SWE2004: Principles in Programming(Spring 2014)

Programming Lab #1

Due-date: March 13, 11:59PM

Description

Your task is to write a program that reads a chessboard configuration and identifies whether a king is under attack (in check). A king is in check if it is on square which can be taken by the opponent on his next move.

White pieces will be represented by uppercase letters, and black pieces by lowercase letters. The white side will always be on the bottom of the board, with the black side always on the top.

For those unfamiliar with chess, here are the movements of each piece:

Pawn (p **or** P**):** can only move straight ahead, one square at a time. However, it takes pieces diagonally, and that is what concerns you in this problem.
Knight (n **or** N**)** : has an L-shaped movement shown below. It is the only piece that can jump over other pieces.
Bishop (b **or** B**)** : can move any number of squares diagonally, either forward or backward.
Rook (r **or** R**)** : can move any number of squares vertically or horizontally, either forward or backward.
Queen (q **or** Q**)** : can move any number of squares in any direction (diagonally, horizontally, or vertically) either forward or backward.
King (k **or** K**)** : can move one square at a time in any direction (diagonally, horizontally, or vertically) either forward or backward.

Movement examples are shown below, where "*" indicates the positions where the piece can capture another piece:

```
Pawn        Rook        Bishop      Queen       King        Knight
........    ...*....    .......*    ...*...*    ........    ........
........    ...*....    *.....*.    *..*..*.    ........    ........
........    ...*....    .*...*..    .*.*.*..    ........    ..*.*...
........    ...*....    ..*.*...    ..***...    ..***...    .*...*..
...p....    ***r****    ...b....    ***q****    ..*k*...    ...n....
..*.*...    ...*....    ..*.*...    ..***...    ..***...    .*...*..
........    ...*....    .*...*..    .*.*.*..    ........    ..*.*...
........    ...*....    *.....*.    *..*..*.    ........    ........
```

Remember that the knight is the only piece that can jump over other pieces. The pawn movement will depend on its side. If it is a black pawn, it can only move one square diagonally down the board. If it is a white pawn, it can only move one square diagonally up the board. The example above is a black pawn, described by a lowercase "p". We use "*move*" to indicate the squares where the pawn can capture another piece.

Input

There will be an arbitrary number of board configurations in the input, each consisting of eight lines of eight characters each. A "." denotes an empty square, while upperand lowercase letters represent the pieces as defined above. There will be no invalid characters and no configurations where both kings are in check. You must read until you find an empty board consisting only of "." characters, which should not be processed. There will be an empty line between each pair of board configurations. All boards, except for the empty one, will contain exactly one white king and one black king.

Output

For each board configuration read you must output one of the following answers:
Game #*d*: white king is in check.
Game #*d*: black king is in check.
Game #*d*: no king is in check.
where *d* stands for the game number starting from 1.

Sample Iuput

```
..k.....
PPP·PPPP
........
.R...B..
........
........
PPPPPPPP
K.......

rnbqk.nr
PPP··PPP
....p...
...p....
.bPP....
.....N..
PP..PPPP
RNBQKB.R
```

Sample Output

Game #1: black king is in check.

Game #2: white king is in check.