## CHANDIGARH UNIVERSITY

## UNIVERSITY INSTITUTE OF NGINEERING

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**



|  |  |
| --- | --- |
| **Submitted By: Submitted To:**  Vivek Kumar(21BCS8129) Mamta Punia(E12337) | |
| **Subject Name** | Competitive Coding - I |
| **Subject Code** | 20CSP-314 |
| **Branch** | Computer Science and Engineering |
| **Semester** | 5th |

**Experiment - 2**

**Student Name: Vivek Kumar UID: 21BCS8129**

**Branch: BE-CSE(LEET) Section/Group: WM-20BCS-616/A**

**Semester: 5th Date of Performance: 12/08/2022**

**Subject Name: Competitive coding - I Subject Code: 20CSP-314**

**Game of Two Stack:**

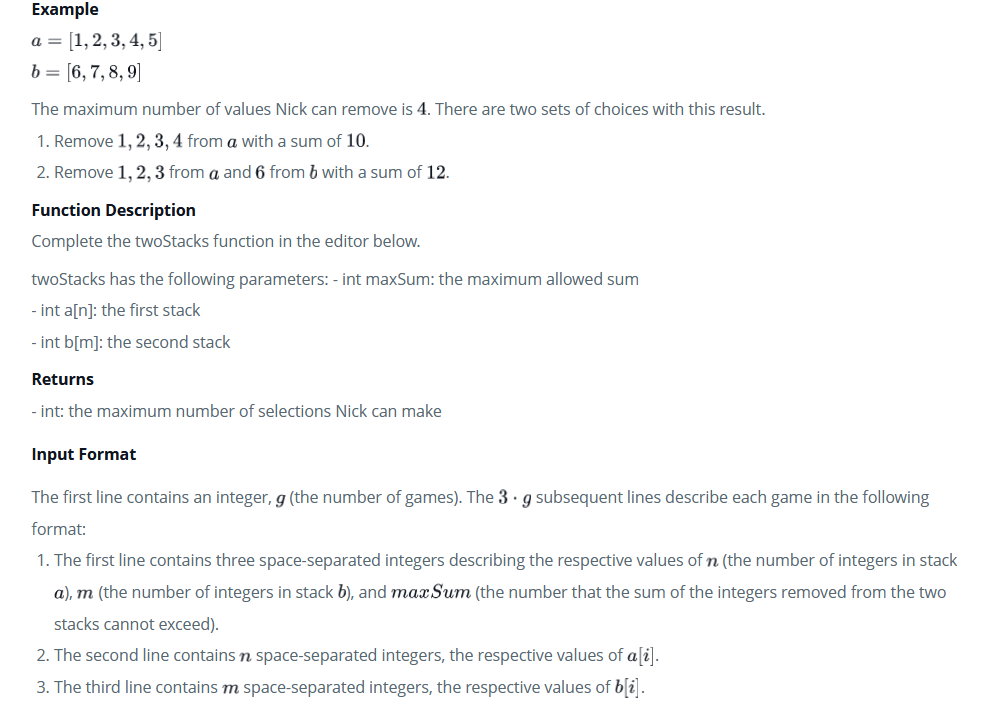
**1. Aim/Overview of the practical:**

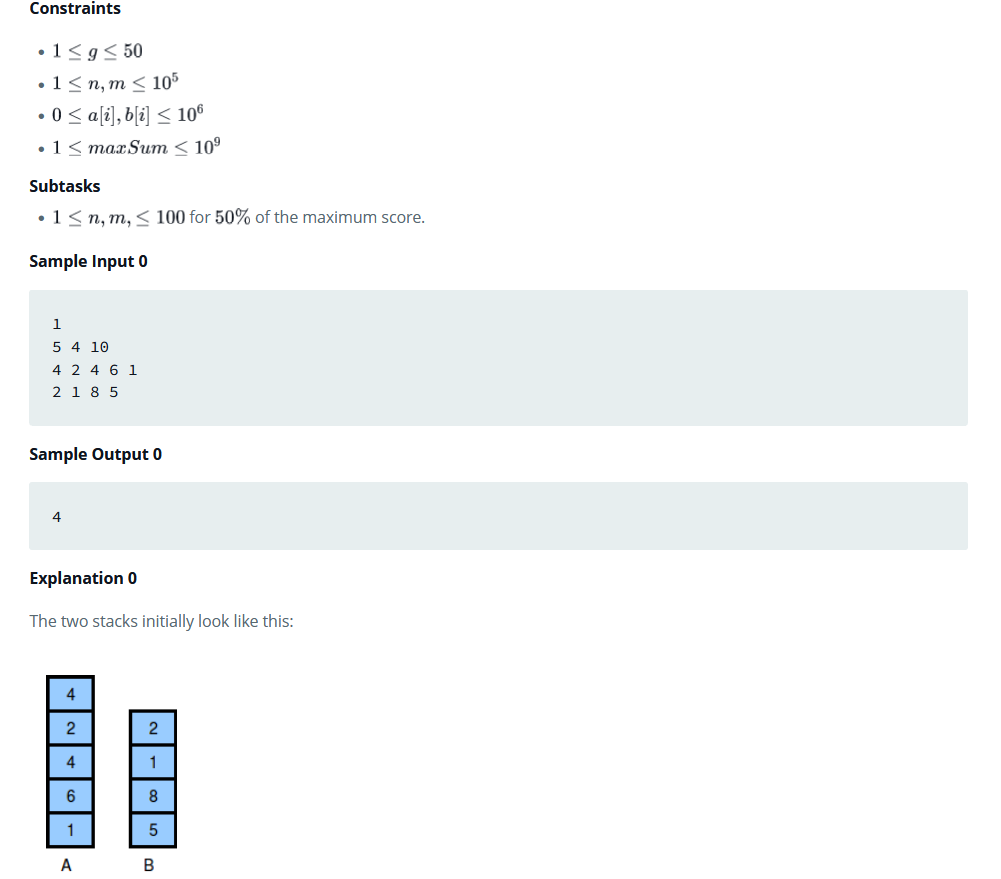
Alexa has two stacks of non-negative integers, stack a[n] and stack b[m] where index 0 denotes the top of the stack. Alexa challenges Nick to play the following game:

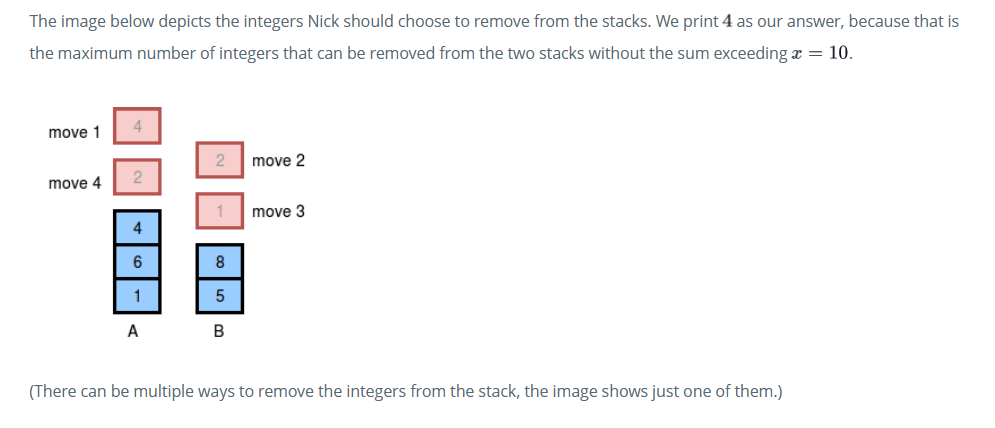
* In each move, Nick can remove one integer from the top of either stack a or stack b.
* Nick keeps a running sum of the integers he removes from the two stacks.
* Nick is disqualified from the game if, at any point, his running sum becomes greater than some integer max-Sum given at the beginning of the game.
* Nick's *final score* is the total number of integers he has removed from the two stacks.

Given a, b and max-Sum for g games, find the maximum possible score Nick can achieve.

**2. Task to be done/ Which logistics used:**







**3. Hardware and Software Requirements (For programming-based labs):**

* Laptop or Desktop
* Hacker-Rank Account

**4. Steps for experiment/practical/Code:**

import java.io.\*;

import java.math.\*;

import java.security.\*;

import java.text.\*;

import java.util.\*;

import java.util.concurrent.\*;

import java.util.regex.\*;

class Result {

/\*

\* Complete the 'twoStacks' function below.

\*

\* The function is expected to return an INTEGER.

\* The function accepts following parameters:

\* 1. INTEGER maxSum

\* 2. INTEGER\_ARRAY a

\* 3. INTEGER\_ARRAY b

\*/

public static int twoStacks(int maxSum, List<Integer> a, List<Integer> b) {

// Write your code here

int sum = 0;

int count = 0;

int i = 0;

int j = 0;

while (i < a.size() && (sum + a.get(i)) <= maxSum) {

sum += a.get(i);

i++;

}

count = i;

while (j < b.size() && i >= 0) {

sum += b.get(j);

j++;

while (sum > maxSum && i > 0) {

i--;

sum -= a.get(i);

}

if (sum <= maxSum && (i + j) > count)

count = i + j;

}

//System.out.println("count "+count);

return count;

}

}

public class Solution {

public static void main(String[] args) throws IOException {

BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(System.in));

BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT\_PATH")));

int g = Integer.parseInt(bufferedReader.readLine().trim());

for (int gItr = 0; gItr < g; gItr++) {

String[] firstMultipleInput = bufferedReader.readLine().replaceAll("\\s+$", "").split(" ");

int n = Integer.parseInt(firstMultipleInput[0]);

int m = Integer.parseInt(firstMultipleInput[1]);

int maxSum = Integer.parseInt(firstMultipleInput[2]);

String[] aTemp = bufferedReader.readLine().replaceAll("\\s+$", "").split(" ");

List<Integer> a = new ArrayList<>();

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

int aItem = Integer.parseInt(aTemp[i]);

a.add(aItem);

}

String[] bTemp = bufferedReader.readLine().replaceAll("\\s+$", "").split(" ");

List<Integer> b = new ArrayList<>();

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

int bItem = Integer.parseInt(bTemp[i]);

b.add(bItem);

}

int result = Result.twoStacks(maxSum, a, b);

bufferedWriter.write(String.valueOf(result));

bufferedWriter.newLine();

}

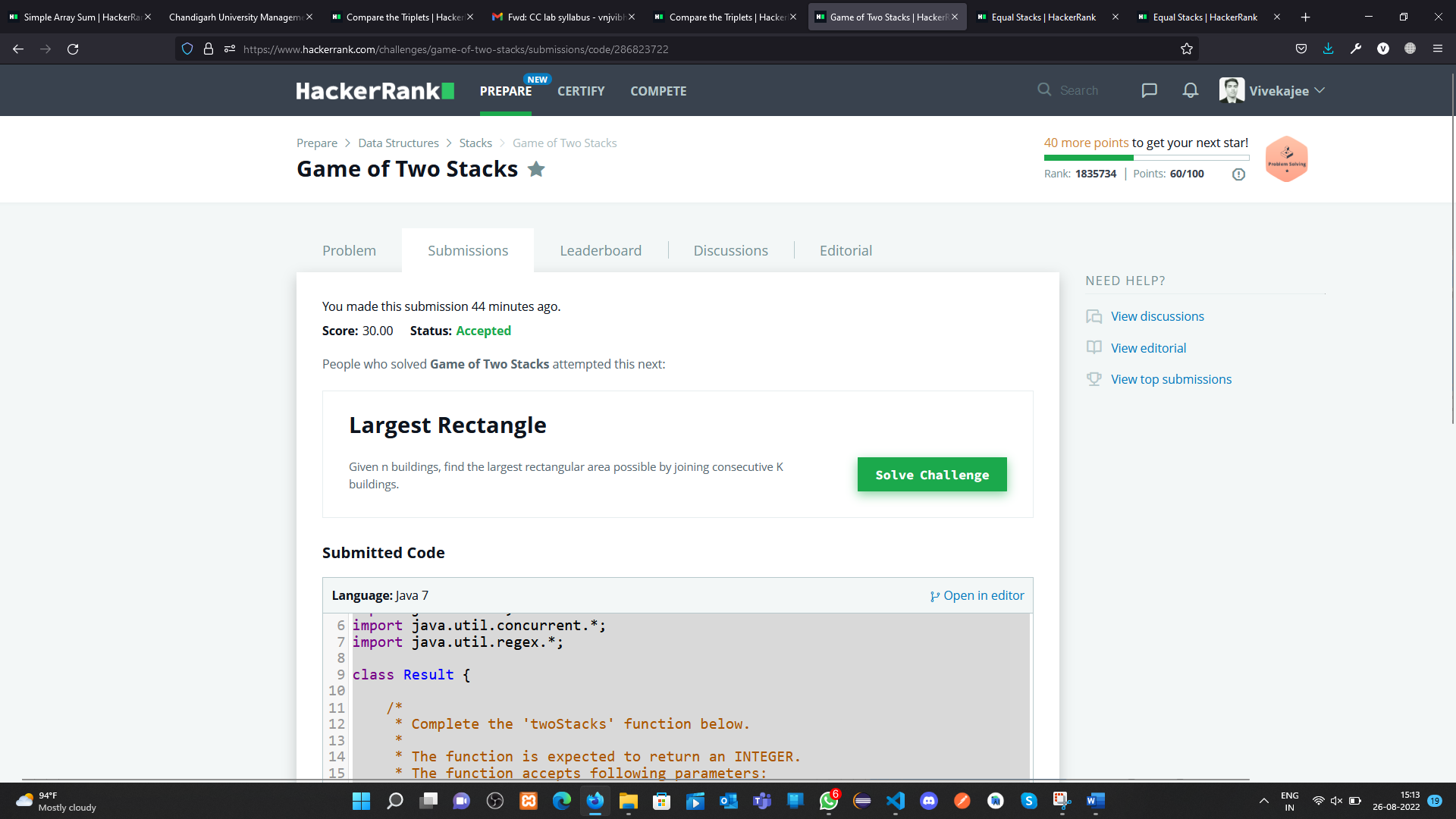
bufferedReader.close();

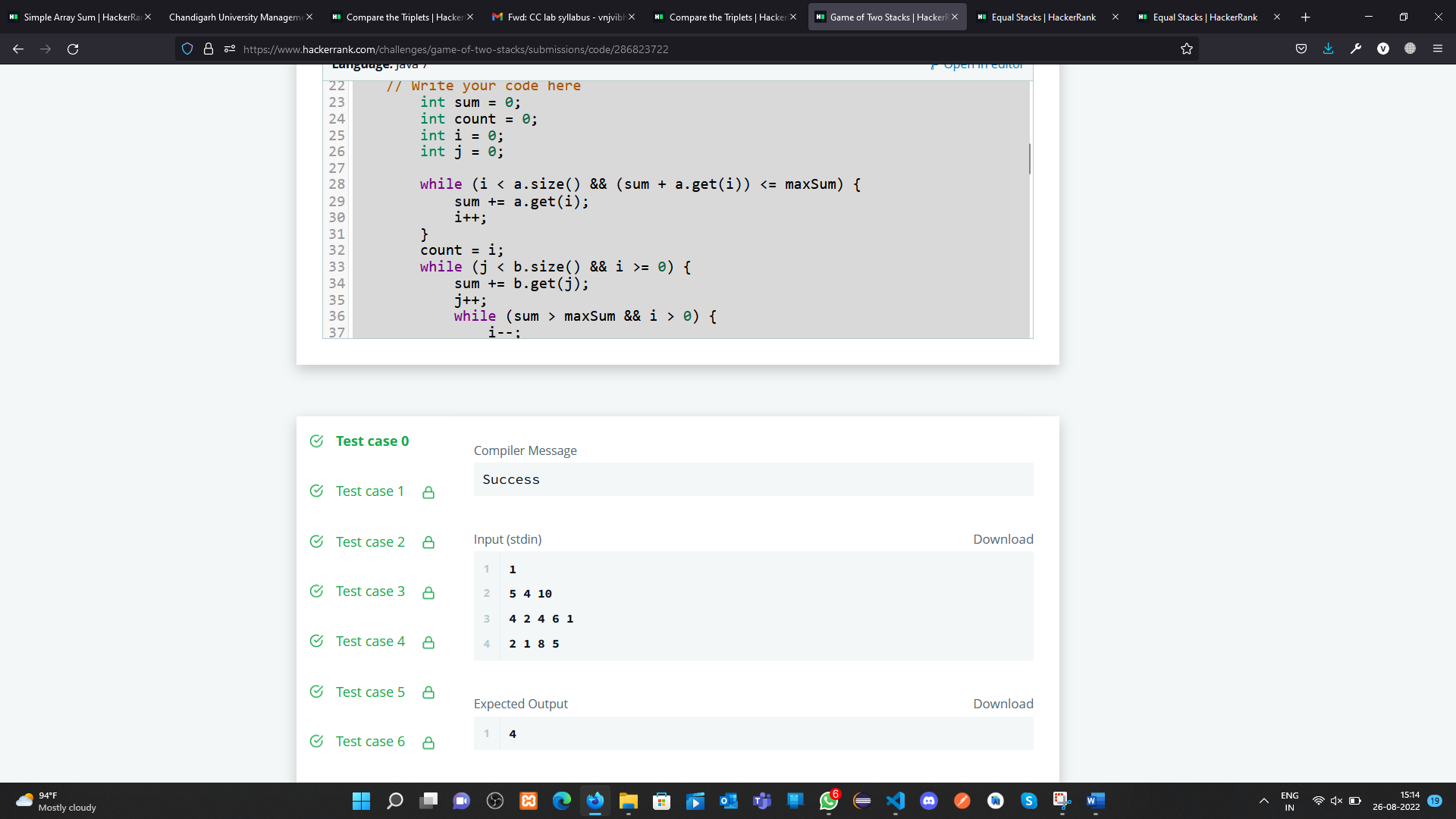
bufferedWriter.close();

}

}

**5. Result/Output/Writing Summary:**





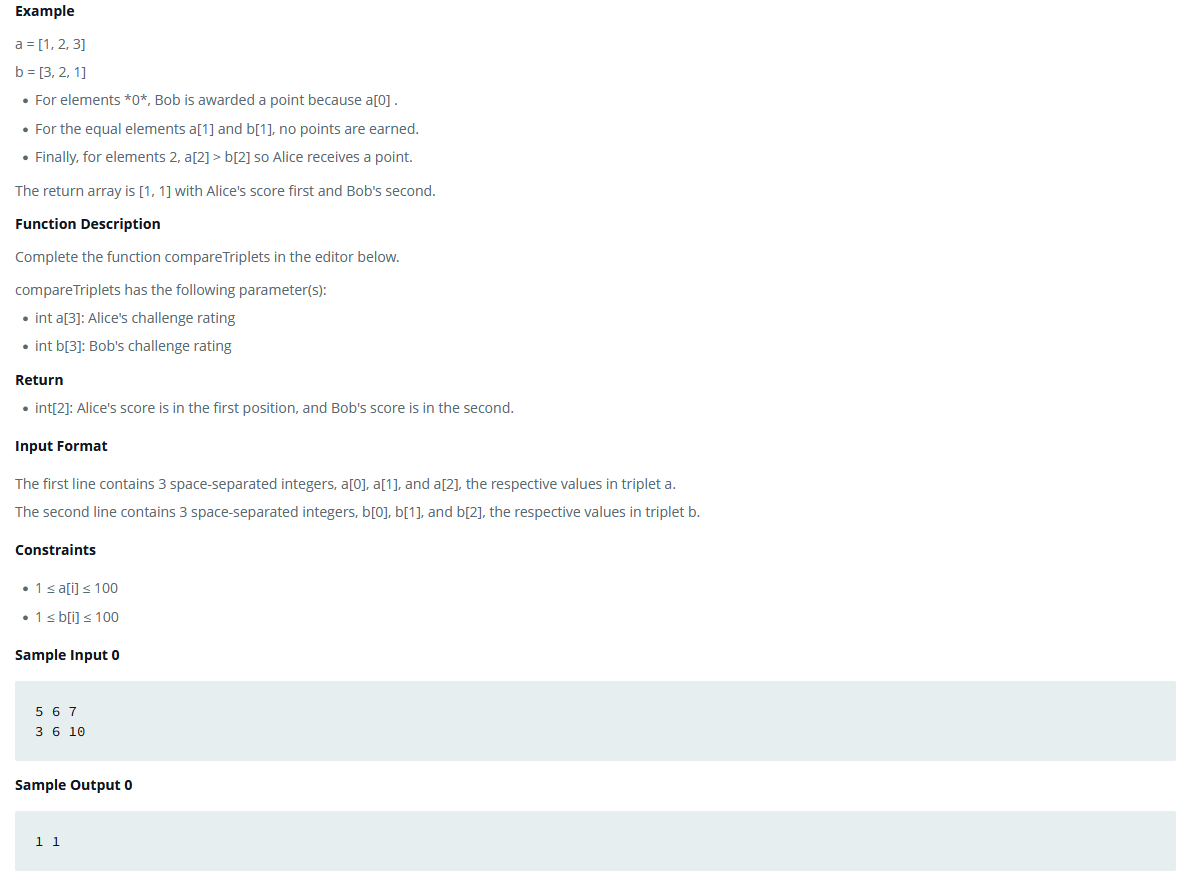
**Compare the Triplets:**

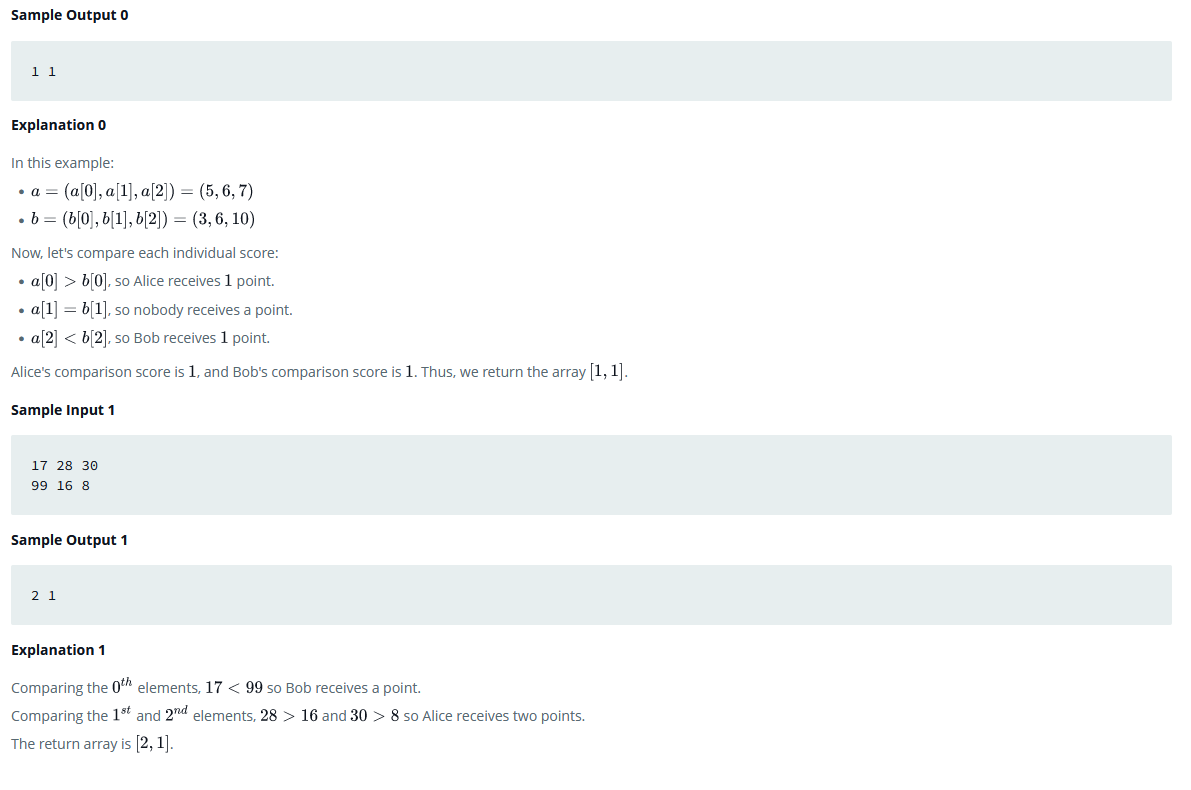
**1. Aim/Overview of the practical:**

Given an array of integers, find the sum of its elements.

For example, if the array arr=[1,2,3], 1+2+3=6 , so return 6.

**2. Task to be done/ Which logistics used:**





**3. Hardware and Software Requirements (For programming-based labs):**

* Laptop or Desktop
* Hacker-Rank Account

**4. Steps for experiment/practical/Code:**

import java.io.\*;

import java.math.\*;

import java.security.\*;

import java.text.\*;

import java.util.\*;

import java.util.concurrent.\*;

import java.util.regex.\*;

class Result {

/\*

\* Complete the 'compareTriplets' function below.

\*

\* The function is expected to return an INTEGER\_ARRAY.

\* The function accepts following parameters:

\* 1. INTEGER\_ARRAY a

\* 2. INTEGER\_ARRAY b

\*/

public static List<Integer> compareTriplets(List<Integer> a, List<Integer> b) {

// Write your code here

int alice = 0;

int bob = 0;

List<Integer> answer = new ArrayList<>();

for(int i = 0; i < 3; i++) {

if (a.get(i) > b.get(i)) alice++;

if (a.get(i) < b.get(i)) bob++;

}

answer.add(0,alice);

answer.add(1,bob);

return answer;

}

}

public class Solution {

public static void main(String[] args) throws IOException {

BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(System.in));

BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT\_PATH")));

String[] aTemp = bufferedReader.readLine().replaceAll("\\s+$", "").split(" ");

List<Integer> a = new ArrayList<>();

for (int i = 0; i < 3; i++) {

int aItem = Integer.parseInt(aTemp[i]);

a.add(aItem);

}

String[] bTemp = bufferedReader.readLine().replaceAll("\\s+$", "").split(" ");

List<Integer> b = new ArrayList<>();

for (int i = 0; i < 3; i++) {

int bItem = Integer.parseInt(bTemp[i]);

b.add(bItem);

}

List<Integer> result = Result.compareTriplets(a, b);

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

bufferedWriter.write(String.valueOf(result.get(i)));

if (i != result.size() - 1) {

bufferedWriter.write(" ");

}

}

bufferedWriter.newLine();

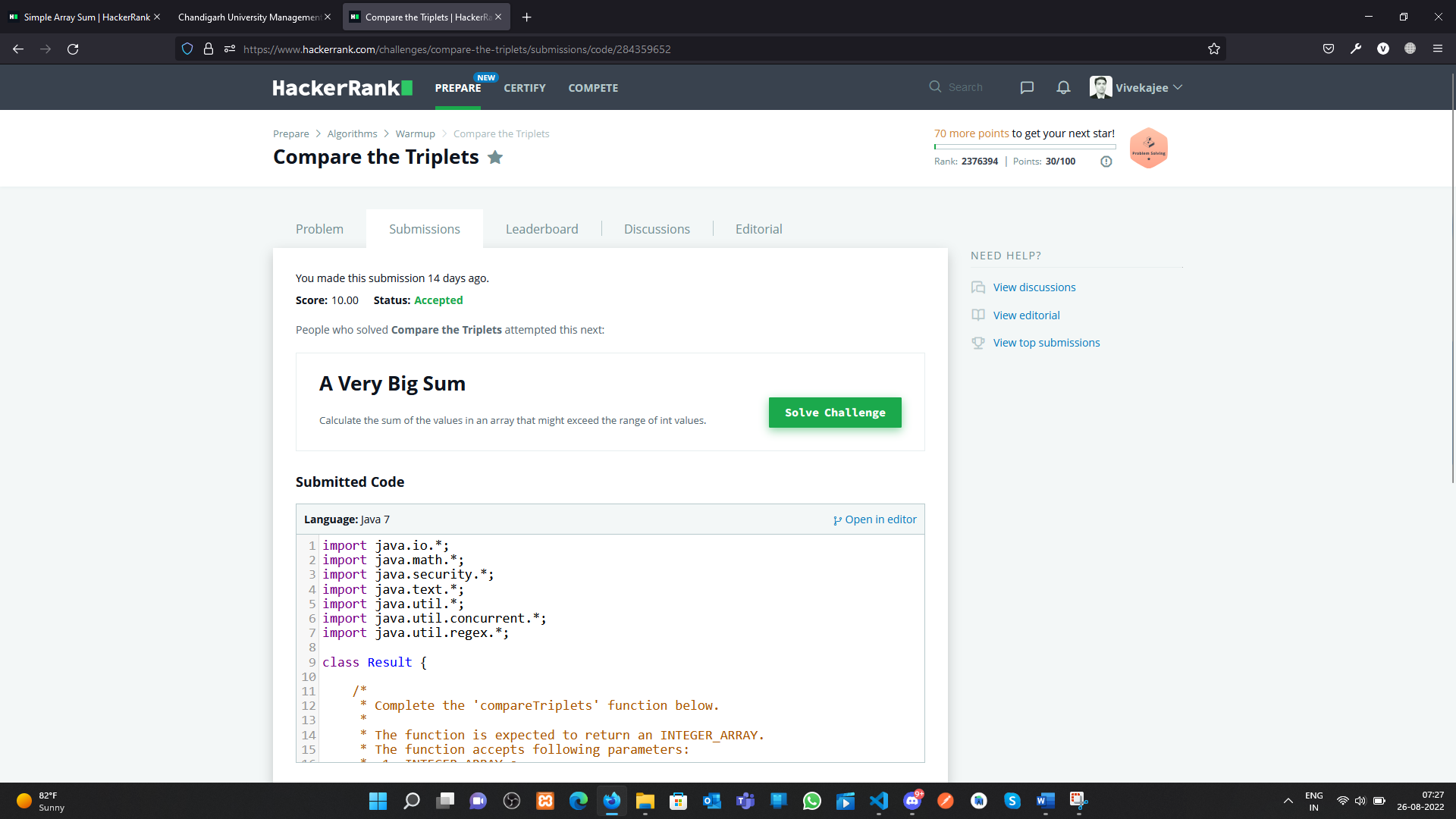
bufferedReader.close();

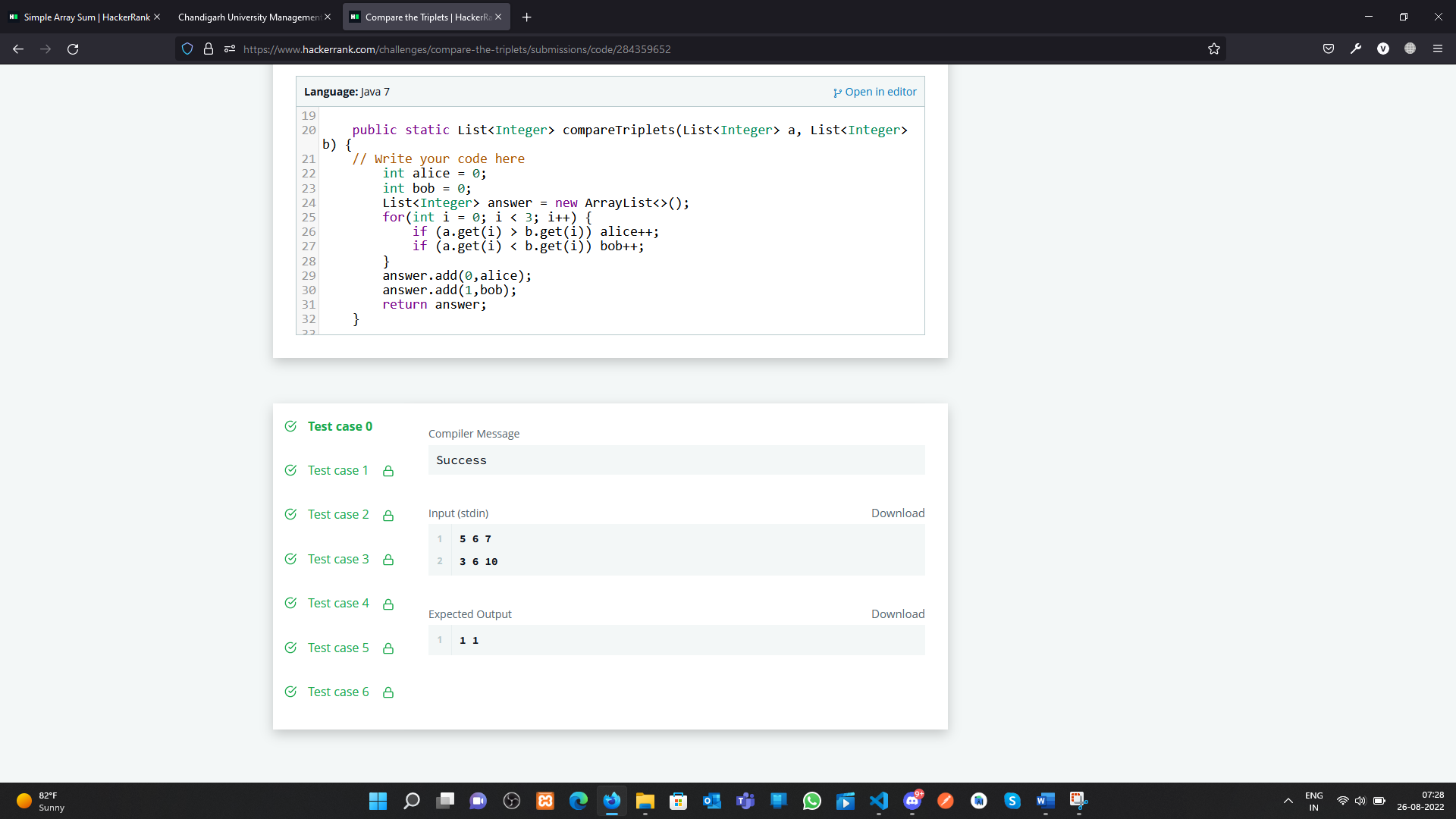
bufferedWriter.close();

}

}

**6. Result/Output/Writing Summary:**





**Learning outcomes (What I have learnt):**

**1.** Array concept in Java

**2.** Sum of the all-item present in an Array

**3.** Compare the triplets and show the results.

**Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):**

|  |  |  |  |
| --- | --- | --- | --- |
| Sr. No. | Parameters | Marks Obtained | Maximum Marks |
| 1. |  |  |  |
| 2. |  |  |  |
| 3. |  |  |  |
|  |  |  |  |