# 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.**

```
import java.util.Scanner;


public class ArrayExample1 {

    public static void main(String[] args) {

        int[] array = new int[5];  // Declare an array of 5 integers


        // Print default values (which are 0 for integers)

        System.out.println("Default values in the array:");

        for (int value : array) {

            System.out.println(value);

        }


        // Accept records from user

        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter 5 integers:");

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

            array[i] = scanner.nextInt();

        }


        // Print updated values

        System.out.println("Updated values in the array:");

        for (int value : array) {
            System.out.println(value);

        }



        scanner.close();

    }

}
```

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.

```
import java.util.Scanner;



public class ArrayExample2 {

    public static void main(String[] args) {

        int[] array = new int[5];  // Declare an array of 5 integers



        // Accept records from the user

        acceptRecord(array);



        // Print the state of the array

        printRecord(array);

    }



    // 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:");
        for (int i = 0; i < array.length; i++) {
            array[i] = scanner.nextInt();
        }
    }

    // Method to print the state of the array
    public static void printRecord(int[] array) {
        System.out.println("Array values:");
        for (int value : array) {
            System.out.println(value);
        }
    }
}
```

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

```
import java.util.Scanner;

public class ArrayExample3 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int[] array = new int[5];

        System.out.println("Enter 5 integers:");
        for (int i = 0; i < array.length; i++) {
            array[i] = scanner.nextInt();
        }


        int max = findMax(array);
        int min = findMin(array);


        System.out.println("Maximum value: " + max);
        System.out.println("Minimum value: " + min);


        scanner.close();
    }


    public static int findMax(int[] array) {
        int max = array[0];
        for (int value : array) {
            if (value > max) {
                max = value;
            }
        }
        return max;
    }


    public static int findMin(int[] array) {
        int min = array[0];
        for (int value : array) {
          if (value < min) {

              min = value;

          }

        }

        return min;

    }

}
```

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

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

public class ArrayExample4 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int[] array = new int[5];

        System.out.println("Enter 5 integers:");
        for (int i = 0; i < array.length; i++) {
            array[i] = scanner.nextInt();
        }

        int[] uniqueArray = removeDuplicates(array);
        System.out.println("Array after removing duplicates: " +
Arrays.toString(uniqueArray));

        scanner.close();
    }

    public static int[] removeDuplicates(int[] array) {
        return Arrays.stream(array).distinct().toArray();
    }
}
```

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

```
import java.util.Arrays;

import java.util.HashSet;

import java.util.Scanner;
import java.util.Set;


public class ArrayExample5 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int[] array1 = new int[5];
        int[] array2 = new int[5];


        System.out.println("Enter 5 integers for the first array:");
        for (int i = 0; i < array1.length; i++) {
            array1[i] = scanner.nextInt();
        }


        System.out.println("Enter 5 integers for the second array:");
        for (int i = 0; i < array2.length; i++) {
            array2[i] = scanner.nextInt();
        }


        int[] intersection = findIntersection(array1, array2);
        System.out.println("Intersection of the two arrays: " + Arrays.toString(intersection));


        scanner.close();
    }


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

        for (int value : array1) {

            set1.add(value);

        }



        Set<Integer> intersection = new HashSet<>();

        for (int value : array2) {

            if (set1.contains(value)) {

                intersection.add(value);

            }

        }



        return intersection.stream().mapToInt(Integer::intValue).toArray();

    }

}
```

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

```
import java.util.Scanner;



public class ArrayExample6 {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        int n = 5; // Number of elements expected (1 to N)

        int[] array = new int[n];
        System.out.println("Enter " + (n - 1) + " integers (1 to " + n + ") with one missing:");
        for (int i = 0; i < n - 1; i++) {
            array[i] = scanner.nextInt();
        }

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

        scanner.close();
    }

    public static int findMissingNumber(int[] array, int n) {
        int totalSum = n * (n + 1) / 2;
        int arraySum = 0;
        for (int value : array) {
            arraySum += value;
        }
        return totalSum - arraySum;
    }
}
```

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.**

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

        // Constructor to initialize the array
        public ArrayHandler(int size) {
            array = new int[size];
        }

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

        // Method to print the state of the array
        public void printRecord() {
            System.out.println("Array values:");
            for (int value : array) {
                System.out.println(value);
            }
        }
    }

    public class ArrayExample7 {
        public static void main(String[] args) {
            ArrayHandler handler = new ArrayHandler(5);
            handler.acceptRecord();
            handler.printRecord();
        }
    }
```

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

```
import java.util.Scanner;


class ArrayHandler {

    private int[] array;



    // Constructor to initialize the array

    public ArrayHandler(int size) {
        array = new int[size];
    }

    // Getter for the array
    public int[] getArray() {
        return array;
    }

    // Setter for the array
    public void setArray(int[] array) {
        this.array = array;
    }

    // Method to print the state of the array
    public void printRecord() {
        System.out.println("Array values:");
        for (int value : array) {
            System.out.println(value);
        }
    }
}

public class ArrayExample8 {
    public static void main(String[] args) {
        ArrayHandler handler = new ArrayHandler(5);
        Scanner scanner = new Scanner(System.in);

        int[] inputArray = new int[5];

        System.out.println("Enter 5 integers:");

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

            inputArray[i] = scanner.nextInt();

        }


        handler.setArray(inputArray);

        handler.printRecord();


        scanner.close();

    }

}
```

**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.

```
import java.util.Scanner;



class AirplaneSeats {

    private boolean[][] seats;
```

```
// Constructor to initialize seating arrangement
public AirplaneSeats(int rows, int columns) {
    seats = new boolean[rows][columns];
    // All seats are initially available (false)
}

// Method to book a seat
public void bookSeat(int row, int column) {
    if (isValidSeat(row, column) && !seats[row][column]) {
        seats[row][column] = true;
        System.out.println("Seat booked successfully.");
    } else {
        System.out.println("Seat is already booked or invalid.");
    }
}

// Method to cancel a booking
public void cancelBooking(int row, int column) {
    if (isValidSeat(row, column) && seats[row][column]) {
        seats[row][column] = false;
        System.out.println("Booking canceled.");
    } else {
        System.out.println("Seat is already available or invalid.");
    }
    }

    // Method to check seat availability
    public boolean isSeatAvailable(int row, int column) {
        return isValidSeat(row, column) && !seats[row][column];
    }

    // 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] ? "X " : "O ");
            }
            System.out.println();
        }
    }

    // Helper method to check if a seat is valid
    private boolean isValidSeat(int row, int column) {
        return row >= 0 && row < seats.length && column >= 0 && column <
seats[row].length;
    }
}
```

```
public class AirplaneSeatingSystem {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        AirplaneSeats airplane = new AirplaneSeats(5, 5); // 5 rows and 5 columns

        while (true) {
            System.out.println("\nMenu:");
            System.out.println("1. 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("Enter your choice: ");
            int choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    System.out.print("Enter row and column to book (0-based index): ");
                    int bookRow = scanner.nextInt();
                    int bookColumn = scanner.nextInt();
                    airplane.bookSeat(bookRow, bookColumn);
                    break;
                case 2:
                    System.out.print("Enter row and column to cancel (0-based index): ");
                    int cancelRow = scanner.nextInt();
                    int cancelColumn = scanner.nextInt();

                    airplane.cancelBooking(cancelRow, cancelColumn);

                    break;

                case 3:

                    System.out.print("Enter row and column to check (0-based index): ");

                    int checkRow = scanner.nextInt();

                    int checkColumn = scanner.nextInt();

                    boolean available = airplane.isSeatAvailable(checkRow, checkColumn);

                    System.out.println("Seat availability: " + (available ? "Available" : "Not
Available"));

                    break;

                case 4:

                    airplane.displaySeatingChart();

                    break;

                case 5:

                    System.out.println("Exiting...");

                    scanner.close();

                    System.exit(0);

                    break;

                default:

                    System.out.println("Invalid choice. Try again.");

            }

        }

    }

}
```

ASSIGNMENT NO.7