1. **Find the maximum and minimum element in an array.**

import java.util.Scanner;

class Main {

public static void MaxMinArrayElement(int[] elements){

int Max = elements[0],Min = elements[0];

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

if(elements[i] > Max){

Max = elements[i];

}else if(elements[i]< Min){

Min = elements[i];

}

}

System.out.println("Maximum Element of an array : " + Max) ;

System.out.println("Minimum Element of an array : " + Min) ;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter Total number of array element : ");

int size = sc.nextInt();

System.out.println("Enter Array Element : ");

int[] arr = new int[size];

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

arr[i] = sc.nextInt();

}

// System.out.println("Elements Of an Array is : " + arr.toString());

MaxMinArrayElement(arr);

}

}

**Alternate Method (Using Collections):**

import java.util.Arrays;

import java.util.Collections;

class Main {

public static void main(String[] args) {

Integer arr[] = {10, 25, 5, 90, 45, 2};

int max = Collections.max(Arrays.asList(arr));

int min = Collections.min(Arrays.asList(arr));

System.out.println("Max Elelemt of an Array : " + max);

System.out.println("Min Element of an Array : " + min);

}

}

====================================================================================================================================================================

1. **Reverse an array in-place**

**Or**

**Reverse an array without using another array**

class Main {

public static void reverseArrayInPlace(int [] arr , int start, int end){

while(start< end){

int temp = arr[start];

arr[start] = arr[end];

arr[end] = temp;

start++;

end--;

}

System.out.print ("Reverse Array : ");

for(int num : arr)

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

}

public static void main(String[] args) {

int[] arr = {1, 2, 3, 4, 5};

int start = 0;

int end = arr.length-1;

reverseArrayInPlace(arr,start,end);

}

}

**==========================================================================================**

**3. Check if an array is a palindrome.**

An array is a palindrome if it reads the same forward and backward.

class Main {

public static void PalindromeArrayCheck(int [] arr , int start, int end){

boolean isPlindomeArray = true;

while(start<end){

if(arr[start]!= arr[end]){

isPlindomeArray = false;

break;

}

start++;

end--;

}

if(isPlindomeArray){

System.out.println("Given Array is an Plaindrom ");

}else{

System.out.println("Given Array is NOT an Plaindrom ");

}

}

public static void main(String[] args) {

int[] arr = {1, 2, 3, 0, 1};

int start = 0;

int end = arr.length-1;

PalindromeArrayCheck(arr,start,end);

}

}

====================================================================================================================================================================

1. **Find duplicates in an array.**

import java.util.Set;

import java.util.HashSet;

class Main {

public static void main(String[] args) {

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

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

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

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

if(!unique.add(arr[i])){

duplicate.add(arr[i]);

}

}

System.out.println("Duplicate array element : " + duplicate);

System.out.println("Unique array element : " + unique);

}

}

======================================================================================================================================================

1. **Remove duplicates from a sorted array**

import java.util.Arrays;

class Main {

public static void main(String[] args) {

int [] arr = {1, 1, 2, 2, 3, 4, 4, 5};

int count =0;

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

if(i<arr.length-1 && arr[i]==arr[i+1]){

continue;

}else{

arr[count]=arr[i];

count++;

}

}

for(int i=0 ;i<count ; i++)

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

}

}

======================================================================================================================================================

1. **Find the missing number in a sequence**

class Main {

public static void main(String[] args) {

int[] arr = {1, 2, 4, 5};

int n = arr.length+1;

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

int actualsum =0;

for(int num : arr){

actualsum += num;

}

int missingsum = expectedsum - actualsum ;

System.out.println("Missing Number is : " + missingsum);

}

}

======================================================================================================================================================

1. **Find the first non-repeating element in an array**

class Main {

public static void main(String[] args) {

int[] arr = {1, 2, 3, 2, 1, 4, 5};

boolean found = false;

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

boolean isRepeating = false;

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

if(i<j && arr[i]==arr[j]){

isRepeating = true;

break;

}

}

if(!isRepeating){

found=true;

System.out.println ("First Non Repeating Element Of an Array Found as : " + arr[i]);

break;

}

}

if(!found){

System.out.println ("No First Non Repeating Element Of array Found ");

}

}

}

**OR**

**Alternative Solution**

import java.util.HashMap;

import java.util.Map;

public class FirstNonRepeatingElement {

public static void main(String[] args) {

int[] arr = {2, 3, 4, 2, 3, 5, 4};

// Step 1: Create a HashMap to store frequency of each element

Map<Integer, Integer> countMap = new HashMap<>();

for (int num : arr) {

countMap.put(num, countMap.getOrDefault(num, 0) + 1);

}

// Step 2: Traverse array again to find first element with frequency = 1

int firstNonRepeating = -1;

for (int num : arr) {

if (countMap.get(num) == 1) {

firstNonRepeating = num;

break;

}

}

// Step 3: Print result

if (firstNonRepeating != -1)

System.out.println("First non-repeating element: " + firstNonRepeating);

else

System.out.println("No non-repeating element found");

}

}

======================================================================================================================================================