Laboratory work # 5

Student: HU Riqian
Student ID: 20321114
Timus Name: hduads2022_20321114
Mail: jhlxhrq@163.com

Problem # 1494. Monobilliards

Screenshot from Timus:

9848491	11:32:12 24 Apr 2022	hduads2022_20321114	1494. Monobilliards	Java 1.8	Accepted		0.156	1 480 KB

Explanation of algorithm:

1. The problem, as I think, is designed like the data structure ‘stack’. What I need to do is to check out whether the way to take out the billiard balls is like popping out the stack.

2. So I design the algorithm to judge the process.

Computational complexity of algorithm:

$$O(N \log N)$$

Source code:

```
import java.io.*;

public class Monobilliards {
    public static void main(String[] args) throws IOException {
        new Monobilliards().run();
    }

    StreamTokenizer in;
    PrintWriter out;
    static int[] BALLS;
    static int[] TAKEN;
    static int SIZE;

    int nextInt() throws IOException {
        in.nextToken();
```

```
        return (int)in.nval;
    }

    void run() throws IOException {
        in = new StreamTokenizer(new BufferedReader(new
InputStreamReader(System.in)));
        out = new PrintWriter(System.out);
        prepareData();
        checkOut();
        out.println(checkOut() == 0 ? "Not a proof" : "Cheater");
        out.flush();
    }

    void prepareData() throws IOException {
        SIZE = nextInt();
        BALLS = new int[SIZE + 1];
        TAKEN = new int[SIZE + 1];
        for (int i = 0 ; i < SIZE; i++) {
            TAKEN[i] = nextInt();
        }
    }

    int checkOut() {
        int topBalls = 0;
        int topTaken = 0;
        for (int i = 0; i < SIZE; i++) {
            BALLS[++topBalls] = i + 1;
            while (topBalls != 0 && BALLS[topBalls] == TAKEN[topTaken]) {
                topTaken++;
                topBalls--;
            }
        }
        return topBalls;
    }
}
```