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.

**package** main;

**import** java.util.Scanner;

**public** **class** ArrayExample {

**public** **static** **void** main(String[] args) {

**int**[] numbers = **new** **int**[5];

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]);

}

Scanner scanner = **new** Scanner(System.***in***);

System.***out***.println("\nEnter 5 integers to update the array:");

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

System.***out***.print("Enter value for element at index " + i + ": ");

numbers[i] = scanner.nextInt();

}

System.***out***.println("\nUpdated values of the array:");

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

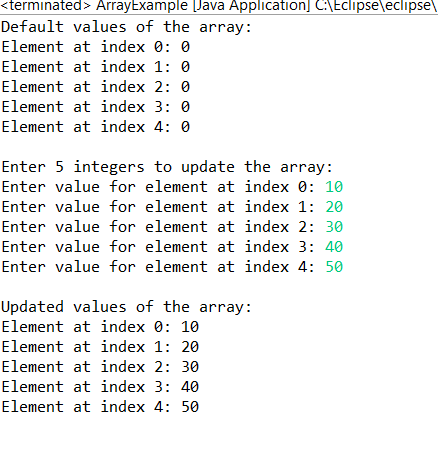
System.***out***.println("Element at index " + i + ": " + numbers[i]);

}

scanner.close();

}

}



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.

**package** main;

**import** java.util.Scanner;

**public** **class** ArrayHandler {

**public** **static** **void** acceptRecord(**int**[] array) {

Scanner scanner = **new** Scanner(System.***in***);

System.***out***.println("Enter " + array.length + " integers:");

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

System.***out***.print("Enter value for element at index " + i + ": ");

array[i] = scanner.nextInt();

}

scanner.close();

}

**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]);

}

}

**public** **static** **void** main(String[] args) {

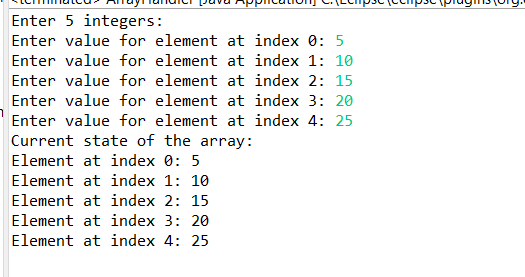
**int**[] numbers = **new** **int**[5];

*acceptRecord*(numbers);

*printRecord*(numbers);

}

}



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

**package** main;

**public** **class** MixMinFinder {

**public** **static** **void** main(String[] args) {

**int**[] arr = **new** **int**[] { 10, 45, 5, 78, 99 };

**int** min = arr[0];

**int** max = arr[0];

**for** (**int** i = 1; i < arr.length; i++) {

**if** (max < arr[i]) {

max = arr[i];

}

**if** (min > arr[i]) {

min = arr[i];

}

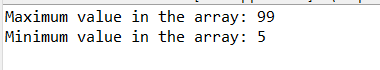
}

System.***out***.println("Maximum value in the array: " + max);

System.***out***.println("Minimum value in the array: " + min);

}

}



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

**package** main;

**public** **class** ArrayRemoveDuplicateElements {

**static** **int** removeDuplicates(**int** arr[], **int** n) {

**if** (n == 0 || n == 1)

**return** n;

**int** j = 0;

**for** (**int** i = 0; i < n - 1; i++)

**if** (arr[i] != arr[i + 1])

arr[j++] = arr[i];

arr[j++] = arr[n - 1];

**return** j;

}

**public** **static** **void** main(String[] args) {

**int** arr[] = { 10, 20, 20, 30, 40, 40, 40, 50, 50 };

**int** n = arr.length;

n = *removeDuplicates*(arr, n);

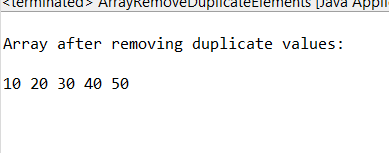
System.***out***.println("\nArray after removing duplicate values:\n");

**for** (**int** i = 0; i < n; i++)

System.***out***.print(arr[i] + " ");

}

}



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

**package** main;

**public** **class** ArrayIntersection {

**public** **static** **void** main(String[] args) {

**int** arr1[] = { 4, 3, 7, 9, 2 };

**int** arr2[] = { 5, 1, 4, 8, 3 };

**for** (**int** i = 0; i < arr1.length; i++) {

**for** (**int** j = 0; j < arr2.length; j++) {

**if** (arr1[i] == arr2[j]) {

System.***out***.println("Common element found: " + arr1[i]);

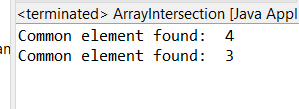
}

}

}

}

}



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

**package** main;

**public** **class** MissingNumber {

**public** **static** **int** findMissingNumber(**int**[] array, **int** n) {

**int** expectedSum = n \* (n + 1) / 2;

**int** actualSum = 0;

**for** (**int** num : array) {

actualSum += num;

}

**return** expectedSum - actualSum;

}

**public** **static** **void** main(String[] args) {

**int**[] array = {1, 2, 4, 5, 6};

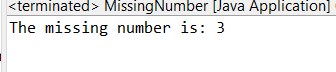
**int** n = 6;

**int** missingNumber = *findMissingNumber*(array, n);

System.***out***.println("The missing number is: " + missingNumber);

}

}



1. 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.

**package** main;

**import** java.util.Scanner;

**class** ArrayManager {

**private** **int**[] numbers;

**public** ArrayManager(**int** size) {

numbers = **new** **int**[size];

}

**public** **void** acceptRecord() {

Scanner scanner = **new** Scanner(System.***in***);

System.***out***.println("Enter " + numbers.length + " integers:");

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

System.***out***.print("Enter value for element " + (i + 1) + ": ");

numbers[i] = scanner.nextInt();

}

}

**public** **void** printRecord() {

System.***out***.println("Array elements are:");

**for** (**int** number : numbers) {

System.***out***.print(number + " ");

}

System.***out***.println();

}

**public** **static** **void** main(String[] args) {

Scanner scanner = **new** Scanner(System.***in***);

System.***out***.print("Enter the size of the array: ");

**int** size = scanner.nextInt();

ArrayManager arrayManager = **new** ArrayManager(size);

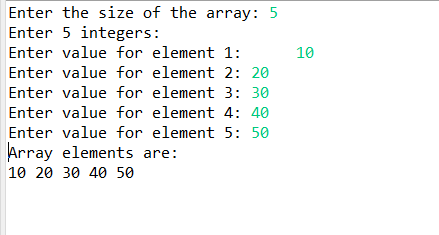
arrayManager.acceptRecord();

arrayManager.printRecord();

scanner.close();

}

}



1. 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.

**package** main;

* **import** java.util.Scanner;
* **public** **class** AirplaneSeating {
* **private** **char**[][] seats;
* **public** AirplaneSeating(**int** rows, **int** columns) {
* seats = **new** **char**[rows][columns];
* initializeSeats();
* }
* **private** **void** initializeSeats() {
* **for** (**int** i = 0; i < seats.length; i++) {
* **for** (**int** j = 0; j < seats[i].length; j++) {
* seats[i][j] = 'A';
* }
* }
* }
* **public** **boolean** bookSeat(**int** row, **int** column) {
* **if** (isValidSeat(row, column) && seats[row][column] == 'A') {
* seats[row][column] = 'O';
* System.***out***.println("Seat booked successfully.");
* **return** **true**;
* } **else** {
* System.***out***.println("Seat is not available for booking.");
* **return** **false**;
* }
* }
* **public** **boolean** cancelBooking(**int** row, **int** column) {
* **if** (isValidSeat(row, column) && seats[row][column] == 'O') {
* seats[row][column] = 'A';
* System.***out***.println("Booking cancelled successfully.");
* **return** **true**;
* } **else** {
* System.***out***.println("No booking found to cancel.");
* **return** **false**;
* }
* }
* **public** **boolean** checkAvailability(**int** row, **int** column) {
* **if** (isValidSeat(row, column)) {
* **return** seats[row][column] == 'A';
* }
* System.***out***.println("Invalid seat number.");
* **return** **false**;
* }
* **public** **void** displaySeatingChart() {
* System.***out***.println("Current 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();
* }
* }
* **private** **boolean** isValidSeat(**int** row, **int** column) {
* **return** row >= 0 && row < seats.length && column >= 0 && column < seats[0].length;
* }
* **public** **static** **void** main(String[] args) {
* Scanner scanner = **new** Scanner(System.***in***);
* System.***out***.print("Enter the number of rows: ");
* **int** rows = scanner.nextInt();
* System.***out***.print("Enter the number of columns: ");
* **int** columns = scanner.nextInt();
* AirplaneSeating airplaneSeating = **new** AirplaneSeating(rows, columns);
* airplaneSeating.displaySeatingChart();
* **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 number to book: ");
* **int** bookRow = scanner.nextInt();
* System.***out***.print("Enter column number to book: ");
* **int** bookColumn = scanner.nextInt();
* airplaneSeating.bookSeat(bookRow, bookColumn);
* **break**;
* **case** 2:
* System.***out***.print("Enter row number to cancel booking: ");
* **int** cancelRow = scanner.nextInt();
* System.***out***.print("Enter column number to cancel booking: ");
* **int** cancelColumn = scanner.nextInt();
* airplaneSeating.cancelBooking(cancelRow, cancelColumn);
* **break**;
* **case** 3:
* System.***out***.print("Enter row number to check availability: ");
* **int** checkRow = scanner.nextInt();
* System.***out***.print("Enter column number to check availability: ");
* **int** checkColumn = scanner.nextInt();
* **boolean** isAvailable = airplaneSeating.checkAvailability(checkRow, checkColumn);
* System.***out***.println("Seat availability: " + (isAvailable ? "Available" : "Occupied"));
* **break**;
* **case** 4:
* airplaneSeating.displaySeatingChart();
* **break**;
* **case** 5:
* System.***out***.println("Exiting the system.");
* scanner.close();
* **return**;
* **default**:
* System.***out***.println("Invalid choice. Please try again.");
* }
* }
* }
* }

