4148 Rank and File

Programs and algorithms for playing the game of chess have been around as long as computers themselves, with the first chess playing program being developed in the 50's by Alan Turing. Computer chess has come a long way since then, and in 1997 IBM's Deep Blue defeated chess grandmaster Garry Kasparov. One thing all these chess programs have in common though is a need to determine when the winning move, or *checkmate*, is reached. Your goal for this problem is to implement an algorithm such that, given the current layout of the chessboard, it will detect if a checkmate has occurred during that turn.

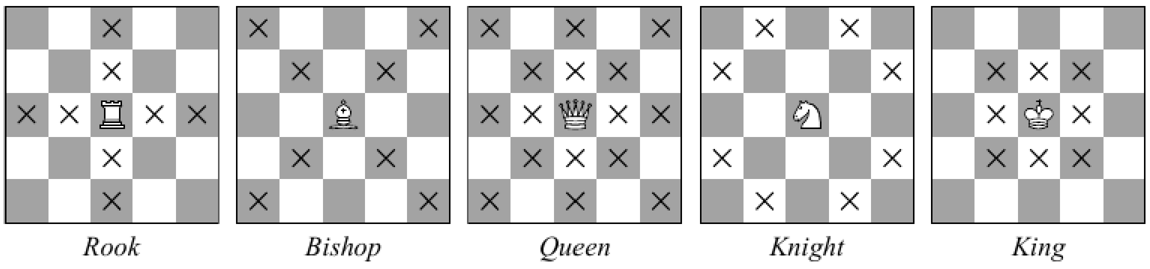
Chess is played on a board divided into a 8×8 grid of 64 squares. On a real chessboard, the 64 squares have alternating light and dark colors. For this problem the individual square colors are irrelevant and the entire board is simply treated as a uniform 8x8 grid.

The game is played by two opposing sides, white and black, with each side controlling up to six kinds of pieces: a *king*, *queen*, *rook*, *bishop*, *knight*, and one or more *pawns*. For simplicity's sake, however, this problem will only consider the first five and not make any use of the pawn chess piece. The two players take turns moving one piece at a time on every turn. It is up to each player to decide which piece they wish to move on their turn, but it is not possible for a player to "skip" or "pass"; each player **must** move one of their pieces in some fashion.

Each kind of chess piece moves in a distinct way as explained in the list below, and *Figure 1* gives an example using an X to show each square that a chess piece can move to.

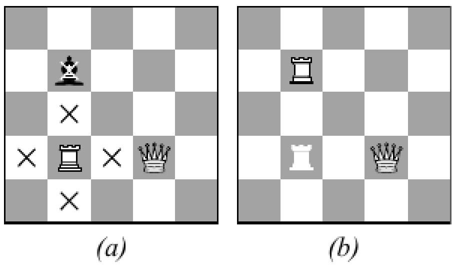
- *Rook*: The rook moves in a straight line by any number of squares in any of the four cardinal (horizontal and vertical) directions.
- *Bishop*: The bishop moves in a line by any number of squares in any of the four diagonal directions.
- *Queen*: The queen can move in a line by any number of squares in any of the eight cardinal and diagonal directions. As such, it is considered to be the most powerful piece in the game.
- *Knight*: The knight always moves by "jumping" two squares in one cardinal direction and one square in a direction perpendicular to the first. There are 8 possible squares that a knight can move to from a given position, and these are shown in the figure below.
- *King*: The king can move in any of the eight cardinal and diagonal directions but by one square only. Put another way, the king can only move into the eight immediately adjacent squares. As such, this makes the king one of the weakest pieces on the board.

Figure 1: Valid moves for each kind of chess piece



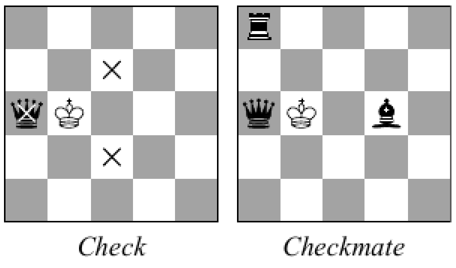
On every turn, a chess piece may be moved either into a vacant square or into a square already occupied by an opposing piece. In the latter case, the opposing chess piece is said to be *captured* and is permanently removed from the game. However, a chess piece **may not** move into a square already occupied by another friendly piece, because each square can be occupied by at most one piece at a time. Most chess pieces move by "sliding" across vacant squares on the board. In other words, any other chess piece (be it friendly or foe) in the path of the moving piece will block any further movement of that piece. The only exception to this rule is the knight which "jumps" directly to its final destination square, and therefore its movement cannot be blocked by any surrounding pieces between it and the destination square. See *Figure 2* for an example: image (a) shows a white rook's movement blocked by two other pieces, and image (b) shows the same white rook capturing a black bishop (the rook's previous position before the capture is shown as a white outline).

Figure 2: (a) rook's movement blocked by other pieces; (b) rook capturing the black bishop



Although the king may be one of the weakest pieces on the board, it is also the most important. A king cannot be captured directly by another piece but it can be threatened. When the king is threatened, or *checked*, it means that on the **next** turn, the opposing side would be able to capture the king. Putting a king in check forces the checked side to defend their king by moving the king out of harm's way, blocking the threat with another piece, or capturing the threatening piece. If the checked player has no valid move that they can make to protect the king, then the king is in fact *checkmated*, and that player has lost the game. Also note that a player may never put their own king in check. *Figure 3* shows the difference between a check and checkmate. In image (a), the imminent capture of the white king can still be avoided by either moving the king out of the way or by capturing the black queen. Image (b) shows an example of checkmate, where no valid moves exist for the white side that would eliminate the threat to the king.

Figure 3: Example of white king in (a) check and (b) checkmate



The rules laid out above are the only ones that should be considered for the purposes of this problem. Any other special rules or moves present in a real game of chess, such as *castling*, are **not** considered valid in the context of this problem. For the chessboard layout in each data set, you may assume it will contain exactly one white king and exactly one black king, although either side may have any number of rooks, bishops, queens, and knights.

Input

Input to this problem will begin with a line containing a single integer D $(1 \leq D \leq 100)$ indicating the number of data sets. Each data set consists of the following components:

- A line containing a single lower case '`w`' or upper case '`B`' indicating which side is to be analyzed for a check or checkmate (and which side is about to move on this turn), with '`w`' specifying the white side and '`B`' specifying black.
- A series of eight lines, with each line containing eight characters. These lines specify the state of the chessboard to be analyzed in this data set. The characters used on these eight lines are:
  - A '`.`' (period) to indicate an empty square
  - The lower case letters '`r`', '`b`', '`q`', '`n`', and '`k`', to respectively indicate the **white** side's rook, bishop, queen, knight, and king pieces
  - The capital letters '`R`', '`B`', '`Q`', '`N`', and '`K`', to indicate the same respective pieces but for the **black** side

Output

For each data set in the input, print a single line. Begin the line with either '`WHITE IS `' or '`BLACK IS `' depending on which side was analyzed in the data set. Finally, complete the line with '`CHECKED`' or '`CHECKMATED`' if either is detected, or complete the line with '`SAFE`' if neither condition holds. If both check and checkmate are detected, print '`CHECKMATED`'.

Note: *The chess piece images below were created by Colin M.L. Burnett and are used under the auspices of the BSD license, the text of which follows.*

Sample Input

```
3
w
........
........
........
.Qk.K...
........
........
........
........
B
........
........
........
```

```
.qK.k...
........
........
........
.r......
w
........
..k.....
........
.Q..K...
........
........
........
........
```

Sample Output

```
WHITE IS CHECKED
BLACK IS CHECKMATED
WHITE IS SAFE
```