Czech ACM Student Chapter
CTU in Prague, Dept. of Computer Science and Engineering
VŠB TU Ostrava, FEI, Dept. of Computer Science
SUT Bratislava, FEI, Dept. of Computer Science and Engineering

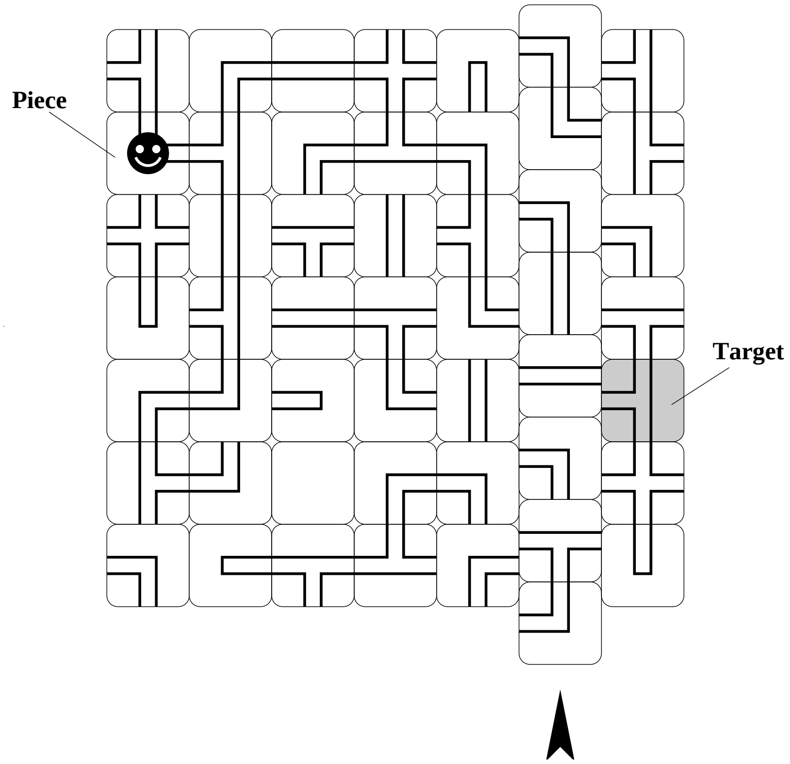
CTU Open Contest 2002

Interesting Maze Game

`maze.p`, `maze.c`, `maze.C`

The Police President has recently bought a new game — the famous Ravensburger's aMAZEing Labyrinth. Now, he is really keen on it, he spends any free time playing this game. While we want the Police President during the Summit to perform much more important decisions, we need a program that would substitute him in playing the game.

The game is played on the field of 7×7 squares with equal-sized cards lying on each square. Various path patterns are drawn on the cards, these paths join arbitrary subset of the four edges of a single square. The patterns can form longer paths leading through the whole game field. When following these paths, it is possible to move from a square to its neighbouring square, if both squares contain the path pattern leading to their common edge. It is impossible to travel between squares diagonally. See the picture for a better idea about the game appearance.



In the beginning of the move, the player has one game piece on some of the square cards and his/her goal is to move the piece to some other square card (target) following the valid paths. Before the walk, the player alters the maze state by inserting one extra square card into it.

The extra card can only be inserted to the position at the field margin. The insertion causes the whole row or column of cards to be shifted one position further, which makes another card to fall out at the other end of the game field. (This card becomes a new extra card for the next move, but we will care of a single move only in this problem.)

Since the cards with both coordinates odd are stuck firmly to the playing desk, only the even rows and columns can be shifted. Thus, the extra card can be inserted into an even row or column only. If we number rows and columns with numbers 1 to 7, there are 12 possible positions where the new card can be inserted: $(1,2)$, $(1,4)$, $(1,6)$, $(7,2)$, $(7,4)$, $(7,6)$, $(2,1)$, $(4,1)$, $(6,1)$, $(2,7)$, $(4,7)$, and $(6,7)$. For instance, insertion into the position $(7,6)$ causes the following shift:

$$(7,6) \to (6,6) \to (5,6) \to (4,6) \to (3,6) \to (2,6) \to (1,6)$$

The extra card comes to the position $(7,6)$ and the card formerly being at the position $(1,6)$ is removed from the game field for the rest of the move.

Before insertion, the extra card can be rotated to any of the four possible directions. No other card in the game can be rotated. This makes the total maximal number of 48 possible moves (if the extra card is asymmetric).

Another important rule considers the case when the target card or the card with the player's piece appears in the row or column being shifted. In such case, the position of the piece or the target is shifted too. This makes it possible to move the target to some more appropriate place.

Note that if the target is shifted away from the game (the target card falls out from the game), it is no more possible to reach it in the same move — the piece cannot leave the game field. On the other hand, if the game piece is located on the card which is moved away from the game field, the piece position is "wrapped" to the opposite end of the field, i.e., to the just inserted card.

Therefore, a valid move consists of two parts: insertion of the extra card into the game (this action must always be made) and walking the path of an arbitrary length (including zero, i.e., staying on the same square). Your task is to determine, whether it is possible to reach the target in a single move. In other words, if it is possible to insert the extra card into the game and then to walk to the target position.

Input Specification

The input consists of several game descriptions. The first line of each description contains four integer numbers $R1$, $C1$, $R2$, and $C2$ separated with a space, $1 \le R1, C1, R2, C2 \le 7$. $(R1, C1)$ is the position (row and column) of the game piece, $(R2, C2)$ is the position of the target. Note that these positions can sometimes be shifted during the move, as specified above. After these numbers, there is one blank line.

The next three lines describe the first row of the game field. Each of these lines contains 27 characters: three for the card in the first column, one space, three characters for the card in the second column, etc. Thus, every card is represented by a square of nine (3×3) characters. The middle one of these nine characters is always capital letter "`O`". The four characters in the corners are always dots ("`.`"). The both left and right characters are either a dot ("`.`") or a dash ("`-`"). Dashes mean the path pattern leading to the left or right edge. The top and bottom characters are either a dot or a pipe ("`|`"). The pipe means the path pattern leading to the corresponding edge.

After the first row, there is one blank line and three other lines describing the second row. Then follow one other blank line and three lines for the third row, etc. After the seventh row, there is a blank line and three other lines containing exactly three characters each. This is the description of the extra card, given in the same way as the cards in the field.

The input is terminated by a line containing four zeros instead of piece and target coordinates.

Output Specification

For each game, output a single line. If it is possible to insert the extra card in such a way that there is a path from the game piece to the target, print the sentence "`You can win in one move.`". Otherwise, print the sentence "`Bad luck!`".

Sample Input

```
1 1 7 7

... ... ... .|. ... ... ...
-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.
.|.

1 1 7 7

... ... ... ... ... ... ...
-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
```

Output for Sample Input

```
You can win in one move.
Bad luck!
```