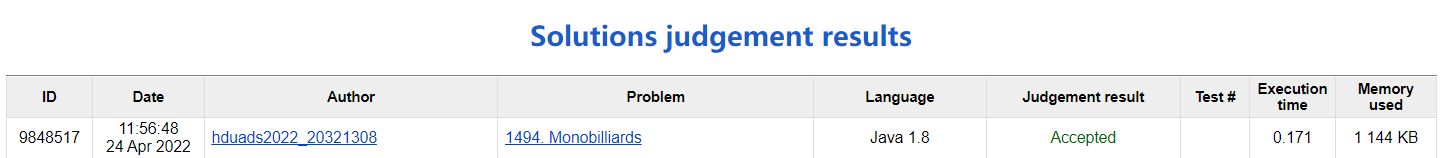
Laboratory work #5

Student: CAO Xinyang  
Student ID: 20321308  
Timus Name: hduads2022\_20321308

Mail: c.x\_yang@foxmail.com

Problem #1494

Screenshot from Timus:



Explanation of algorithm:

When the element at the top of the stack is equal to the number of elements removed, N, remove the element and continue; When the number of elements at the top of the stack is less than N, increasing elements are added to the top of the stack until the top element is equal to N; When the top element of the stack is greater than N, it is considered cheating.

Computational complexity of algorithm:

T(N) = O(N^2)

Source code:

import java.io.\*;

public class App {

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

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

PrintWriter out = new PrintWriter(new OutputStreamWriter(System.out));

in.nextToken();

int N = (int) in.nval;

int[] stack = new int[100001];

int current = 0;

int top = 0;

int count = 1;

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

in.nextToken();

current = (int) in.nval;

while (stack[top] != current && count <= N) {

top++;

stack[top] = count;

count++;

}

if (stack[top] == current && top != 0) {

top--;

}

}

if (top != 0) {

out.print("Cheater");

} else {

out.print("Not a proof");

}

out.flush();

}

}