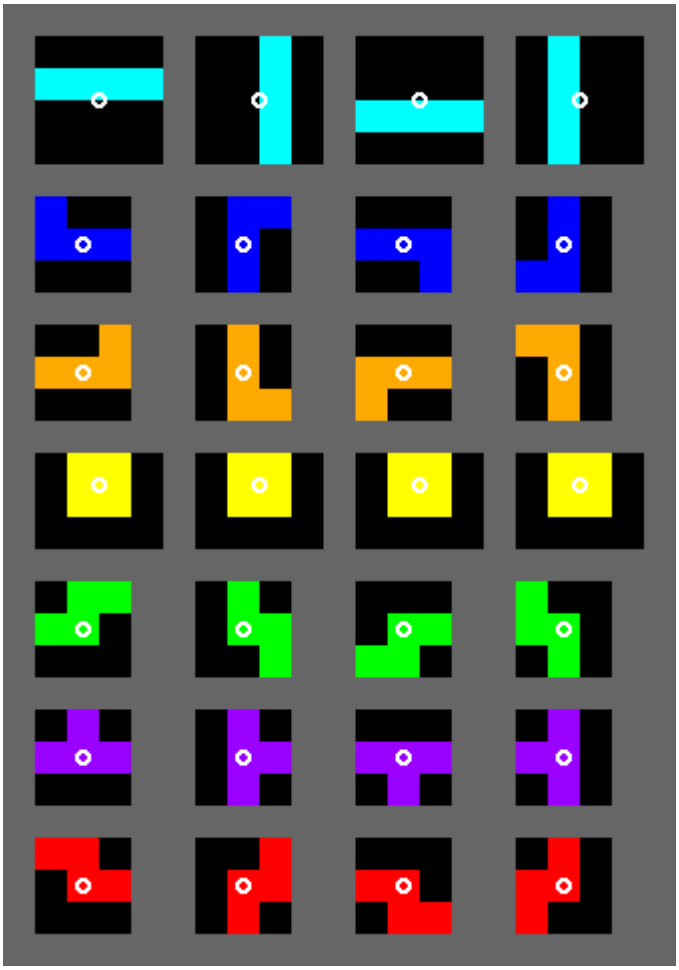
tetrart

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

You've hacked into a special version of Tetris that lets you freely choose which piece comes next. Your goal is no longer survival — it's now to create pixel art.

- The game field is a $h \times w$ rectangular grid, initially empty. Top left is $(0, 0)$ and bottom right is (h, w).**It is guaranteed that w is odd.**
- **Blocks can stack up to 20 rows above the visible grid before triggering a game over.**
- **There are 7 tetrominos, each made of 4 connected blocks, named after their distinctive shapes: I, J, L, O, S, T and Z. The figure below illustrates the 4 orientations of the 7 tetrominoes. The first column displays the initial orientation, and each subsequent column is obtained by rotating the previous column 90 degrees clockwise around the tetrominoes' center (indicated by the white circles).**



- **In each move, you can specify a triple (t, x, d), where t is an uppercase letter and x, d are integers. Then a tetromino t will appear, rotated $d \times 90°$ counterclockwise from its initial orientation, and its center coordinates, when rounded to nearest integers, are initially $(-\infty, x)$. The tetromino will drop down until it hits the bottom or any other block in the field and turn into blocks.**

- **Line Clear: When a horizontal row is completely filled with blocks, it disappears. All blocks above the cleared row shift down.**

You are given a grid the same size as the game field, representing a pixel art pattern. Your task is to construct a sequence of moves so that after performing them, all the '#'s in the grid are occupied by a block, and '.'s are empty.

Input

Each test contains multiple test cases.

The first line contains the number of test cases T ($1 \le T \le 10^4$).

The first line of each test case contains two integers, $1 \le h \le 100$ and $w \le 100$, where w is guaranteed to be odd. The following h lines each contain a string of length w, consisting of '.' and '#', representing the target pattern.

It is guaranteed that the sum of hw over all test cases does not exceed 10^5.

Output

For each test case:

If no solution exists, output -1. Otherwise, the first line should contain an integer $0 \le k \le 10hw$, denoting the number of operations in your construction. The next k lines each contain an uppercase character t, followed by two integers x and $0 \le d \le 3$, representing a move (t, x, d).

It can be proven that if a solution exists, there is one with no more than $10hw$ operations.

Example

standard input	standard output
2 3 5 ....#.. .###. 3 5 ..#.. ..... #...#	1 T 3 0 -1

Note

A checker is provided to help you verify the correctness of your solution locally. (hint: You may copy some code snippets from 'checker.cpp' for writing your own solution.)

`Compilation`

Place checker.cpp and testlib.h in the same directory, then compile with:

```
g++ checker.cpp -o checker
```

To enable verbose mode (which prints the game field after each move), add the -DVERBOSE flag:

```
g++ checker.cpp -o checker -DVERBOSE
```

`Usage`

Run the checker with the following command:

```
./checker <input-file> <output-file> <answer-file>
```