# **Tower Breakers**



Two players are playing a game of Tower Breakers! Player  ${\bf 1}$  always moves first, and both players always play optimally. The rules of the game are as follows:

- Initially there are n towers.
- Each tower is of height m.
- The players move in alternating turns.
- In each turn, a player can choose a tower of height x and reduce its height to y, where  $1 \le y < x$  and y evenly divides x.
- If the current player is unable to make a move, they lose the game.

Given the values of n and m, determine which player will win. If the first player wins, return 1. Otherwise, return 2.

Example. 
$$n=2$$
  $m=6$ 

There are  $\mathbf{2}$  towers, each  $\mathbf{6}$  units tall. Player  $\mathbf{1}$  has a choice of two moves:

- remove 3 pieces from a tower to leave 3 as  $6 \ modulo \ 3 = 0$
- remove  ${f 5}$  pieces to leave  ${f 1}$

Let Player 1 remove 3. Now the towers are 3 and 6 units tall.

Player 2 matches the move. Now the towers are both 3 units tall.

Now Player 1 has only one move.

Player 1 removes 2 pieces leaving 1. Towers are 1 and 2 units tall.

Player 2 matches again. Towers are both 1 unit tall.

Player 1 has no move and loses. Return 2.

### **Function Description**

Complete the *towerBreakers* function in the editor below.

towerBreakers has the following paramter(s):

- int n: the number of towers
- int m: the height of each tower

#### Returns

• *int:* the winner of the game

#### **Input Format**

The first line contains a single integer t, the number of test cases.

Each of the next t lines describes a test case in the form of 2 space-separated integers, n and m.

#### **Constraints**

- $1 \le t \le 100$
- $1 \le n, m \le 10^6$

### **Sample Input**

```
STDIN Function
-----2
t = 2
2 2 n = 2, m = 2
1 4 n = 1, m = 4
```

#### Sample Output

```
2
1
```

## **Explanation**

We'll refer to player  ${f 1}$  as  ${f P1}$  and player  ${f 2}$  as  ${f P2}$ 

In the first test case, P1 chooses one of the two towers and reduces it to 1. Then P2 reduces the remaining tower to a height of 1. As both towers now have height 1, P1 cannot make a move so P2 is the winner.

In the second test case, there is only one tower of height 4. P1 can reduce it to a height of either 1 or 2. P1 chooses 1 as both players always choose optimally. Because P2 has no possible move, P1 wins.