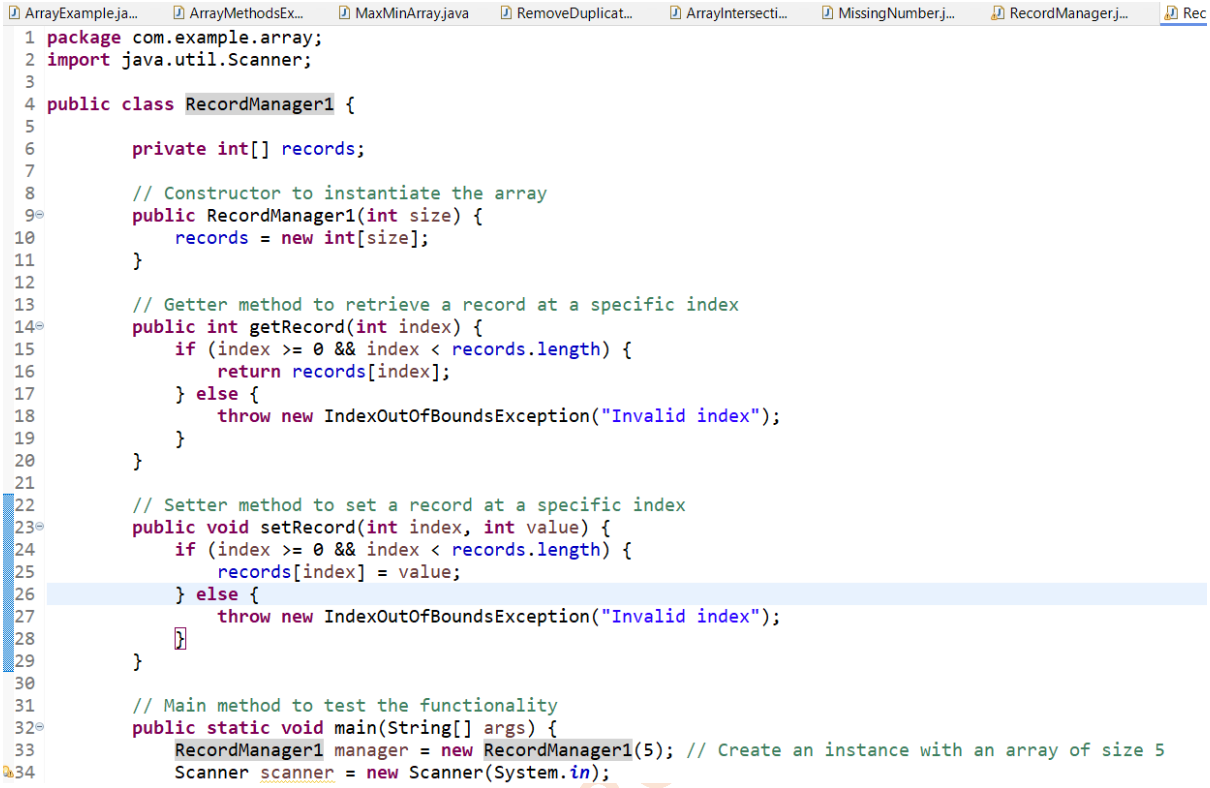
8. Modify the previous assignment to use getter and setter methods instead of `acceptRecord` and `printRecord`.

Ans:

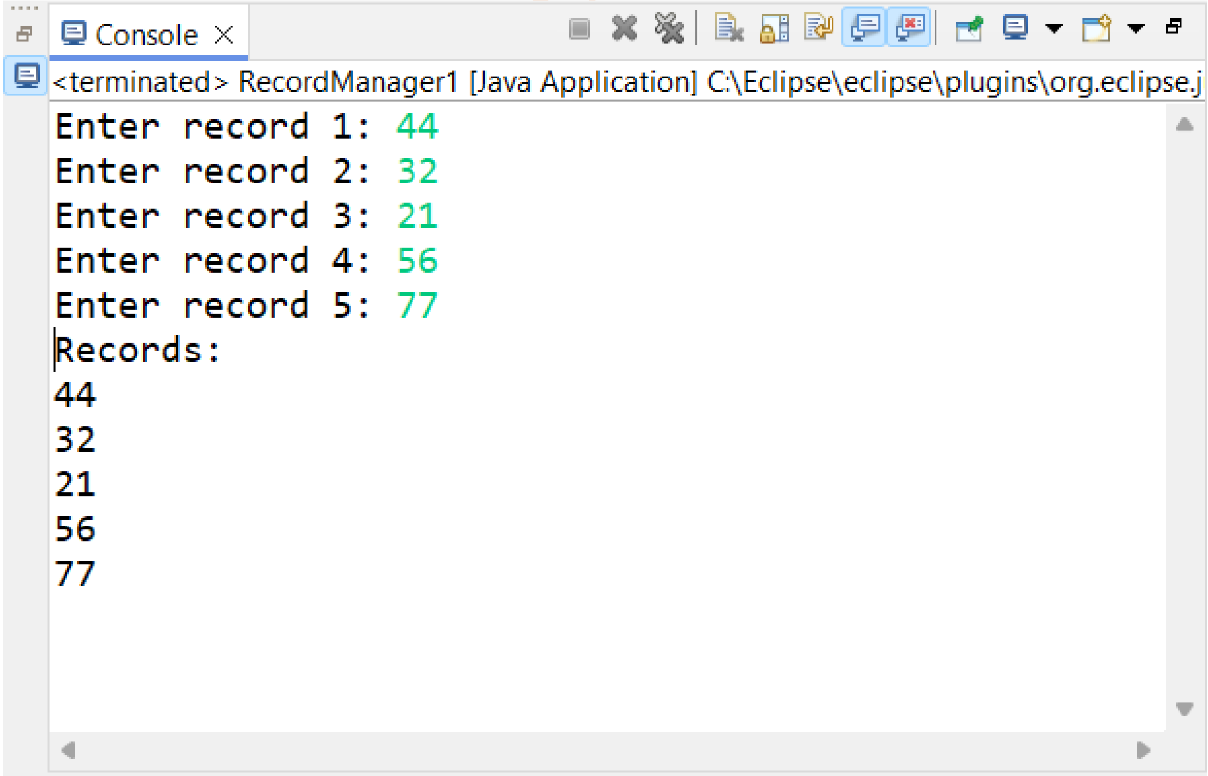
package com.example.array;

ASSIGNMENT NO.7

```
import java.util.Scanner;

public class RecordManager1 {

        private int[] records;

        // Constructor to instantiate the array
        public RecordManager1(int size) {
           records = new int[size];
        }

        // Getter method to retrieve a record at a specific index
        public int getRecord(int index) {
           if (index >= 0 && index < records.length) {
              return records[index];
           } else {
              throw new IndexOutOfBoundsException("Invalid index");
           }
        }

        // Setter method to set a record at a specific index
        public void setRecord(int index, int value) {
           if (index >= 0 && index < records.length) {
              records[index] = value;
           } else {
              throw new IndexOutOfBoundsException("Invalid index");
           }
        }

        // Main method to test the functionality
        public static void main(String[] args) {
           RecordManager manager = new RecordManager(5); // Create an instance with
an array of size 5
           Scanner scanner = new Scanner(System.in);

           // Accept records from the user using setter method
           for (int i = 0; i < 5; i++) {
              System.out.print("Enter record " + (i + 1) + ": ");
              int value = scanner.nextInt();
              manager.setRecord(i, value);
           }

           // Print the records using getter method
           System.out.println("Records:");
           for (int i = 0; i < 5; i++) {
              System.out.println(manager.getRecord(i));
           }
        }

}
```



```java
package com.example.array;
import java.util.Scanner;

public class RecordManager1 {

        private int[] records;

        // Constructor to instantiate the array
        public RecordManager1(int size) {
            records = new int[size];
        }

        // Getter method to retrieve a record at a specific index
        public int getRecord(int index) {
            if (index >= 0 && index < records.length) {
                return records[index];
            } else {
                throw new IndexOutOfBoundsException("Invalid index");
            }
        }

        // Setter method to set a record at a specific index
        public void setRecord(int index, int value) {
            if (index >= 0 && index < records.length) {
                records[index] = value;
            } else {
                throw new IndexOutOfBoundsException("Invalid index");
            }
        }

        // Main method to test the functionality
        public static void main(String[] args) {
            RecordManager1 manager = new RecordManager1(5); // Create an instance with an array of size 5
            Scanner scanner = new Scanner(System.in);
```



```
<terminated> RecordManager1 [Java Application] C:\Eclipse\eclipse\plugins\org.eclipse.j
Enter record 1: 44
Enter record 2: 32
Enter record 3: 21
Enter record 4: 56
Enter record 5: 77
Records:
44
32
21
56
77
```

9. 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.

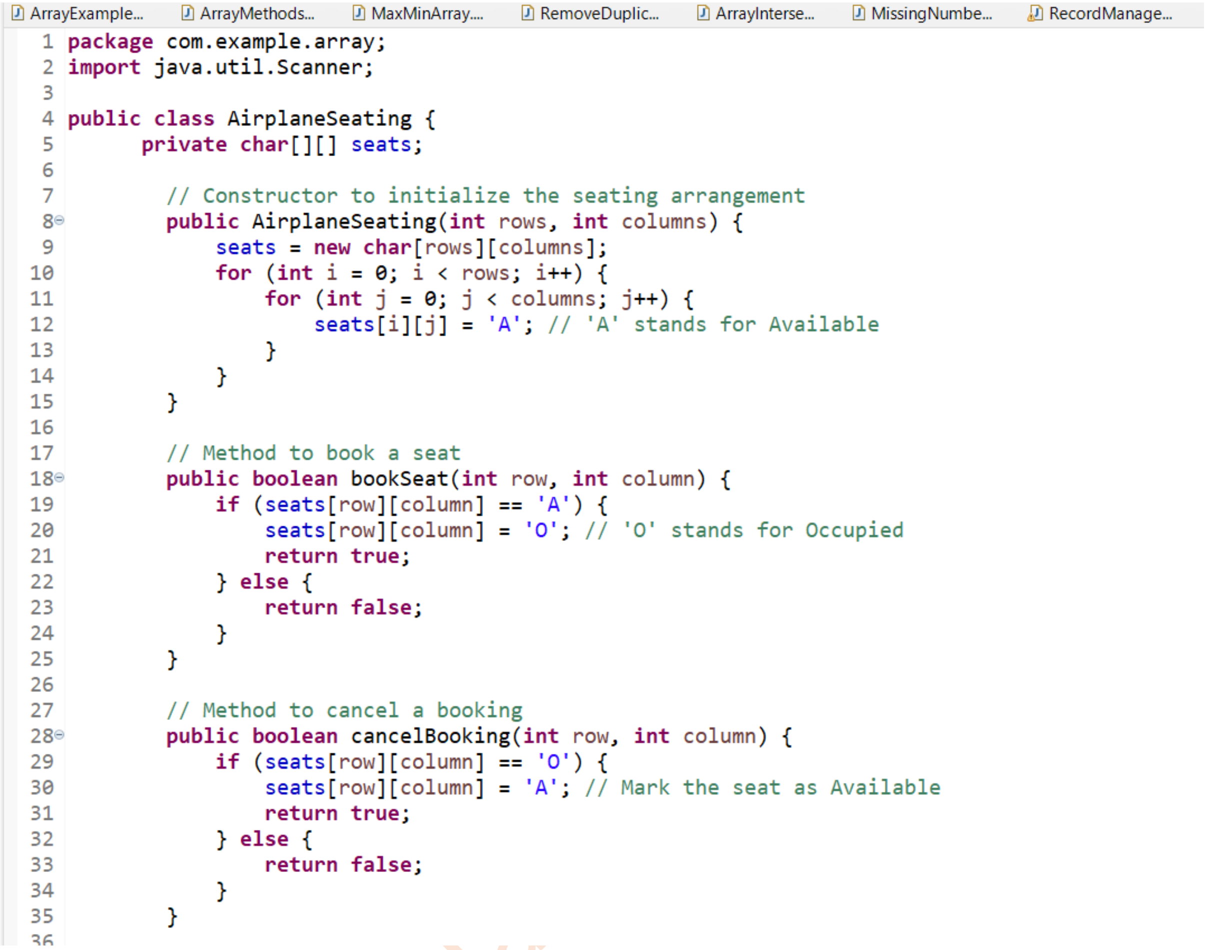
Ans:

```
import java.util.Scanner;

public class AirplaneSeating {

private char[][] seats;

    // Constructor to initialize the seating arrangement
    public AirplaneSeating(int rows, int columns) {
        seats = new char[rows][columns];
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < columns; j++) {
                seats[i][j] = 'A'; // 'A' stands for Available
            }
        }
    }

    // Method to book a seat
    public boolean bookSeat(int row, int column) {
        if (seats[row][column] == 'A') {
            seats[row][column] = 'O'; // 'O' stands for Occupied
            return true;
        } else {
            return false;
        }
    }

    // Method to cancel a booking
    public boolean cancelBooking(int row, int column) {
        if (seats[row][column] == 'O') {
            seats[row][column] = 'A'; // Mark the seat as Available
            return true;
        } else {
            return false;
        }
    }
```

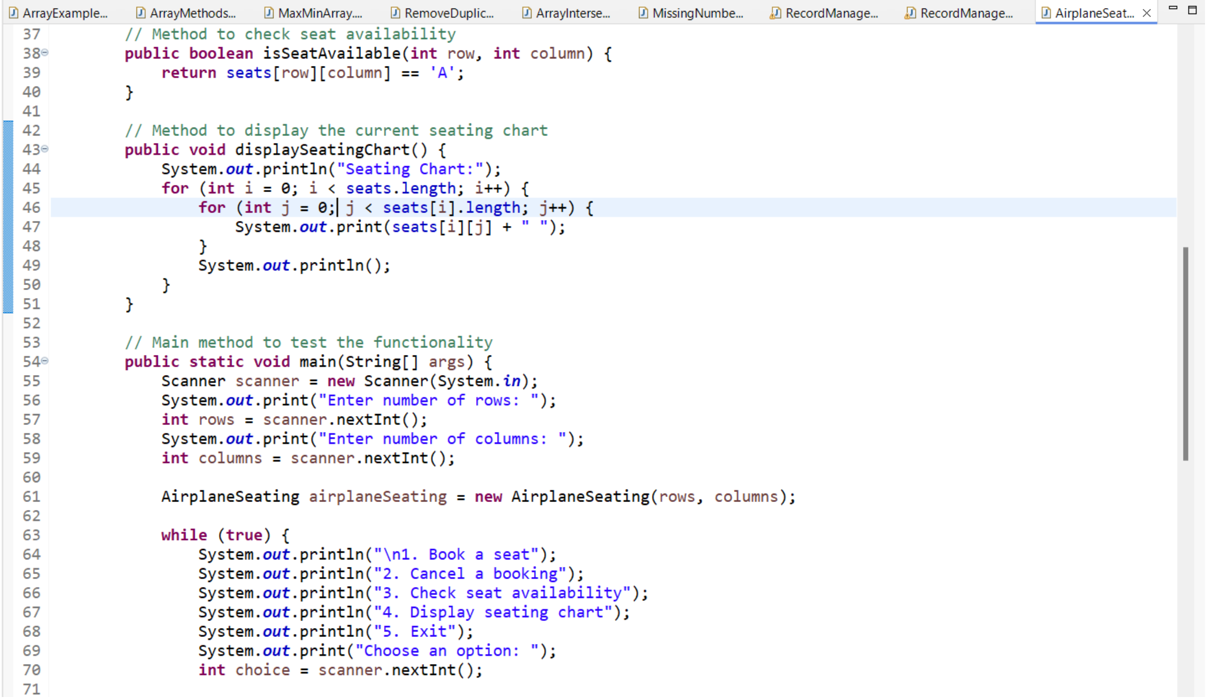
```
// Method to check seat availability
public boolean isSeatAvailable(int row, int column) {
    return seats[row][column] == 'A';
}

// Method to display the current 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();
    }
}

// Main method to test the functionality
public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter number of rows: ");
    int rows = scanner.nextInt();
    System.out.print("Enter number of columns: ");
    int columns = scanner.nextInt();

    AirplaneSeating airplaneSeating = new AirplaneSeating(rows, columns);

    while (true) {
        System.out.println("\n1. Book a seat");
        System.out.println("2. Cancel a booking");
        System.out.println("3. Check seat availability");
        System.out.println("4. Display seating chart");
        System.out.println("5. Exit");
        System.out.print("Choose an option: ");
        int choice = scanner.nextInt();

        switch (choice) {
            case 1:
                System.out.print("Enter row to book: ");
                int bookRow = scanner.nextInt();
                System.out.print("Enter column to book: ");
                int bookColumn = scanner.nextInt();
                if (airplaneSeating.bookSeat(bookRow, bookColumn)) {
                    System.out.println("Seat booked successfully.");
                } else {
                    System.out.println("Seat is already occupied.");
                }
```

```
          break;
        case 2:
          System.out.print("Enter row to cancel: ");
          int cancelRow = scanner.nextInt();
          System.out.print("Enter column to cancel: ");
          int cancelColumn = scanner.nextInt();
          if (airplaneSeating.cancelBooking(cancelRow, cancelColumn)) {
            System.out.println("Booking cancelled successfully.");
          } else {
            System.out.println("Seat is not occupied.");
          }
          break;
        case 3:
          System.out.print("Enter row to check: ");
          int checkRow = scanner.nextInt();
          System.out.print("Enter column to check: ");
          int checkColumn = scanner.nextInt();
          if (airplaneSeating.isSeatAvailable(checkRow, checkColumn)) {
            System.out.println("Seat is available.");
          } else {
            System.out.println("Seat is occupied.");
          }
          break;
        case 4:
          airplaneSeating.displaySeatingChart();
          break;
        case 5:
          System.out.println("Exiting...");
          scanner.close();
          return;
        default:
          System.out.println("Invalid option. Please try again.");
      }
    }
```

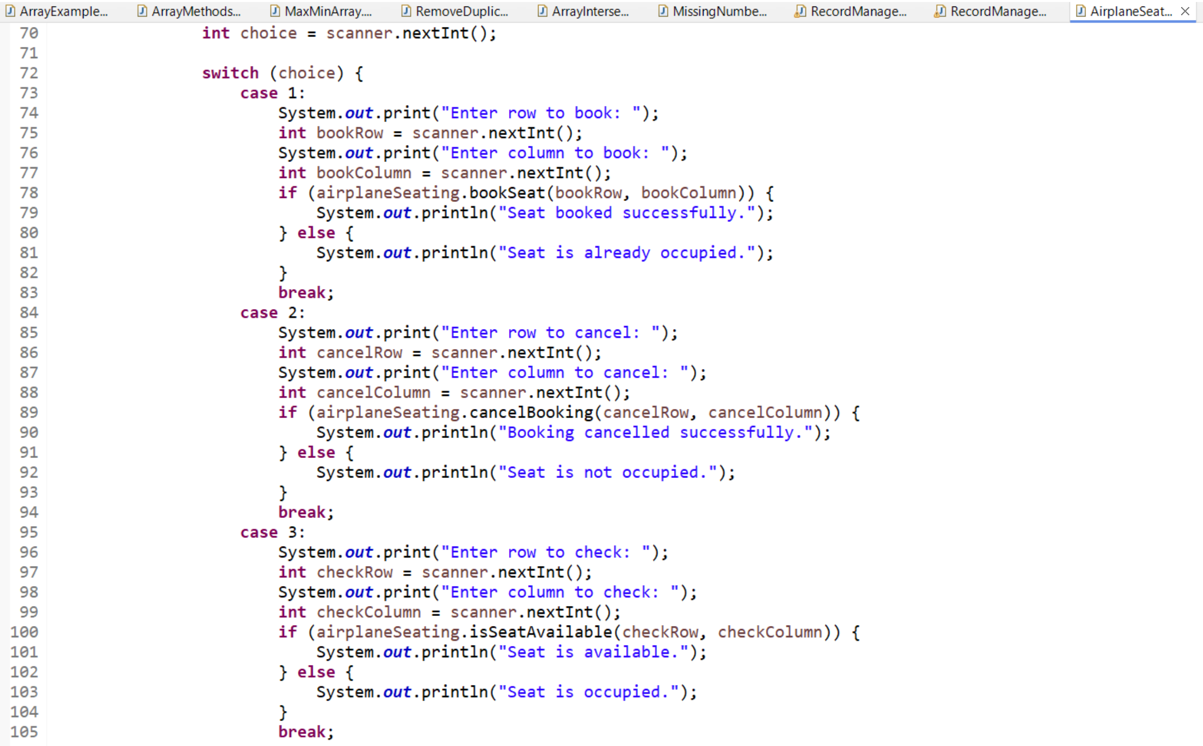
```
  }
}
```



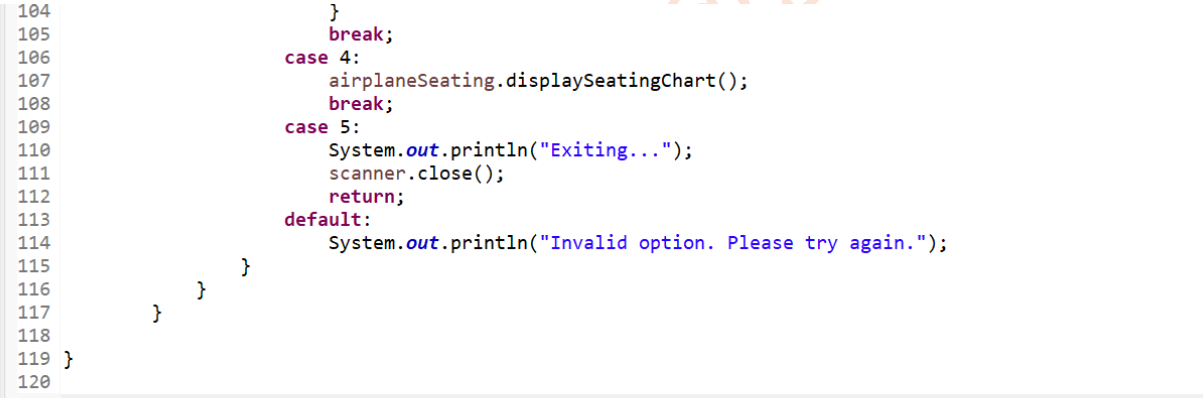
```
package com.example.array;
import java.util.Scanner;

public class AirplaneSeating {
      private char[][] seats;

        // Constructor to initialize the seating arrangement
        public AirplaneSeating(int rows, int columns) {
            seats = new char[rows][columns];
            for (int i = 0; i < rows; i++) {
                for (int j = 0; j < columns; j++) {
                    seats[i][j] = 'A'; // 'A' stands for Available
                }
            }
        }

        // Method to book a seat
        public boolean bookSeat(int row, int column) {
            if (seats[row][column] == 'A') {
                seats[row][column] = 'O'; // 'O' stands for Occupied
                return true;
            } else {
                return false;
            }
        }

        // Method to cancel a booking
        public boolean cancelBooking(int row, int column) {
            if (seats[row][column] == 'O') {
                seats[row][column] = 'A'; // Mark the seat as Available
                return true;
            } else {
                return false;
            }
        }
```



```
        // Method to check seat availability
        public boolean isSeatAvailable(int row, int column) {
            return seats[row][column] == 'A';
        }

        // Method to display the current 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();
            }
        }

        // Main method to test the functionality
        public static void main(String[] args) {
            Scanner scanner = new Scanner(System.in);
            System.out.print("Enter number of rows: ");
            int rows = scanner.nextInt();
            System.out.print("Enter number of columns: ");
            int columns = scanner.nextInt();

            AirplaneSeating airplaneSeating = new AirplaneSeating(rows, columns);

            while (true) {
                System.out.println("\n1. Book a seat");
                System.out.println("2. Cancel a booking");
                System.out.println("3. Check seat availability");
                System.out.println("4. Display seating chart");
                System.out.println("5. Exit");
                System.out.print("Choose an option: ");
                int choice = scanner.nextInt();

```

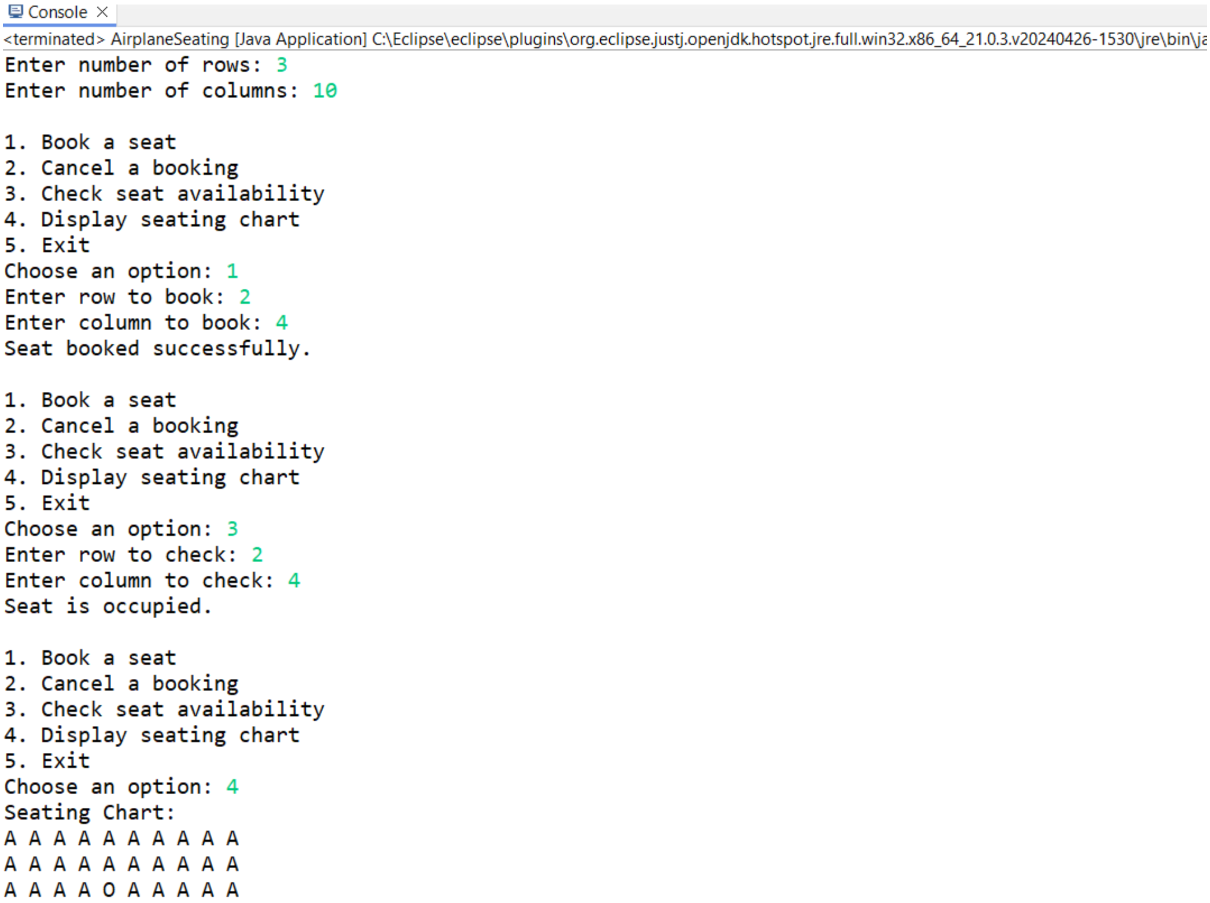


```java
                int choice = scanner.nextInt();

                switch (choice) {
                    case 1:
                        System.out.print("Enter row to book: ");
                        int bookRow = scanner.nextInt();
                        System.out.print("Enter column to book: ");
                        int bookColumn = scanner.nextInt();
                        if (airplaneSeating.bookSeat(bookRow, bookColumn)) {
                            System.out.println("Seat booked successfully.");
                        } else {
                            System.out.println("Seat is already occupied.");
                        }
                        break;
                    case 2:
                        System.out.print("Enter row to cancel: ");
                        int cancelRow = scanner.nextInt();
                        System.out.print("Enter column to cancel: ");
                        int cancelColumn = scanner.nextInt();
                        if (airplaneSeating.cancelBooking(cancelRow, cancelColumn)) {
                            System.out.println("Booking cancelled successfully.");
                        } else {
                            System.out.println("Seat is not occupied.");
                        }
                        break;
                    case 3:
                        System.out.print("Enter row to check: ");
                        int checkRow = scanner.nextInt();
                        System.out.print("Enter column to check: ");
                        int checkColumn = scanner.nextInt();
                        if (airplaneSeating.isSeatAvailable(checkRow, checkColumn)) {
                            System.out.println("Seat is available.");
                        } else {
                            System.out.println("Seat is occupied.");
                        }
                        break;
```



```java
                        }
                        break;
                    case 4:
                        airplaneSeating.displaySeatingChart();
                        break;
                    case 5:
                        System.out.println("Exiting...");
                        scanner.close();
                        return;
                    default:
                        System.out.println("Invalid option. Please try again.");
                }
            }
        }

}

```

ASSIGNMENT NO.7



```
<terminated> AirplaneSeating [Java Application] C:\Eclipse\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.3.v20240426-1530\jre\bin\ja
Enter number of rows: 3
Enter number of columns: 10

1. Book a seat
2. Cancel a booking
3. Check seat availability
4. Display seating chart
5. Exit
Choose an option: 1
Enter row to book: 2
Enter column to book: 4
Seat booked successfully.

1. Book a seat
2. Cancel a booking
3. Check seat availability
4. Display seating chart
5. Exit
Choose an option: 3
Enter row to check: 2
Enter column to check: 4
Seat is occupied.

1. Book a seat
2. Cancel a booking
3. Check seat availability
4. Display seating chart
5. Exit
Choose an option: 4
Seating Chart:
A A A A A A A A A A
A A A A A A A A A A
A A A A O A A A A A
```