Problem: Bugs, bugs everywhere

Input file:	`standard input`
Output file:	`standard output`
Time limit:	10 seconds for 10 moves
Memory limit:	64 megabytes

This problem name is devoted to functioning of the testing system during previous game programming contest

Inspired by flash game Bug War 2.

http://www.bubblebox.com/play/adventure/1757.htm

In this problem you are to manage a bugs colone, struggling to survive on 15×15 field. Some celll of the field contain hives. Every hive can be neutral or can belong to one of two players. In every cell of the field there can be any amount of bugs (but only bugs belonging to one player). Some cells of the field contain rocks.

In his turn player can order several (possibly, all) his bugs to move in one of four directions: up, down, left or right. Main game loop looks like following:

1. First player orders his bugs to move.
2. Orders are executed and field is updated (more about this later).
3. Second player orders his bugs to move.
4. Orders are executed and field is updated (more about this later).

How orders are executed and field is updated:

1. All bugs, which are ordered to move, move in specified directions (this can lead that some field cells will contain both players' bugs simultaneously). Orders which result in bug moving outside the field or to the cell with rock in it, are ignored).
2. In every hive (except neutral hives) appears one more bug, belonging to the player, which owns the hive.
3. In every cell of the field, which contains bugs of both players, battle happens. All battles are independent of each other, so we will consider battle in one cell. Let first player have a bugs in this cell and second player have b bugs. One of two events happen:
   (a) one bug of the second player dies (this happen with probability $\frac{a}{a+b}$)
   (b) one bug of the first player dies (this happen with probability $\frac{b}{a+b}$)

   These events happen until there's only one player's bugs in the cell. You can assume, that battle happens with infinite speed (so before every player's move there will be only one player's bugs in every cell).
4. Every hive begins to belong to player, who has bugs in the cell with the hive. If there's no bugs in that cell, hive's ownership does not change.

Players score points to determine a winner: one point for every produced bug and one point for each opponent's bug destroyed. Player, who has more points is declared a winner. If both players score the same amount of points, game is declared a draw.

Game ends when one of the players is out of bugs and hives of after 1000 moves (so, 500 moves for every player).

Input

You will receive the map before every move. Map consists of 15 lines with 15 characters in each (not including newline). Empty cells are denoted by `'.'`, first player's hives are denoted by `'1'`, second player's hives are denoted by `'2'`, neutral hives are denoted by `'H'`, rocks are denoted by `'#'`. The map represents hives' ownership before your move.

You can determine if you are first or second player via command line option passed to your program (it will be `"1"` or `"2"`).

After the map you will receive current field state: 15 lines with 15 numbers in it. Every number a_{ij} corresponds to one field cell. If $a_{ij} > 0$, the cell contains a_{ij} your bugs, if $a_{ij} < 0$, then the cell contains $-a_{ij}$ your opponent's bugs. If $a_{ij} = 0$, there are no bugs in the cell.

Output

You should output a list of orders at every move. The list should contain a number k and k lines with orders after that. The list should contain no more than $15 * 15 * 4 = 900$ order. Every order should be denoted by four numbers $i_p j_p d_p n_p$. This means that n_p bugs from the cell (i_p, j_p) should move into directio d_p. Directions are denoted by following numbers:

- 0 — left (decreasing j_p)
- 1 — up (decreasing i_p)
- 2 — right (increasing j_p)
- 3 — down (increasing i_p)

Output an empty line after the list of order. Before processing your orders, checking program will verify that every cell (i_p, j_p) contains not less bugs, than sum of amounts of bugs in all orders from that cell. If this is wrong for the cell, all orders from that cell will be ignored.

Please remember, that your program should not terminate by itself, so make a dummy read operation after last move.

Example shows output for one move:

Examples

standard input

```
1...H.....H...H
...............
...............
...............
H...H.....H...H
...............
...............
.......H.......
...............
...............
H...H.....H...H
...............
...............
...............
H...H.....H...2
10 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 -10
```

standard output

```
2
0 0 2 5
0 0 3 5
```