**Crossword Puzzle**

**Problem**

You are in the process of creating a crossword puzzle. You've already designed the board, but now you need to come up with words of various sizes to put in the puzzle.

An empty crossword puzzle consists of filled squares ('X') and empty squares ('.').

Here is an example of a board with 5 rows and 6 columns:

X....X

X.XX.X

...X..

X.XX.X

..X...

A "slot" of length N is defined as exactly N empty squares in a row, surrounded on either side by either a filled square or the edge of the board. N must be at least 2. There are five horizontal slots in the example puzzle above:

First row, length = 4

Third row, length = 3

Third row, length = 2

Fifth row, length = 2

Fifth row, length = 3

Given a ‘char[,] board’, representing an empty crossword puzzle, and an ‘int size’, your method should return the number of horizontal slots in the puzzle that are exactly size characters in length. Each element of board represents one row of the puzzle.

**Definition**

Class : CrossWordPuzzle

Method : CountWords

Parameters : char[,], int

Returns : int

Method signature : public int countWords(char[,] board, int size)

**Constraints**

* board will contain between 3 and 50 elements, inclusive.
* Each element of board will contain between 3 and 50 characters, inclusive.
* Each element of board will be the same length.
* board will consist of only '.' and 'X' characters, and will contain at least two '.' characters.
* All '.' characters in board will be connected horizontally or vertically.
* size will be between 2 and 50, inclusive.

**Sample**

3

X....X

X.XX.X

...X..

X.XX.X

..X...

Returns: 2

The example from above. Note that there are two horizontal slots with length = 3.

3

...X...

.X...X.

..X.X..

X..X..X

..X.X..

.X...X.

...X...

Returns: 6

There are two slots of length 3 on both the first and seventh rows, and one slot each on the second and sixth rows, for a total of 6.

5

.....X....X....

.....X....X....

..........X....

....X....X.....

...X....X....XX

XXX...X....X...

.....X....X....

.......X.......

....X....X.....

...X....X...XXX

XX....X....X...

.....X....X....

....X..........

....X....X.....

....X....X.....

Returns: 8

50

...

...

...

Returns: 0

3

....

....

....

Returns: 0