Understand

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

This is an interactive problem.

In the simplified version of the game "Understand," you find yourself in a 256×256 grid. There are n items within this grid whose precise positions are yet unknown. It is possible that multiple items are in the same grid cell.

Your task is to determine the locations of each item through a series of interactive queries, with a limit of **no more than** 16 **queries**. Each query allows you to draw a **simple path** on the grid, and the interactor will respond with a binary string of length n, representing whether each item is on that path.

It is guaranteed that the interactor is non-adaptive, meaning the positions of the items are already determined before you start querying.

Input

The first line of input contains an integer n ($1 \leq n \leq 10\,000$), denoting the number of items in the grid.

Interaction Protocol

The interaction begins after reading the integer n.

Then, make at most 16 queries. To make a query: output "? x_0 y_0 k s" in a line. Here, $1 \leq x_0, y_0 \leq 256$ represents that the path starts at the x_0-th row from top to bottom and the y_0-th column from left to right. $1 \leq k \leq 256 \times 256$ represents the length of the simple path. s is a string of length $k-1$ consisting of characters 'L,' 'R,'' U', and 'D.' The i-th character represents the direction of movement from the i-th cell to the $(i+1)$-th cell along the path, where

- 'L' means moving from (x, y) to $(x, y-1)$,
- 'R' means moving from (x, y) to $(x, y+1)$,
- 'U' means moving from (x, y) to $(x-1, y)$,
- 'D' means moving from (x, y) to $(x+1, y)$.

If $k = 1$, you can simply output "? x_0 y_0 k" and may not output the additional blank space (It would be OK doing so, though).

Here, the path you output must be **simple**, meaning your path should not pass the same grid more than once. Also, the path must lie in the 256×256 grid, meaning each grid (x, y) in the path must satisfy $1 \leq x, y \leq 256$.

After you output the query in the correct format, read in a binary string t of length n, where the i-th ($1 \leq i \leq n$) character of t is 1 if the i-th item is on your path, and the i-th ($1 \leq i \leq n$) character of t is 0 otherwise. If your query is invalid or you have made more than 16 queries, you will receive a single character '$-$' instead. Upon reading it, your program must terminate immediately to receive a "Wrong Answer" verdict, or you could get any possible verdict.

At any point of the interaction, if you want to guess the locations of each item, output "! x_1 y_1 x_2 $y_2 \cdots x_n$ y_n" in a line, where (x_i, y_i) represents the location of the i-th item in your guess. **This guess does not count as a query.** After doing that, you should terminate your program immediately.

After printing each query and the answer, do not forget to output the end of the line and flush the output. Otherwise, you may not get the correct verdict. To flush the output, use:

- `fflush(stdout)` or `cout.flush()` in C++;
- `System.out.flush()` in Java;
- `stdout.flush()` in Python;

Example

standard input	standard output
4 0101 1111 1100 0001	 ? 3 1 9 RRRUULLD ? 2 1 4 RRR ? 2 1 6 URDDR ? 2 4 1 ! 2 1 2 2 2 3 2 4

Note

Note that additional empty lines in the sample test are for better understanding, and you should not output any additional empty lines in your program.

Here, we provide graphical illustrations of the interaction in the sample test. In the picture below, we only show the upper-left 4×4 part of the board, but the actual size of the board is 256×256. The locations of all four items are labeled with stars. The starting point of each query path is labeled with a square, and the result is shown below the path (filled circles represent 1, and unfilled circles represent 0).