DSA CODE

**Most Frequent Element**

import java.util.\*;

public class Source {

public static int mostFrequentElement(int[] arr) {

// Write code here

}

public static void main(String[] args) {

int n;

Scanner sc = new Scanner(System.in);

n = sc.nextInt();

int arr[] = new int[n];

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

arr[i] = sc.nextInt();

}

System.out.println(mostFrequentElement(arr));

}

}

**Check Whether an Undirected Graph is a Tree or Not**

import java.util.\*;{

private int vertexCount;

private static LinkedList<Integer> adj[];

Source(int vertexCount) {

this.vertexCount = vertexCount;

this.adj = new LinkedList[vertexCount];

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

adj[i] = new LinkedList<Integer>();

}

}

public void addEdge(int v, int w) {

if (!isValidIndex(v) || !isValidIndex(w)) {

return;

}

adj[v].add(w);

adj[w].add(v);

}

**Find kth Largest Element in a Stream**

V import java.util.PriorityQueue;

class KthLargestElement {

public static int findKthLargest(int[] nums, int k) {

PriorityQueue<Integer> minHeap = new PriorityQueue<>();

for (int num : nums) {

minHeap.add(num);

if (minHeap.size() > k) {

minHeap.poll();

}

}

return minHeap.peek();

}

public static void main(String[] args) {

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

int k = 2;

System.out.println("The " + k + "th largest element is: " + findKthLargest(nums, k)); // Output: 5

}

}

**Sort Nearly Sorted Array**

import java.util.\*;

public class Source {

private static void sortArray(int[] arr, int k) {

// Write code here

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

int k = sc.nextInt();

int arr[] = new int[n];

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

arr[i] = sc.nextInt();

}

sortArray(arr, k);

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

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

}

}

}

**Find Sum Between pth and qth Smallest Elements**

import java.util.\*;

public class Source {

public static int sumBetweenPthToQthSmallestElement(int[] arr, int p, int q) {

// Write code here

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

int arr[] = new int[n];

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

arr[i] = sc.nextInt();

}

int p = sc.nextInt();

int q = sc.nextInt();

System.out.println(sumBetweenPthToQthSmallestElement(arr, p, q));

}

}

**Find All Symmetric Pairs in an Array**

import java.util.\*;

public class Source {

public static void symmetricPair(int[][] arr) {

// Write code here

}

public static void main(String arg[]) {

Scanner sc = new Scanner(System.in);

int row = sc.nextInt();

int arr[][] = new int[row][2];

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

for(int j = 0 ; j < 2 ; j++){

arr[i][j] = sc.nextInt();

}

}

symmetricPair(arr);

}

}

**Find All Common Element in All Rows of Matrix**

import java.util.\*;

public class Source {

public static void printElementInAllRows(int mat[][]) {

// Write code here

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int row = sc.nextInt();

int col = sc.nextInt();

int matrix[][] = new int[row][col];

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

for(int j = 0 ; j < col ; j++){

matrix[i][j] = sc.nextInt();

}

}

printElementInAllRows(matrix);

}

}

**Find Itinerary in Order**

import java.util.\*;

public class Source {

public static void findItinerary(Map<String, String> tickets) {

// Write code here

}

public static void main(String[] args) {

Map<String, String> tickets = new HashMap<String, String>();

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

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

tickets.put(sc.next(),sc.next());

}

findItinerary(tickets);

}

}

**Search Element in a Rotated Array**

import java.util.\*;

public class Source {

public static int search(int arr[], int left, int right, int key) {

// Write code here

}

public static void main(String args[]) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

int arr[] = new int[n];

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

arr[i] = sc.nextInt();

}

int key = sc.nextInt();

int i = search(arr, 0, n - 1, key);

if (i != -1) {

System.out.println(i);

} else {

System.out.println("-1");

}

}

}

**Find Median After Merging Two Sorted Arrays**

import java.util.\*;

public class Source {

public static int median(int[] arr1, int[] arr2 , int n){

// Write code here

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

int arr1[] = new int[n];

int arr2[] = new int[n];

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

arr1[i] = sc.nextInt();

}

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

arr2[i] = sc.nextInt();

}

System.out.println(median(arr1, arr2, n));

}

}