

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();
}
```

}

ArrayExample.java ×

```
1 package com.example.array;
2 import java.util.Scanner;
3
4 public class ArrayExample {
5     public static void main(String[] args) {
6         // Declare a single-dimensional array of 5 integers
7         int[] numbers = new int[5];
8
9         // Print the default values of the array
10        System.out.println("Default values of the array:");
11        for (int i = 0; i < numbers.length; i++) {
12            System.out.println("Element at index " + i + ": " + numbers[i]);
13        }
14
15        // Create a Scanner object to accept user input
16        Scanner scanner = new Scanner(System.in);
17
18        // Accept records from the user
19        System.out.println("Enter 5 integers to update the array:");
20        for (int i = 0; i < numbers.length; i++) {
21            System.out.print("Enter value for index " + i + ": ");
22            numbers[i] = scanner.nextInt();
23        }
24
25        // Print the updated values of the array
26        System.out.println("Updated values of the array:");
27        for (int i = 0; i < numbers.length; i++) {
28            System.out.println("Element at index " + i + ": " + numbers[i]);
29        }
30
31        // Close the scanner
32        scanner.close();
33    }
34 }
35
```

Console ×

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.

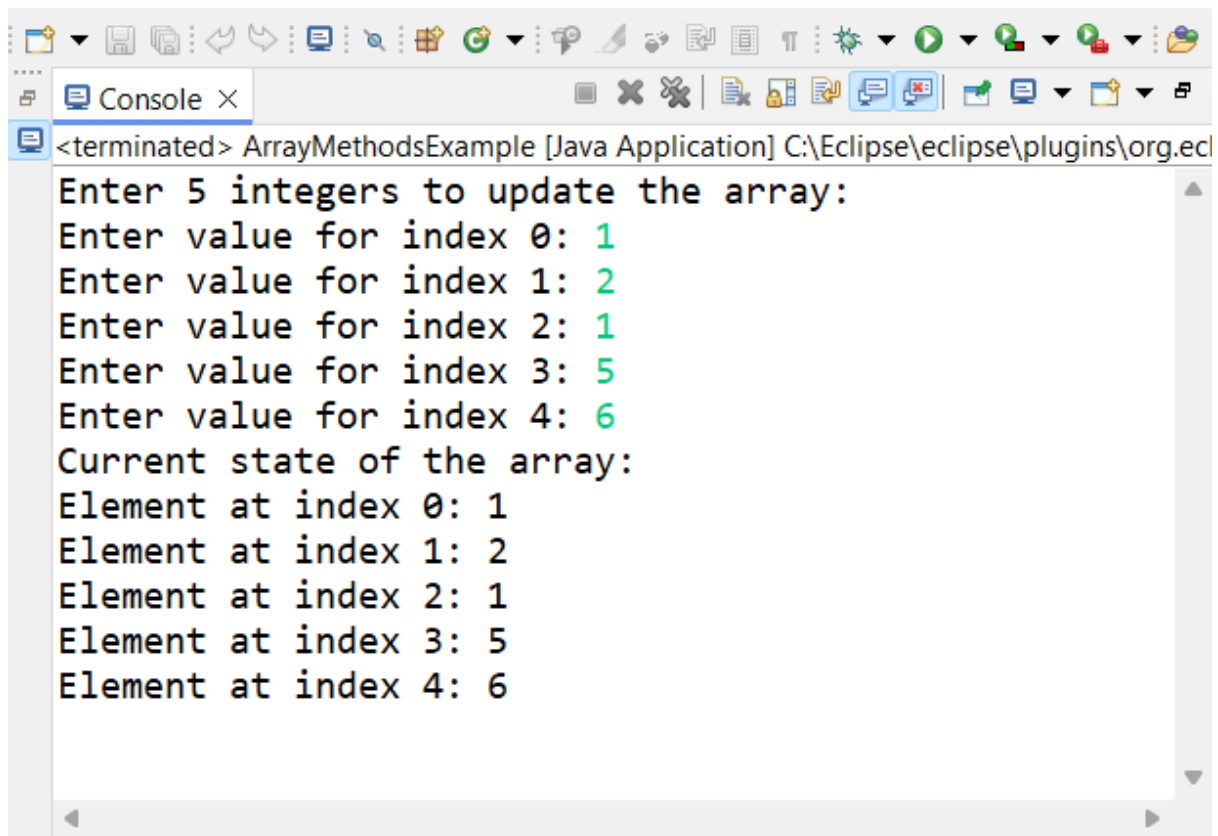
Ans:

```
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]);  
        }  
    }  
}
```

}

```
ArrayExample.java  ArrayMethodsExample.java ×
1 package com.example.array;
2 import java.util.Scanner;
3 public class ArrayMethodsExample {
4     public static void main(String[] args) {
5         // Declare a single-dimensional array of 5 integers
6         int[] numbers = new int[5];
7
8         // Call the method to accept records from the user
9         acceptRecord(numbers);
10
11        // Call the method to print the state of the array
12        printRecord(numbers);
13    }
14
15    // Method to accept records from the user
16    public static void acceptRecord(int[] array) {
17        Scanner scanner = new Scanner(System.in);
18        System.out.println("Enter 5 integers to update the array:");
19        for (int i = 0; i < array.length; i++) {
20            System.out.print("Enter value for index " + i + ": ");
21            array[i] = scanner.nextInt();
22        }
23        scanner.close();
24    }
25
26    // Method to print the state of the array
27    public static void printRecord(int[] array) {
28        System.out.println("Current state of the array:");
29        for (int i = 0; i < array.length; i++) {
30            System.out.println("Element at index " + i + ": " + array[i]);
31        }
32    }
33 }
34
```



```
<terminated> ArrayMethodsExample [Java Application] C:\Eclipse\eclipse\plugins\org.ec
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

```

}
ArrayExample.java  ArrayMethodsExample.java  MaxMinArray.java ×
1  package com.example.array;
2
3  import java.util.Scanner;
4
5  public class MaxMinArray {
6
7      public static void main(String[] args) {
8          Scanner scanner = new Scanner(System.in);
9
10         // Input the size of the array
11         System.out.println("Enter the number of elements in the array: ");
12         int n = scanner.nextInt();
13
14         // Initialize the array
15         int[] array = new int[n];
16
17         // Input the elements of the array
18         System.out.println("Enter the elements of the array:");
19         for (int i = 0; i < n; i++) {
20             array[i] = scanner.nextInt();
21         }
22
23         // Initialize max and min with the first element of the array
24         int max = array[0];
25         int min = array[0];
26
27         // Traverse the array to find the max and min values
28         for (int i = 1; i < n; i++) {
29             if (array[i] > max) {
30                 max = array[i];
31             }
32             if (array[i] < min) {
33                 min = array[i];
34             }
35         }
36
37         // Output the results

```

Console ×

<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] + " ");  
        }  
    }  
}
```



```

}
ArrayExample.java  ArrayMethodsExample.java  MaxMinArray.java  RemoveDuplicates.java ×
4 public class RemoveDuplicates {
5     public static int removeDuplicates(int[] array) {
6         if (array.length == 0) {
7             return 0;
8         }
9
10        // Sort the array to bring duplicates together
11        Arrays.sort(array);
12
13        // Index of the last unique element
14        int j = 0;
15
16        // Traverse the array
17        for (int i = 1; i < array.length; i++) {
18            // If current element is different from the last unique element
19            if (array[i] != array[j]) {
20                j++;
21                array[j] = array[i];
22            }
23        }
24
25        // Return the count of unique elements
26        return j + 1;
27    }
28
29    public static void main(String[] args) {
30        int[] array = {1, 2, 2, 3, 4, 4, 5};
31        int n = removeDuplicates(array);
32
33        // Print the unique elements
34        for (int i = 0; i < n; i++) {
35            System.out.print(array[i] + " ");
36        }
37    }
38
39 }

```

Console ×

<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:

```
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;
}
```

ASSIGNMENT NO.7

```

}
ArrayExample.java  ArrayMethodsExample.java  MaxMinArray.java  RemoveDuplicates.java  ArrayIntersection.java x
1 package com.example.array;
2 import java.util.HashSet;
3 import java.util.Set;
4
5 public class ArrayIntersection {
6     public static void main(String[] args) {
7         int[] array1 = {1, 2, 3, 4, 5};
8         int[] array2 = {3, 4, 5, 6, 7};
9
10        // Find intersection
11        Set<Integer> intersection = findIntersection(array1, array2);
12
13        // Print the intersection
14        System.out.println("Intersection of the two arrays: " + intersection);
15    }
16
17    public static Set<Integer> findIntersection(int[] array1, int[] array2) {
18        Set<Integer> set1 = new HashSet<>();
19        Set<Integer> intersection = new HashSet<>();
20
21        // Add elements of the first array to the set
22        for (int num : array1) {
23            set1.add(num);
24        }
25
26        // Check elements of the second array against the set
27        for (int num : array2) {
28            if (set1.contains(num)) {
29                intersection.add(num);
30            }
31        }
32
33        return intersection;
34    }
35 }
36
37

```

n

Console x

<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:

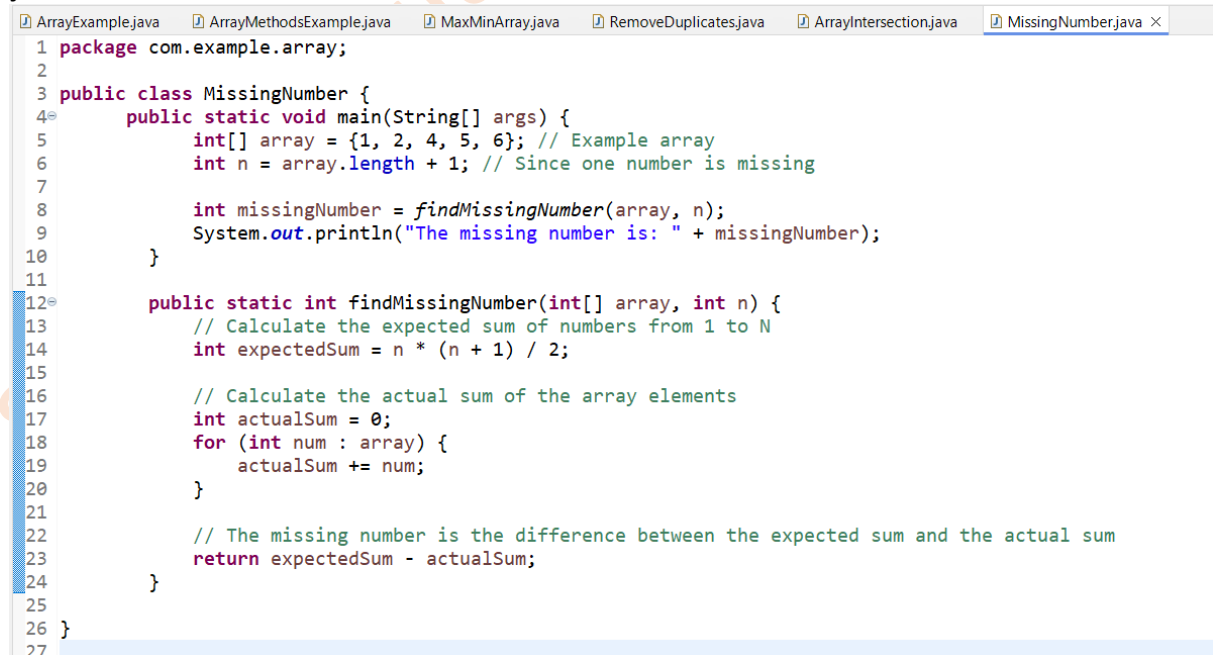
```
public class MissingNumber {
    public static void main(String[] args) {
        int[] array = {1, 2, 4, 5, 6}; // Example array
        int n = array.length + 1; // Since one number is missing

        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;
    }
}
```



```
1 package com.example.array;
2
3 public class MissingNumber {
4     public static void main(String[] args) {
5         int[] array = {1, 2, 4, 5, 6}; // Example array
6         int n = array.length + 1; // Since one number is missing
7
8         int missingNumber = findMissingNumber(array, n);
9         System.out.println("The missing number is: " + missingNumber);
10    }
11
12    public static int findMissingNumber(int[] array, int n) {
13        // Calculate the expected sum of numbers from 1 to N
14        int expectedSum = n * (n + 1) / 2;
15
16        // Calculate the actual sum of the array elements
17        int actualSum = 0;
18        for (int num : array) {
19            actualSum += num;
20        }
21
22        // The missing number is the difference between the expected sum and the actual sum
23        return expectedSum - actualSum;
24    }
25 }
26
27
```

```

Console ×
<terminated> MissingNumber [Java Application] C:\Eclipse\eclipse\plugins\org.eclipse.justj.openjdkhotspot.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
    }
}

```

ASSIGNMENT NO.7

```
}
1 package com.example.array;
2 import java.util.Scanner;
3
4 public class RecordManager {
5
6     private int[] records;
7
8     // Constructor to instantiate the array
9     public RecordManager(int size) {
10         records = new int[size];
11     }
12
13     // Method to accept records from the user
14     public void acceptRecord() {
15         Scanner scanner = new Scanner(System.in);
16         for (int i = 0; i < records.length; i++) {
17             System.out.print("Enter record " + (i + 1) + ": ");
18             records[i] = scanner.nextInt();
19         }
20     }
21
22     // Method to print the records
23     public void printRecord() {
24         System.out.println("Records:");
25         for (int record : records) {
26             System.out.println(record);
27         }
28     }
29
30     // Main method to test the functionality
31     public static void main(String[] args) {
32         RecordManager manager = new RecordManager(5); // Create an instance with an array of size 5
33         manager.acceptRecord(); // Accept records from the user
34         manager.printRecord(); // Print the records
35     }
36 }
37
```

Console ×

```
<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;
```

```

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));
        }
    }
}

```

ASSIGNMENT NO.7

```

    }
}

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);
    }
}

```

Console X

<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.");
}
}
```

ASSIGNMENT NO.7

```

    }
}

1 package com.example.array;
2 import java.util.Scanner;
3
4 public class AirplaneSeating {
5     private char[][] seats;
6
7     // Constructor to initialize the seating arrangement
8     public AirplaneSeating(int rows, int columns) {
9         seats = new char[rows][columns];
10        for (int i = 0; i < rows; i++) {
11            for (int j = 0; j < columns; j++) {
12                seats[i][j] = 'A'; // 'A' stands for Available
13            }
14        }
15    }
16
17    // Method to book a seat
18    public boolean bookSeat(int row, int column) {
19        if (seats[row][column] == 'A') {
20            seats[row][column] = 'O'; // 'O' stands for Occupied
21            return true;
22        } else {
23            return false;
24        }
25    }
26
27    // Method to cancel a booking
28    public boolean cancelBooking(int row, int column) {
29        if (seats[row][column] == 'O') {
30            seats[row][column] = 'A'; // Mark the seat as Available
31            return true;
32        } else {
33            return false;
34        }
35    }
36
37    // Method to check seat availability
38    public boolean isSeatAvailable(int row, int column) {
39        return seats[row][column] == 'A';
40    }
41
42    // Method to display the current seating chart
43    public void displaySeatingChart() {
44        System.out.println("Seating Chart:");
45        for (int i = 0; i < seats.length; i++) {
46            for (int j = 0; j < seats[i].length; j++) {
47                System.out.print(seats[i][j] + " ");
48            }
49            System.out.println();
50        }
51    }
52
53    // Main method to test the functionality
54    public static void main(String[] args) {
55        Scanner scanner = new Scanner(System.in);
56        System.out.print("Enter number of rows: ");
57        int rows = scanner.nextInt();
58        System.out.print("Enter number of columns: ");
59        int columns = scanner.nextInt();
60
61        AirplaneSeating airplaneSeating = new AirplaneSeating(rows, columns);
62
63        while (true) {
64            System.out.println("\n1. Book a seat");
65            System.out.println("2. Cancel a booking");
66            System.out.println("3. Check seat availability");
67            System.out.println("4. Display seating chart");
68            System.out.println("5. Exit");
69            System.out.print("Choose an option: ");
70            int choice = scanner.nextInt();
71

```

ASSIGNMENT NO.7

```

70         int choice = scanner.nextInt();
71
72         switch (choice) {
73             case 1:
74                 System.out.print("Enter row to book: ");
75                 int bookRow = scanner.nextInt();
76                 System.out.print("Enter column to book: ");
77                 int bookColumn = scanner.nextInt();
78                 if (airplaneSeating.bookSeat(bookRow, bookColumn)) {
79                     System.out.println("Seat booked successfully.");
80                 } else {
81                     System.out.println("Seat is already occupied.");
82                 }
83                 break;
84             case 2:
85                 System.out.print("Enter row to cancel: ");
86                 int cancelRow = scanner.nextInt();
87                 System.out.print("Enter column to cancel: ");
88                 int cancelColumn = scanner.nextInt();
89                 if (airplaneSeating.cancelBooking(cancelRow, cancelColumn)) {
90                     System.out.println("Booking cancelled successfully.");
91                 } else {
92                     System.out.println("Seat is not occupied.");
93                 }
94                 break;
95             case 3:
96                 System.out.print("Enter row to check: ");
97                 int checkRow = scanner.nextInt();
98                 System.out.print("Enter column to check: ");
99                 int checkColumn = scanner.nextInt();
100                 if (airplaneSeating.isSeatAvailable(checkRow, checkColumn)) {
101                     System.out.println("Seat is available.");
102                 } else {
103                     System.out.println("Seat is occupied.");
104                 }
105                 break;
106
107                 }
108                 break;
109             case 4:
110                 airplaneSeating.displaySeatingChart();
111                 break;
112             case 5:
113                 System.out.println("Exiting...");
114                 scanner.close();
115                 return;
116             default:
117                 System.out.println("Invalid option. Please try again.");
118         }
119     }
120 }

```

ASSIGNMENT NO.7

```
Console X
<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\j
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
```