**Competitive Coding Lab 2**

**Student Name: Sahul Kr. Parida UID: 20BCS4919**

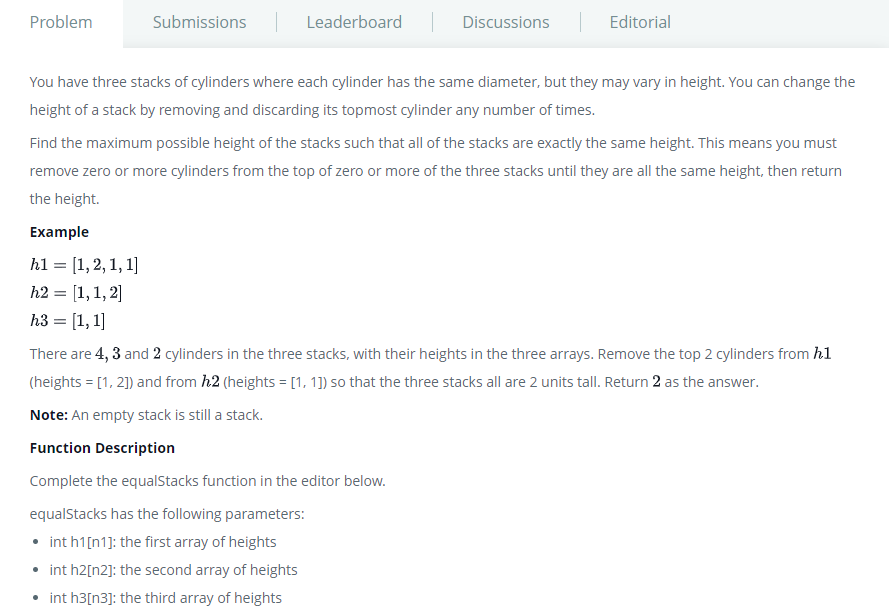
**Branch: CSE Section/Group: WM-904/B**

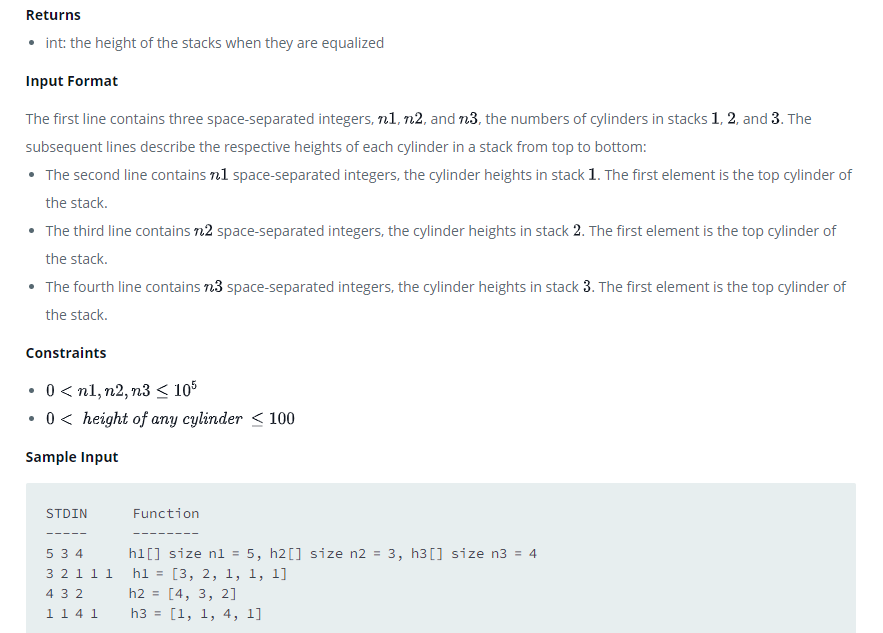
**Semester: 5th Date of Performance: 16/08/22**

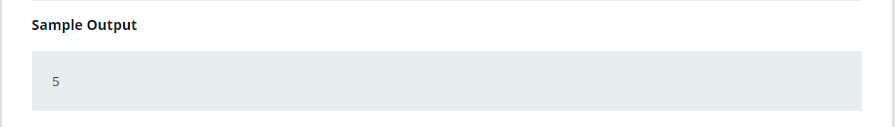
**Subject Name: Competitive Coding(CC) Subject Code: 20CSP-314**

**PROBLEM STATEMENT 2.1: -**

[**https://www.hackerrank.com/challenges/equal-stacks/problem?isFullScreen=false**](https://www.hackerrank.com/challenges/equal-stacks/problem?isFullScreen=false)







**SOLUTION:**

import java.io.BufferedWriter;

import java.io.FileWriter;

import java.io.IOException;

import java.util.Scanner;

import java.util.Stack;

/\* 20BCS4919\_Sahul Kumar Parida \*/

public class EqualStacks {

    static int equalStacks(int[] ha1, int[] ha2, int[] ha3) {

        Stack<Integer> s1 = new Stack<Integer>(); //First stack of cylinders

        Stack<Integer> s2 = new Stack<Integer>(); //Second stack of cylinders

        Stack<Integer> s3 = new Stack<Integer>(); //Third stack of cylinders

        int h1 = 0; //Height of the first stack

        int h2 = 0; //Height of the second stack

        int h3 = 0; //Height of the third stack

        int minStack; //The stack with the smallest height

        //Initialize the stacks and their heights

        for(int i = ha1.length-1; i >= 0 ; i--){

            s1.push(ha1[i]);

            h1 += ha1[i];

        }

        for(int i = ha2.length-1; i >= 0 ; i--){

            s2.push(ha2[i]);

            h2 += ha2[i];

        }

        for(int i = ha3.length-1; i >= 0 ; i--){

            s3.push(ha3[i]);

            h3 += ha3[i];

        }

        minStack = Math.min(h1,Math.min(h2,h3)); //Initialize minStack with the minimum height

        //Heights are not all equal enter the while loop

        while(h1 != h2 || h1 != h3) {

            //Remove cylinders from stack 1 until your height is <= the smallest

            while(h1 > minStack){

                h1 -= s1.pop();

            }

            minStack = Math.min(h1,Math.min(h2,h3)); //Recalculate min

            //Remove cylinders from stack 2 until your height is <= the smallest

            while(h2 > minStack){

                h2 -= s2.pop();

            }

            minStack = Math.min(h1,Math.min(h2,h3)); //Recalculate min

            //Remove cylinders from stack 3 until your height is <= the smallest

            while(h3 > minStack){

                h3 -= s3.pop();

            }

            minStack = Math.min(h1,Math.min(h2,h3)); //Recalculate min

        }

        return minStack;

    }

    private static final Scanner scanner = new Scanner(System.in);

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

        String[] n1N2N3 = scanner.nextLine().split(" ");

        int n1 = Integer.parseInt(n1N2N3[0].trim());

        int n2 = Integer.parseInt(n1N2N3[1].trim());

        int n3 = Integer.parseInt(n1N2N3[2].trim());

        int[] h1 = new int[n1];

        String[] h1Items = scanner.nextLine().split(" ");

        for (int h1Itr = 0; h1Itr < n1; h1Itr++) {

            int h1Item = Integer.parseInt(h1Items[h1Itr].trim());

            h1[h1Itr] = h1Item;

        }

        int[] h2 = new int[n2];

        String[] h2Items = scanner.nextLine().split(" ");

        for (int h2Itr = 0; h2Itr < n2; h2Itr++) {

            int h2Item = Integer.parseInt(h2Items[h2Itr].trim());

            h2[h2Itr] = h2Item;

        }

        int[] h3 = new int[n3];

        String[] h3Items = scanner.nextLine().split(" ");

        for (int h3Itr = 0; h3Itr < n3; h3Itr++) {

            int h3Item = Integer.parseInt(h3Items[h3Itr].trim());

            h3[h3Itr] = h3Item;

        }

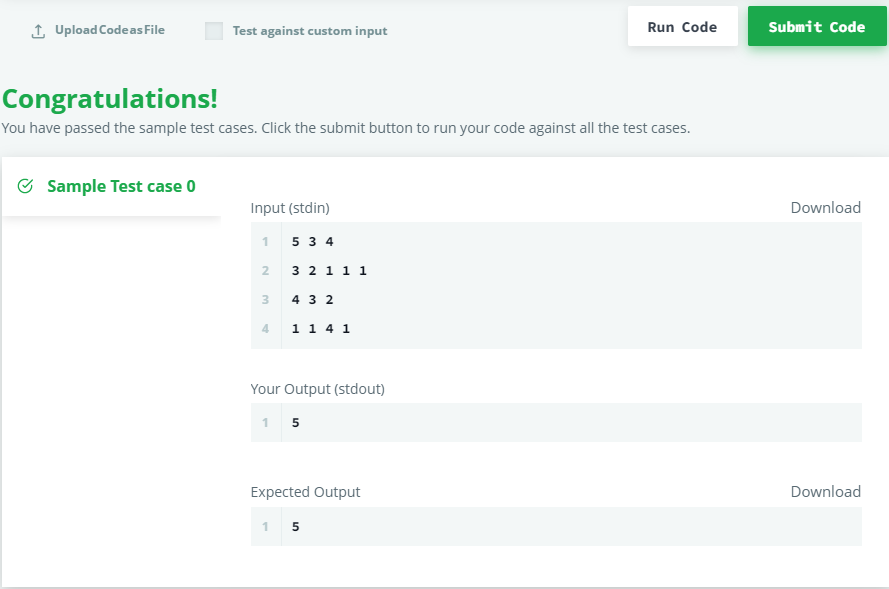
        int result = equalStacks(h1, h2, h3);

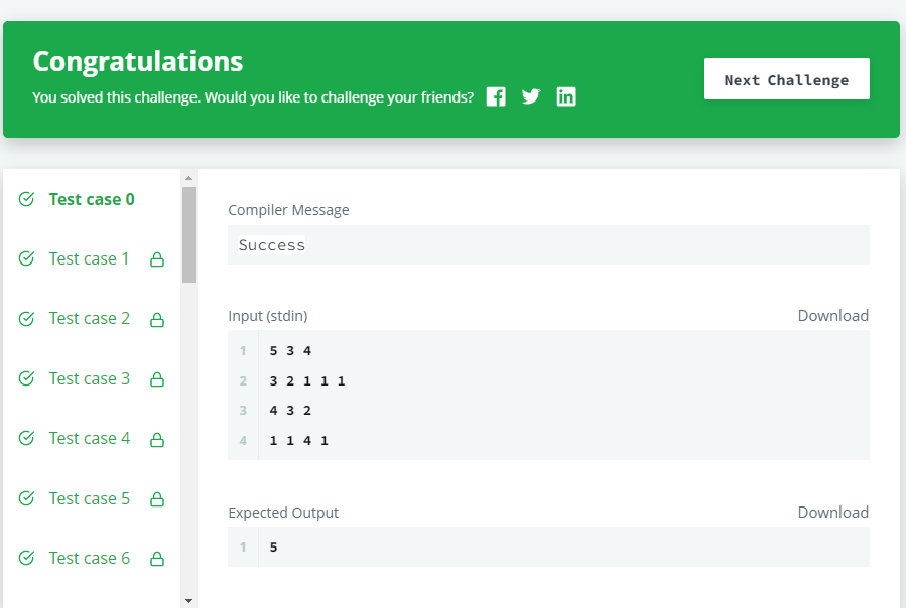
        System.out.println(result);

    }

}

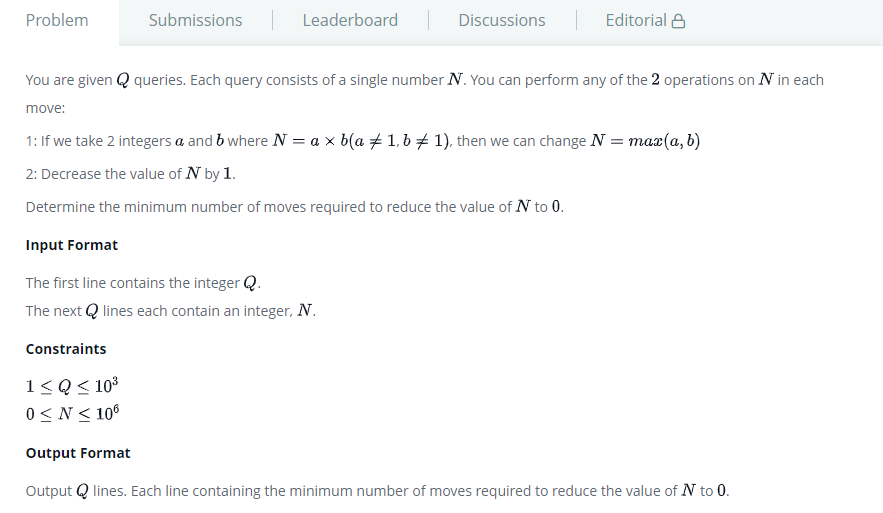
# TEST CASES:

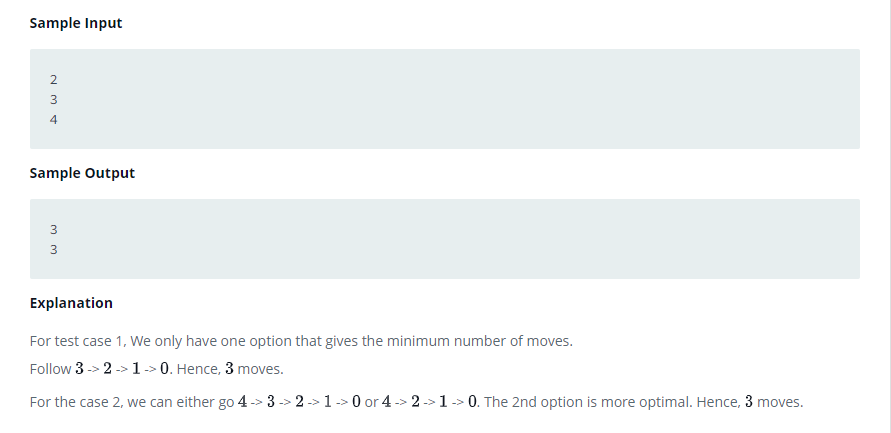
****

****

**PROBLEM STATEMENT 2.2: -**

[**https://www.hackerrank.com/challenges/down-to-zero-ii/problem?isFullScreen=false**](https://www.hackerrank.com/challenges/down-to-zero-ii/problem?isFullScreen=false)





# SOLUTION:

import java.io.\*;

import java.util.\*;

/\* 20BCS4919\_Sahul Kumar Parida \*/

public class Solution {

  static int[] moves = new int[1000001];

  public static void main(String[] args) {

    for (int i = 1; i <= 1000000; ++i) {

      int least = moves[i - 1];

      for (int j = 2; j \* j <= i; ++j) {

        if (i % j == 0) {

          least = Math.min(least, moves[i / j]);

        }

      }

      moves[i] = ++least;

    }

    Scanner in = new Scanner(System.in);

    int t = in.nextInt();

    while (t-- > 0) {

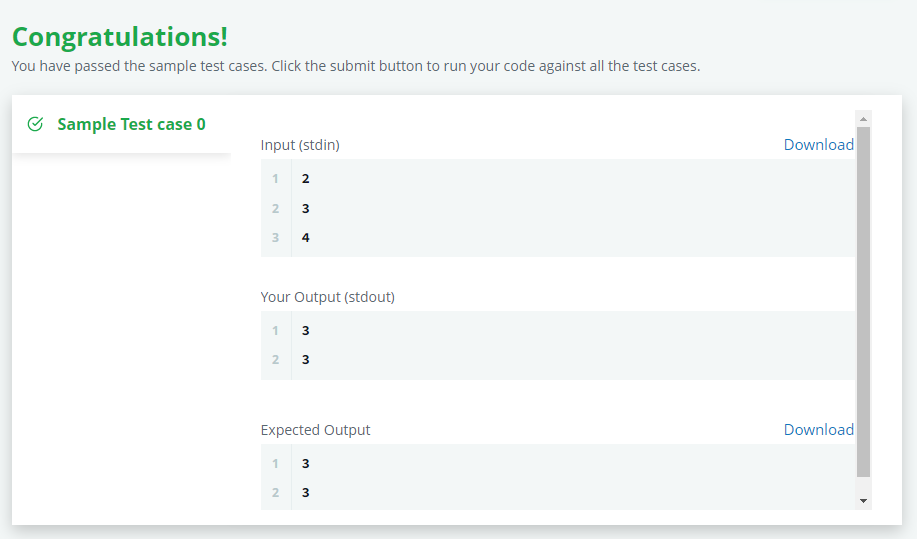
      System.out.println(moves[in.nextInt()]);

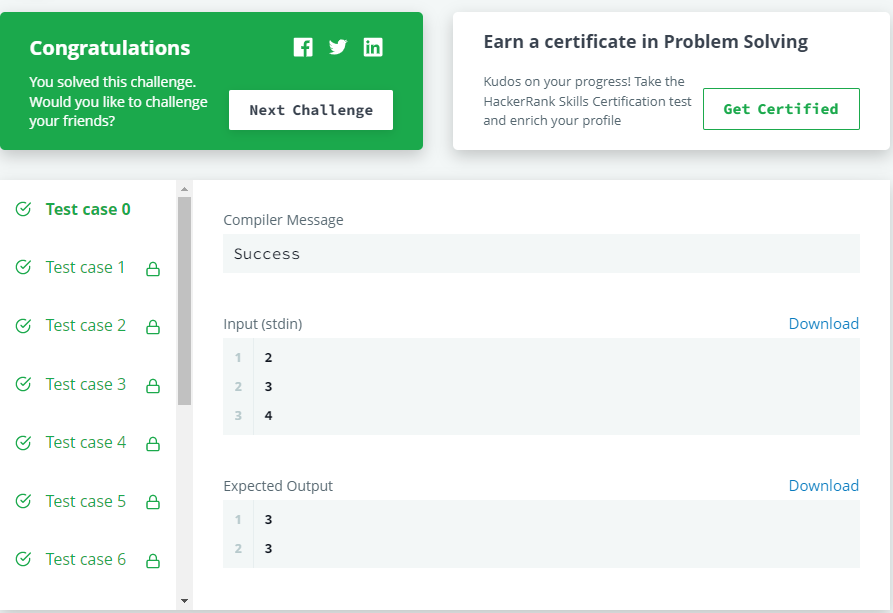
    }

  }

}

# TEST CASES:

****

****

**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. |  |  |  |
|  |  |  |  |