Array Subtraction

i/p : arr\_1[] = {9,8,7,5}, arr\_2[] = {1,2,3,8}

CODE :

public class ArraySubtraction{

    public static void main(String[] args) {

        int arr1[] = {9,1,8,9};

        int arr2[] = {1,0,2,7};

        int n1 = arr1.length;

        int n2 = arr2.length;

        int n3 = n1>n2 ? n1:n2;

        int result[] = new int[n3];

        int diff = 0;

        int borrow =0;

        int i = n1-1;

        int j = n2-1;

        int k= n3-1;

        while(k>=0){

            if(i>=0 && j>=0){

                diff = arr1[i] - arr2[j] - borrow;

            }

            else if(i>=0 && j<0){

                diff = arr1[i] - borrow;

            }

            else if(i<0 && j>=0){

                diff = arr2[i] - borrow;

            }

            if(diff <0){

                diff+=10;

                borrow=1;

            }

            else{

                borrow=0;

            }

            result[k] = diff;

            i--;

            j--;

            k--;

        }

        int startIndex = 0;

        while (startIndex < n3 - 1 && result[startIndex] == 0) {

            startIndex++;

        }

        System.out.print("Result: ");

        if (startIndex == n3) {

            System.out.print("0");

        } else {

            for (int l = startIndex; l < n3; l++) {

                System.out.print(result[l] + " ");

            }

        }

    }

}

OUTPUT :



Missing Number : LC - 268

i/p : arr[] = {1,2,4,5,6}

o/p : 3

CODE:

import java.util.Arrays;

public class MissingNumber {

    static void missing(int arr[]){

        Arrays.sort(arr);

        int n = arr.length;

        int missingNumber = -1;

        for (int i = 0; i < n; i++) {

            if (arr[i] != i+1) { // if arr start or havig 0 in it then arr[i] != i

                missingNumber = i+1; // missingnum - i

                continue;

            }

        }

        // if (missingNumber == -1) {

        //     missingNumber = n; // Handles the case when the missing number is at the end of the array.

        // }

        System.out.println(missingNumber);

    }

    public static void main(String[] args) {

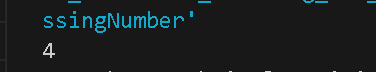
        int arr[] = {0,1,2,3,5};

        missing(arr);

    }

}

OUTPUT :



Remove Duplicate from sorted array : LC - 26

CODE :

public class removeDuplicate {

    static int duplicateremove(int [] arr){

        int l = 1;

        int n = arr.length;

        for(int i=1;i<n;i++){

            if(arr[i]!=arr[i-1]){

                arr[l]=arr[i];

                l+=1;

            }

        }

        return l;

    }

    public static void main(String[] args) {

        int arr[] = {1,2,2,4,5};

        int length = duplicateremove(arr);

        System.out.println("New Length: " + length);

        System.out.print("Array without Duplicates: ");

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

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

        }

    }

}

OUTPUT :

A screen shot of a computer

Description automatically generated with low confidence

Odd Element in array

i/p : arr[] = {1,1,2,2,3,3,4,5,5,6,6};

o/p : 4

CODE:

import javax.sound.sampled.SourceDataLine;

public class oddElement {

    static void oddele(int [] arr){

        int ans =0;

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

            ans = ans^arr[i];

            }

        System.out.println(ans);

        }

    public static void main(String[] args) {

        int [] arr= {1,1,2,2,3,3,4,5,5,6,6};

        oddele(arr);

    }

}

OUTPUT :

