1. Rotate the array

class Solution {

    public void reverse(int nums[],int low,int high){

        while(low<high){

            int temp=nums[low];

            nums[low]=nums[high];

            nums[high] = temp;

            low++;

            high--;

        }

    }

    public void rotate(int[] nums, int k) {

        k = k%nums.length;

        if(k==0)

            return;

        reverse(nums,0,nums.length-1);

        reverse(nums,0,k-1);

        reverse(nums,k,nums.length-1);

    }

}

1. Find out the missing number

class Solution {

int missingNumber(int array[], int n) {

// Your Code Here

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

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

sum-=array[i];

return sum;

}

}

1. Addition and subtraction between two arrays and place it in 3rd array

ADD

public class SumOfArrays{

    public static int[] addArrays(int[] arr1, int[] arr2) {

        int maxLength = Math.max(arr1.length, arr2.length);

        int[] result = new int[maxLength];

        int carry = 0;

        int i = arr1.length-1;

        int j = arr2.length-1;

        int k = result.length-1;

        while(i>=0 || j>=0){

            int val1 = (i >= 0) ? arr1[i] : 0;

            int val2 = (j>=0) ? arr2[j] : 0;

            int sum = val1 + val2 + carry;

            result[k] = sum % 10;

            carry = sum / 10;

            i--;

            j--;

            k--;

        }

        if (carry != 0) {

            int[] extendedResult = new int[result.length + 1];

            System.arraycopy(result, 0, extendedResult, 1, result.length);

            extendedResult[0] = carry;

            return extendedResult;

        }

        return result;

    }

    public static void main(String[] args) {

        int[] array1 = {9, 9, 9, 9};

        int[] array2 = {1, 2, 3, 4};

        int[] sumArray = addArrays(array1, array2);

        // Print the sumArray

        for (int num : sumArray) {

            System.out.print(num);

        }

        System.out.println();

    }

}

SUBTRACT

public class DifferenceOfArrays {

    public static int[] subtractArrays(int[] arr1, int[] arr2) {

        int maxLength = Math.max(arr1.length, arr2.length);

        int[] result = new int[maxLength];

        int borrow = 0;

        int i = arr1.length - 1;

        int j = arr2.length - 1;

        int k = result.length - 1;

        while (i >= 0 || j >= 0) {

            int val1 = (i >= 0) ? arr1[i] : 0;

            int val2 = (j >= 0) ? arr2[j] : 0;

            // Subtract the borrow from val1

            val1 -= borrow;

            if (val1 < val2) {

                // Need to borrow from the previous digit

                borrow = 1;

                val1 += 10;

            } else {

                // No borrowing required

                borrow = 0;

            }

            int diff = val1 - val2;

            result[k] = diff;

            i--;

            j--;

            k--;

        }

        // Remove leading zeros if any

        int startIndex = 0;

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

            startIndex++;

        }

        int[] trimmedResult = new int[result.length - startIndex];

        System.arraycopy(result, startIndex, trimmedResult, 0, trimmedResult.length);

        return trimmedResult;

    }

    public static void main(String[] args) {

        int[] array1 = {9, 1, 9, 9};

        int[] array2 = {1, 2, 3, 4};

        int[] diffArray = subtractArrays(array1, array2);

        // Print the diffArray

        for (int num : diffArray) {

            System.out.print(num);

        }

        System.out.println();

    }

}