

# CPSC 2150 Project Report

Jason Rodgers

## Requirements Analysis

### Functional Requirements:

1. As a player, I need to know if I am X's or O's, so that I know if I am going first or not.
2. As a player, I need to know where I can move on the board, so that I can make the best possible next move.
3. As a player, I need to know if it is my turn to make a move, so that I can so that I can place my token on the board.
4. As a player, I want to be able see the board, so that I can see how I am doing
5. As a player, I need to know what spots are taken on the board, so that I don't move there on the board.
6. As a player, I need to know if I won or loss, so that I know when the game is over.
7. As a player, I want to know if I can play again, so that I can restart the game.
8. As a player, I want to be able to see all the moves made so far on the board, so that I can see if I need to play defense or offense.
9. As a player, I need to know if there are no possible moves left on the board, so that I know the game ended in a tie.
10. As a player, I want to see the winning combination that won the game, so that I can see how I won or loss.
11. As a player, I want to know what the columns are listed as, so that I ensure I place my token in the right column.
12. As a player, I need to identify where my tokens are, so that I can spot out the many ways for me to win the game.
13. As a player, I need to the know different combinations in which I can win, so that I know how to play the game.
14. As a player, I want to know how many rows and columns there are, so that I can make a strategy of how I am going win.
15. As a player, I want to know if I can change the rows and columns of the game, so that I can change how many tokens I would need in a row to win the game.
16. As a player, I want to know if I can change my token position after I place it, so that I can change my mistake if I misplace my token.
17. As a player, I want to know If I can change the token names from X and O, so that I can have some customizability in the game.
18. As a player, I want to enter the number of rows and columns that the game board will have, so that I have some control over the game.

19. As a player, I want to be able to see why that I cannot place my token at a certain position, so that I can put it at another valid position.
20. I want to get a winning message when I win the game, so that I feel rewarded after winning.
21. As a player, I can place a marker in a column, so I can claim a space.
22. As a player, if I get 5 in a row horizontally, I will win the game so that I can win the game.
23. As a player, if I get 5 in a row vertically, I will win the game so that I can win the game.
24. As a player, if I get 5 in a row diagonally, I will win the game so that I can win the game.
25. As a player, I can choose to play again, so I can play again.
26. As a player, I can choose the number of rows, so that I can play with the game board that I want.
27. As a player, I can choose the number of columns, so that I can play extended connect X the way that I want.
28. As a player, I can choose how many numbers are needed to win, so that I can play the game how I want to play.
29. As a player, I can choose the number of players I want to play the game, so that I can play with as many people as I want to.
30. As a player, I want to choose the token character that I want to play with, so that I am able to easily identify which token is my token.

### **Non-Functional Requirements**

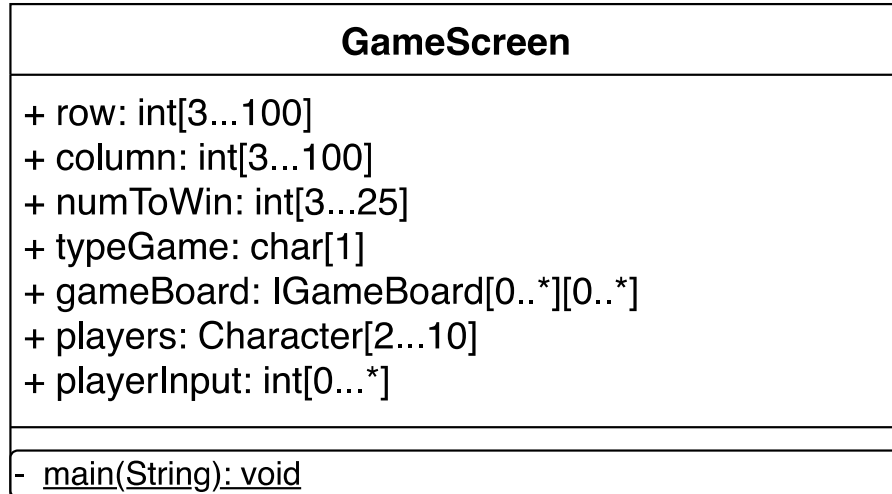
1. The system must be written in Java.
2. The system must be able to gather input from the user.
3. The system must be able to generate a 6 x 9 game board.
4. The system must be able to handle two players.
5. The system must utilize algorithms to decrease run time and increase performance and efficiency.
6. Position 0,0 is at the bottom left of the game board.

### **Deployment Instructions**

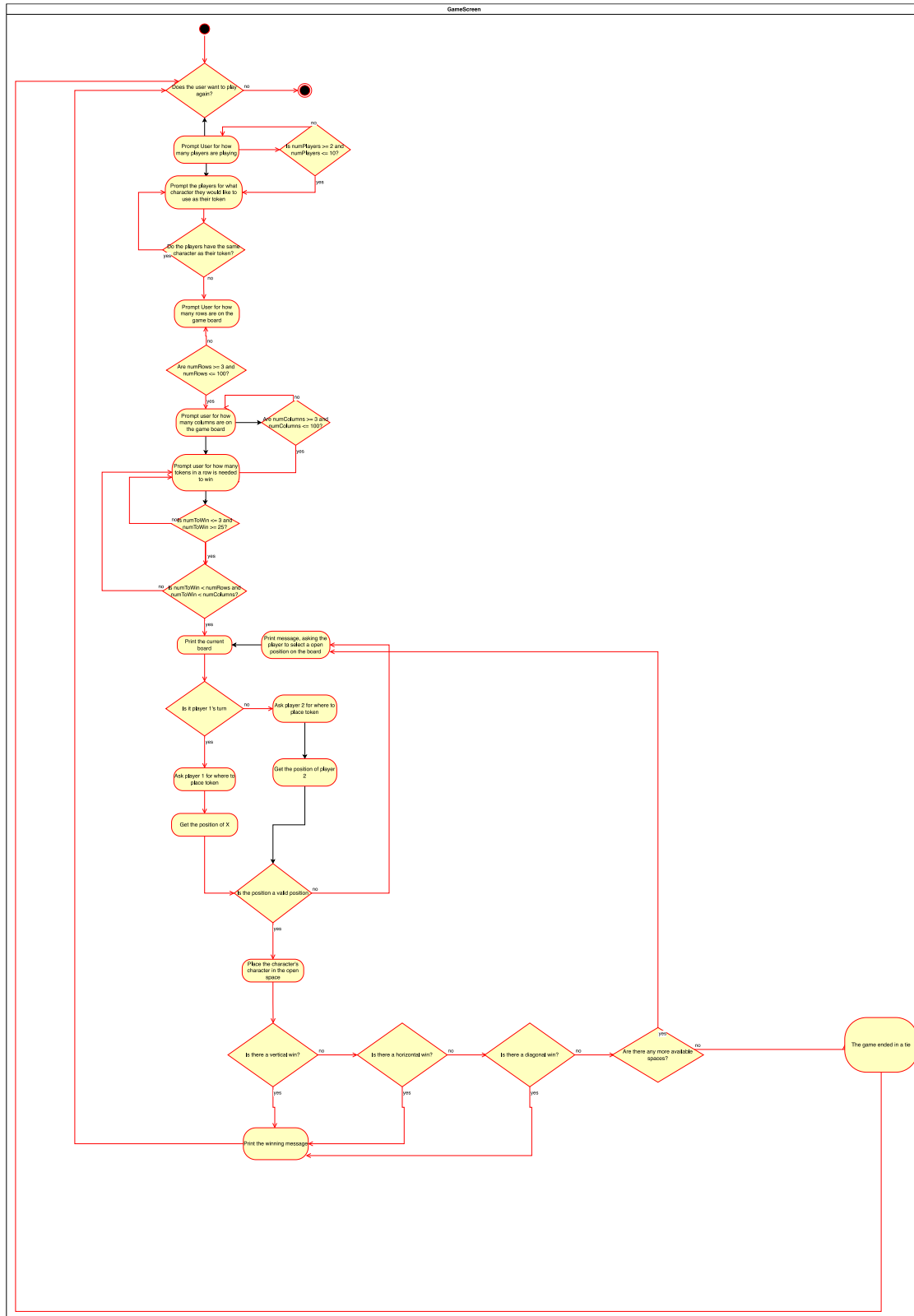
Details in Projects 2-5.

## Class 1: GameScreen

### Class Diagram:

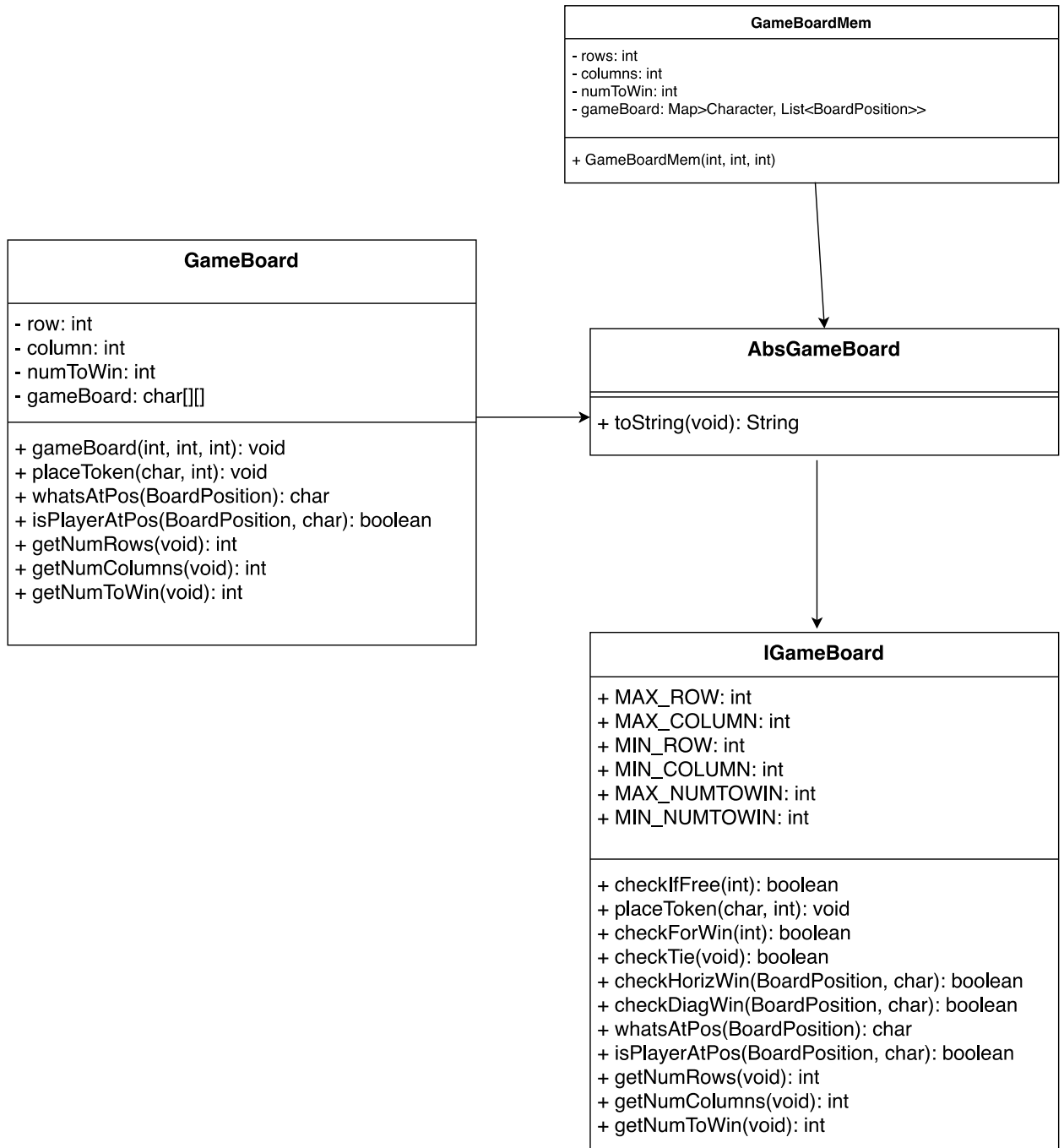


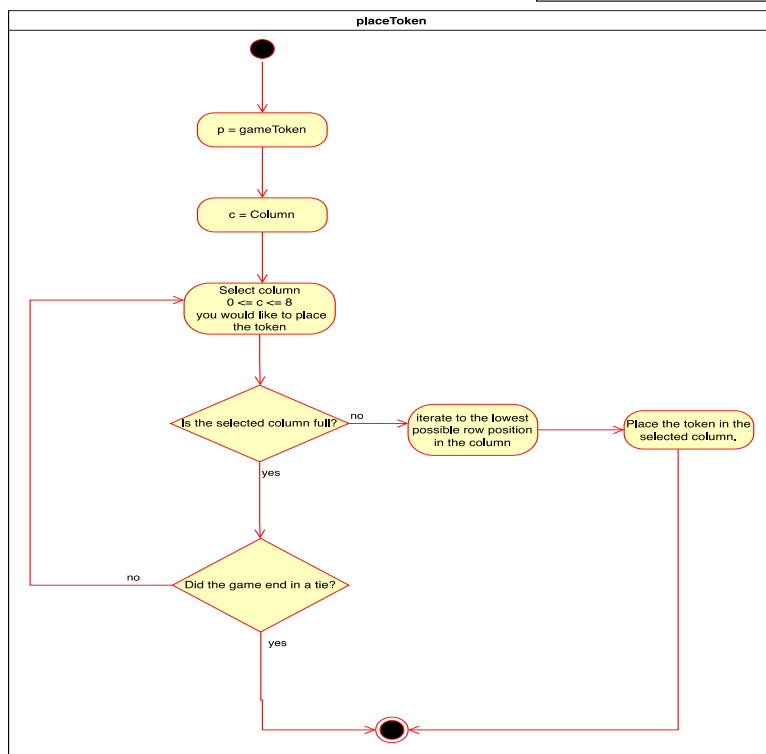
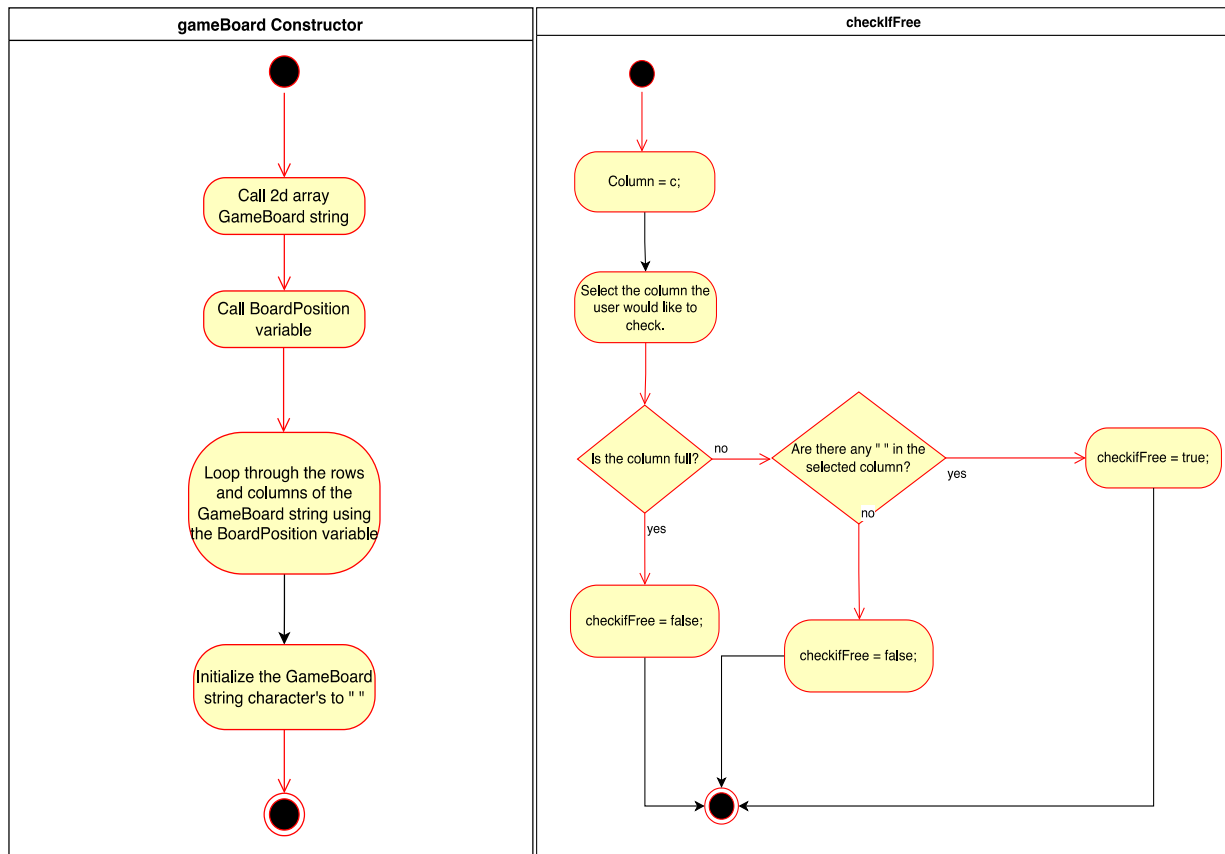
## Activity diagrams (UML Diagram)

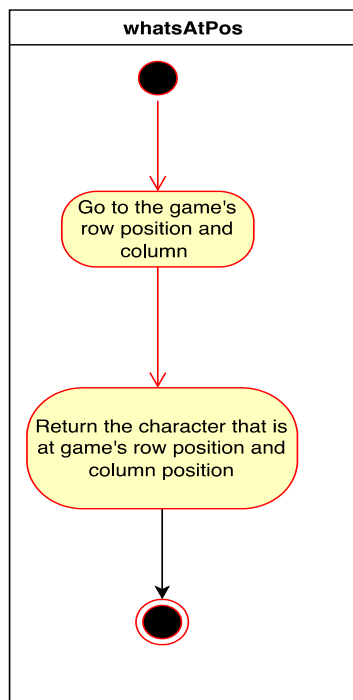
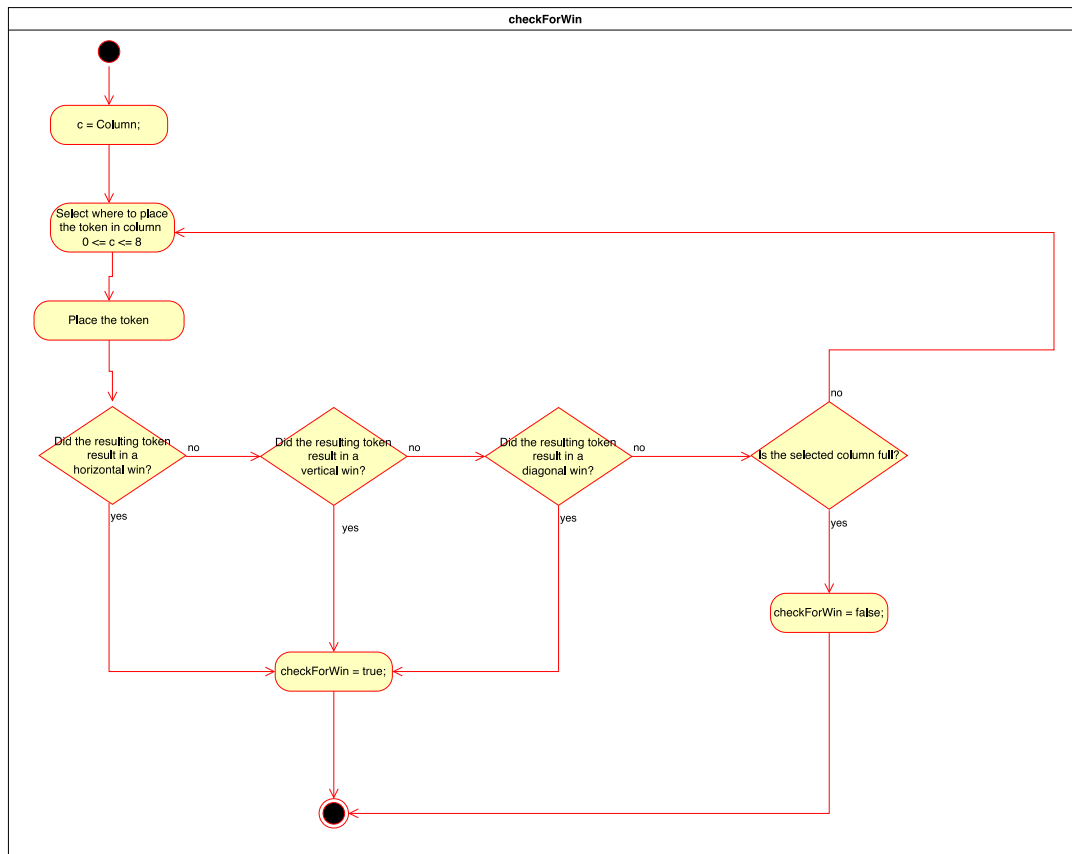


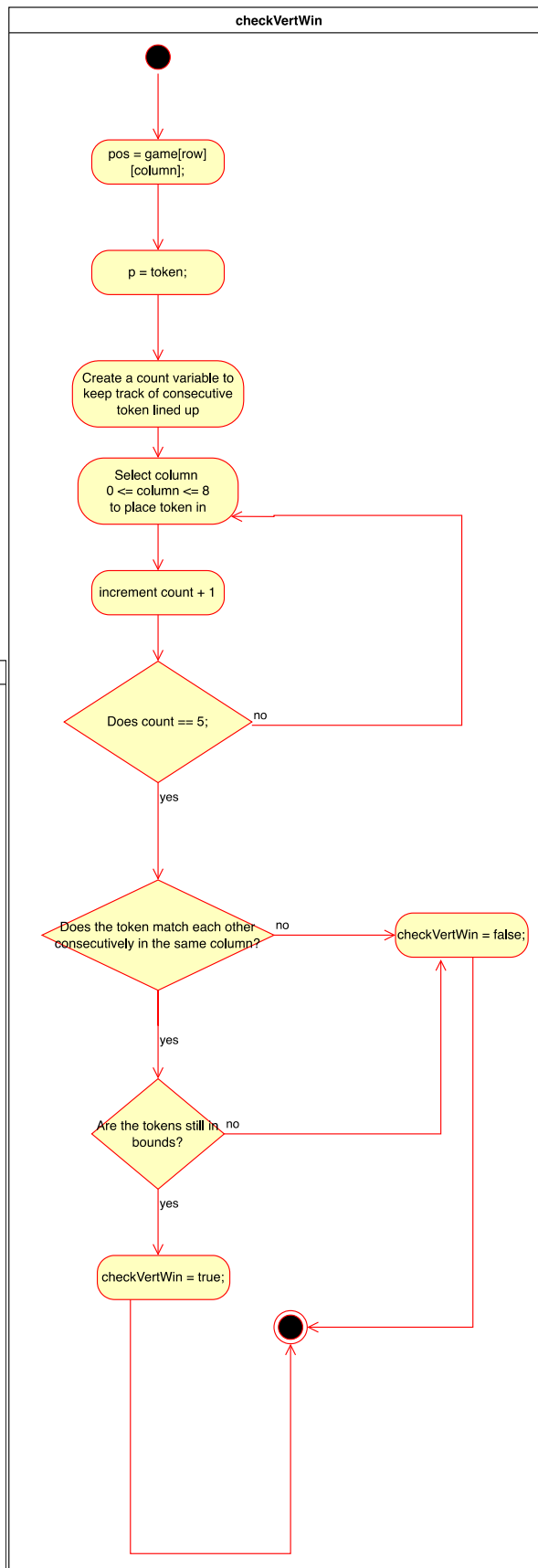
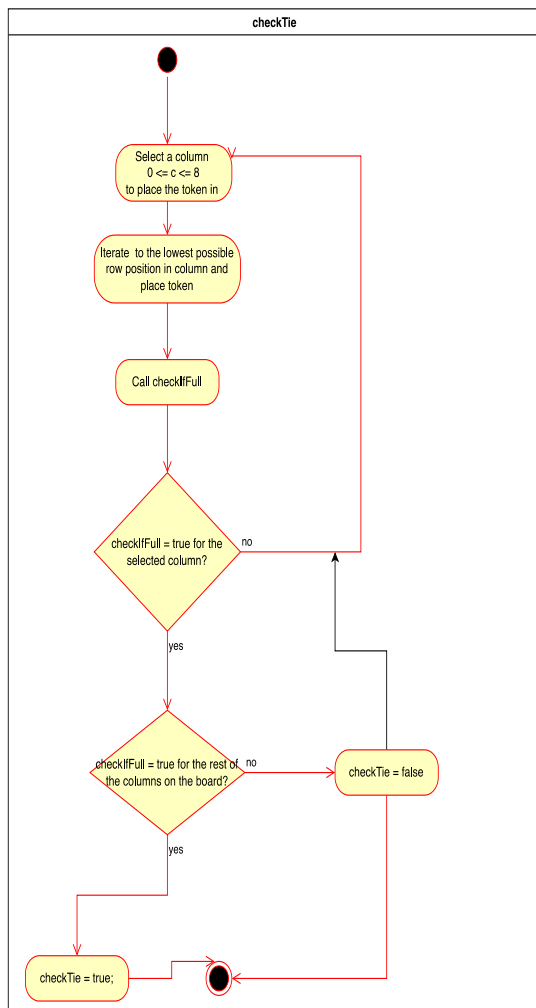
## Class 2: GameBoard

Class diagram:

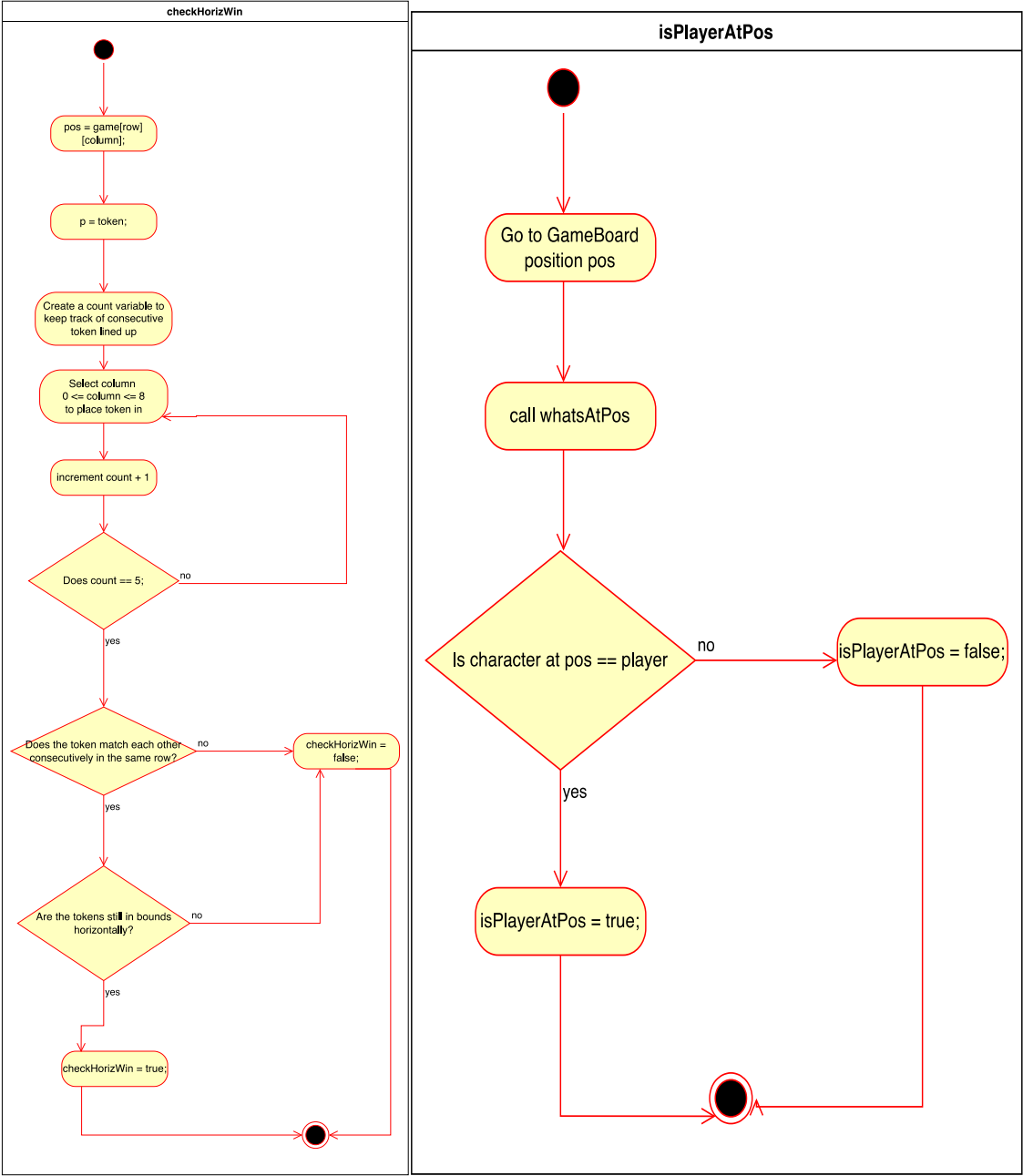


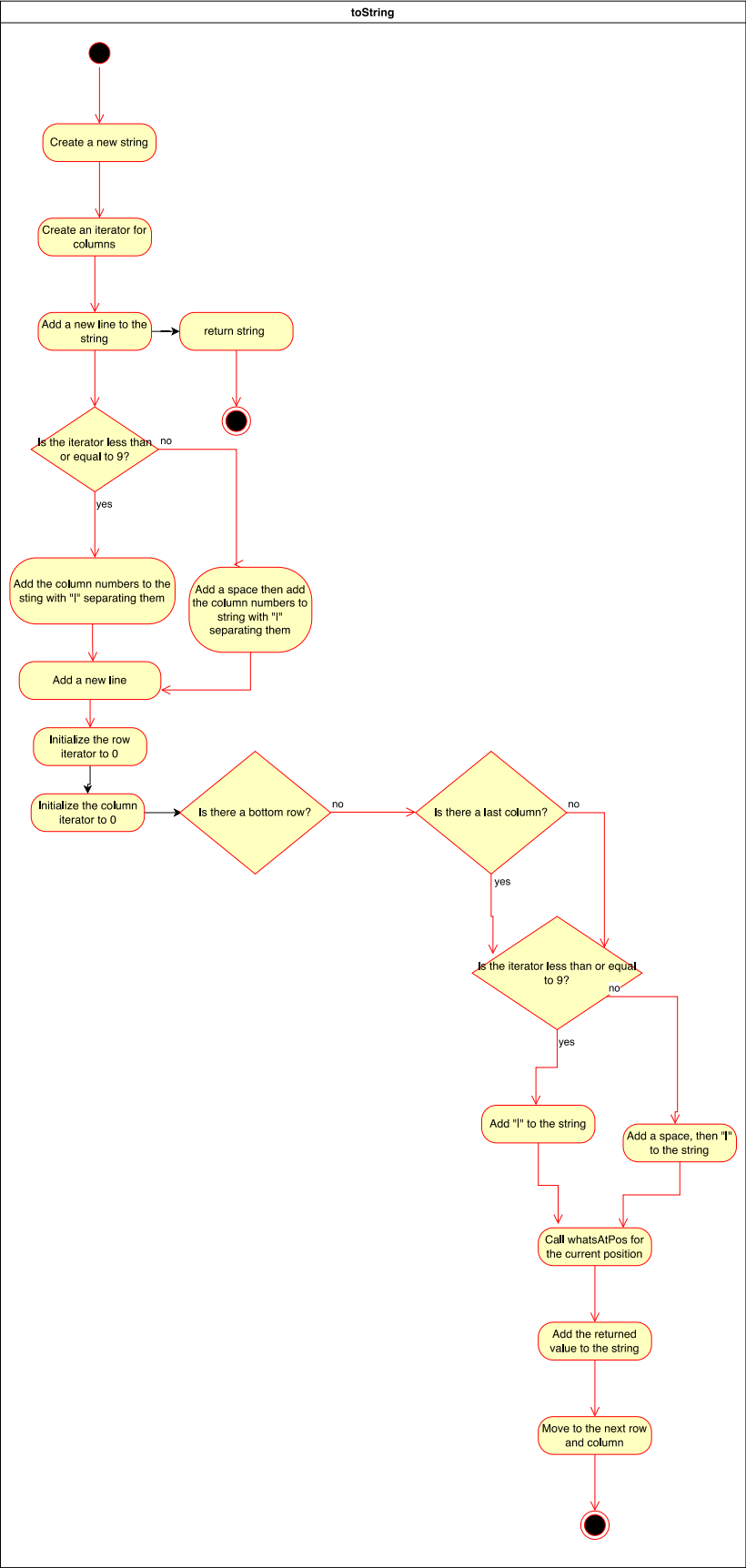




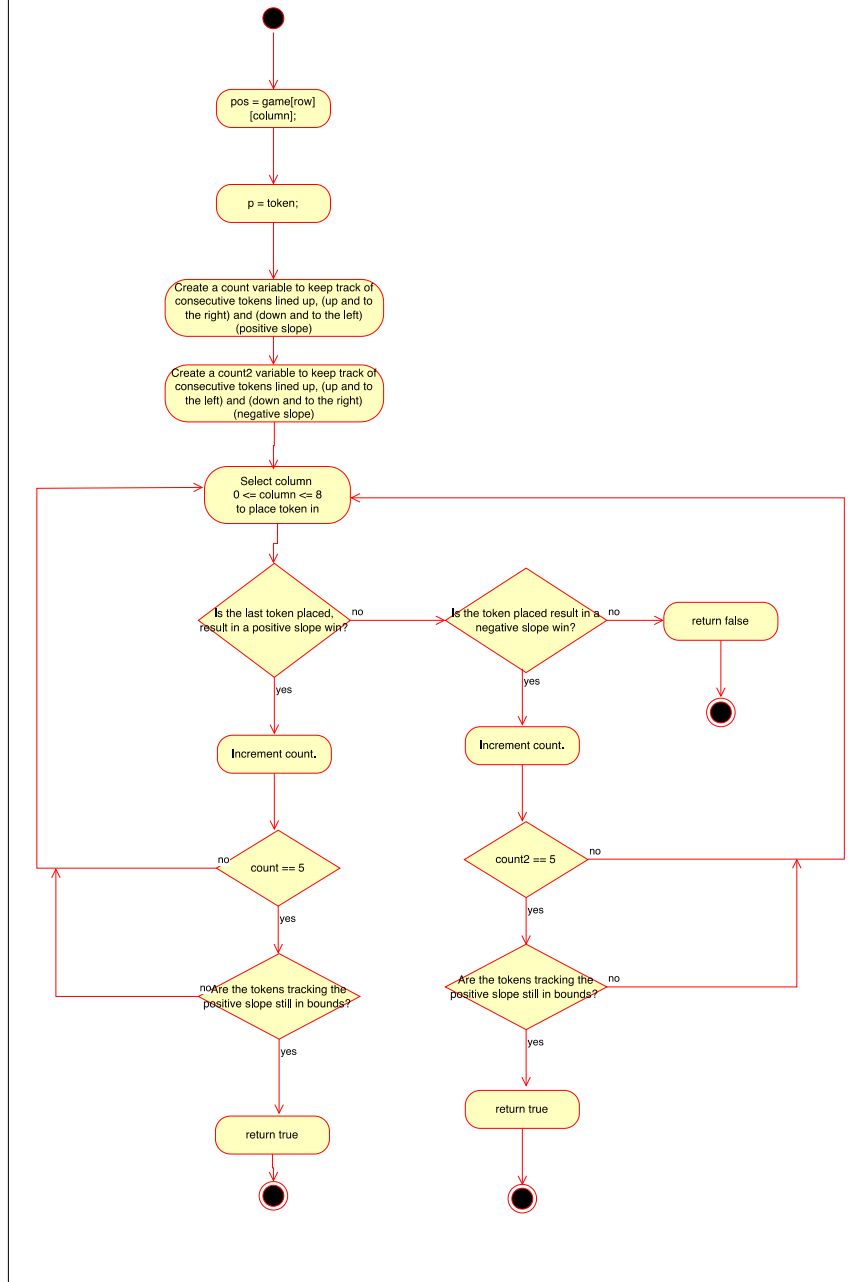


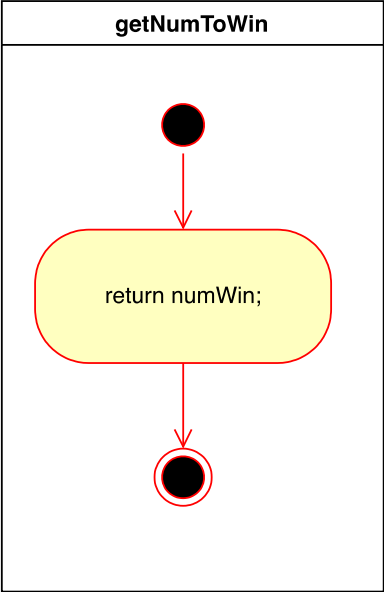
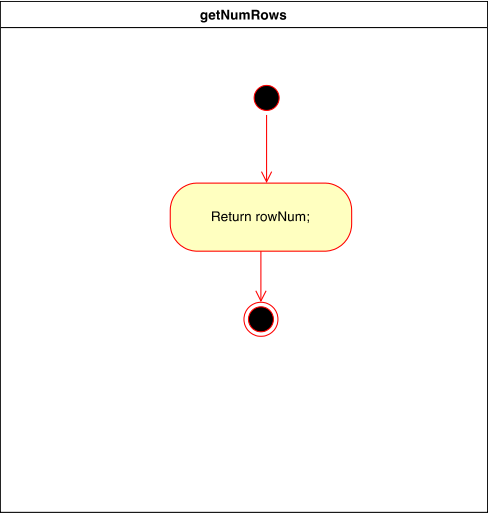
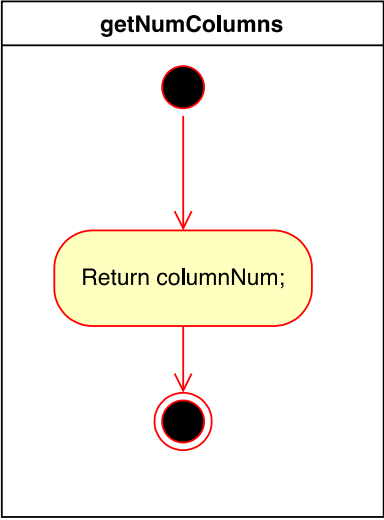






# checkDiagWin





### Class 3: BoardPosition

Class diagram:

| BoardPosition   |
|---|
| <u>+ Row: int[1]</u><br><u>+ Column: int[1]</u>   |
| + BoardPosition(int, int): void<br>+ getRow(void): int<br>+ getColumn(void): int<br>+ equals(Object): boolean<br>+ toString(void): String |

## Test Cases

public GameBoard(int row, int column, int numWin)

|                      |   |                |   |   |   |   |  |  |  |   |  |  |  |   |  |  |  |   |
|----------------------|---|----------------|---|---|---|---|--|--|--|---|--|--|--|---|--|--|--|---|
| <b>Input:</b>        | <b>Output:</b>  | <b>Reason:</b> |   |   |   |   |  |  |  |   |  |  |  |   |  |  |  |   |
| State: Uninitialized | <table><tr><td>State:</td><td>0</td><td>1</td><td>2</td></tr><tr><td>2</td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td></tr><tr><td>0</td><td></td><td></td><td></td></tr></table> | State:         | 0 | 1 | 2 | 2 |  |  |  | 1 |  |  |  | 0 |  |  |  | <p>This test case is unique and distinct because it tests that the board row, column, and numToWin works with the minimum precondition case.</p> <p><b>Function Name:</b></p> <p>test_constructor_SquareMinimum_BoardSize</p> |
| State:               | 0   | 1              | 2 |   |   |   |  |  |  |   |  |  |  |   |  |  |  |   |
| 2                    |   |                |   |   |   |   |  |  |  |   |  |  |  |   |  |  |  |   |
| 1                    |   |                |   |   |   |   |  |  |  |   |  |  |  |   |  |  |  |   |
| 0                    |   |                |   |   |   |   |  |  |  |   |  |  |  |   |  |  |  |   |

|  |   |   |   |   |     |       |     |       |    |    |  |  |  |  |  |  |  |       |  |  |  |  |  |  |  |     |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |
|--|---|---|---|---|-----|-------|-----|-------|----|----|--|--|--|--|--|--|--|-------|--|--|--|--|--|--|--|-----|--|--|--|--|--|--|--|---|--|--|--|--|--|--|--|---|--|--|--|--|--|--|--|---|--|--|--|--|--|--|--|--|
| <b>Input:</b>                            | <b>Output:</b>  | <b>Reason:</b>  |   |   |     |       |     |       |    |    |  |  |  |  |  |  |  |       |  |  |  |  |  |  |  |     |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |
| State:<br>Uninitialized                  | <table><tr><td>State:</td><td>0</td><td>1</td><td>2</td><td>3</td><td>...</td><td>50...</td><td>99</td></tr><tr><td>99</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>50...</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>...</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table> | State:  | 0 | 1 | 2   | 3     | ... | 50... | 99 | 99 |  |  |  |  |  |  |  | 50... |  |  |  |  |  |  |  | ... |  |  |  |  |  |  |  | 2 |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  | This test case is unique and distinct because it tests that the board row, column and numToWin works with the maximum precondition case. |
| State:                                   | 0   | 1   | 2 | 3 | ... | 50... | 99  |       |    |    |  |  |  |  |  |  |  |       |  |  |  |  |  |  |  |     |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |
| 99                                       |   |   |   |   |     |       |     |       |    |    |  |  |  |  |  |  |  |       |  |  |  |  |  |  |  |     |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |
| 50...                                    |   |   |   |   |     |       |     |       |    |    |  |  |  |  |  |  |  |       |  |  |  |  |  |  |  |     |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |
| ...                                      |   |   |   |   |     |       |     |       |    |    |  |  |  |  |  |  |  |       |  |  |  |  |  |  |  |     |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |
| 2  |   |   |   |   |     |       |     |       |    |    |  |  |  |  |  |  |  |       |  |  |  |  |  |  |  |     |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |
| 1  |   |   |   |   |     |       |     |       |    |    |  |  |  |  |  |  |  |       |  |  |  |  |  |  |  |     |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |
| 0  |   |   |   |   |     |       |     |       |    |    |  |  |  |  |  |  |  |       |  |  |  |  |  |  |  |     |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |
| Row = 100<br>Column = 100<br>numWin = 25 |   | <b>Function Name:</b><br><br>test_constructor_SquareMax_BoardSize |   |   |     |       |     |       |    |    |  |  |  |  |  |  |  |       |  |  |  |  |  |  |  |     |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |

| Input:  | Output:   | Reason: |   |     |       |     |       |    |    |  |  |  |  |  |  |    |  |  |  |  |  |  |       |  |  |  |  |  |  |     |  |  |  |  |  |  |   |  |  |  |  |  |  |   |  |  |  |  |  |  |   |  |  |  |  |  |  |  |
|---|---|---------|---|-----|-------|-----|-------|----|----|--|--|--|--|--|--|----|--|--|--|--|--|--|-------|--|--|--|--|--|--|-----|--|--|--|--|--|--|---|--|--|--|--|--|--|---|--|--|--|--|--|--|---|--|--|--|--|--|--|--|
| State:<br>Uninitialized<br><br>Row = 50<br>Column = 50<br>numWin = 25 | <table><tr><td>State:</td><td>0</td><td>1</td><td>2</td><td>...</td><td>25...</td><td>49</td></tr><tr><td>49</td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>48</td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>25...</td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>...</td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table> | State:  | 0 | 1   | 2     | ... | 25... | 49 | 49 |  |  |  |  |  |  | 48 |  |  |  |  |  |  | 25... |  |  |  |  |  |  | ... |  |  |  |  |  |  | 2 |  |  |  |  |  |  | 1 |  |  |  |  |  |  | 0 |  |  |  |  |  |  | <p>This test case is unique and distinct because it tests that the board works a random row input and column input.</p> <p><b>Function Name:</b></p> <p>test_constructor_Square50x50_BoardSize</p> |
| State:  | 0   | 1       | 2 | ... | 25... | 49  |       |    |    |  |  |  |  |  |  |    |  |  |  |  |  |  |       |  |  |  |  |  |  |     |  |  |  |  |  |  |   |  |  |  |  |  |  |   |  |  |  |  |  |  |   |  |  |  |  |  |  |  |
| 49  |   |         |   |     |       |     |       |    |    |  |  |  |  |  |  |    |  |  |  |  |  |  |       |  |  |  |  |  |  |     |  |  |  |  |  |  |   |  |  |  |  |  |  |   |  |  |  |  |  |  |   |  |  |  |  |  |  |  |
| 48  |   |         |   |     |       |     |       |    |    |  |  |  |  |  |  |    |  |  |  |  |  |  |       |  |  |  |  |  |  |     |  |  |  |  |  |  |   |  |  |  |  |  |  |   |  |  |  |  |  |  |   |  |  |  |  |  |  |  |
| 25...   |   |         |   |     |       |     |       |    |    |  |  |  |  |  |  |    |  |  |  |  |  |  |       |  |  |  |  |  |  |     |  |  |  |  |  |  |   |  |  |  |  |  |  |   |  |  |  |  |  |  |   |  |  |  |  |  |  |  |
| ...   |   |         |   |     |       |     |       |    |    |  |  |  |  |  |  |    |  |  |  |  |  |  |       |  |  |  |  |  |  |     |  |  |  |  |  |  |   |  |  |  |  |  |  |   |  |  |  |  |  |  |   |  |  |  |  |  |  |  |
| 2   |   |         |   |     |       |     |       |    |    |  |  |  |  |  |  |    |  |  |  |  |  |  |       |  |  |  |  |  |  |     |  |  |  |  |  |  |   |  |  |  |  |  |  |   |  |  |  |  |  |  |   |  |  |  |  |  |  |  |
| 1   |   |         |   |     |       |     |       |    |    |  |  |  |  |  |  |    |  |  |  |  |  |  |       |  |  |  |  |  |  |     |  |  |  |  |  |  |   |  |  |  |  |  |  |   |  |  |  |  |  |  |   |  |  |  |  |  |  |  |
| 0   |   |         |   |     |       |     |       |    |    |  |  |  |  |  |  |    |  |  |  |  |  |  |       |  |  |  |  |  |  |     |  |  |  |  |  |  |   |  |  |  |  |  |  |   |  |  |  |  |  |  |   |  |  |  |  |  |  |  |

boolean checkIfFree(int c)

|   |                |                |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |
|---|----------------|----------------|---|---|---|---|---|---|---|---|--|--|--|--|--|--|--|--|---|--|--|--|--|--|--|--|--|---|--|--|--|--|--|--|--|--|---|--|--|--|--|--|--|--|--|---|--|--|--|--|--|--|--|--|---|--|--|--|--|--|--|--|--|---|--|--|--|--|--|--|--|--|---|--|--|--|--|--|--|--|--|---|--|
| <b>Input:</b>   | <b>Output:</b> | <b>Reason:</b> |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |
| State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr><tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>5</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table><br>c = 0; |                | 0              | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 7 |  |  |  |  |  |  |  |  | 6 |  |  |  |  |  |  |  |  | 5 |  |  |  |  |  |  |  |  | 4 |  |  |  |  |  |  |  |  | 3 |  |  |  |  |  |  |  |  | 2 |  |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  |  | checkIfFree = true;<br><br>The state of the board is unchanged. | This test case is unique and distinct because it tests that the checkIfFree is able to detect that there is not a character at the selected position.<br><br><b>Function Name:</b><br><br>test_checkIfFree_columnEmpty |
|   | 0              | 1              | 2 | 3 | 4 | 5 | 6 | 7 |   |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |
| 7   |                |                |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |
| 6   |                |                |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |
| 5   |                |                |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |
| 4   |                |                |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |
| 3   |                |                |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |
| 2   |                |                |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |
| 1   |                |                |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |
| 0   |                |                |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |  |  |  |  |  |  |  |   |  |

|  |                |                |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |  |
|--|----------------|----------------|---|---|---|---|---|---|---|---|--|--|--|--|--|--|--|--|---|---|--|--|--|--|--|--|--|---|---|--|--|--|--|--|--|--|---|---|--|--|--|--|--|--|--|---|---|--|--|--|--|--|--|--|---|---|--|--|--|--|--|--|--|---|---|--|--|--|--|--|--|--|---|---|--|--|--|--|--|--|--|---|--|
| <b>Input:</b>  | <b>Output:</b> | <b>Reason:</b> |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |  |
| State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr><tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>6</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>5</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>4</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table><br>c = 0; |                | 0              | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 7 |  |  |  |  |  |  |  |  | 6 | X |  |  |  |  |  |  |  | 5 | X |  |  |  |  |  |  |  | 4 | X |  |  |  |  |  |  |  | 3 | X |  |  |  |  |  |  |  | 2 | X |  |  |  |  |  |  |  | 1 | X |  |  |  |  |  |  |  | 0 | X |  |  |  |  |  |  |  | checkIfFree = true;<br><br>The state of the board is unchanged. | This test case is unique and distinct because it tests that the checkIfFree is able to detect that the column is not full at the selected number when the column is not full.<br><br><b>Function Name:</b><br><br>test_checkIfFree_column_not_full |
|  | 0              | 1              | 2 | 3 | 4 | 5 | 6 | 7 |   |   |  |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |  |
| 7  |                |                |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |  |
| 6  | X              |                |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |  |
| 5  | X              |                |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |  |
| 4  | X              |                |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |  |
| 3  | X              |                |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |  |
| 2  | X              |                |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |  |
| 1  | X              |                |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |  |
| 0  | X              |                |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |  |

|   |                |                |   |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |  |   |
|---|----------------|----------------|---|---|---|---|---|---|---|---|---|--|--|--|--|--|--|--|---|---|--|--|--|--|--|--|--|---|---|--|--|--|--|--|--|--|---|---|--|--|--|--|--|--|--|---|---|--|--|--|--|--|--|--|---|---|--|--|--|--|--|--|--|---|---|--|--|--|--|--|--|--|---|---|--|--|--|--|--|--|--|--|---|
| <b>Input:</b>   | <b>Output:</b> | <b>Reason:</b> |   |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |  |   |
| State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr><tr><td>7</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>6</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>5</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>4</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table><br>c = 0; |                | 0              | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 7 | X |  |  |  |  |  |  |  | 6 | X |  |  |  |  |  |  |  | 5 | X |  |  |  |  |  |  |  | 4 | X |  |  |  |  |  |  |  | 3 | X |  |  |  |  |  |  |  | 2 | X |  |  |  |  |  |  |  | 1 | X |  |  |  |  |  |  |  | 0 | X |  |  |  |  |  |  |  | checkIfFree = false;<br><br>The state of the board is unchanged. | This test case is unique and distinct because it tests that the checkIfFree is able to detect that there is a character at the selected position when the selected is full.<br><br><b>Function Name:</b><br><br>test_checkIfFree_columnFull |
|   | 0              | 1              | 2 | 3 | 4 | 5 | 6 | 7 |   |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |  |   |
| 7   | X              |                |   |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |  |   |
| 6   | X              |                |   |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |  |   |
| 5   | X              |                |   |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |  |   |
| 4   | X              |                |   |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |  |   |
| 3   | X              |                |   |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |  |   |
| 2   | X              |                |   |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |  |   |
| 1   | X              |                |   |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |  |   |
| 0   | X              |                |   |   |   |   |   |   |   |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |   |   |  |  |  |  |  |  |  |  |   |

boolean checkHorizWin(BoardPosition pos, char p)

| Input:   | Output: | Reason: |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |  |  |  |   |
|--|---------|---------|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|---|---|---|--|--|--|---|
| <p>State:</p> <table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td>X</td><td>X</td><td>X</td><td></td><td></td></tr></table> <p>pos.getRow = 0<br/>pos.getColumn = 0<br/>p = 'X'</p> |         | 0       | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 |  |  |  |  |  | 1 |  |  |  |  |  | 0 | X | X | X |  |  | <p>checkHorizWin = true;</p> <p>The state of the board is unchanged.</p> | <p>This test case is unique and distinct because it tests that the checkHorizWin is true and able to detect character X on the first three rows of the game board. This also test the minimum case of numToWin.</p> <p><b>Function Name:</b></p> <p>test_checkHorizWin_beginning_row0</p> |
|  | 0       | 1       | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |  |  |  |   |
| 4  |         |         |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |  |  |  |   |
| 3  |         |         |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |  |  |  |   |
| 2  |         |         |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |  |  |  |   |
| 1  |         |         |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |  |  |  |   |
| 0  | X       | X       | X |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |  |  |  |   |

| Input:   | Output: | Reason: |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |   |
|--|---------|---------|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|---|---|---|---|---|--|---|
| <p>State:</p> <table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr></table> <p>pos.getRow = 0<br/>pos.getColumn = 0<br/>p = 'X'</p> |         | 0       | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 |  |  |  |  |  | 1 |  |  |  |  |  | 0 | X | X | X | X | X | <p>checkHorizWin = true;</p> <p>The state of the board is unchanged.</p> | <p>This test case is unique and distinct because it tests that the checkHorizWin is true and able to detect character X on the first five rows of the game board. This test case also tests a random numToWin number.</p> <p><b>Function Name:</b></p> <p>test_checkHorizWin_beginningRow</p> |
|  | 0       | 1       | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |   |
| 4  |         |         |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |   |
| 3  |         |         |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |   |
| 2  |         |         |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |   |
| 1  |         |         |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |   |
| 0  | X       | X       | X | X | X |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |   |

| Input:   | Output: | Reason: |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |   |  |
|--|---------|---------|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|---|---|---|---|---|---|--|
| <p>State:</p> <table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td>X</td><td>X</td><td>O</td><td>O</td><td>O</td></tr></table> <p>pos.getRow = 0<br/>pos.getColumn = 0<br/>p = 'X'</p> |         | 0       | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 |  |  |  |  |  | 1 |  |  |  |  |  | 0 | X | X | O | O | O | <p>checkHorizWin = false;</p> <p>The state of the board is unchanged.</p> | <p>This test case is unique and distinct because it tests that the checkHorizWin is false for character X even though checkHorizWin for character O returns true.</p> <p><b>Function Name:</b></p> <p>test_checkHorizWin_beginningRowFalse</p> |
|  | 0       | 1       | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |   |  |
| 4  |         |         |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |   |  |
| 3  |         |         |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |   |  |
| 2  |         |         |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |   |  |
| 1  |         |         |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |   |  |
| 0  | X       | X       | O | O | O |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |   |  |



|  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |   |   |   |  |   |
|--|---|---|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|---|---|---|---|--|---|
| <b>Input:</b><br><br>State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td></td><td>X</td><td>X</td><td>O</td><td>O</td></tr></table><br><br>pos.getRow = 0<br>pos.getColumn = 4<br>p = 'o' |   | 0 | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 |  |  |  |  |  | 1 |  |  |  |  |  | 0 |  | X | X | O | O | <b>Output:</b><br><br>checkHorizWin = false;<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it tests that the checkHorizWin is false at a position that is not at the beginning of the row. Makes sure that the boundary of checkHorizWin is correct.<br><br><b>Function Name:</b><br><br>test_checkHorizWin_beginningRowFalseEnd |
|  | 0 | 1 | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |   |   |   |  |   |
| 4  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |   |   |   |  |   |
| 3  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |   |   |   |  |   |
| 2  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |   |   |   |  |   |
| 1  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |   |   |   |  |   |
| 0  |   | X | X | O | O |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |   |   |   |  |   |

boolean checkVertWin(BoardPosition pos, char p)

|   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|---|--|--|--|--|---|---|--|--|--|--|---|---|--|--|--|--|--|--|
| <b>Input:</b><br><br>State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td>X</td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>X</td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td>X</td><td></td><td></td><td></td><td></td></tr></table><br><br>pos.getRow = 2<br>pos.getColumn = 0<br>p = 'X' |   | 0 | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 | X |  |  |  |  | 1 | X |  |  |  |  | 0 | X |  |  |  |  | <b>Output:</b><br><br>checkVertWin = true;<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it test that checkVertWin is true at the column 0 after placing 3 X characters in the first column.<br><br><b>Function Name:</b><br><br>test_checkVertWin_from_top |
|   | 0 | 1 | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |  |  |
| 4   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |  |  |
| 3   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |  |  |
| 2   | X |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |  |  |
| 1   | X |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |  |  |
| 0   | X |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |  |  |

|  |   |   |   |   |   |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |
|--|---|---|---|---|---|---|---|--|--|--|--|--|---|---|--|--|--|--|---|---|--|--|--|--|---|---|--|--|--|--|---|---|--|--|--|--|---|---|
| <b>Input:</b><br><br>State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td>X</td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td>X</td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>O</td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td>O</td><td></td><td></td><td></td><td></td></tr></table><br><br>pos.getRow = 0<br>pos.getColumn = 0<br>p = 'X' |   | 0 | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 | X |  |  |  |  | 2 | X |  |  |  |  | 1 | O |  |  |  |  | 0 | O |  |  |  |  | <b>Output:</b><br><br>checkVertWin = false;<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it tests that checkVertWin is false at the column 0 after placing both X and O in the first column of the board. Makes sure that the boundary of checkVertWin is correct.<br><br><b>Function Name:</b><br><br>test_checkVertWin_false_with_empty_spot_above |
|  | 0 | 1 | 2 | 3 | 4 |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |
| 4  |   |   |   |   |   |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |
| 3  | X |   |   |   |   |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |
| 2  | X |   |   |   |   |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |
| 1  | O |   |   |   |   |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |
| 0  | O |   |   |   |   |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |

|  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |   |
|--|---|---|---|---|---|---|---|--|--|--|--|--|---|--|--|---|--|--|---|--|--|---|--|--|---|--|--|---|--|--|---|--|--|---|--|--|---|---|
| <b>Input:</b><br><br>State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td>X</td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td>X</td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td>O</td><td></td><td></td></tr><tr><td>0</td><td></td><td></td><td>O</td><td></td><td></td></tr></table><br><br>pos.getRow = 3<br>pos.getColumn = 0<br>p = 'X' |   | 0 | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  | X |  |  | 2 |  |  | X |  |  | 1 |  |  | O |  |  | 0 |  |  | O |  |  | <b>Output:</b><br><br>checkVertWin = false;<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it tests that checkVertWin is false with the 2 characters in the middle column to show that checkVertWin is able to work the same in a different column.<br><br><b>Function Name:</b><br><br>test_checkVertWin_false_top_with_spots_below |
|  | 0 | 1 | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |   |
| 4  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |   |
| 3  |   |   | X |   |   |   |   |  |  |  |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |   |
| 2  |   |   | X |   |   |   |   |  |  |  |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |   |
| 1  |   |   | O |   |   |   |   |  |  |  |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |   |
| 0  |   |   | O |   |   |   |   |  |  |  |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |  |  |   |   |

boolean checkDiagWin(BoardPosition pos, char p)

|   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |
|---|---|---|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|
| <div>Input:</div> <div>State:</div> <table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td></td><td></td><td></td><td></td><td></td></tr></table> <div>pos.getRow = 0<br/>pos.getColumn = 0<br/>p = 'X'</div> |   | 0 | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 |  |  |  |  |  | 1 |  |  |  |  |  | 0 |  |  |  |  |  | <div>Output:</div> <div>checkDiagWin = false;</div> <div>The state of the board is unchanged.</div> | <div>Reason:</div> <div>This test case is unique and distinct because it tests that checkDiagWin is able to return false with no tokens on the board and an empty board.</div> <div>Function Name:</div> <div>test_checkDiagWin_false_with_empty board</div> |
|   | 0 | 1 | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |
| 4   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |
| 3   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |
| 2   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |
| 1   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |
| 0   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |

|  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |   |  |  |   |  |   |   |  |  |   |   |   |   |  |  |  |  |
|--|---|---|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|---|--|--|---|--|---|---|--|--|---|---|---|---|--|--|--|--|
| <b>Input:</b><br><br>State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td>X</td><td></td><td></td></tr><tr><td>1</td><td></td><td>X</td><td>X</td><td></td><td></td></tr><tr><td>0</td><td>X</td><td>X</td><td>X</td><td></td><td></td></tr></table><br><br>pos.getRow = 2<br>pos.getColumn = 2<br>p = 'X' |   | 0 | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 |  |  | X |  |  | 1 |  | X | X |  |  | 0 | X | X | X |  |  | <b>Output:</b><br><br>checkDiagWin = true;<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it tests that checkDiagWin is able to detect character X in a win from left to top right.<br><br><b>Function Name:</b><br><br>test_checkDiagWin_left_to_topRight |
|  | 0 | 1 | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |   |  |  |   |  |   |   |  |  |   |   |   |   |  |  |  |  |
| 4  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |   |  |  |   |  |   |   |  |  |   |   |   |   |  |  |  |  |
| 3  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |   |  |  |   |  |   |   |  |  |   |   |   |   |  |  |  |  |
| 2  |   |   | X |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |   |  |  |   |  |   |   |  |  |   |   |   |   |  |  |  |  |
| 1  |   | X | X |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |   |  |  |   |  |   |   |  |  |   |   |   |   |  |  |  |  |
| 0  | X | X | X |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |   |  |  |   |  |   |   |  |  |   |   |   |   |  |  |  |  |

|  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |   |  |  |  |   |   |   |   |  |  |  |   |
|--|---|---|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|---|--|--|--|--|---|---|---|--|--|--|---|---|---|---|--|--|--|---|
| <b>Input:</b><br><br>State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td>X</td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>X</td><td>X</td><td></td><td></td><td></td></tr><tr><td>0</td><td>X</td><td>X</td><td>X</td><td></td><td></td></tr></table><br><br>pos.getRow = 2<br>pos.getColumn = 0<br>p = 'X' |   | 0 | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 | X |  |  |  |  | 1 | X | X |  |  |  | 0 | X | X | X |  |  | <b>Output:</b><br><br>checkDiagWin = true;<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it tests that checkDiagWin is able to detect character X in a diagonal win from right to top left.<br><br><b>Function Name:</b><br><br>test_checkDiagWin_right_to_topLeft |
|  | 0 | 1 | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |   |  |  |  |   |   |   |   |  |  |  |   |
| 4  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |   |  |  |  |   |   |   |   |  |  |  |   |
| 3  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |   |  |  |  |   |   |   |   |  |  |  |   |
| 2  | X |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |   |  |  |  |   |   |   |   |  |  |  |   |
| 1  | X | X |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |   |  |  |  |   |   |   |   |  |  |  |   |
| 0  | X | X | X |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |   |  |  |  |   |   |   |   |  |  |  |   |

|  |   |   |   |   |   |   |   |  |  |  |   |  |   |  |  |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |  |
|--|---|---|---|---|---|---|---|--|--|--|---|--|---|--|--|---|---|---|---|--|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|--|--|
| <b>Input:</b><br><br>State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td>X</td><td></td></tr><tr><td>3</td><td></td><td></td><td>X</td><td>X</td><td>X</td></tr><tr><td>2</td><td></td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>1</td><td>O</td><td>O</td><td>O</td><td>O</td><td>O</td></tr><tr><td>0</td><td>O</td><td>O</td><td>O</td><td>O</td><td>O</td></tr></table><br><br>pos.getRow = 4<br>pos.getColumn = 4<br>p = 'X' |   | 0 | 1 | 2 | 3 | 4 | 4 |  |  |  | X |  | 3 |  |  | X | X | X | 2 |  | X | X | X | X | 1 | O | O | O | O | O | 0 | O | O | O | O | O | <b>Output:</b><br><br>checkDiagWin = true;<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it shows that checkDiagWin works with X being in the middle and top of the game board.<br><br><b>Function Name:</b><br><br>test_checkDiagWin_bottom_left_to_top_right_filled_under |
|  | 0 | 1 | 2 | 3 | 4 |   |   |  |  |  |   |  |   |  |  |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |  |
| 4  |   |   |   | X |   |   |   |  |  |  |   |  |   |  |  |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |  |
| 3  |   |   | X | X | X |   |   |  |  |  |   |  |   |  |  |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |  |
| 2  |   | X | X | X | X |   |   |  |  |  |   |  |   |  |  |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |  |
| 1  | O | O | O | O | O |   |   |  |  |  |   |  |   |  |  |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |  |
| 0  | O | O | O | O | O |   |   |  |  |  |   |  |   |  |  |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |  |

|  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |  |  |  |   |   |   |  |  |  |   |  |
|--|---|---|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|---|--|--|--|---|---|---|--|--|--|---|--|
| <b>Input:</b><br><br>State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td>X</td><td></td><td></td><td></td></tr><tr><td>0</td><td>X</td><td>X</td><td></td><td></td><td></td></tr></table><br><p>pos.getRow = 0<br/>pos.getColumn = 0<br/>p = 'X'</p> |   | 0 | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 |  |  |  |  |  | 1 |  | X |  |  |  | 0 | X | X |  |  |  | <b>Output:</b><br><br>checkDiagWin = false;<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it tests that checkDiagWin is false when there are not enough tokens to fill the criteria of checkDiagWin top left to bottom right. It shows that the boundaries of checkDiagWin top left to bottom right are valid.<br><br><b>Function Name:</b><br><br>test_checkDiagWin_top_left_to_bottom_right_insufficient_chars |
|  | 0 | 1 | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |  |  |  |   |   |   |  |  |  |   |  |
| 4  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |  |  |  |   |   |   |  |  |  |   |  |
| 3  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |  |  |  |   |   |   |  |  |  |   |  |
| 2  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |  |  |  |   |   |   |  |  |  |   |  |
| 1  |   | X |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |  |  |  |   |   |   |  |  |  |   |  |
| 0  | X | X |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |  |  |  |   |   |   |  |  |  |   |  |

|   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |  |  |  |   |  |   |   |  |  |   |  |
|---|---|---|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|---|--|--|--|---|--|---|---|--|--|---|--|
| <b>Input:</b><br><br>State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td>X</td><td></td><td></td><td></td></tr><tr><td>0</td><td></td><td>X</td><td>X</td><td></td><td></td></tr></table><br><br>pos.getRow = 1<br>pos.getColumn = 1<br>p = 'X' |   | 0 | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 |  |  |  |  |  | 1 |  | X |  |  |  | 0 |  | X | X |  |  | <b>Output:</b><br><br>checkDiagWin = false;<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it tests that checkDiagWin is false when there are not enough tokens to fill the criteria of checkDiagWin top left to bottom right. It shows that the boundaries of checkDiagWin top left to bottom right are valid.<br><br><b>Function Name:</b><br><br>test_checkDiagWin_false_top_left_to_bottom_right_insufficient_chars |
|   | 0 | 1 | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |  |  |  |   |  |   |   |  |  |   |  |
| 4   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |  |  |  |   |  |   |   |  |  |   |  |
| 3   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |  |  |  |   |  |   |   |  |  |   |  |
| 2   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |  |  |  |   |  |   |   |  |  |   |  |
| 1   |   | X |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |  |  |  |   |  |   |   |  |  |   |  |
| 0   |   | X | X |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |   |  |  |  |   |  |   |   |  |  |   |  |

|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|--|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| <b>Input:</b><br><br>State: <table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>3</td><td>O</td><td>O</td><td>O</td><td>O</td><td>O</td></tr><tr><td>2</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>1</td><td>O</td><td>O</td><td>O</td><td>O</td><td>O</td></tr><tr><td>0</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr></table><br><br>pos.getRow = 2<br>pos.getColumn = 2<br>p = 'X' |   | 0 | 1 | 2 | 3 | 4 | 4 | X | X | X | X | X | 3 | O | O | O | O | O | 2 | X | X | X | X | X | 1 | O | O | O | O | O | 0 | X | X | X | X | X | <b>Output:</b><br><br>checkDiagWin = false;<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it tests that checkDiagWin is false when there is a full tied board. X fills all of the even numbered rows while O fill all of the odd numbered rows which will not result in a diagonal win.<br><br><b>Function Name:</b><br><br>test_checkDiagWin_false_full_tied_board |
|  | 0 | 1 | 2 | 3 | 4 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 4  | X | X | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 3  | O | O | O | O | O |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 2  | X | X | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1  | O | O | O | O | O |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0  | X | X | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |

boolean checkTie()

|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
|--|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|--|---|
| <b>Input:</b><br><br>State: <table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>3</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>2</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>1</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>0</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr></table> |   | 0 | 1 | 2 | 3 | 4 | 4 | X | X | X | X | X | 3 | X | X | X | X | X | 2 | X | X | X | X | X | 1 | X | X | X | X | X | 0 | X | X | X | X | X | <b>Output:</b><br><br>checkTie = true;<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it tests that checkTie recognizes that there are no more empty spaces on the board and that it recognizes the token that is filling up the board.<br><br><b>Function Name:</b><br><br>test_checkTie_true_full_board |
|  | 0 | 1 | 2 | 3 | 4 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 4  | X | X | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 3  | X | X | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 2  | X | X | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 1  | X | X | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 0  | X | X | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|--|---|
| <b>Input:</b><br><br>State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>3</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>2</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>1</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>0</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr></table> |   | 0 | 1 | 2 | 3 | 4 | 4 | X | X | X | X | X | 3 | X | X | X | X | X | 2 | X | X | X | X | X | 1 | X | X | X | X | X | 0 | X | X | X | X | X | <b>Output:</b><br><br>checkTie = true;<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it tests that checkTie recognizes that there are no more empty spaces on the board and that it recognizes the token that is filling up the board.<br><br><b>Function Name:</b><br><br>test_checkTie_true_full_board |
|   | 0 | 1 | 2 | 3 | 4 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 4   | X | X | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 3   | X | X | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 2   | X | X | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 1   | X | X | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 0   | X | X | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |

|  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |
|--|---|---|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|
| <b>Input:</b><br><br>State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td></td><td></td><td></td><td></td><td></td></tr></table> |   | 0 | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 |  |  |  |  |  | 1 |  |  |  |  |  | 0 |  |  |  |  |  | <b>Output:</b><br><br>checkTie = false;<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it tests that checkTie recognizes that there are empty spaces on the game board.<br><br><b>Function Name:</b><br><br>test_checkTie_false_empty_board |
|  | 0 | 1 | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |
| 4  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |
| 3  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |
| 2  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |
| 1  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |
| 0  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |

|   |   |   |   |   |   |   |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |  |
|---|---|---|---|---|---|---|---|---|---|---|--|--|---|---|---|---|--|--|---|---|---|---|--|--|---|---|---|---|--|--|---|---|---|---|--|--|---|--|
| <b>Input:</b><br><br>State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td>X</td><td>X</td><td>X</td><td></td><td></td></tr><tr><td>3</td><td>X</td><td>X</td><td>X</td><td></td><td></td></tr><tr><td>2</td><td>X</td><td>X</td><td>X</td><td></td><td></td></tr><tr><td>1</td><td>X</td><td>X</td><td>X</td><td></td><td></td></tr><tr><td>0</td><td>X</td><td>X</td><td>X</td><td></td><td></td></tr></table> |   | 0 | 1 | 2 | 3 | 4 | 4 | X | X | X |  |  | 3 | X | X | X |  |  | 2 | X | X | X |  |  | 1 | X | X | X |  |  | 0 | X | X | X |  |  | <b>Output:</b><br><br>checkTie = false;<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it tests that checkTie is able to recognize that although there are some full columns filled with the character X, empty spaces still exist on the game board and the board has not filled up yet.<br><br><b>Function Name:</b><br><br>test_checkTie_some_full_columns |
|   | 0 | 1 | 2 | 3 | 4 |   |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |  |
| 4   | X | X | X |   |   |   |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |  |
| 3   | X | X | X |   |   |   |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |  |
| 2   | X | X | X |   |   |   |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |  |
| 1   | X | X | X |   |   |   |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |  |
| 0   | X | X | X |   |   |   |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |   |   |   |  |  |   |  |

|  |                |                |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
|--|----------------|----------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|--|---|
| <b>Input:</b>  | <b>Output:</b> | <b>Reason:</b> |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>3</td><td>O</td><td>O</td><td>O</td><td>O</td><td>O</td></tr><tr><td>2</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>1</td><td>O</td><td>O</td><td>O</td><td>O</td><td>O</td></tr><tr><td>0</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr></table> |                | 0              | 1 | 2 | 3 | 4 | 4 | X | X | X | X | X | 3 | O | O | O | O | O | 2 | X | X | X | X | X | 1 | O | O | O | O | O | 0 | X | X | X | X | X | checkTie = true;<br><br>The state of the board is unchanged. | <p>This test case is unique and distinct because it tests that checkTie recognizes that there are no empty spaces on the game board with more than one token on the board. X is placed in the even numbered rows and O is placed in the odd numbered rows.</p> <p><b>Function Name:</b></p> <p>test_checkTie_full_alternating_board</p> |
|  | 0              | 1              | 2 | 3 | 4 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 4  | X              | X              | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 3  | O              | O              | O | O | O |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 2  | X              | X              | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 1  | O              | O              | O | O | O |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 0  | X              | X              | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |

char whatsAtPos(BoardPosition pos)

|   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
|---|---|---|---|---|---|---|---|---|---|---|---|--|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|--|---|
| <div>Input:</div> <div>State:</div> <table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td></td><td></td><td></td><td></td><td></td></tr></table> <div>pos.getRow = 0<br/>pos.getColumn = 0</div>                         |   | 0 | 1 | 2 | 3 | 4 | 4 |   |   |   |   |  | 3 |   |   |   |   |   | 2 |   |   |   |   |   | 1 |   |   |   |   |   | 0 |   |   |   |   |   | <div>Output:</div> <div>whatsAtPos == ' ';</div> <div>The state of the board is unchanged.</div> | <div>Reason:</div> <div>This test case is unique and distinct because it tests that whatsAtPos recognizes that there is an empty space at the current position of the board.</div> <div>Function Name:</div> <div>test_whatsAtPos_empty_space_empty_board</div>   |
|   | 0 | 1 | 2 | 3 | 4 |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 4   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 3   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 2   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 1   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 0   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| <div>Input:</div> <div>State:</div> <table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr></table> <div>pos.getRow = 1<br/>pos.getColumn = 0</div>                    |   | 0 | 1 | 2 | 3 | 4 | 4 |   |   |   |   |  | 3 |   |   |   |   |   | 2 |   |   |   |   |   | 1 |   |   |   |   |   | 0 | X | X | X | X | X | <div>Output:</div> <div>whatsAtPos == ' ';</div> <div>The state of the board is unchanged.</div> | <div>Reason:</div> <div>This test case is unique and distinct because it tests that whatsAtPos correctly traverses through the game board to get to the correct location. The bottom row of the game board is filled with X.</div> <div>Function Name:</div> <div>test_whatsAtPos_one_full_row_empty_space</div>                                    |
|   | 0 | 1 | 2 | 3 | 4 |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 4   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 3   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 2   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 1   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 0   | X | X | X | X | X |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| <div>Input:</div> <div>State:</div> <table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td>X</td><td>X</td><td>X</td><td>X</td><td></td></tr><tr><td>3</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>2</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>1</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>0</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr></table> <div>pos.getRow = 4<br/>pos.getColumn = 4</div> |   | 0 | 1 | 2 | 3 | 4 | 4 | X | X | X | X |  | 3 | X | X | X | X | X | 2 | X | X | X | X | X | 1 | X | X | X | X | X | 0 | X | X | X | X | X | <div>Output:</div> <div>whatsAtPos = ' ';</div> <div>The state of the board is unchanged.</div>  | <div>Reason:</div> <div>This test case is unique and distinct because it tests that whatsAtPos correctly identifies what is at the last row and column of the game board with the whole board except for the last row position and column position filled.</div> <div>Function Name:</div> <div>test_whatsAtPos_almost_full_board_empty_space</div> |
|   | 0 | 1 | 2 | 3 | 4 |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 4   | X | X | X | X |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 3   | X | X | X | X | X |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 2   | X | X | X | X | X |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 1   | X | X | X | X | X |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |
| 0   | X | X | X | X | X |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |

|  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |   |  |   |   |
|--|---|---|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|---|--|---|---|
| <b>Input:</b><br><br>State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td></td><td></td><td></td><td>X</td><td></td></tr></table><br>pos.getRow = 0<br>pos.getColumn = 3 |   | 0 | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 |  |  |  |  |  | 1 |  |  |  |  |  | 0 |  |  |  | X |  | <b>Output:</b><br><br>whatsAtPos = 'X';<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it tests that whatsAtPos recognizes that there is only one character on the board and is able to recognize the correct placement of the character token and return the correct character token present.<br><br><b>Function Name:</b><br><br>test_whatsAtPos_one_char_on_board |
|  | 0 | 1 | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |   |  |   |   |
| 4  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |   |  |   |   |
| 3  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |   |  |   |   |
| 2  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |   |  |   |   |
| 1  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |   |  |   |   |
| 0  |   |   |   | X |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |   |  |   |   |

|  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |
|--|---|---|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|---|--|--|--|--|---|---|--|--|--|--|---|---|--|--|--|--|---|---|
| <b>Input:</b><br><br>State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td>O</td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>X</td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td>O</td><td></td><td></td><td></td><td></td></tr></table><br><br>pos.getRow = 1<br>pos.getColumn = 0 |   | 0 | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 | O |  |  |  |  | 1 | X |  |  |  |  | 0 | O |  |  |  |  | <b>Output:</b><br><br>whatsAtPos = 'X';<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it tests that whatsAtPos recognizes that there is an empty space at the current position of the board.<br><br><b>Function Name:</b><br><br>test_whatsAtPos_spot_surrounded_by_chars |
|  | 0 | 1 | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |
| 4  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |
| 3  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |
| 2  | O |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |
| 1  | X |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |
| 0  | O |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |

boolean isPlayerAtPos(BoardPosition pos, char player)

|   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|--|--|
| <b>Input:</b><br><br>State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td></td><td></td><td></td><td></td><td></td></tr></table><br>pos.getRow = 0<br>pos.getColumn = 0<br>player = X |   | 0 | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 |  |  |  |  |  | 1 |  |  |  |  |  | 0 |  |  |  |  |  | <b>Output:</b><br><br>IsPlayerAtPos = false;<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it tests that isPlayerAtPos recognizes that there is not a token at the current board position of the board. The game board is empty.<br><br><b>Function Name:</b><br><br>test_isPlayerAtPos_false_empty_space_empty_board |
|   | 0 | 1 | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |
| 4   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |
| 3   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |
| 2   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |
| 1   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |
| 0   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |

|   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |   |   |   |
|---|---|---|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|---|---|---|
| <b>Input:</b><br><br>State: <table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td></td><td></td><td></td><td></td><td>O</td></tr></table><br>pos.getRow = 0<br>pos.getColumn = 4<br>player = O |   | 0 | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 |  |  |  |  |  | 1 |  |  |  |  |  | 0 |  |  |  |  | O | <b>Output:</b><br><br>IsPlayerAtPos = true;<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it tests that isPlayerAtPos recognizes that there is a token at the current board position of the board. isPlayerAtPos has one valid position on the board where it can return true.<br><br><b>Function Name:</b><br><br>test_isPlayerAtPos_one_char_on_board |
|   | 0 | 1 | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |   |   |   |
| 4   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |   |   |   |
| 3   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |   |   |   |
| 2   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |   |   |   |
| 1   |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |   |   |   |
| 0   |   |   |   |   | O |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |   |   |   |

|  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |   |  |
|--|---|---|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|---|---|---|---|---|---|--|
| <b>Input:</b><br><br>State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr></table><br><br>pos.getRow = 0<br>pos.getColumn = 1<br>player = X |   | 0 | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 |  |  |  |  |  | 1 |  |  |  |  |  | 0 | X | X | X | X | X | <b>Output:</b><br><br>IsPlayerAtPos = true;<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it tests that isPlayerAtPos recognizes that there is a token at the current board position of the board. With the first row of the game board filled, isPlayerAtPos is able to recognize that the current position of the game board is filled with a token.<br><br><b>Function Name:</b><br><br>test_isPlayerAtPos_one_filled_row |
|  | 0 | 1 | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |   |  |
| 4  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |   |  |
| 3  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |   |  |
| 2  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |   |  |
| 1  |   |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |   |  |
| 0  | X | X | X | X | X |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |   |  |

|   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |  |
|---|---|---|---|---|---|---|---|---|---|---|---|--|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|--|--|
| <b>Input:</b><br><br>State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td>X</td><td>X</td><td>X</td><td>X</td><td></td></tr><tr><td>3</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>2</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>1</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>0</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr></table><br><br>pos.getRow = 4<br>pos.getColumn = 4<br>player = X |   | 0 | 1 | 2 | 3 | 4 | 4 | X | X | X | X |  | 3 | X | X | X | X | X | 2 | X | X | X | X | X | 1 | X | X | X | X | X | 0 | X | X | X | X | X | <b>Output:</b><br><br>IsPlayerAtPos = false;<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it tests that isPlayerAtPos recognizes that there is a token at the current board position of the board. With every position on the board filled with a token, isPlayerAtPos is able to recognize the only character on the board that is not filled with token O which makes isPlayerAtPos return false.<br><br><b>Function Name:</b><br><br>test_isPlayerAtPos_false_almost_full_board_empty_space |
|   | 0 | 1 | 2 | 3 | 4 |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |  |
| 4   | X | X | X | X |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |  |
| 3   | X | X | X | X | X |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |  |
| 2   | X | X | X | X | X |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |  |
| 1   | X | X | X | X | X |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |  |
| 0   | X | X | X | X | X |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |  |



|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|--|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| <b>Input:</b><br><br>State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>3</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>2</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>1</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr><tr><td>0</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr></table><br><br>pos.getRow = 4<br>pos.getColumn = 4<br>player = X |   | 0 | 1 | 2 | 3 | 4 | 4 | X | X | X | X | X | 3 | X | X | X | X | X | 2 | X | X | X | X | X | 1 | X | X | X | X | X | 0 | X | X | X | X | X | <b>Output:</b><br><br>IsPlayerAtPos = true;<br><br>The state of the board is unchanged. | <b>Reason:</b><br><br>This test case is unique and distinct because it tests that isPlayerAtPos recognizes that there is a token at the current board position of the board when the game board is completely full. The token X fills the whole game board.<br><br><b>Function Name:</b><br><br>test_isPlayerAtPos_full_board |
|  | 0 | 1 | 2 | 3 | 4 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 4  | X | X | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 3  | X | X | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 2  | X | X | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1  | X | X | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0  | X | X | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |

void placeToken(char p, int c)

|  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |
|--|----------------|----------------|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|--|--|---|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|---|--|--|--|--|--|
| <b>Input:</b>  | <b>Output:</b> | <b>Reason:</b> |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |
| State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td></td><td></td><td></td><td></td><td></td></tr></table><br><p>p = 'X'</p> <p>c = 0</p> |                | 0              | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 |  |  |  |  |  | 1 |  |  |  |  |  | 0 |  |  |  |  |  | State:<br><table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td>X</td><td></td><td></td><td></td><td></td></tr></table> |  | 0 | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 |  |  |  |  |  | 1 |  |  |  |  |  | 0 | X |  |  |  |  | <p>This test case is unique and distinct because it tests that placeToken I am placing token X in an empty board, and after placing the token in column 0, it was the only token that was in the board.</p> <p><b>Function Name:</b></p> <p>test_placeToken_on_empty_board</p> |
|  | 0              | 1              | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |
| 4  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |
| 3  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |
| 2  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |
| 1  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |
| 0  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |
|  | 0              | 1              | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |
| 4  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |
| 3  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |
| 2  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |
| 1  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |
| 0  | X              |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |

|  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
|--|----------------|----------------|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|---|--|--|--|--|--|--|---|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|---|--|--|--|--|---|---|--|--|--|--|---|
| <b>Input:</b>  | <b>Output:</b> | <b>Reason:</b> |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| <div>State:<table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td>O</td><td></td><td></td><td></td><td></td></tr></table></div> <div>p = 'X'<br/>c = 0</div> |                | 0              | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 |  |  |  |  |  | 1 |  |  |  |  |  | 0 | O |  |  |  |  | <div>State:<table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>X</td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td>O</td><td></td><td></td><td></td><td></td></tr></table></div> |  | 0 | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 |  |  |  |  |  | 1 | X |  |  |  |  | 0 | O |  |  |  |  | <div>This test case is unique and distinct because it tests that placeToken I am placing token X in a column that is already filled with a token, and also the column was not close to being full.</div> <div>Function Name:</div> <div>test_placeToken_in_partly_filled_column</div> |
|  | 0              | 1              | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| 4  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| 3  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| 2  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| 1  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| 0  | O              |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
|  | 0              | 1              | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| 4  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| 3  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| 2  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| 1  | X              |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| 0  | O              |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |  |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |

|  |                |                |   |   |   |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |  |   |   |   |   |   |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
|--|----------------|----------------|---|---|---|---|---|--|--|--|--|--|---|---|--|--|--|--|---|---|--|--|--|--|---|---|--|--|--|--|---|---|--|--|--|--|---|--|---|---|---|---|---|---|---|--|--|--|--|---|---|--|--|--|--|---|---|--|--|--|--|---|---|--|--|--|--|---|---|--|--|--|--|---|
| <b>Input:</b>  | <b>Output:</b> | <b>Reason:</b> |   |   |   |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |  |   |   |   |   |   |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| State: <table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td>O</td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td>O</td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>O</td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td>O</td><td></td><td></td><td></td><td></td></tr></table> <p>p = 'X'</p> <p>c = 0</p> |                | 0              | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 | O |  |  |  |  | 2 | O |  |  |  |  | 1 | O |  |  |  |  | 0 | O |  |  |  |  | State: <table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td>X</td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td>O</td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td>O</td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>O</td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td>O</td><td></td><td></td><td></td><td></td></tr></table> |  | 0 | 1 | 2 | 3 | 4 | 4 | X |  |  |  |  | 3 | O |  |  |  |  | 2 | O |  |  |  |  | 1 | O |  |  |  |  | 0 | O |  |  |  |  | <p>This test case is unique and distinct because it tests that placeToken I am placing token X in a column that is almsot full, and also after placing the token in column 0, it filled up the column.</p> <p><b>Function Name:</b></p> <p>test_placeToken_fill_up_column</p> |
|  | 0              | 1              | 2 | 3 | 4 |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |  |   |   |   |   |   |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| 4  |                |                |   |   |   |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |  |   |   |   |   |   |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| 3  | O              |                |   |   |   |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |  |   |   |   |   |   |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| 2  | O              |                |   |   |   |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |  |   |   |   |   |   |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| 1  | O              |                |   |   |   |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |  |   |   |   |   |   |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| 0  | O              |                |   |   |   |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |  |   |   |   |   |   |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
|  | 0              | 1              | 2 | 3 | 4 |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |  |   |   |   |   |   |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| 4  | X              |                |   |   |   |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |  |   |   |   |   |   |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| 3  | O              |                |   |   |   |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |  |   |   |   |   |   |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| 2  | O              |                |   |   |   |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |  |   |   |   |   |   |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| 1  | O              |                |   |   |   |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |  |   |   |   |   |   |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |
| 0  | O              |                |   |   |   |   |   |  |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |  |   |   |   |   |   |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |   |  |  |  |  |   |

|  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |  |   |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |
|--|----------------|----------------|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|---|---|---|---|--|---|--|---|---|---|---|---|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|--|--|--|--|--|---|---|---|---|---|---|--|
| <b>Input:</b>  | <b>Output:</b> | <b>Reason:</b> |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |  |   |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |
| State: <table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td>O</td><td>O</td><td>O</td><td>O</td><td></td></tr></table> <p>p = 'X'</p> <p>c = 4</p> |                | 0              | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 |  |  |  |  |  | 1 |  |  |  |  |  | 0 | O | O | O | O |  | State: <table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td>O</td><td>O</td><td>O</td><td>O</td><td>X</td></tr></table> |  | 0 | 1 | 2 | 3 | 4 | 4 |  |  |  |  |  | 3 |  |  |  |  |  | 2 |  |  |  |  |  | 1 |  |  |  |  |  | 0 | O | O | O | O | X | <p>This test case is unique and distinct because it tests that placeToken I am placing token X in a row that is almsot full, and also after placing the token in row 4, it fills up the row.</p> <p><b>Function Name:</b></p> <p>test_placeToken_fill_up_row</p> |
|  | 0              | 1              | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |  |   |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |
| 4  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |  |   |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |
| 3  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |  |   |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |
| 2  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |  |   |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |
| 1  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |  |   |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |
| 0  | O              | O              | O | O |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |  |   |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |
|  | 0              | 1              | 2 | 3 | 4 |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |  |   |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |
| 4  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |  |   |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |
| 3  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |  |   |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |
| 2  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |  |   |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |
| 1  |                |                |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |  |   |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |
| 0  | O              | O              | O | O | X |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |  |   |  |   |   |   |   |   |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |  |  |  |  |  |   |   |   |   |   |   |  |

|   |                |                |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |
|---|----------------|----------------|---|---|---|---|---|---|---|---|---|--|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|--|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|--|
| <b>Input:</b>   | <b>Output:</b> | <b>Reason:</b> |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |
| State: <table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td>O</td><td>O</td><td>O</td><td>O</td><td></td></tr><tr><td>3</td><td>O</td><td>O</td><td>O</td><td>O</td><td>O</td></tr><tr><td>2</td><td>O</td><td>O</td><td>O</td><td>O</td><td>O</td></tr><tr><td>1</td><td>O</td><td>O</td><td>O</td><td>O</td><td>O</td></tr><tr><td>0</td><td>O</td><td>O</td><td>O</td><td>O</td><td>O</td></tr></table> <p>p = 'X'<br/>c = 4</p> |                | 0              | 1 | 2 | 3 | 4 | 4 | O | O | O | O |  | 3 | O | O | O | O | O | 2 | O | O | O | O | O | 1 | O | O | O | O | O | 0 | O | O | O | O | O | State: <table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>4</td><td>O</td><td>O</td><td>O</td><td>O</td><td>X</td></tr><tr><td>3</td><td>O</td><td>O</td><td>O</td><td>O</td><td>O</td></tr><tr><td>2</td><td>O</td><td>O</td><td>O</td><td>O</td><td>O</td></tr><tr><td>1</td><td>O</td><td>O</td><td>O</td><td>O</td><td>O</td></tr><tr><td>0</td><td>O</td><td>O</td><td>O</td><td>O</td><td>O</td></tr></table> |  | 0 | 1 | 2 | 3 | 4 | 4 | O | O | O | O | X | 3 | O | O | O | O | O | 2 | O | O | O | O | O | 1 | O | O | O | O | O | 0 | O | O | O | O | O | <p>This test case is unique and distinct because it tests that placeToken I am placing token X in a board that is almost full, and after placing the token in column 4, the entire board will be full.</p> <p><b>Function Name:</b></p> <p>test_placeToken_to_fill_board</p> |
|   | 0              | 1              | 2 | 3 | 4 |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |
| 4   | O              | O              | O | O |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |
| 3   | O              | O              | O | O | O |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |
| 2   | O              | O              | O | O | O |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |
| 1   | O              | O              | O | O | O |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |
| 0   | O              | O              | O | O | O |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |
|   | 0              | 1              | 2 | 3 | 4 |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |
| 4   | O              | O              | O | O | X |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |
| 3   | O              | O              | O | O | O |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |
| 2   | O              | O              | O | O | O |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |
| 1   | O              | O              | O | O | O |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |
| 0   | O              | O              | O | O | O |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |