South African Computer Olympiad
Third Round Future Stars 2008
Day 1

Overview

Author(s)	**Ben Steenhuisen and Richard Baxter**	**Marco Gallotta and Mark Danoher**	**Richard Baxter and Julian Kenwood**
Problem	**cashier**	**cleaning**	**hanoi**
Source	cashier.java cashier.py cashier.c cashier.cpp cashier.pas cashier.hs	cleaning.java cleaning.py cleaning.c cleaning.cpp cleaning.pas cleaning.hs	hanoi.java hanoi.py hanoi.c hanoi.cpp hanoi.pas hanoi.hs
Input file	stdin	stdin	stdin
Output file	stdout	stdout	stdout
Time limit	1 second	1 second	1 second
Number of tests	10	20	10
Points per test	10	5	10
Detailed feedback	Yes	Yes	No
Total points	**100**	**100**	**100**

The maximum total score is 300 points.

Cashier

Ben Steenhuisen and Richard Baxter

Introduction

Sybil wants Basil to upgrade the reception's cashier, claiming it is old and cannot handle the amount of money going through it each day. Help Basil prove that their cashier is still good enough.

Task

The cashier only remembers the last K bills. After receiving each bill, it should print the total of the last K bills (since that is all it can remember). To help determine whether the cashier is working correctly, Basil has asked you to calculate the maximum such total it should be printing. In other words, the maximum sum of any K consecutive bills.

Example

The cashier has received $N = 5$ bills and can only remember the last $K = 3$. The following bills received were 9, 13, 2, 11 and 6. Table 1 lists the numbers the cashier remembers after receiving each bill and the total it should be printing.

Bill received	Cashier's memory			Total printed
9			9	9
13		9	13	22
2	9	13	2	24
11	13	2	11	26

Table 1: Numbers for the example.

After the cashier receives the third bill, it remembers all of them (9, 13 and 2) and should print a total of 24. It then receives the fourth bill of 11, forgets the first bill of 9 and should print a total of 26. The highest total the cashier should print is therefore 26.

Input (stdin)

The first line contains two space-separated integers, N and K. N represents the number of bills. K represents the number of bills that the cashier keeps in memory. The next N lines each hold a single integer: the value of a bill.

Sample input

```
4 3
9
13
2
11
```

Output (stdout)

Output a single integer: the largest sum found.

Sample output

```
26
```

Constraints

- $3 \leq K < N \leq 300\,000$

Additionally, in 50% of the test cases:

- $3 \leq K < N \leq 100\,000$

Time limit

1 second. Python: 10 seconds.

Detailed feedback

Detailed feedback is enabled for this problem.

Scoring

A correct solution will score 100% while an incorrect solution will score 0%.

Spring Cleaning

Marco Gallotta and Mark Danoher

Introduction

Fred the manic storekeeper has a large storeroom filled with piles of boxes of explosives that he has been unable to sell. Spring has come and Fred needs to clear out his storeroom so he has space for the next popular product. He will do this by detonating the boxes and he wants you to help him minimise the cost to do so.

Task

Due to the odd shape of his storeroom, Fred has stored the boxes in N piles, which have been arranged in a single straight line. Each time a box is detonated, it is destroyed and the explosion spreads sideways, destroying the top box of any neighbouring piles *of the same height* (all boxes are of identical size and shape). The explosion continues spreading in both directions, until it reaches a pile of a different height. For safety reasons, only the top box of a pile can be exploded (either by being detonated or by its neighbour exploding).

Fred will only consider his storeroom clear when all boxes have been destroyed. He would like you to calculate the minimum number of detonations required to achieve this so that he can budget accordingly. Fred isn't good at working with large numbers so he would like you to give him only the last six digits of the number of detonations, i.e. the remainder after division by 1 000 000.

Example

Suppose there are four piles, containing 1, 3, 2 and 5 boxes respectively. One way to destroy all the boxes in the minimum number of detonations would be:

- Detonate the fourth pile three times, destroying the top three boxes. (three detonations)
- Detonate the second pile once, destroying the top box. (one detonation)
- Detonate the fourth pile once, destroying the top box of the fourth pile and then the top boxes of the third and second piles. (one detonation)
- Detonate any pile, destroying the last box in that pile and then the last box in all of the other piles. (one detonation)

This would clear out the storeroom in six detonations.

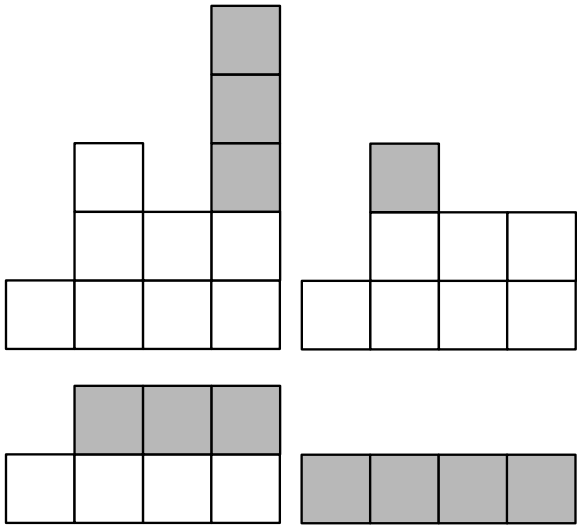

Figure 1: *The sequence of detonations. The grey boxes are to be exploded next. Note that the first step requires three detonations.*

Input (stdin)

The first line of input contains a single integer, N, the number of piles. The next N lines each contain a single integer, H_i, the number of boxes in the i^{th} pile.

Sample input

```
4
1
3
2
5
```

Output (stdout)

Output a single integer: the remainder when the minimum number of detonations is divided by 1 000 000.

Sample output

```
6
```

Constraints

- $1 \leq N \leq 750\,000$
- $0 \leq H_i \leq 1\,000\,000$ for all i

Additionally, in 70% of the test cases:

- $N \leq 200\,000$

Additionally, in 25% of the test cases:

- $N \leq 1\,000$
- $H_i \leq 1\,000$ for all i

Time limit

1 second. Python: 10 seconds.

Detailed feedback

Detailed feedback is enabled for this problem.

Scoring

A correct solution will score 100%, while an incorrect solution will score 0%.

Hanoi Reconstruction

Richard Baxter and Julian Kenwood

Introduction

The legend of the Towers of Hanoi says that a group of monks was given three pegs. The first peg had 64 discs, of different sizes, stacked from largest to smallest. The monks had to move all the discs to another peg, one at a time. However, they could only move a disc either onto an empty peg or onto a larger disc.

The optimal recursive solution to the puzzle is:

```
hanoi(NUM, FROM, HELPER, TO):
    if NUM == 1:
        Move disc from peg FROM to peg TO
    else:
        hanoi(NUM-1, FROM,   TO,     HELPER)
        hanoi(1,     FROM,   HELPER, TO)
        hanoi(NUM-1, HELPER, FROM,   TO)
hanoi(N, 1, 2, 3)
```

Task

The Guji tribe are attempting to solve a smaller version of the puzzle with N discs, *using the above solution.* After many long days of solving the puzzle, they took a well-deserved rest.

Upon returning to the puzzle, they discovered that they had been raided by their enemy, the Burji tribe, who moved all the discs back to the first peg. They really did not feel like starting all over again, but could not remember which discs were on which pegs. The only information they remembered was, T, the number of moves they had made, where a "move" is the act of taking the upper disk from one of the pegs and putting it onto another peg. They have asked you to determine on which peg each disc was before they took their rest, using N and T provided.

Example

The Guji tribe had made four moves ($T = 4$) in the puzzle with three discs before they were raided. Working from the start, with Disc 1 the largest, you determine the four moves were:

1. Move Disc 3 onto Peg 3
2. Move Disc 2 onto Peg 2
3. Move Disc 3 onto Peg 2
4. Move Disc 1 onto Peg 3

The three discs were therefore on pegs 3, 2 and 2 respectively. This example is illustrated in Figure 2.

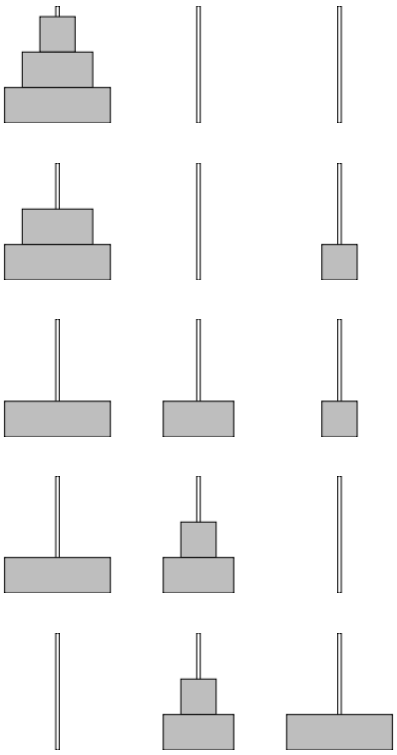

Figure 2: *The initial configuration and first four moves of the puzzle with $N = 3$ discs. The leftmost peg is Peg 1.*

Input (stdin)

The input contains two space-separated integers, N and T.

Sample input

```
3 4
```

Output (stdout)

Output N lines, with line i containing a single integer representing the current peg on which the i^{th} disc is on (where disc 1 is the largest and N the smallest). Each number should be one of:

- 1, if the disc is on the first peg
- 2, if the disc is on the middle (helper) peg

- 3, if the disc is on the last peg

Sample output

```
3
2
2
```

Constraints

- $1 \leq N \leq 20$
- $0 \leq T < 2^N$

Time limit

1 second. Python: 10 seconds.

Scoring

A correct solution will score 100%, while an incorrect solution will score 0%.