Problem 181: Plink

Difficulty: Hard

Author: Brett Reynolds, Annapolis Junction, Maryland, United States

Originally Published: Code Quest 2022

Problem Background

Plink is a common carnival game in which a player drops a ball or disc into a slot at the top of a board filled with pegs arranged in offset rows. The ball rolls down the board, bouncing off the pegs as it drops, causing it to change course as it falls. Eventually the ball will fall into one of several buckets positioned at the bottom of the board. Carnival workers assign each bucket a point value, and the goal of the game is to try to get the ball to fall into the bucket worth the most points.

While targeting a specific bucket is the goal of the game, actually targeting a specific bucket is very difficult. The bouncing of the ball causes it to move almost randomly; each time the ball hits a peg, it has a nearly-equal chance to move to either the right or left side of the peg. As a result, the buckets in the middle of the board tend to catch the ball more often, because there are more paths leading to them, and thus a higher chance that the ball will land there. Buckets near the edges have fewer paths, and so catch the ball less frequently. As a result, the buckets near the edges tend to have a higher value than those in the middle.

You're working with a gaming company to test a new form of the game of Plink, a version that requires more skill and eliminates some of the randomness. There are three key differences to this version of the game:

- The ball can only be started in the middle of the board, rather than anywhere across the top.
- The round pegs have been replaced with flippers that can be tilted left or right by the player, allowing them to direct the course of the ball.
- Each peg has a point value; as the ball falls, the player adds the value of each peg the ball hits to their score. This is then added to the value of the bucket in which the ball landed to determine the player's final score.

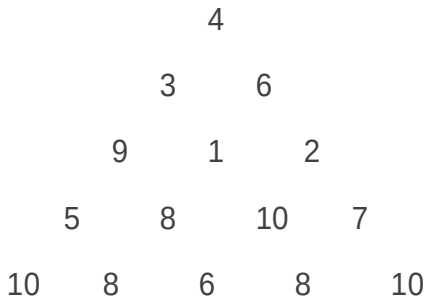
As a result of the changes in format, the player can manipulate the path of the ball to maximize their score. Aiming for a high-value bucket on the edge of the board may not earn the best score, if the only path leading to it contains only low-value pegs. In order to get a high score, a player must find the optimal path for the ball as it falls through the board.

Problem Description

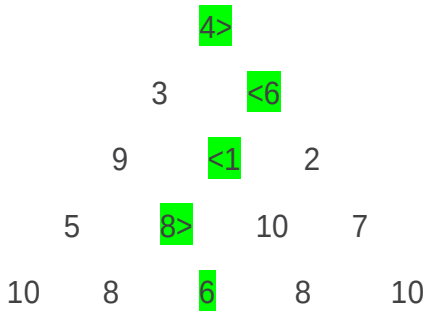
You must design a program that can find the optimal path for a given Plink board; that is, the path that is worth the greatest number of points given the specific arrangement of point values and bucket

values. Your path will be expressed as a series of letters - L or R - indicating if the ball should move to the left or right, respectively, of each flipper it hits as it falls. The ball will only hit one flipper on each level of the board, so the number of directions given will be equal to the number of levels in the board.

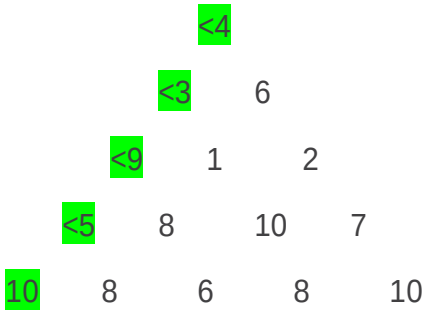
For example, consider the board below. Each number in the triangle represents the point value of the flipper in that position. The bold numbers along the bottom represent the values of the buckets at the bottom of the board.

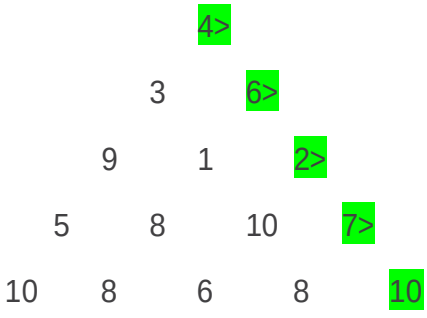


Despite the small size of this board, there are 16 possible paths through it. The path "RLLR," for example, results in a score of 25:

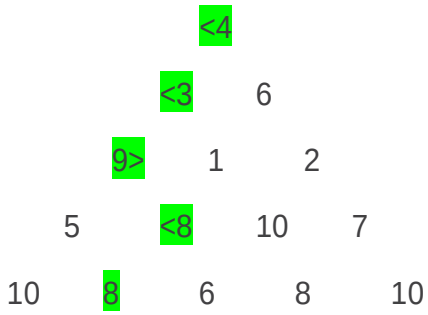


This is not the optimal path for this board. It lands in a low-value bucket in the middle of the board, and also hits a flipper worth only one point. In traditional Plink, it's best to aim for the edges of the board; here, that would be achieved with the paths "LLLL" and "RRRR."





The all-left path earns a score of 31 points, and the all-right path earns a score of 29 points. These are both better than the path we looked at earlier. However, neither is the optimal path. There is one path that earns 32 points; LLRL:



Even though we lose two points by landing in a different bucket, we gain three points by aiming for the 8-point paddle in the middle, rather than staying on the edge.

Your task is to write a program that can calculate the path that earns the maximum possible score for a given Plink board. Be warned, however; these examples covered a very small board consisting of only four rows of paddles. Your team is hoping to design much larger boards, in order to increase the game's difficulty. Each additional row of paddles increases the number of possible solutions by another power of two. Your program must be certain it's found the single best path out of all possibilities.

As a reminder, your program must complete execution within two minutes, or it will be marked as incorrect (even if it would have eventually generated the correct response).

Sample Input

The first line of your program's input, received from the standard input channel, will contain a positive integer representing the number of test cases. Each test case will include:

- A line containing a positive integer, X, representing the number of rows of paddles in the Plink board.
- X lines, containing the integer point values of the paddles in the corresponding row of the table in order from left to right. Values are separated by spaces, and lines will contain a number of values equal to the row number of the board (the first row is row 1, and will contain 1 value).
- A line containing X+1 integers separated by spaces representing the point values of the buckets at the bottom of the board in order from left to right.

```
1
4
4
3 6
9 1 2
5 8 10 7
10 8 6 8 10
```

Sample Output

For each test case, your program must print a line containing:

- A string with X uppercase letters L or R, representing the optimal path through the given Plink board, as described above.
- A space.
- An equals sign.
- A space.
- An integer representing the number of points obtained from that path.

```
LLRL = 32
```