1. `GameBoard(int rows, int columns, int numToWin)`

| | | |
|---|---|---|
| **Input: 3, 3, 3**<br>State: | **Output:** `getNumRows()=3`<br>`getNumColumns()=3`<br>`getNumToWin() = 3`<br>State of the board is unchanged | **Reason:** Testing the minimum boundary of values to make sure they are properly assigned<br>**Function Name:** `constructorTestAssignmentMin()` |

2. `GameBoard(int rows, int columns, int numToWin)`

| | | |
|---|---|---|
| **Input: 100, 100, 25**<br>State: | **Output:**<br>`getNumRows()=100`<br>`getNumColumns()=100`<br>`getNumToWin() = 25`<br>State of the board is unchanged | **Reason:** Testing the maximum value to make sure they are properly assigned<br>**Function Name:** `constructorTestAssignmentMax()` |

3. `GameBoard(int rows, int columns, int numToWin)`

| | | |
|---|---|---|
| **Input: 8, 8, 4**<br>State: | **Output:** `getNumRows()=8`<br>`getNumColumns()=8`<br>`getNumToWin() = 4`<br>State of the board is unchanged | **Reason:** Testing a normal value (Normal defined as not a boundary) to make sure they are properly assigned<br>**Function Name:** `constructorTestAssignmentNormal()` |

1. `placeMarker(BoardPosition pos, char player)`

<table>
<tr>
<td>

**Input:**

State:

<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td>0</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>2</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

player = ‘X’<br>
Pos.getRow = 2<br>
Pos.getCol = 2

</td>
<td>

**Output:**

<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td>0</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>2</td><td></td><td></td><td>X</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

</td>
<td>

**Reason:** This test case is unique because we are testing on an empty board where no values exist. So this method tests to make sure the board was properly initialized

**Function Name:** `placeMarkerOnEmptyBoard()`

</td>
</tr>
</table>

2. `placeMarker(BoardPosition pos, char player)`

<table>
<tr>
<td><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><td>5</td><td>6</td><td>7</td></tr>
<tr><td>0</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>2</td><td></td><td></td><td>X</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
<br>player = ‘X’<br>Pos.getRow = 0<br>Pos.getCol = 0</td>
<td><b>Output:</b><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>0</td><td>X</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>2</td><td></td><td></td><td>X</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
</td>
<td><b>Reason:</b> This test case is unique because we are testing to make sure the board's (0,0) position is the top left and make sure there are no out of bounds problems.<br><br><b>Function Name:</b><br><code>placeMarkerTopLeft()</code></td>
</tr>
</table>

3. `placeMarker(BoardPosition pos, char player)`

<table>
<tr>
<td><b>Input:</b><br><br>State:<br>
<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td>0</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>2</td><td></td><td></td><td>X</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
player = ‘X’<br>Pos.getRow = 7<br>Pos.getCol = 7</td>
<td><b>Output:</b><br>
<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td>0</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>2</td><td></td><td></td><td>X</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td></tr>
</table></td>
<td><b>Reason:</b> This test case is unique because we are testing to make sure the board's (7,7) position is the bottom right and make sure there are no out of bounds problems.<br><br><b>Function Name:</b><br><code>placeMarkerTopLeft()</code></td>
</tr>
</table>

4. `placeMarker(BoardPosition pos, char player)`

<table>
<tr>
<td>

**Input:**

State:

<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>0</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>2</td><td></td><td></td><td>X</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

player = ‘X’<br>Pos.getRow = 7<br>Pos.getCol = 0

</td>
<td>

**Output:**

<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>0</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>2</td><td></td><td></td><td>X</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

</td>
<td>

**Reason:** This test case is unique because we are testing to make sure the board's (7,0) position is the bottom left and make sure there are no out of bounds problems. On top of this we are also testing to make sure row and column correspond to their proper location on the board (row,column)

**Function Name:** `placeMarkerOnRowBoundary()`

</td>
</tr>
</table>

5. `placeMarker(BoardPosition pos, char player)`

<table>
<tr>
<td>

**Input:**

State:

<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td>0</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>2</td><td></td><td></td><td>X</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

player = ‘X’<br>Pos.getRow = 0<br>Pos.getCol = 7

</td>
<td>

**Output:**

<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</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>2</td><td></td><td></td><td>X</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

</td>
<td>

**Reason:** This test case is unique because we are testing to make sure the board's (0,7) position is the top right and make sure there are no out of bounds problems. On top of this we are also testing to make sure row and column correspond to their proper location on the board (row,column)

**Function Name:** `placeMarkerOnColumnBoundary()`

</td>
</tr>
</table>

1. whatsAtPos(BoardPosition pos)

<table>
<tr>
<td><b>Input:</b><br><br>State:<br>
<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td>0</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>2</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
<br>player = ‘X’<br>Pos.getRow = 4<br>Pos.getCol = 4</td>
<td><b>Output:</b><br><br>whatsAtPos = ‘ ‘</td>
<td><b>Reason:</b> This test case is unique because we are testing to make sure whatsAtPos functions on an empty board.<br><br><b>Function Name:</b><br>whatsAtPosOnEmptyBoard()</td>
</tr>
</table>

2. whatsAtPos(BoardPosition pos)

<table>
<tr>
<td><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><td>5</td><td>6</td><td>7</td></tr>
<tr><td>0</td><td>X</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>2</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
<br>player = ‘X’<br>Pos.getRow = 0<br>Pos.getCol = 0</td>
<td><b>Output:</b><br><br>whatsAtPos = ‘X‘</td>
<td><b>Reason:</b> This test case is unique because we are testing to make sure whatsAtPos is working with the board (0,0) being at the top left and that it's properly grabbing the character that was placed there<br><br><b>Function Name:</b><br>whatsAtPosOnMarkedTopLeft()</td>
</tr>
</table>

3. `whatsAtPos(BoardPosition pos)`

<table>
<tr>
<td><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><td>5</td><td>6</td><td>7</td></tr>
<tr><td>0</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>2</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td></tr>
</table>
<br>player = ‘X’<br>Pos.getRow = 7<br>Pos.getCol = 7</td>
<td><b>Output:</b><br><br>whatsAtPos = ‘X‘</td>
<td><b>Reason:</b> This test case is unique because we are testing to make sure whatsAtPos is working with the board (7,7) being bottom right and that it's properly grabbing the character that was placed there without going out of bounds. This is the boundary for both row and column<br><br><b>Function Name:</b><br><code>whatsAtPosOnMarkedBottomRight()</code></td>
</tr>
</table>

4. `whatsAtPos(BoardPosition pos)`

<table>
<tr>
<td><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><td>5</td><td>6</td><td>7</td></tr>
<tr><td>0</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>2</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
<br>player = ‘X’<br>Pos.getRow = 7<br>Pos.getCol = 0</td>
<td><b>Output:</b><br><br>whatsAtPos = ‘X‘</td>
<td><b>Reason:</b> This test case is unique because we are testing to make sure whatsAtPos is working with the board (7,0) being bottom right and that it's properly grabbing the character that was placed there without going out of bounds. This is the upper boundary for row and lower boundary for column<br><br><b>Function Name:</b><br><code>whatsAtPosOnMarkedBottomLeft()</code></td>
</tr>
</table>

5. `whatsAtPos(BoardPosition pos)`

<table>
<tr>
<td><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><td>5</td><td>6</td><td>7</td></tr>
<tr><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</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>2</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
<br>player = ‘X’<br>Pos.getRow = 0<br>Pos.getCol = 7</td>
<td><b>Output:</b><br><br>whatsAtPos = ‘X‘</td>
<td><b>Reason:</b> This test case is unique because we are testing to make sure whatsAtPos is working with the board (0,7) being bottom right and that it's properly grabbing the character that was placed there without going out of bounds. This is the lower boundary for row and upper boundary for column<br><br><b>Function Name:</b><br><code>whatsAtPosOnMarkedTopRight()</code></td>
</tr>
</table>

1. `checkSpace(BoardPosition pos)`

<table>
<tr>
<td><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><td>5</td><td>6</td><td>7</td></tr>
<tr><td>0</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>2</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
<br>Pos.getRow = 4<br>Pos.getCol = 4</td>
<td><b>Output:</b><br><code>checkSpace = true</code><br><br>State of the board is unchanged</td>
<td><b>Reason:</b> This test case is unique because we are testing on a board that is empty. Therefore checkSpace must return true every single time.<br><br><b>Function Name:</b><br><code>checkSpacePlacementEmptyBoard()</code></td>
</tr>
</table>

2. `checkSpace(BoardPosition pos)`

<table>
<tr>
<td><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><td>5</td><td>6</td><td>7</td></tr>
<tr><td>0</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td></tr>
<tr><td>1</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td></tr>
<tr><td>2</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td></tr>
<tr><td>3</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td></tr>
<tr><td>4</td><td>X</td><td>O</td><td>X</td><td>O</td><td></td><td>O</td><td>X</td><td>O</td></tr>
<tr><td>5</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td></tr>
<tr><td>6</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td></tr>
<tr><td>7</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td></tr>
</table>
<br>Pos.getRow = 4<br>Pos.getCol = 4</td>
<td><b>Output:</b><br><code>checkSpace = true</code><br><br>State of the board is unchanged</td>
<td><b>Reason:</b> This test case is unique because we are testing on a board that is almost full with one remaining location. The method can only return true for this particular value and if it does it's a good indication it's working properly<br><br><b>Function Name:</b><br><code>checkSpacePlacementOneAwayFromFullBoard()</code></td>
</tr>
</table>

`3. checkSpace(BoardPosition pos)`

<table>
<tr>
<td><b>Input:</b><br><br>State:<br>
<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td>0</td><td>x</td><td>O</td><td></td><td>X</td><td></td><td>O</td><td>X</td><td>O</td></tr>
<tr><td>1</td><td></td><td>O</td><td>X</td><td>O</td><td>X</td><td>X</td><td></td><td>X</td></tr>
<tr><td>2</td><td>x</td><td>X</td><td>X</td><td>O</td><td></td><td>O</td><td>X</td><td>O</td></tr>
<tr><td>3</td><td>x</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td></td><td>O</td></tr>
<tr><td>4</td><td>X</td><td>X</td><td></td><td>O</td><td></td><td>X</td><td></td><td>X</td></tr>
<tr><td>5</td><td></td><td>O</td><td></td><td>O</td><td></td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>6</td><td>x</td><td>X</td><td>X</td><td>O</td><td>X</td><td>X</td><td></td><td>X</td></tr>
<tr><td>7</td><td>x</td><td>X</td><td>X</td><td>X</td><td></td><td>X</td><td></td><td>X</td></tr>
</table>
<br>Pos.getRow = 3<br>Pos.getCol = 3</td>
<td><b>Output:</b><br><code>checkSpace = false</code><br><br>State of the board is unchanged</td>
<td><b>Reason:</b> This test case is unique because we are testing on a board that represents a game board that could happen during live play. We are checking a position already taken to make sure it returns false<br><br><b>Function Name:</b><br><code>checkSpacePlacementNormalBoard()</code></td>
</tr>
</table>

1. `checkHorizontalWin(BoardPosition lastPos, char player)`

<table>
<tr>
<td>

**Input:**

State:

<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td>0</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</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>2</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

player = ‘X’<br>NumToWin = 4<br>Pos.getRow = 2<br>Pos.getCol = 2

</td>
<td>

**Output:**
`checkHorizontalWin = true`

State of the board is unchanged

</td>
<td>

**Reason:** This test case is unique because we are testing to make sure that if the last mark we placed connected into a horizontal line that the method can properly detect the new horizontal line that was found through the middle “position”.

**Function Name:**
`checkHorizontalWinPlacedInMiddle()`

</td>
</tr>
</table>

2. checkHorizontalWin(BoardPosition lastPos, char player)

<table>
<tr>
<td>

**Input:**

State:

<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>0</td><td>X</td><td>X</td><td>X</td><td></td><td>X</td><td>X</td><td>X</td><td></td></tr>
<tr><td>1</td><td>X</td><td>X</td><td>X</td><td></td><td>X</td><td>X</td><td>X</td><td></td></tr>
<tr><td>2</td><td>X</td><td>X</td><td>X</td><td></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></td><td>X</td><td>X</td><td>X</td><td></td></tr>
<tr><td>4</td><td>X</td><td>X</td><td>X</td><td></td><td>X</td><td>X</td><td>X</td><td></td></tr>
<tr><td>5</td><td>X</td><td>X</td><td>X</td><td></td><td>X</td><td>X</td><td>X</td><td></td></tr>
<tr><td>6</td><td>X</td><td>X</td><td>X</td><td></td><td>X</td><td>X</td><td>X</td><td></td></tr>
<tr><td>7</td><td>X</td><td>X</td><td>X</td><td></td><td>X</td><td>X</td><td>X</td><td></td></tr>
</table>

player = ‘X’<br>numToWin = 4<br>Pos.getRow = 2<br>Pos.getCol = 2

</td>
<td>

**Output:**<br>checkHorizontalWin = false

State of the board is unchanged

</td>
<td>

**Reason:** This test case is unique because we are testing to make sure that there are no false positives detected. We are filling every row to one away from it's win and then putting a space in between it so it's no longer a consecutive string of characters

**Function Name:**<br>checkHorizontalWin PlacedInMiddle()

</td>
</tr>
</table>

3. `checkHorizontalWin(BoardPosition lastPos, char player)`

<table>
<tr>
<td>
<b>Input:</b><br><br>State:<br>
<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td>0</td><td>X</td><td>X</td><td>X</td><td></td><td>X</td><td>X</td><td>X</td><td></td></tr>
<tr><td>1</td><td>X</td><td>X</td><td>X</td><td></td><td>X</td><td>X</td><td>X</td><td></td></tr>
<tr><td>2</td><td>X</td><td>X</td><td>X</td><td></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></td><td>X</td><td>X</td><td>X</td><td></td></tr>
<tr><td>4</td><td>X</td><td>X</td><td>X</td><td></td><td>X</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>5</td><td>X</td><td>X</td><td>X</td><td></td><td>X</td><td>X</td><td>X</td><td></td></tr>
<tr><td>6</td><td>X</td><td>X</td><td>X</td><td></td><td>X</td><td>X</td><td>X</td><td></td></tr>
<tr><td>7</td><td>X</td><td>X</td><td>X</td><td></td><td>X</td><td>X</td><td>X</td><td></td></tr>
</table>
player = ‘X’<br>numToWin = 4<br>Pos.getRow = 4<br>Pos.getCol = 7
</td>
<td>
<b>Output:</b><br><code>checkHorizontalWin = true</code><br><br>State of the board is unchanged
</td>
<td>
<b>Reason:</b> This test case is unique because we are testing to make sure that there are no false positives as well that the board does detect the one valid horizontal win. On top of detecting the win we are checking to make sure the fact that the horizontal reaches the column's boundary does not affect the win detection<br><br><b>Function Name:</b><br><code>checkHorizontalDetectingProperWin()</code>
</td>
</tr>
</table>

4. `checkHorizontalWin(BoardPosition lastPos, char player)`

<table>
<tr>
<td><b>Input:</b><br><br>State:<br>
<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td>0</td><td>O</td><td>O</td><td>O</td><td></td><td>O</td><td>O</td><td>O</td><td></td></tr>
<tr><td>1</td><td>X</td><td>X</td><td>X</td><td></td><td>X</td><td>X</td><td>X</td><td></td></tr>
<tr><td>2</td><td>O</td><td>O</td><td>O</td><td></td><td>O</td><td>O</td><td>O</td><td></td></tr>
<tr><td>3</td><td>X</td><td>X</td><td>X</td><td></td><td>X</td><td>X</td><td>X</td><td></td></tr>
<tr><td>4</td><td>O</td><td>O</td><td>O</td><td>X</td><td>O</td><td>O</td><td>O</td><td>O</td></tr>
<tr><td>5</td><td>X</td><td>X</td><td>X</td><td></td><td>X</td><td>X</td><td>X</td><td></td></tr>
<tr><td>6</td><td>O</td><td>O</td><td>O</td><td></td><td>O</td><td>O</td><td>O</td><td></td></tr>
<tr><td>7</td><td>X</td><td>X</td><td>X</td><td></td><td>X</td><td>X</td><td>X</td><td></td></tr>
</table>
<br>player = ‘X’<br>numToWin = 4<br>Pos.getRow = 4<br>Pos.getCol = 7</td>
<td><b>Output:</b><br><code>checkHorizontalWin = true</code><br><br>State of the board is unchanged</td>
<td><b>Reason:</b> This test case is unique because we are testing to make sure that there are no false positives from having alternating characters as well as if another character interrupts a particular character it's horizontal streak is no longer counted.<br><br><b>Function Name:</b><br><code>checkHorizontalWinWithMixedChars()</code></td>
</tr>
</table>

1. `checkVerticalWin(BoardPosition lastPos, char player)`

<table>
<tr>
<td><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><td>5</td><td>6</td><td>7</td></tr>
<tr><td>0</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>2</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>4</td><td>X</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
player = ‘X’<br>NumToWin = 4<br>Pos.getRow = 0<br>Pos.getCol = 0</td>
<td><b>Output:</b><br><code>checkVerticalWin= true</code><br><br>State of the board is unchanged</td>
<td><b>Reason:</b> This test case is unique because we are testing to make sure that if the last mark we placed connected into a vertical line that the method can properly detect the new vertical line given the middle position in between the line as the lastPos<br><br><b>Function Name:</b><br><code>checkVerticalWinPlacedInMiddle()</code></td>
</tr>
</table>

2. `checkVerticalWin(BoardPosition lastPos, char player)`

<table>
<tr>
<td>
<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><td>5</td><td>6</td><td>7</td></tr>
<tr><td>0</td><td>X</td><td>X</td><td>X</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><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><td>X</td><td>X</td><td>X</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>4</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>5</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>6</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
player = ‘X’<br>numToWin = 4<br>Pos.getRow = 2<br>Pos.getCol = 2
</td>
<td>
<b>Output:</b><br><code>checkVerticalWin = false</code><br><br>State of the board is unchanged
</td>
<td>
<b>Reason:</b> This test case is unique because we are testing to make sure that there are no false positives detected. We are filling every column to one away from it's win and then putting a space in between it so it's no longer a consecutive string of characters<br><br><b>Function Name:</b><br><code>checkVerticalWinNoFalsePositive()</code>
</td>
</tr>
</table>

3. `checkVerticalWin(BoardPosition lastPos, char player)`

<table>
<tr>
<td><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><td>5</td><td>6</td><td>7</td></tr>
<tr><td>0</td><td>X</td><td>X</td><td>X</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><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><td>X</td><td>X</td><td>X</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>4</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>5</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>6</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>7</td><td></td><td></td><td></td><td>X</td><td></td><td></td><td></td><td></td></tr>
</table>
player = ‘X’<br>numToWin = 4<br>Pos.getRow = 7<br>Pos.getCol = 4</td>
<td><b>Output:</b><br><code>checkVerticalWin = true</code><br><br>State of the board is unchanged</td>
<td><b>Reason:</b> This test case is unique because we are testing to make sure that there are no false positives as well that the board does detect the one valid vertical win. On top of detecting the win we are checking to make sure the fact that the vertical reaches the rows boundary does not affect the win detection<br><br><b>Function Name:</b><br><code>checkHorizontal DetectingProperWin()</code></td>
</tr>
</table>

4. `checkVerticalWin(BoardPosition lastPos, char player)`

<table>
<tr>
<td><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><td>5</td><td>6</td><td>7</td></tr>
<tr><td>0</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td></tr>
<tr><td>1</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td></tr>
<tr><td>2</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td></tr>
<tr><td>3</td><td></td><td></td><td></td><td></td><td>X</td><td></td><td></td><td></td></tr>
<tr><td>4</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td></tr>
<tr><td>5</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td></tr>
<tr><td>6</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td><td>X</td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td>O</td><td></td><td></td><td></td></tr>
</table>
player = ‘O’<br>numToWin = 4<br>Pos.getRow = 7<br>Pos.getCol = 4</td>
<td><b>Output:</b><br><code>checkVerticalWin = true</code><br><br>State of the board is unchanged</td>
<td><b>Reason:</b> This test case is unique because we are testing to make sure that there are no false positives from having alternating characters as well as if another character interrupts a particular character it's vertical streak is no longer counted.<br><br><b>Function Name:</b><br><code>checkHorizontalDetectingProperWin()</code></td>
</tr>
</table>

1. `checkDiagonalWin(BoardPosition lastPos, char player)`

<table>
<tr>
<td><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><td>5</td><td>6</td><td>7</td></tr>
<tr><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td></tr>
<tr><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td></tr>
<tr><td>2</td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td><td></td></tr>
<tr><td>3</td><td></td><td></td><td></td><td></td><td>X</td><td></td><td></td><td></td></tr>
<tr><td>4</td><td></td><td></td><td></td><td>X</td><td></td><td></td><td></td><td></td></tr>
<tr><td>5</td><td></td><td></td><td>X</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>6</td><td></td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
player = ‘X’<br>NumToWin = 4<br>Pos.getRow = 4<br>Pos.getCol = 3</td>
<td><b>Output:</b><br><code>checkDiagonalWin= true</code><br><br>State of the board is unchanged</td>
<td><b>Reason:</b> This test case is unique because we are testing to make sure that the right diagonal is being properly detected when it cuts through the center board<br><br><b>Function Name:</b><br><code>checkDiagonalWin StandardLeftDiagonal()</code></td>
</tr>
</table>

2. `checkDiagonalWin(BoardPosition lastPos, char player)`

<table>
<tr>
<td><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><td>5</td><td>6</td><td>7</td></tr>
<tr><td>0</td><td>X</td><td></td><td></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><td></td><td></td><td></td></tr>
<tr><td>2</td><td></td><td></td><td>X</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>3</td><td></td><td></td><td></td><td>X</td><td></td><td></td><td></td><td></td></tr>
<tr><td>4</td><td></td><td></td><td></td><td></td><td>X</td><td></td><td></td><td></td></tr>
<tr><td>5</td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td><td></td></tr>
<tr><td>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td></tr>
</table>
<br>player = ‘X’<br>NumToWin = 4<br>Pos.getRow = 4<br>Pos.getCol = 4</td>
<td><b>Output:</b><br><code>checkDiagonalWin= true</code><br><br>State of the board is unchanged</td>
<td><b>Reason:</b> This test case is unique because we are testing to make sure that the left diagonal is being properly detected when it cuts through the center board<br><br><b>Function Name:</b><br><code>checkDiagonalWin StandardLeftDiagonal()</code></td>
</tr>
</table>

3. checkDiagonalWin(BoardPosition lastPos, char player)

<table>
<tr>
<td><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><td>5</td><td>6</td><td>7</td><td>8</td></tr>
<tr><td>0</td><td></td><td></td><td></td><td></td><td>X</td><td></td><td></td><td></td><td></td></tr>
<tr><td>1</td><td></td><td></td><td></td><td>X</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><td></td><td></td><td></td><td></td></tr>
<tr><td>3</td><td></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><td></td></tr>
</table>
player = ‘X’<br>NumToWin = 4<br>Pos.getRow = 1<br>Pos.getCol = 3</td>
<td><b>Output:</b><br>checkDiagonalWin= true<br><br>State of the board is unchanged</td>
<td><b>Reason:</b> This test case is unique because we are testing to make sure that the right diagonal is not affected when board size changes<br><br><b>Function Name:</b><br>checkDiagonalWinOnRightDB()</td>
</tr>
</table>

4. `checkDiagonalWin(BoardPosition lastPos, char player)`

<table>
<tr>
<td><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><td>5</td><td>6</td><td>7</td><td>8</td></tr>
<tr><td>0</td><td></td><td></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><td></td></tr>
<tr><td>2</td><td></td><td>X</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>X</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>X</td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
player = ‘X’<br>NumToWin = 4<br>Pos.getRow = 4<br>Pos.getCol = 4</td>
<td><b>Output:</b><br><code>checkDiagonalWin= true</code><br><br>State of the board is unchanged</td>
<td><b>Reason:</b> This test case is unique because we are testing to make sure that the left diagonal is not affected when board size changes<br><br><b>Function Name:</b><br><code>checkDiagonalWinOnLeftDB()</code></td>
</tr>
</table>

5. `checkDiagonalWin(BoardPosition lastPos, char player)`

<table>
<tr>
<td><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>0</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>2</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>4</td><td></td><td></td><td></td><td></td><td>X</td></tr>
<tr><td>5</td><td></td><td></td><td></td><td>X</td><td></td></tr>
<tr><td>6</td><td></td><td></td><td>X</td><td></td><td></td></tr>
<tr><td>7</td><td></td><td>X</td><td></td><td></td><td></td></tr>
<tr><td>8</td><td>X</td><td></td><td></td><td></td><td></td></tr>
</table>
player = ‘X’<br>NumToWin = 4<br>Pos.getRow = 8<br>Pos.getCol = 0</td>
<td><b>Output:</b><br><code>checkDiagonalWin= true</code><br><br>State of the board is unchanged</td>
<td><b>Reason:</b> This test case is unique because we are testing to make sure that the right diagonal is not affected when board size changes. Specifically when total rows is greater than columns as this affects how the diagonal is calculated<br><br><b>Function Name:</b><br><code>checkDiagonalWinOnRightWithRHigherThanC()</code></td>
</tr>
</table>

6. `checkDiagonalWin(BoardPosition lastPos, char player)`

<table>
<tr>
<td><b>Input:</b><br><br>State:<br>
<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th></tr>
<tr><td>0</td><td>X</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>2</td><td></td><td></td><td>X</td><td></td><td></td></tr>
<tr><td>3</td><td></td><td></td><td></td><td>X</td><td></td></tr>
<tr><td>4</td><td></td><td></td><td></td><td></td><td>X</td></tr>
<tr><td>5</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></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>8</td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
player = ‘X’<br>NumToWin = 4<br>Pos.getRow = 0<br>Pos.getCol = 0</td>
<td><b>Output:</b><br><code>checkDiagonalWin= true</code><br><br>State of the board is unchanged</td>
<td><b>Reason:</b> This test case is unique because we are testing to make sure that the left diagonal is not affected when board size changes. Specifically when total rows is greater than columns as this affects how the diagonal is calculated<br><br><b>Function Name:</b><br><code>checkDiagonalWinOnLeftWithRHigherThanC()</code></td>
</tr>
</table>

7. `checkDiagonalWin(BoardPosition lastPos, char player)`

<table>
<tr>
<td><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><td>5</td><td>6</td><td>7</td></tr>
<tr><td>0</td><td>x</td><td>O</td><td></td><td>X</td><td></td><td>O</td><td>X</td><td>O</td></tr>
<tr><td>1</td><td></td><td>O</td><td>X</td><td>O</td><td>X</td><td>X</td><td></td><td>X</td></tr>
<tr><td>2</td><td>x</td><td>X</td><td>X</td><td>O</td><td></td><td>O</td><td>X</td><td>O</td></tr>
<tr><td>3</td><td>x</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td></td><td>O</td></tr>
<tr><td>4</td><td>X</td><td>X</td><td></td><td>O</td><td></td><td>X</td><td></td><td>X</td></tr>
<tr><td>5</td><td></td><td>O</td><td></td><td>O</td><td></td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>6</td><td>x</td><td>X</td><td>X</td><td>O</td><td>X</td><td>X</td><td></td><td>X</td></tr>
<tr><td>7</td><td>x</td><td>X</td><td>X</td><td>X</td><td></td><td>X</td><td></td><td>X</td></tr>
</table>
<br>player = ‘X’<br>Pos.getRow = 6<br>Pos.getCol = 7</td>
<td><b>Output:</b><br><code>checkDiagonalWin = false</code><br><br>State of the board is unchanged</td>
<td><b>Reason:</b> This test case is unique because we are testing on a board that represents a game board that could happen during live play. There are different players and many one away wins but only one diagonal that's valid on the entire board. Gives good indication that the diagonal is being calculated properly<br><br><b>Function Name:</b><br><code>checkSpacePlacementNormalBoard()</code></td>
</tr>
</table>

1. `checkForDraw()`

<table>
<tr>
<td><b>Input:</b><br><br>State:<br>
<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td>0</td><td>X</td><td>O</td><td>O</td><td>O</td><td>X</td><td>O</td><td>O</td><td>O</td></tr>
<tr><td>1</td><td>O</td><td>X</td><td>X</td><td>X</td><td>O</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>2</td><td>X</td><td>O</td><td>O</td><td>O</td><td>X</td><td>O</td><td>O</td><td>O</td></tr>
<tr><td>3</td><td>O</td><td>X</td><td>X</td><td>X</td><td>O</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>4</td><td>X</td><td>O</td><td>O</td><td>O</td><td>X</td><td>O</td><td>O</td><td>O</td></tr>
<tr><td>5</td><td>O</td><td>X</td><td>X</td><td>X</td><td>O</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>6</td><td>x</td><td>O</td><td>O</td><td>O</td><td>X</td><td>O</td><td>O</td><td>O</td></tr>
<tr><td>7</td><td>O</td><td>X</td><td>X</td><td>X</td><td>O</td><td>X</td><td>X</td><td>X</td></tr>
</table>
</td>
<td><b>Output:</b><br><code>checkForDraw = false</code><br><br>State of the board is unchanged</td>
<td><b>Reason:</b> This test case is relatively simple it's testing if a full board is being properly detected as a tied game<br><br><b>Function Name:</b><br><code>checkForDrawNormal()</code></td>
</tr>
</table>

2. `checkForDraw()`

<table>
<tr>
<td><b>Input:</b><br><br>State:<br>
<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td>0</td><td>X</td><td>O</td><td>O</td><td>O</td><td>X</td><td>O</td><td>O</td><td>O</td></tr>
<tr><td>1</td><td>O</td><td>X</td><td>X</td><td>X</td><td>O</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>2</td><td>X</td><td>O</td><td>O</td><td>O</td><td>X</td><td>O</td><td>O</td><td>O</td></tr>
<tr><td>3</td><td>O</td><td>X</td><td>X</td><td>X</td><td>O</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>4</td><td>X</td><td>O</td><td>O</td><td>O</td><td>X</td><td>O</td><td>O</td><td>O</td></tr>
<tr><td>5</td><td>O</td><td>X</td><td>X</td><td>X</td><td>O</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>6</td><td>x</td><td>O</td><td>O</td><td>O</td><td>X</td><td>O</td><td>O</td><td>O</td></tr>
<tr><td>7</td><td>O</td><td>X</td><td>X</td><td>X</td><td>O</td><td>X</td><td>X</td><td></td></tr>
</table>
</td>
<td><b>Output:</b><br><code>checkForDraw = false</code><br><br>State of the board is unchanged</td>
<td><b>Reason:</b> This test case is relatively simple; it's testing if a board one away from being tied is detected as a tied board. If this method works it's a very good indication checkForDraw() behaves properly. Missing char is 7,7 (row,col)<br><br><b>Function Name:</b><br><code>checkForDrawOneCharMissing()</code></td>
</tr>
</table>

3. `checkForDraw()`

<table>
<tr>
<td><b>Input:</b><br><br>State:<br>
<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td>0</td><td></td><td>O</td><td>O</td><td>O</td><td>X</td><td>O</td><td>O</td><td>O</td></tr>
<tr><td>1</td><td>O</td><td>X</td><td>X</td><td>X</td><td>O</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>2</td><td>X</td><td>O</td><td>O</td><td>O</td><td>X</td><td>O</td><td>O</td><td>O</td></tr>
<tr><td>3</td><td>O</td><td>X</td><td>X</td><td>X</td><td>O</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>4</td><td>X</td><td>O</td><td>O</td><td>O</td><td>X</td><td>O</td><td>O</td><td>O</td></tr>
<tr><td>5</td><td>O</td><td>X</td><td>X</td><td>X</td><td>O</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>6</td><td>x</td><td>O</td><td>O</td><td>O</td><td>X</td><td>O</td><td>O</td><td>O</td></tr>
<tr><td>7</td><td>O</td><td>X</td><td>X</td><td>X</td><td>O</td><td>X</td><td>X</td><td>X</td></tr>
</table>
</td>
<td><b>Output:</b><br><code>checkForDraw = false</code><br><br>State of the board is unchanged</td>
<td><b>Reason:</b> This test case is making sure the draw is covering all corners of the board. Previous tests covered the bottom right and this one covers top left in order to ensure all corners are being counted<br><br><b>Function Name:</b><br><code>checkForDrawOneCharMissingOnTop()</code></td>
</tr>
</table>

4. checkForDraw()

<table>
<tr>
<td><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><td>5</td><td>6</td><td>7</td></tr>
<tr><td>0</td><td>X</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>2</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
</td>
<td><b>Output:</b><br><code>checkForDraw = false</code><br><br>State of the board is unchanged</td>
<td><b>Reason:</b> This test case is testing the minimum possible marks on a board at the time of checkDraw being called.<br><br><b>Function Name:</b><br><code>checkForDrawOnNearlyEmptyBoard()</code></td>
</tr>
</table>

1. `isPlayerAtPos(BoardPosition pos, char player)`

<table>
<tr>
<td>
<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><td>5</td><td>6</td><td>7</td></tr>
<tr><td>0</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>2</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
player = ‘X'<br>pos.getNumCol = 0<br>pos.getNumRow = 0
</td>
<td>
<b>Output:</b><br><code>isPlayerAtPos = false</code><br><br>State of the board is unchanged
</td>
<td>
<b>Reason:</b> This test case is unique because it's testing if a player is present on the board while the board is empty. This should always return false in this instance<br><br><b>Function Name:</b><br><code>isPlayerAtPosEmptyBoard()</code>
</td>
</tr>
</table>

2. isPlayerAtPos(BoardPosition pos, char player)

**Input:**

State:

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| 0 | X | O | O | O | X | O | O | O |
| 1 | O | X | X | X | O | X | X | X |
| 2 | X | O | O | O | X | O | O | O |
| 3 | O | X | X | X | O | X | X | X |
| 4 | X | O | O | O | X | O | O | O |
| 5 | O | X | X | X | O | X | X | |
| 6 | x | O | O | O | X | O | O | O |
| 7 | O | X | X | X | O | X | X | |

**Output:**
`isPlayerAtPos = false`

State of the board is unchanged

**Reason:** This test case is unique because it's testing a board with only one available position. We are checking to see if this position is available. This gives us a very good indication if the method is behaving properly.

**Function Name:**
`isPlayerAtPosFullBoard()`

3. `isPlayerAtPos(BoardPosition pos, char player)`

<table>
<tr>
<td>

**Input:**

State:

<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>0</td><td>X</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>2</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

player = ‘X'<br>pos.getNumCol = 0<br>pos.getNumRow = 0

</td>
<td>

**Output:**<br>`isPlayerAtPos = true`

State of the board is unchanged

</td>
<td>

**Reason:** This test case is unique because it's testing if a player is present on the board while the board contains one player. We are checking the only spot where that player belongs. On top of that we also checking to make sure (0,0) lines up with the top left

**Function Name:**<br>`isPlayerAtPosOneCharOnB()`

</td>
</tr>
</table>

4. `isPlayerAtPos(BoardPosition pos, char player)`

<table>
<tr>
<td><b>Input:</b><br><br>State:<br>
<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td>0</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>2</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
player = ‘X'<br>pos.getNumCol = 0<br>pos.getNumRow = 0</td>
<td><b>Output:</b><br><code>isPlayerAtPos = true</code><br><br>State of the board is unchanged</td>
<td><b>Reason:</b> This test case is unique because it's testing if a player is present on the board while the board contains one player. We are checking the only spot where that player belongs. On top of that we also checking to make sure (7,0) lines up with the bottom left<br><br><b>Function Name:</b><br><code>isPlayerAtPosBottomLeft()</code></td>
</tr>
</table>

5. `isPlayerAtPos(BoardPosition pos, char player)`

<table>
<tr>
<td><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><td>5</td><td>6</td><td>7</td></tr>
<tr><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</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>2</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>4</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>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>7</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
player = ‘X'<br>pos.getNumCol = 0<br>pos.getNumRow = 0</td>
<td><b>Output:</b><br><code>isPlayerAtPos = true</code><br><br>State of the board is unchanged</td>
<td><b>Reason:</b> This test case is unique because it's testing if a player is present on the board while the board contains one player. We are checking the only spot where that player belongs. On top of that we also checking to make sure (0,7) lines up with the top right<br><br><b>Function Name:</b><br><code>isPlayerAtPosTopRight()</code></td>
</tr>
</table>