**13-11-2024**

**CODING PRACTICE PROBLEMS**

**1.Kth smallest element**

import java.util.\*;

class Problem1 {

public static int kthSmallest(int[] arr, int n, int k) {

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

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

pq.add(arr[i]);

}

int result = -1;

for (int i = 1; i <= k; i++) {

result = pq.poll();

}

return result;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter the size of the array: ");

int n = sc.nextInt();

int[] arr = new int[n];

System.out.println("Enter the elements of the array: ");

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

arr[i] = sc.nextInt();

}

System.out.print("Enter the value of k: ");

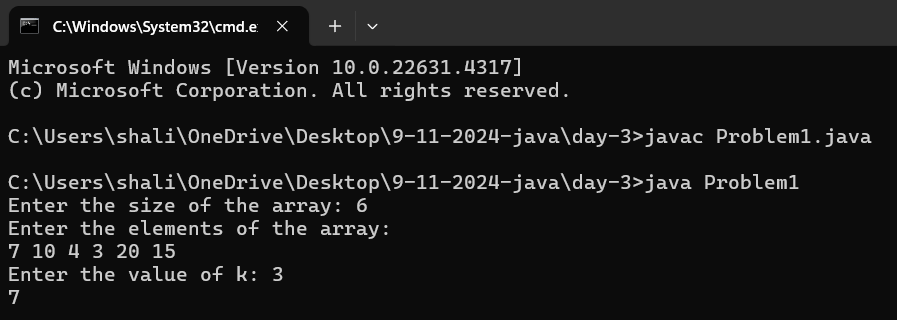
int k = sc.nextInt();

System.out.println(kthSmallest(arr, n, k));

sc.close();

}

}



**2.Minimize the height**

import java.util.\*;

class Problem2 {

public static int minDifference(int[] arr, int n, int k) {

Arrays.sort(arr);

int ans = arr[n - 1] - arr[0];

for (int i = 1; i < n; i++) {

int min = Math.min(arr[0] + k, arr[i] - k);

int max = Math.max(arr[i - 1] + k, arr[n - 1] - k);

ans = Math.min(ans, max - min);

}

return ans;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter the size of the array: ");

int n = sc.nextInt();

int[] arr = new int[n];

System.out.println("Enter the elements of the array: ");

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

arr[i] = sc.nextInt();

}

System.out.print("Enter the value of k: ");

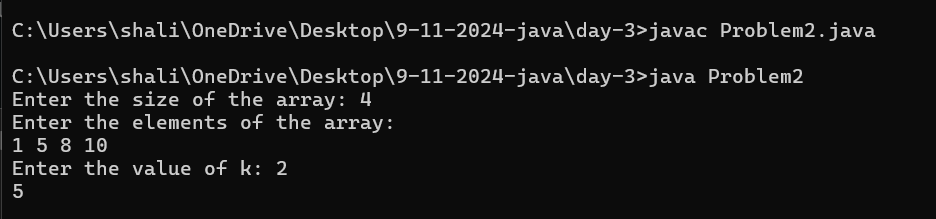
int k = sc.nextInt();

System.out.println(minDifference(arr, n, k));

sc.close();

}

}



**3.Paranthesis Checker**

import java.util.\*;

class Problem3 {

public static boolean isBalanced(String s) {

Stack<Character> stack = new Stack<>();

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

char ch = s.charAt(i);

if (ch == '{' || ch == '(' || ch == '[') {

stack.push(ch);

} else if (ch == '}' && !stack.isEmpty() && stack.peek() == '{') {

stack.pop();

} else if (ch == ')' && !stack.isEmpty() && stack.peek() == '(') {

stack.pop();

} else if (ch == ']' && !stack.isEmpty() && stack.peek() == '[') {

stack.pop();

} else {

return false;

}

}

return stack.isEmpty();

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter the expression: ");

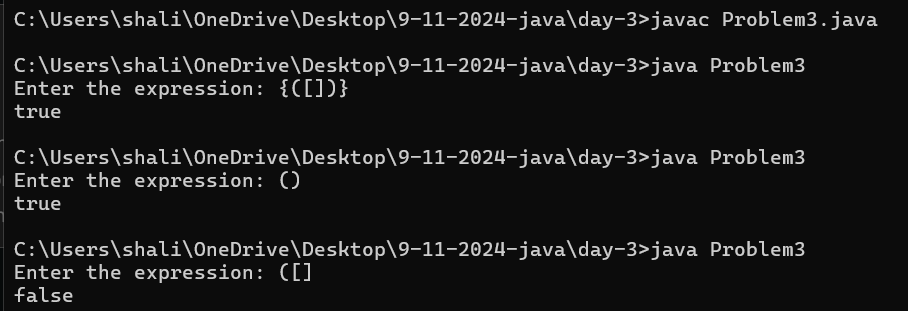
String s = sc.nextLine();

System.out.println(isBalanced(s));

sc.close();

}

}



**4.Equilibrium point**

import java.util.\*;

class Problem4 {

public static int equilibriumPoint(int[] arr, int n) {

int totalSum = 0;

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

totalSum += arr[i];

}

int leftSum = 0;

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

totalSum -= arr[i];

if (leftSum == totalSum) {

return i + 1; // 1-based index

}

leftSum += arr[i];

}

return -1;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter the size of the array: ");

int n = sc.nextInt();

int[] arr = new int[n];

System.out.println("Enter the elements of the array: ");

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

arr[i] = sc.nextInt();

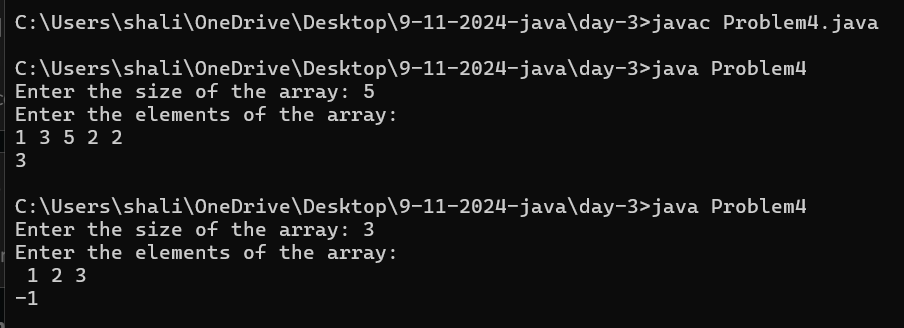
}

System.out.println(equilibriumPoint(arr, n));

sc.close();

}

}



**5.Binary Search**

import java.util.\*;

class Problem5 {

public static int binarySearch(int[] arr, int target) {

int left = 0, right = arr.length - 1;

while (left <= right) {

int mid = left + (right - left) / 2;

if (arr[mid] == target) {

return mid;

}

if (arr[mid] < target) {

left = mid + 1;

} else {

right = mid - 1;

}

}

return -1;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter the size of the array: ");

int n = sc.nextInt();

int[] arr = new int[n];

System.out.println("Enter the elements of the array (sorted): ");

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

arr[i] = sc.nextInt();

}

System.out.print("Enter the target element: ");

int target = sc.nextInt();

int result = binarySearch(arr, target);

if (result == -1) {

System.out.println("Element not found");

} else {

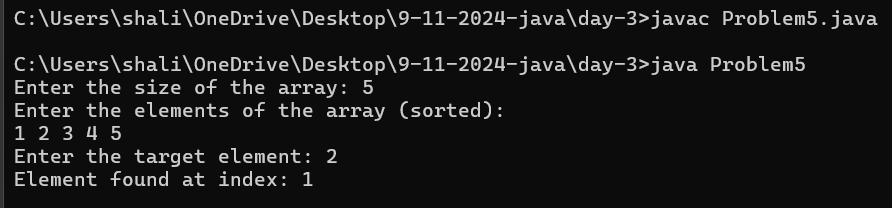
System.out.println("Element found at index: " + result);

}

sc.close();

}

}



**6.Next Greater element**

import java.util.\*;

class Problem6 {

public static int[] nextGreaterElement(int[] arr) {

int n = arr.length;

int[] result = new int[n];

Stack<Integer> stack = new Stack<>();

for (int i = n - 1; i >= 0; i--) {

while (!stack.isEmpty() && stack.peek() <= arr[i]) {

stack.pop();

}

result[i] = stack.isEmpty() ? -1 : stack.peek();

stack.push(arr[i]);

}

return result;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter the size of the array: ");

int n = sc.nextInt();

int[] arr = new int[n];

System.out.println("Enter the elements of the array: ");

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

arr[i] = sc.nextInt();

}

int[] result = nextGreaterElement(arr);

System.out.print("Next greater elements: ");

for (int i : result) {

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

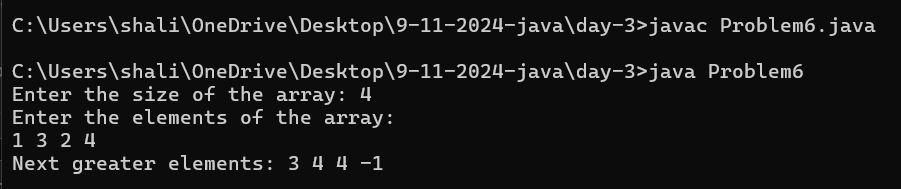
}

System.out.println();

sc.close();

}

}



**7. Union of two arrays(with duplicate elements)**

import java.util.\*;

class Problem7 {

public static int unionCount(int[] a, int[] b) {

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

for (int num : a) {

set.add(num);

}

for (int num : b) {

set.add(num);

}

return set.size();

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter the size of array a: ");

int n = sc.nextInt();

int[] a = new int[n];

System.out.println("Enter elements of array a: ");

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

a[i] = sc.nextInt();

}

System.out.print("Enter the size of array b: ");

int m = sc.nextInt();

int[] b = new int[m];

System.out.println("Enter elements of array b: ");

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

b[i] = sc.nextInt();

}

int result = unionCount(a, b);

System.out.println("Number of elements in the union: " + result);

sc.close();

}

}

