Problem 161: Checkmate

Difficulty: Hard

Author: Louis Ronat, Denver, Colorado, United States

Originally Published: Code Quest 2021

Problem Background

Chess is an ancient game that still remains popular today. Since the early 2000's, however, the best chess players aren't human; they're algorithms programmed into computers. In fact, there are now multiple tournaments whose competitors are various "chess engines" - software programs designed to output valid chess moves in pursuit of a checkmate.

Problem Description

Your goal is to write an application that reads an input representing a chessboard and outputs whether the player set to move is in checkmate. The player to move can be deduced by determining which player is in check - if neither player is in check, no checkmate exists.

While chess is a well-known game, it rules are somewhat complicated, and so the ones relevant to this problem are reiterated below. If you are familiar with the game already, you can skip most of the rest of this description; however, make sure to take note of which letters are used to represent which pieces.

There are six different kinds of pieces in chess, all of which move in different ways. In this problem, they will be represented on the chessboard with upper or lowercase letters, as shown later in this section. The overall goal of the game is to force your opponent into a situation where their king is in danger, and they have no legal way to protect it. When a king is at risk of being captured, it's said to be "in check;" when no options exist to protect it, it's called "checkmate."

The six types of pieces and their rules of movement are as follows:

- P or p - Pawn
    - Can only move onto the space directly in front of it, with three relevant exceptions, explained below
    - A pawn cannot move forward if the space in front of it is occupied by another piece (of either color)
    - If an opposing piece is in either space diagonally ahead of a pawn, the pawn may move onto that space to capture that piece
    - A pawn in its starting position (second row from the player's side of the board) may move two spaces forward, if both spaces are unoccupied
- R or r - Rook

© 2021 Lockheed Martin Corporation. All Rights Reserved.

  - Can move any number of spaces horizontally or vertically
  - Cannot move through pieces of the same color
  - Can capture any piece of the opposing color by moving onto its space
- N or n - Knight
  - Moves two spaces either vertically or horizontally, then one space along the other axis (e.g. two up then one left, or two right then one down, etc.)
  - Can move through pieces of either color
  - Can capture any piece of the opposing color by moving onto its space
- B or b - Bishop
  - Can move any number of spaces diagonally
  - Cannot move through pieces of the same color
  - Can capture any piece of the opposing color by moving onto its space
- Q or q - Queen
  - Can move any number of spaces in any direction - horizontally, vertically, or diagonally
  - Cannot move through pieces of the same color
  - Can capture any piece of the opposing color by moving onto its space
- K or k - King
  - Can move one space in any direction - horizontally, vertically, or diagonally
  - Cannot move onto a space occupied by a piece of the same color
  - Can capture any piece of the opposing color by moving onto its space

The king is a player's most important piece. When a player's king is at risk of being captured by the opponent, it is "in check." The player with a king in check must immediately make a move to protect their king, either by:

- Moving the king to a safe position
- Moving a piece between the king and the threatening piece, blocking the attack (note that since knights can move through other pieces, this is not an option when the king is being threatened by a knight)
- Capturing the threatening piece

Neither player may make any move that puts their own king in check (or allows their king to remain in check). If a player in check is unable to make any move to protect their king, that causes a "checkmate" and they lose the game, since their king would be captured on the opponent's next turn.

Sample Input

The first line of your program's input, received from the standard input channel, will contain a positive integer representing the number of test cases. Each test case will include 8 lines representing the layout of a chessboard during a game of chess. Pieces are represented with uppercase or lowercase letters as shown above; uppercase letters represent white pieces, and lowercase letters

© 2021 Lockheed Martin Corporation. All Rights Reserved.

represent black pieces. Unoccupied spaces are represented by a period. White's position is at the bottom of each board; Black's position is at the top of the board.

```
3
..k.....
.PQP....
........
........
........
........
........
..R....K
........
k.......
........
........
......Q.
.....bnr
......PP
.......K
......k.
........
..rr....
.b......
........
........
........
..RKR...
```

Sample Output

For each test case, if a checkmate exists, your program must print a line containing the word "CHECKMATE" and the winning color. If no checkmate exists (neither player is in check, or the player in check is able to move), print NO CHECKMATE instead.

```
CHECKMATE WHITE
NO CHECKMATE
CHECKMATE BLACK
```

© 2021 Lockheed Martin Corporation. All Rights Reserved.