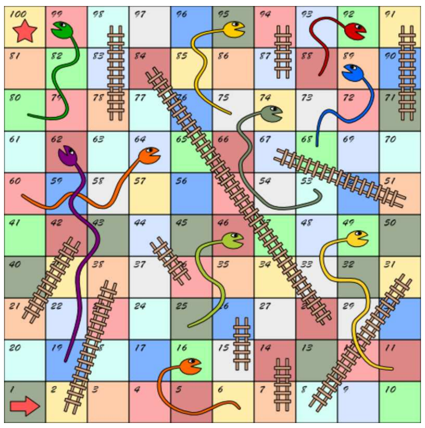
# Lazy Snakes and Ladders (2) P65875_en

Vintè Concurs de Programació de la UPC - Semifinal (2022-06-15)

The author of the previous problem is half a psycho, half a freak. He is well know for his lengthy and weird statements, although *a few* of his problems are not that hard. Consider for instance the following adaptation of an old problem of the same author.

The rules of *Snakes and Ladders* (see the board below) are simple:



- Players red and blue start with their counters on cell number 1, and take turns in rolling a six-sided die, with red going first.
- The counter for the current player moves forward the number of cells rolled in the die (e.g., rolling a 5 when on cell 4 takes the counter to 9).
- The goal is to reach the cell 100. An exact roll is needed: in case of excess, the counter *bounces* and moves the extra count backwards (e.g., rolling a 5 when on cell 97 takes the counter to 98).
- If the landing cell (after potential bouncing) is the bottom of a ladder, the counter is moved to its top, which will be a higher-numbered cell (e.g., rolling a 1 when on cell 1 takes the counter to 38). Nothing happens when the counter directly lands on a top.
- If the landing cell (after potential bouncing) is the head of a snake, the counter is moved to its tail, which will be a lower-numbered one (e.g., rolling a 3 when on cell 98 takes the counter to 80). Nothing happens when the counter directly lands on a tail.
- If the rolled number was six, the player keeps the turn; otherwise, it passes to the other player (irrespective of whether bouncing, snakes, or ladders were involved).

You must simulate several of these games using pseudo-random numbers. In particular, include the `<random>` library, and declare a global

```
mt19937 rng;
```

variable. Every game will be defined by a seed $s$. Just do

```
rng.seed(s);
```

to reset `rng` before every game. Afterwards, every time that you need the result $d$ of the next rolling of the die, use this code:

```
unsigned int r = rng();
int d = r%6 + 1;
```

For instance, with the initial seed 42, we get these values for $r$: 1608637542, 3421126067, 4083286876, 787846414, ... Therefore, the values for $d$ are 1, 6, 5, 5, ... In this game, red goes to 2 (and then to 38), blue goes to 7 (and then to 14), blue moves again (he got a 6) and goes to 19, red goes to 43, ..., and eventually blue wins.

## Input

Input consists of several games, each one defined by an integer seed $s$ between 1 and $10^9$.

## Output

For each game, print “RED” or “BLUE” depending on the winner.

## Sample input

```
42
1
999999999
1000000000
```

## Sample output

```
BLUE
BLUE
RED
BLUE
```

## Problem information

Author : Edgar Gonzàlez
Generation : 2024-05-02 21:28:54