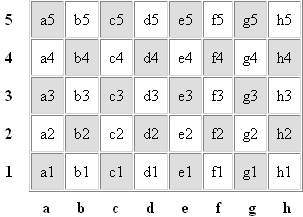
## Problem 2 – Chess Moves KQ

You are given a chess board with size **R**x**C** (**R** rows and **C** columns). Each square (cell) of the chessboard is identified by a unique coordinate pair - a letter and a number. The vertical **columns** of squares from left to right are labeled a, b, c and so on. The horizontal **rows** of squares are numbered 1, 2, 3 and so on, starting from bottom to top. Thus each square has a unique identification of letter followed by number.

Example of board with size 5x8 is given in the picture.

Moves are given by two square (cell) identifications separated by a single space. Examples: “a1 e3”, “d1 f2”, “h5 h1”, “a1 z9”, etc. All given moves will be in this format and **in the range** of the board.

Also you are given 2 types of chess pieces: **knight** and **queen** as explained bellow.

|  |  |
| --- | --- |
| The **knight** moves on an extended diagonal from one corner of any 2×3 rectangle of squares to the furthest opposite corner.  The knight is piece that **is allowed to jump** over any intervening piece(s) when moving. | The **queen** moves any number of vacant (free) squares in any direction: forwards, backwards, left, right, or diagonally, in a straight line. |
| http://www.intuitor.com/chess/images/Knight_Move.jpg | http://www.intuitor.com/chess/images/Queen_Move.jpg |

**Pieces do not capture other pieces.** Move is valid if the figure will be placed on an empty cell.

### Input

On the first line there will be the number **R** (number of rows). On the second line there will be the number **C** (number of columns). On each of the next **R** lines there will be **C** characters representing one cell (square) of the board. Empty cell are denoted with dash (‘**-**‘), Knights with `**K**` and Queens with `**Q**`. See examples for clarification.

On the next line there will be the number **T** representing the number of moves to be checked for validity. At each of the next **T** lines there will be a move. All moves will be in the range of the board.

The input data will always be valid and in the format described. There is no need to check it explicitly.

### Output

For each of the given moves output either ‘**yes**’ or ‘**no**’ whether the move is valid or not. Invalid moves are those which are not possible because of the given restrictions. See examples for clarification.

### Sample solution code (in JavaScript)

**function** *solve*(params) {  
 **var** rows = parseInt(params[0]),  
 cols = parseInt(params[1]),  
 tests = parseInt(params[rows + 2]),  
 knightMoves = [[-2, 1], [-1, 2], [1, 2], [2, 1],

[2, -1], [1, -2], [-1, -2], [-2, -1]], i, move;  
 **for** (i = 0; i < tests; i++) {  
 move = params[rows + 3 + i];  
 *// Your solution here* **console**.log(**'yes'**); *// or console.log('no');* }  
}

### Constraints

* **R** will be between **1** and **9**, inclusive. **C** will be between **1** and **26**, inclusive.
* **T** will be between **5** and **12**, inclusive.
* The board will contain only ‘**-**‘, ‘**K**’ and ‘**Q**’ characters
* The list of moves will contain only strings with 5 characters in the format described above.
* **Some of the test cases are designed to test only specific invalid move types, so partial solutions may also earn points**
* Allowed working time for your program: **0.25 seconds**. Allowed memory: **32 MB**.

### Hints

The **charCodeAt()** method returns the Unicode integer code value of the character at the specified index in a string. The code of the character ‘**a**’ is 97, the code of the character ‘**b**’ is 98, and so on.

The **String.fromCharCode(97)** method returns ‘**a**’, **String.fromCharCode(98)** returns `**b**` and so on.

### Examples

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Input** | **Output** | **Explanation** |  | **Input** | **Output** |
| 3  4  --K-  K--K  Q--Q  12  d1 b3  a1 a3  c3 b2  a1 c1  a1 b2  a1 c3  a2 c1  d2 b1  b1 b2  c3 a3  a2 a3  d1 d3 | yes  no  no  yes  yes  no  yes  yes  no  no  no  no | Do not output empty lines!  Valid move for queen  Non-empty cell on the path (on a2)  Knights cannot move diagonally  Valid move for queen  Valid move for queen  c3 is not a vacant square  Valid move for knight  Valid move for knight  b1 is an empty square (no piece there)  Not a valid move for the knight  Not a valid move for the knight  Another figure in between | 5  5  Q---Q  -----  -K---  --K--  Q---Q  10  a1 a1  a1 d4  e1 b4  a5 d2  e5 b2  b3 d4  b3 c1  b3 d1  c2 a3  c2 b4 | no  yes  yes  yes  yes  yes  yes  no  yes  yes |