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.

Program:

**package** in.re.rs;

**import** java.util.Scanner;

**public** **class** program {

**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 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]);

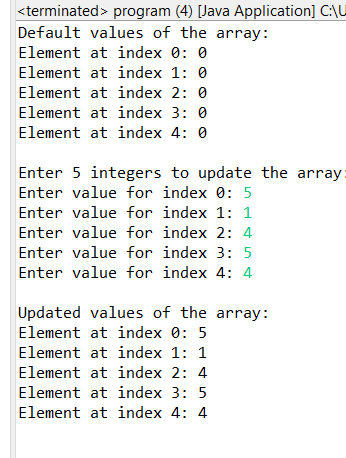
}

scanner.close();

}

}

Output:



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.

Program:

**package** com.in.comlex;

**import** java.util.Scanner;

**public** **class** question2 {

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

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

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

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

arr[i] = scanner.nextInt();

}

scanner.close();

}

**public** **static** **void** printRecord(**int**[] arr) {

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

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

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

}

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

}

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

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

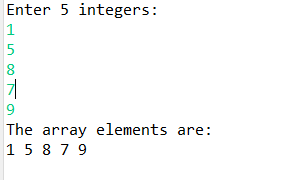
*acceptRecord*(arr);

*printRecord*(arr);

}

}

Output:



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

Program:

**import** java.util.Scanner;

**public** **class** MaxMinArray {

**public** **static** **int** findMax(**int**[] arr) {

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

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

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

max = arr[i];

}

}

**return** max;

}

**public** **static** **int** findMin(**int**[] arr) {

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

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

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

min = arr[i];

}

}

**return** min;

}

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

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

System.***out***.print("Enter the number of elements in the array: ");

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

**int**[] arr = **new** **int**[n];

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

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

arr[i] = scanner.nextInt();

}

**int** max = *findMax*(arr);

**int** min = *findMin*(arr);

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

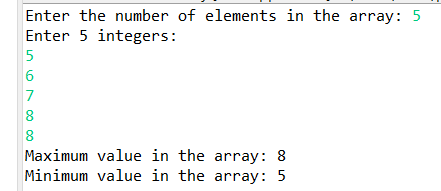
System.***out***.println("Minimum value in the array: " + min);

scanner.close();

}

}

Output:



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

Program:

**package** ass.six.four;

**import** java.util.\*;

**public** **class** dublicates {

**public** **static** **int**[] removeDuplicates(**int**[] arr) {

Arrays.*sort*(arr);

**int**[] temp = **new** **int**[arr.length];

**int** j = 0; // Index for the temp array

**for** (**int** i = 0; i < arr.length - 1; i++) {

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

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

}

}

temp[j++] = arr[arr.length - 1];

**int**[] uniqueArray = **new** **int**[j];

**for** (**int** i = 0; i < j; i++) {

uniqueArray[i] = temp[i];

}

**return** uniqueArray;

}

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

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

System.***out***.print("Enter the number of elements in the array: ");

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

**int**[] arr = **new** **int**[n];

// Accept the elements from the user

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

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

arr[i] = scanner.nextInt();

}

// Remove duplicates and get the new array

**int**[] uniqueArray = *removeDuplicates*(arr);

// Display the array without duplicates

System.***out***.println("Array after removing duplicates:");

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

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

}

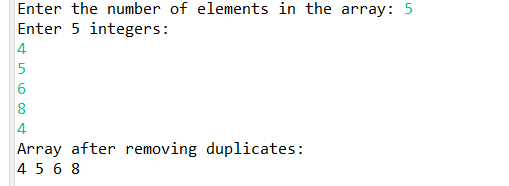
System.***out***.println();

scanner.close();

}

}

Output:



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

Program:

**package** ass.six.five;

**import** java.util.\*;

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

// Method to find the intersection of two arrays

**public** **static** **int**[] findIntersection(**int**[] arr1, **int**[] arr2) {

List<Integer> intersection = **new** ArrayList<>();

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

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

**if** (arr1[i] == arr2[j] && !intersection.contains(arr1[i])) {

intersection.add(arr1[i]);

}

}

}

**int**[] result = **new** **int**[intersection.size()];

**for** (**int** i = 0; i < intersection.size(); i++) {

result[i] = intersection.get(i);

}

**return** result;

}

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

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

System.***out***.print("Enter the number of elements in the first array: ");

**int** n1 = scanner.nextInt();

**int**[] arr1 = **new** **int**[n1];

System.***out***.println("Enter " + n1 + " integers for the first array:");

**for** (**int** i = 0; i < n1; i++) {

arr1[i] = scanner.nextInt();

}

System.***out***.print("Enter the number of elements in the second array: ");

**int** n2 = scanner.nextInt();

**int**[] arr2 = **new** **int**[n2];

System.***out***.println("Enter " + n2 + " integers for the second array:");

**for** (**int** i = 0; i < n2; i++) {

arr2[i] = scanner.nextInt();

}

**int**[] intersection = *findIntersection*(arr1, arr2);

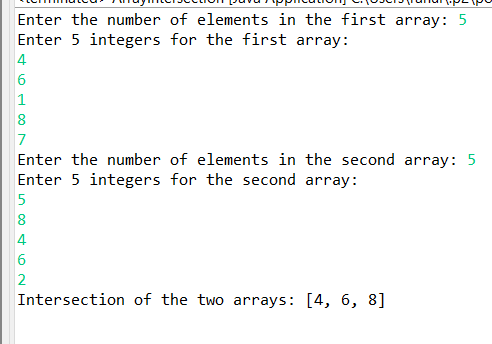
System.***out***.println("Intersection of the two arrays: " + Arrays.*toString*(intersection));

scanner.close();

}

}

Output:



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

Program:

**package** ass.six.six;

**import** java.util.\*;

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

// Method

**public** **static** **int** findMissingNumber(**int**[] arr, **int** N) {

**int** expectedSum = N \* (N + 1) / 2; // Sum of numbers from 1 to N

**int** Sum = 0;

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

Sum += num;

}

**return** expectedSum - Sum;

}

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

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

System.***out***.print("Enter the value of N (the range): ");

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

**int**[] arr = **new** **int**[N - 1]; // Array size is N-1 as one number is missing

System.***out***.println("Enter " + (N - 1) + " integers:");

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

arr[i] = scanner.nextInt();

}

**int** missingNumber = *findMissingNumber*(arr, N);

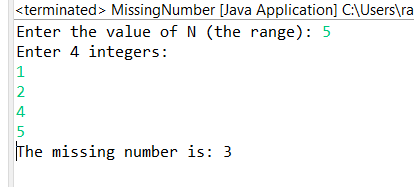
System.***out***.println("The missing number is: " + missingNumber);

scanner.close();

}

}

Output:



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.

Program:

**package** ass.six.seven;

**import** java.util.Scanner;

**class** Array {

**private** **int**[] arr; // Single-dimensional array as a field

// Constructor to instantiate the array

**public** Array(**int** size) {

arr = **new** **int**[size]; // Instantiate the array

}

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

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

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

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

arr[i] = scanner.nextInt();

}

scanner.close();

}

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

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

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

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

}

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

}

}

**public** **class** SingleDimensionalArray {

**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();

Array Rahul= **new** Array(size);

Rahul.acceptRecord();

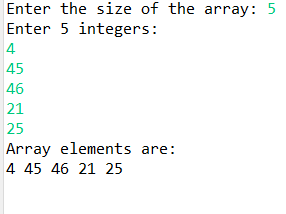
Rahul.printRecord();

scanner.close();

}

}

Output:



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

Program:

**package** ass.six.eight;

**import** java.util.Scanner;

**class** DhamDhum {

**private** **int**[] arr;

**public** DhamDhum(**int** size) {

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

}

**public** **void** setArrayValues() {

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

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

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

arr[i] = scanner.nextInt();

}

scanner.close();

}

**public** **int**[] getArrayValues() {

**return** arr;

}

**public** **void** printArray() {

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

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

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

}

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

}

}

**public** **class** ArrayWithgetset {

**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();

DhamDhum DJ = **new** DhamDhum (size);

DJ.setArrayValues();

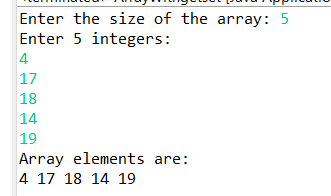
DJ.printArray();

scanner.close();

}

}

Output:



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.

Code : contain 4 class files

File 1:

**package** com.in.RahulBharaskar;

**public** **enum** SeatStatus {

***AVAILABLE***('A'),

***BOOKED***('B');

**private** **final** **char** symbol;

SeatStatus(**char** symbol) {

**this**.symbol = symbol;

}

**public** **char** getSymbol() {

**return** symbol;

}

}

File 2: AirplaneSeatManager

**package** com.in.RahulBharaskar;

**public** **class** AirplaneSeatManager {

**private** SeatStatus[][] seats;

**private** **int** rows;

**private** **int** columns;

**public** AirplaneSeatManager(**int** rows, **int** columns) {

**this**.rows = rows;

**this**.columns = columns;

seats = **new** SeatStatus[rows][columns];

initializeSeats();

}

// Initialize all seats as available

**private** **void** initializeSeats() {

**for** (**int** i = 0; i < rows; i++) {

**for** (**int** j = 0; j < columns; j++) {

seats[i][j] = SeatStatus.***AVAILABLE***;

}

}

}

// Book a seat

**public** **boolean** bookSeat(**int** row, **int** column) {

**if** (isValidSeat(row, column) && seats[row][column] == SeatStatus.***AVAILABLE***) {

seats[row][column] = SeatStatus.***BOOKED***;

**return** **true**;

}

**return** **false**;

}

// Cancel a seat booking (mark it as AVAILABLE)

**public** **boolean** cancelSeat(**int** row, **int** column) {

**if** (isValidSeat(row, column) && seats[row][column] == SeatStatus.***BOOKED***) {

seats[row][column] = SeatStatus.***AVAILABLE***;

**return** **true**;

}

**return** **false**;

}

// Check if a specific seat is available

**public** **boolean** isSeatAvailable(**int** row, **int** column) {

**if** (isValidSeat(row, column)) {

**return** seats[row][column] == SeatStatus.***AVAILABLE***;

}

**return** **false**;

}

// Display the current seating chart

**public** **void** displaySeats() {

System.***out***.println("\nCurrent Seating Chart:");

**for** (**int** i = 0; i < rows; i++) {

**for** (**int** j = 0; j < columns; j++) {

System.***out***.print(seats[i][j].getSymbol() + " ");

}

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

}

}

// Helper method to check if the seat is within valid range

**private** **boolean** isValidSeat(**int** row, **int** column) {

**return** row >= 0 && row < rows && column >= 0 && column < columns;

}

}

File 3: AirplaneSeatManagerUtil

**package** com.in.RahulBharaskar;

**import** java.util.Scanner;

**public** **class** AirplaneSeatManagerUtil {

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

// Method to take input from user

**public** **static** **int** getInput(String prompt) {

System.***out***.print(prompt);

**return** *scanner*.nextInt();

}

// Display menu options

**public** **static** **void** displayMenu() {

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("Choose an option: ");

}

}

File 4: Program

**package** com.in.RahulBharaskar;

**public** **class** Program {

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

System.***out***.println("Welcome to the Airplane Seat Management System!");

**int** rows = AirplaneSeatManagerUtil.*getInput*("Enter number of rows: ");

**int** columns = AirplaneSeatManagerUtil.*getInput*("Enter number of columns: ");

AirplaneSeatManager manager = **new** AirplaneSeatManager(rows, columns);

**boolean** exit = **false**;

**while** (!exit) {

AirplaneSeatManagerUtil.*displayMenu*();

**int** choice = AirplaneSeatManagerUtil.*getInput*("");

**switch** (choice) {

**case** 1: // Book a seat

**int** bookRow = AirplaneSeatManagerUtil.*getInput*("Enter row to book: ");

**int** bookCol = AirplaneSeatManagerUtil.*getInput*("Enter column to book: ");

**if** (manager.bookSeat(bookRow, bookCol)) {

System.***out***.println("Seat booked successfully.");

} **else** {

System.***out***.println("Seat already booked or invalid seat.");

}

**break**;

**case** 2: // Cancel a booking

**int** cancelRow = AirplaneSeatManagerUtil.*getInput*("Enter row to cancel: ");

**int** cancelCol = AirplaneSeatManagerUtil.*getInput*("Enter column to cancel: ");

**if** (manager.cancelSeat(cancelRow, cancelCol)) {

System.***out***.println("Booking canceled successfully.");

} **else** {

System.***out***.println("No booking found or invalid seat.");

}

**break**;

**case** 3: // Check seat availability

**int** checkRow = AirplaneSeatManagerUtil.*getInput*("Enter row to check: ");

**int** checkCol = AirplaneSeatManagerUtil.*getInput*("Enter column to check: ");

**if** (manager.isSeatAvailable(checkRow, checkCol)) {

System.***out***.println("Seat is available.");

} **else** {

System.***out***.println("Seat is not available.");

}

**break**;

**case** 4: // Display seating chart

manager.displaySeats();

**break**;

**case** 5: // Exit

exit = **true**;

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

**break**;

**default**:

System.***out***.println("Invalid option! Please try again.");

}

}

}

}

Output:

