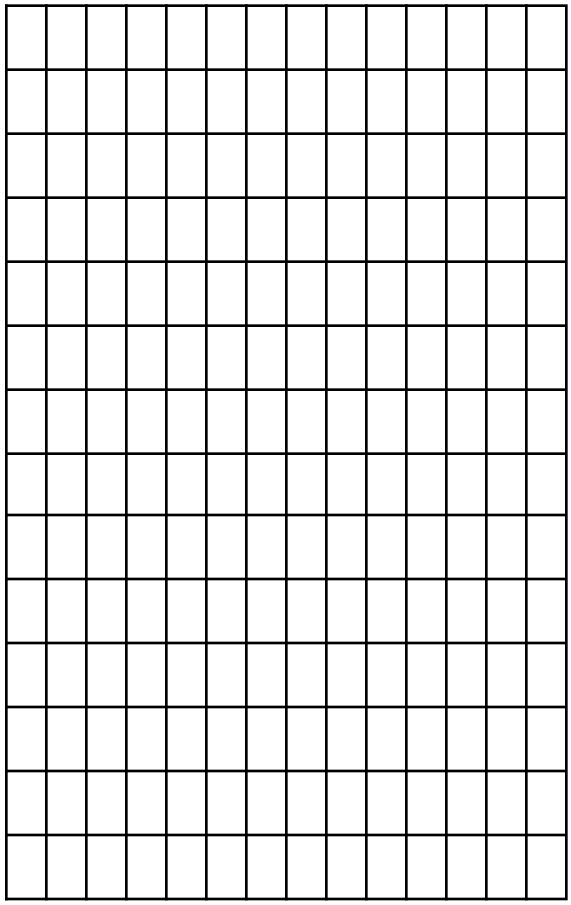
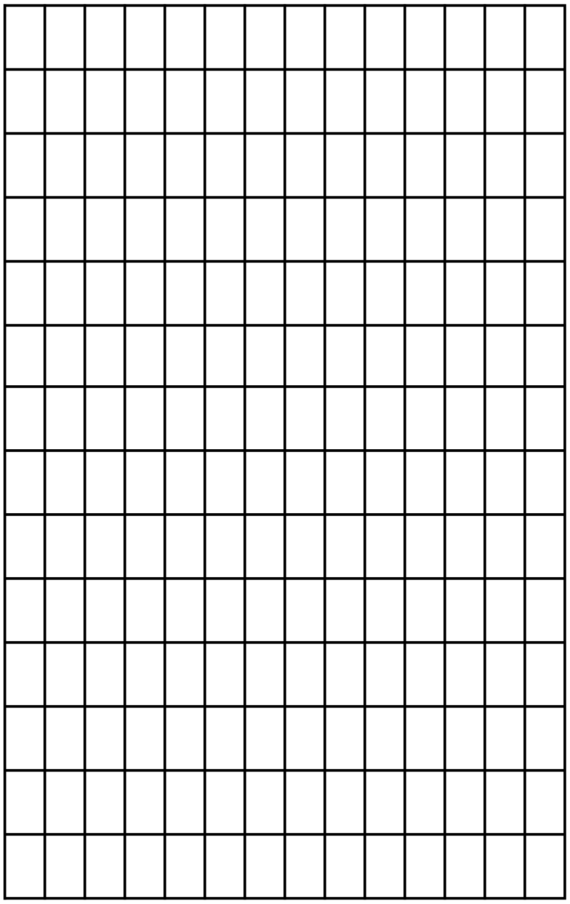
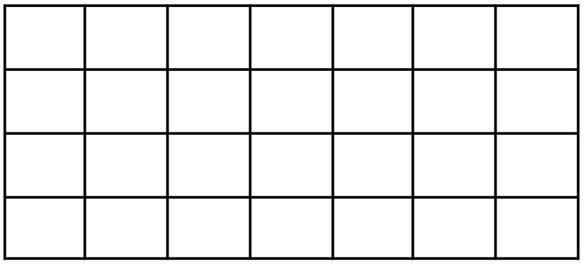
Test Cases

Constructor Input: State: 	Output: Empty game board	Reason: This test case is unique because It checks the game board is constructed with the proper number of rows and columns Function Name: testConstructor_Empty_Game Board()

Constructor Input: State: 	Output: Empty game board of blank characters	Reason: This test case is unique because it checks that the characters making up the board are blank. Function Name: testConstructor_Blank_Character_GameBoard()
Constructor Input: State: 	Output: Empty Board that has been sized down to 7x7 with a Num To Win of 3	Reason: This test case is unique because it shows that the game board can become a different size according to user preference Function Name: testConstructor_Different_Sizes

<table>
<tr>
<td>
<table>
<tr><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></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Boolean checkIfFree
Input: 0

State:
<table>
<tr><td></td><td></td><td></td><td></td><td></td><td></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><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><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><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><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><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><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><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><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><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><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><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
</td>
<td>Output:

checkIF = true
state of the board is unchanged</td>
<td>Reason:

The reason for this test case is unique
because It checks if there is a singular open position in the column for a given token.

Function name:
testCheckIfFree_On_Column_With_Players</td>
</tr>
</table>

<table>
<tr>
<td>Boolean checkIfFree

Input: 0

State:

<table>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
</td>
<td>Output:

CheckIfFree = false
State of the board remained unchanged.</td>
<td>Reason:

The reason this test case is unique is because it checks to see if a given column is full at POS.

Function Name:
testCheckIfFree_On_Full_Column</td>
</tr>
</table>

Boolean checkIfFree Input:0 State:	Output checkIfFree = true State of the game board remains unchanged	Reason: The reason this test is unique is because it tests to see if a column is free at the start of a game. Function Name: testCheckIfFree_On_Empty_Board

Boolean checkHorizWin

Input:(0,0) ,player

State:

	o	o	o	o									
x	x	x	x	x	x	x							

Output:

checkHorizWin = true

State of the game board remained unchanged

Reason:
The reason this test is unique is because the function will need to check to the right 6 more spaces.

Function Name:
testCheckHorizWin__Win_On_Left_Side

<table>
<tr>
<td>Boolean checkHorizWin

Input:(0,7), player

State:

<table>
<tr><td></td><td></td><td></td><td></td><td></td><td></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><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><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><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><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><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><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><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><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><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><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><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>o</td><td>o</td><td>o</td><td>o</td><td>o</td><td>o</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
</td>
<td>Output:

checkHorizWin = true

State of the game board remains unchanged</td>
<td>Reason:
The reason this test case is unique is because it checks to see if the right most token placed resulted in a horizontal win, and the function must check 6 spaces to the right.

Function Name:
testCheckHorizWin_Win_On_Right_Side</td>
</tr>
</table>

Boolean checkHorizWin

Input: (0,3), player

State:

o	o	o		o	o	o							
x	x	x	x	x	x	x							

Output:

checkHorizWin = true

State of the object remains unchanged

Reason:
This test case is unique because it tests to see if the last token placed for a win was in the middle. So the function needs to check to the right and to the left three spaces each.

Function Name:
testCheckHorizWin_Final_Token_Placed_In_Middle

<table>
<tr>
<td>Boolean CheckHorizWin

Input:(13,13), player

State:

<table><tr><td></td><td></td><td></td><td></td><td></td><td></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><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><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><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><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><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><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><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><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><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><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><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><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><td></td><td></td><td></td><td></td><td></td></tr></table></td>
<td>Output:

checkHorizWin = false</td>
<td>Reason:
The reason this test case is unique is because it checks to see if the last token placed is an out of bounds value resulting in the function to return false.

Function Name:
testCheckHorizWin_On_Empty_Board</td>
</tr>
</table>

Boolean checkVertWin

Input: (14,14), player

State:

												O
												O
												O
												O
												O
												O
												O
												X
												O
												X
												O
												X
												O
												X

Output:

checkVertWin = true

State of the object remains unchanged.

Reason:
The reason this test is unique, is because it checks to see if a win has occurred when it is in the top most part of the column.

Function name:

testCheckVertWin_At_Top_Of_Board_Last_Column

Boolean checkVertWin

Input:(10,7), player

State:

							O					
							O					
							O					
							O					
							O					
							O					
							O					
							O					
							X					
							O					
							X					

Output

checkVertWin = true

State of the object remains unchanged.

Reason:

The reason this test case is unique is because it checks to see if the last token placed results in a vertical win, even when it has happened in the center of the board.

Function Name:

testCheckVertWin_Amidst_Characters_In_Middle_Column

<table>
<tr>
<td>

Boolean checkVertWin

Input:(0,1), player

State:

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

</td>
<td>

Output

checkVertWin = true

State of the object remains unchanged

</td>
<td>

Reason:

The reason this test case is unique is because it check to see if the last token placed results in a vertical win, even when it has occurred in the last few columns of the board.

Function Name:

testCheckVertWin_At_Bottom_Of_Column

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

<table>
<tr>
<td>Boolean checkVertWin

Input:(3,1),player

State:

<table>
<tr><td></td><td></td><td></td><td></td><td></td><td></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><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><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><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><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><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><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><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><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>O</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>O</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
</td>
<td>Output

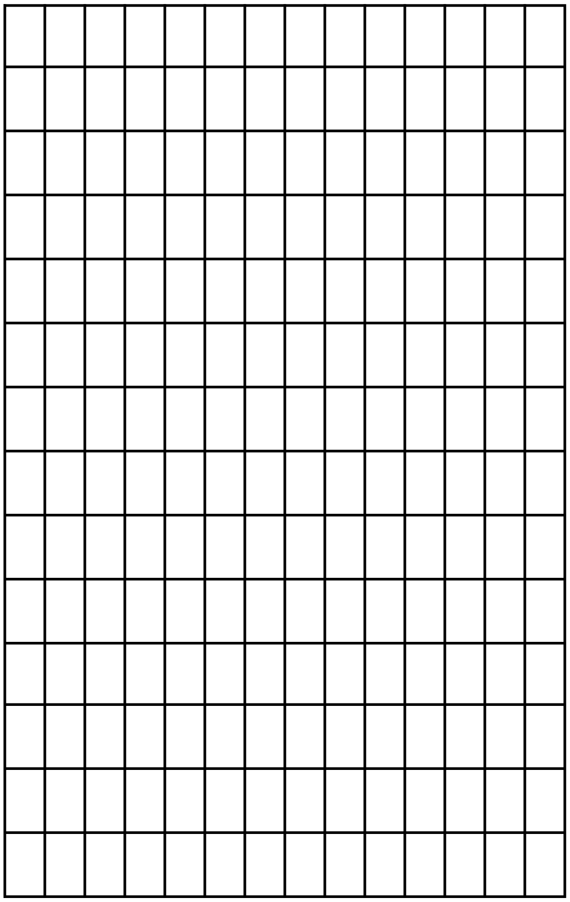
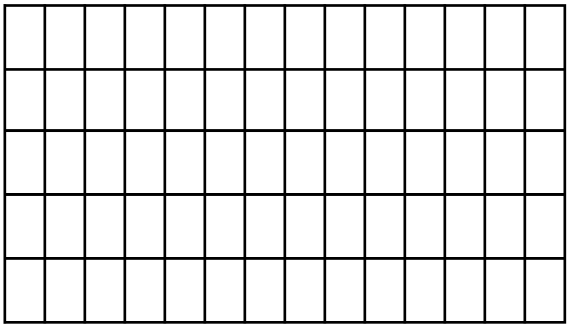
checkVertWin = false

State of the game board remains unchanged.</td>
<td>Reason:

The reason this test is unique is because it checks to see if the user inputted an out of bounds input resulting in an invalid space which would in turn result in this function being false.

Function Name:

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

Boolean CheckDiagWin() Input:(0,0), player State: 	Output: checkDiagWin = false State of the object remains unchanged.	Reason: The reason this test case is unique is because it checks to see an empty boards returns false when checking for a diagonal win Function Name testCheckDiagWin_False_Empty_Board
Boolean CheckDiagWin() Input:(7,7),player State: 	Output checkDiagWin = true State of the object remains unchanged	Reason: The reason this test case is unique is because it checks to see if a win occurs at the top of an Up and Right play

						X							
					X	X							
				X	X	X							
			X	X	X	X							
		X	X	X	X	X							
	X	X	X	X	X	X							
X	X	X	X	X	X	X							

Function Name:
testCheckDiagWin_Up_and_Right_Last_placed_Top

Boolean CheckDiagWin()
Input:
State:

						X							
					X	X							
				X	X	X							
			X	X	X	X							
		X	X	X	X	X							
	X	X	X	X	X	X							
X	X	X	X	X	X	X							

Output:

checkDiagWin = true

State of the object remains unchanged.

Reason:
The reason this test is unique is because it tests to see if the last token placed in the middle of a diagonal string results in a win. It will have to check both up - left, up - right, and down - left or right.

Function Name:
testCheckDiagWin_When_Placed_In_Middle

<table>
<tr>
<td><table><tr><td>o</td><td>x</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><td>o</td><td>x</td><td>o</td><td></td></tr></table></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Boolean CheckDiagWin()
Input:

State:

<table>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></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><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><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><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><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><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><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>X</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>X</td><td>X</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>X</td><td>X</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td>X</td><td>X</td><td>X</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table></td>
<td>Output:

checkDiagWin = true

State of the object remains unchanged</td>
<td>Reason: The reason for this test is unique because it tests to see if the last token placed in the upper position in a diagonal string results in a win.

Function name: testCheckDiagWin_Final_Token_In_Corner</td>
</tr>
</table>

Boolean checkDiagWin

Input:

State:

x													
x	x												
x	x	x											
x	x	x	x										
x	x	x	x	x									
x	x	x	x	x	x								
x	x	x	x	x	x	x							

Output:

checkDiagWin = true

The state of the object remains unchanged

Reason: The reason for this test is to check to see if the function results in a win when the tokens are arranged in the down and to the right position

Function Name: testCheckDiagWin_Down_Right_Last_Token_Middle

<table>
<tr>
<td>Boolean CheckDiagWin

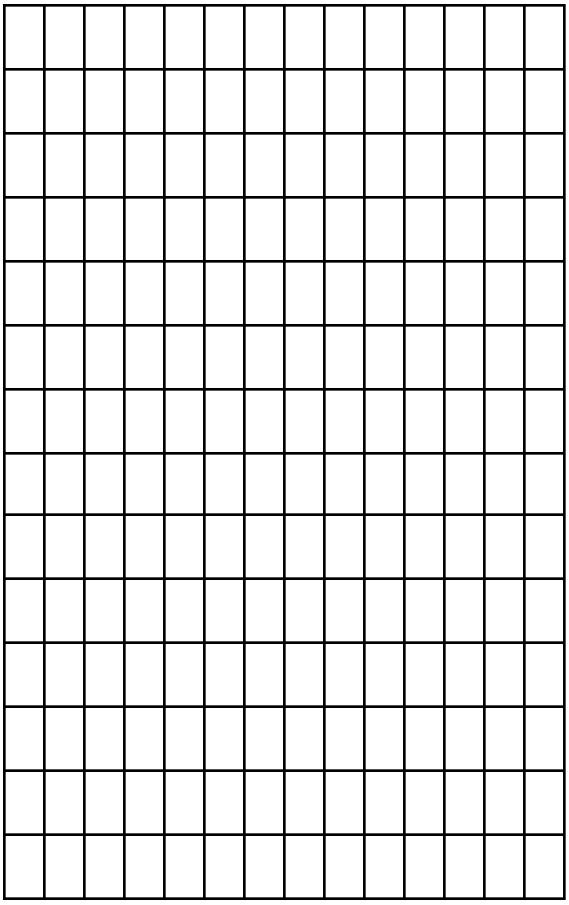
Input:
<table>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><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><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><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><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><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>x</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>x</td><td>x</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>x</td><td>x</td><td>x</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>x</td><td>x</td><td>x</td><td>x</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
</td>
<td>Output:

checkDiagWin = true

The state of the object remains unchanged</td>
<td>Reason:
The reason this test is unique is because it tests to see if the arrangement of tokens down and right results in a win.

Function Name:
testCheckDiagWin_In_Down_Right_Direction_Last_Placed_Top</td>
</tr>
<tr>
<td>Boolean CheckDiagWin

Input:</td>
<td></td>
<td></td>
</tr>
</table>

Boolean checkTie() Input: 	Output: checkTie = false	Reason: The reason that this test is unique is because it check that if the board is empty there is no tie and it returns false Function name: testCheckTie_With_Empty_Board

<table>
<tr>
<td>Boolean checkTie

Input:

<table>
<tr><td>x</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><td>o</td><td>x</td><td>o</td><td>x</td><td>o</td></tr>
<tr><td>o</td><td>x</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><td>o</td><td>x</td><td>o</td><td>x</td></tr>
<tr><td>x</td><td>x</td><td>x</td><td>o</td><td>o</td><td>o</td><td>x</td><td>x</td><td>x</td><td>o</td><td>o</td><td>x</td><td>o</td><td>x</td></tr>
<tr><td>x</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><td>o</td><td>x</td><td>o</td><td>x</td><td>o</td></tr>
<tr><td>o</td><td>x</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><td>o</td><td>x</td><td>o</td><td>x</td></tr>
<tr><td>x</td><td>x</td><td>x</td><td>o</td><td>o</td><td>o</td><td>x</td><td>x</td><td>x</td><td>o</td><td>o</td><td>x</td><td>o</td><td>x</td></tr>
<tr><td>x</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><td>o</td><td>x</td><td>o</td><td>x</td><td>o</td></tr>
<tr><td>o</td><td>x</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><td>o</td><td>x</td><td>o</td><td>x</td></tr>
<tr><td>x</td><td>x</td><td>x</td><td>o</td><td>o</td><td>o</td><td>x</td><td>x</td><td>x</td><td>o</td><td>o</td><td>x</td><td>o</td><td>x</td></tr>
<tr><td>x</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><td>o</td><td>x</td><td>o</td><td>x</td><td>o</td></tr>
<tr><td>o</td><td>x</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><td>o</td><td>x</td><td>o</td><td>x</td></tr>
<tr><td>x</td><td>x</td><td>x</td><td>o</td><td>o</td><td>o</td><td>x</td><td>x</td><td>x</td><td>o</td><td>o</td><td>x</td><td>o</td><td>x</td></tr>
<tr><td>x</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><td>o</td><td>x</td><td>o</td><td>x</td><td>o</td></tr>
<tr><td>o</td><td>x</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><td>o</td><td>x</td><td>o</td><td>x</td></tr>
<tr><td>x</td><td>x</td><td>x</td><td>o</td><td>o</td><td>o</td><td>x</td><td>x</td><td>x</td><td>o</td><td>o</td><td>x</td><td>o</td><td>x</td></tr>
</table>
</td>
<td>Output:

checkTie = True</td>
<td>Reason: The reason that this test is unique is because it checks that if the board is completely full and there is no winner prior to the board being filled and it returns True

Function name: testCheckTie_With_GameBoard_Full</td>
</tr>
</table>

<table>
<tr>
<td>Boolean checkTie

Input:

<table>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></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><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><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><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><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><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><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><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><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><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><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><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
</td>
<td>Output:

checkTie = false</td>
<td>Reason:

The reason that this test is unique is because it checks a partial filled board when only one character has been placed

Function name: testCheckTie_Partial_Filled_Board</td>
</tr>
<tr>
<td>Boolean checkTie

Input:

<table>
<tr><td>O</td><td>X</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><td>O</td><td>X</td><td>O</td><td>X</td></tr>
<tr><td>X</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><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td></tr>
<tr><td>O</td><td>X</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><td>O</td><td>X</td><td>O</td><td>X</td></tr>
<tr><td>X</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><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td></tr>
<tr><td>O</td><td>X</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><td>O</td><td>X</td><td>O</td><td>X</td></tr>
<tr><td>X</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><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td></tr>
</table>
</td>
<td>Output:

checkTie = TRUE</td>
<td>Reason:

The reason this test is unique is because is checks to see that a partial board does not result in a tie.</td>
</tr>
</table>

<table>
<tr>
<td>
<table>
<tr><td>O</td><td>X</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><td>O</td><td>X</td><td>O</td><td>X</td></tr>
<tr><td>X</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><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td></tr>
<tr><td>O</td><td>X</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><td>O</td><td>X</td><td>O</td><td>X</td></tr>
<tr><td>X</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><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td></tr>
<tr><td>O</td><td>X</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><td>O</td><td>X</td><td>O</td><td>X</td></tr>
<tr><td>X</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><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td></tr>
<tr><td>O</td><td>X</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><td>O</td><td>X</td><td>O</td><td>X</td></tr>
<tr><td>X</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><td>O</td><td>X</td><td>O</td><td>X</td><td>O</td></tr>
</table>
</td>
<td></td>
<td>Function name:

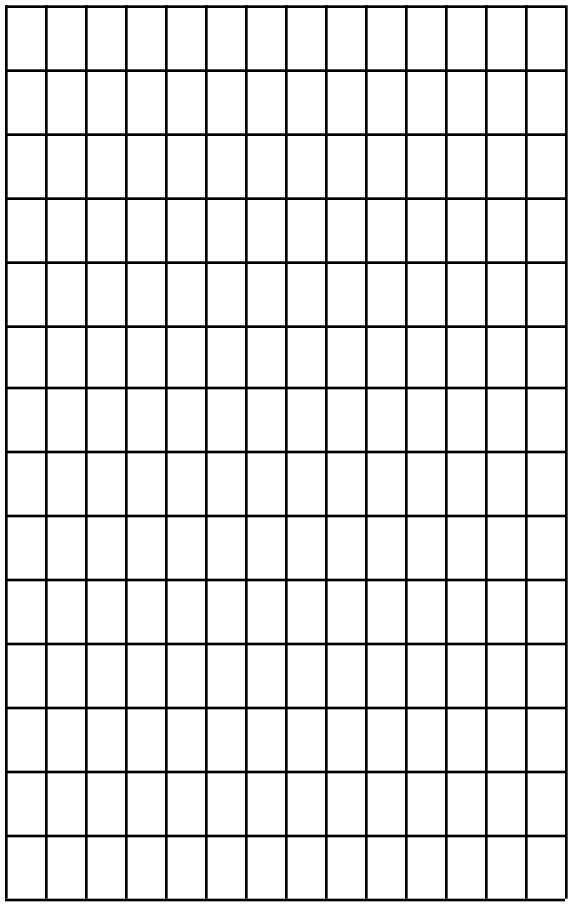
testCheckTie_When_Filled_But_Win_Present</td>
</tr>
<tr>
<td>Char whatsAtPos

Input: (1,0)

<table>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></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><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><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><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><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><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><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><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><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><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><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><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><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><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</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><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr>
</table>
</td>
<td>Output:

whatsAtPos = ‘ ‘
State of the board position remains the same.</td>
<td>Reason:
THe reason this test is unique is because it checks the next row up when the bottom row is full

Function Name:
testWhatsAtPos_One_Row_Full_Characters</td>
</tr>
</table>

Char whatsAtPos Input: (1,1) 	Output whatsAtPos = ‘ ‘ State of the board remained unchanged	Reason: The reason this test is unique is because it will test to make sure that the single digit inputs from the user will return a blank character Function Name: testWhatsAtPos_Empty_Character_1_1

<table>
<tr>
<td>Char whatsAtPos
Input: (0,1)
<table>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
</td>
<td>Output
whatsAtPos = ‘ ’
State of the board remains unchanged</td>
<td>Reason:
The reason this test is unique is because the whatsAtPos function checks the following unfilled column after a filled one
Function Name: testWhatsAtPos_Character_Next_To_Filled_Column</td>
</tr>
<tr>
<td>Char whatsAtPos
Input:(11,12)
<table>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></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><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><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><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><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><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
</td>
<td>Output
whatsAtPos = ‘ ‘
State of the board remains unchanged</td>
<td>Reason:
The reason this test is unique is because the whatsAtPos function must be able to return a blank character when pos is in double digit values.
Function Name: testWhatsAtPos_Empty_Character_In_Double_Digit_Position_11_12</td>
</tr>
</table>

<table>
<tr>
<td>
<table>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></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><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><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><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><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><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><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><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
</td>
<td></td>
<td></td>
</tr>
<tr>
<td>
Char whatsAtPos

Input (11,12)

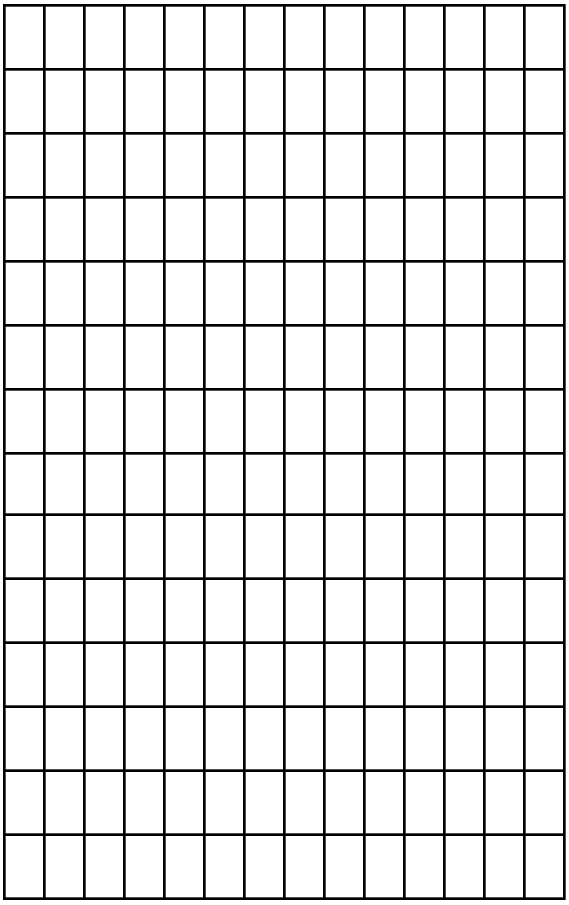
<table>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td></tr>
</table>
</td>
<td>
Output

whatsAtPos = ‘X’

State of the board remains unchanged
</td>
<td>
Reason:

The reason this test is unique is because the whatsAtPos function must be able to return a character at a position when pos is in the double digit values.

Function Name: testWhatsAtPos_Character_In_Double_Digit_Position_11_12
</td>
</tr>
</table>

Boolean isPlayerAtPosition(BoardPosition pos, char player) Input :(15,15) 	Output: isPlayerAtPos = false State of the board remains unchanged	Reason: The reason this test is unique is because it tests to make sure that the BoardPosition parameter of isPlayerAtPostion is a valid number Function Name testIsPlayerAtPos_Out_Of_Bounds_15_15

Boolean isPlayerAtPosition(BoardPosition pos, char player) Input: (1,1) = “ “ <table><tr><td></td><td></td><td></td><td></td><td></td><td></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><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><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><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><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><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><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><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><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><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><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><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><td></td><td></td><td></td><td></td><td></td></tr><tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table>	Output: isPlayerAtPos = false State of the board remains unchanged	Reason: The reason this test is unique is because it tests to make sure that an empty BoardPosition of isPlayerAtPostion returns false Function name: testIsPlayerAtPos_Empty_Position_1_1

<table>
<tr>
<td>Boolean isPlayerAtPos(BoardPosition pos, char player)

Input:
New BoardPosition (0,0)
Player = ‘X’

<table>
<tr><td></td><td></td><td></td><td></td><td></td><td></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><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><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><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><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><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><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><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><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><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><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><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><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td>O</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
</td>
<td>Output:

isPlayerAtPos = True

State of the game board remains unchanged</td>
<td>Reason:
The reason this test is unique is because it tests to make sure that when the same character is in the current selected position it will return true.

Function Name:
testIsPlayerAtPos_Check_Matching_Character_At_Position_0_0</td>
</tr>
</table>

<table>
<tr>
<td>Boolean isPlayerAtPosition(BoardPosition pos, char player)

Input:

Param:
New BoardPosition(0,0)
Player = ‘O’

<table>
<tr><td></td><td></td><td></td><td></td><td></td><td></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><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><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><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><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><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><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><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><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><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><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><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><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><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>X</td><td>O</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
</td>
<td>Output:

isPlayerAtPos = False

State of the game board remains unchanged</td>
<td>Reason:
The reason this test is unique is because it tests to ensure that when a different character is in the current selected spot it will still return false.

Function Name:
testIsPlayerAtPos_Check_Mismatch_Character_At_Position_0_0</td>
</tr>
</table>

<table>
<tr>
<td>isPlayerAtPos(BoardPosition pos, char player)

Input:

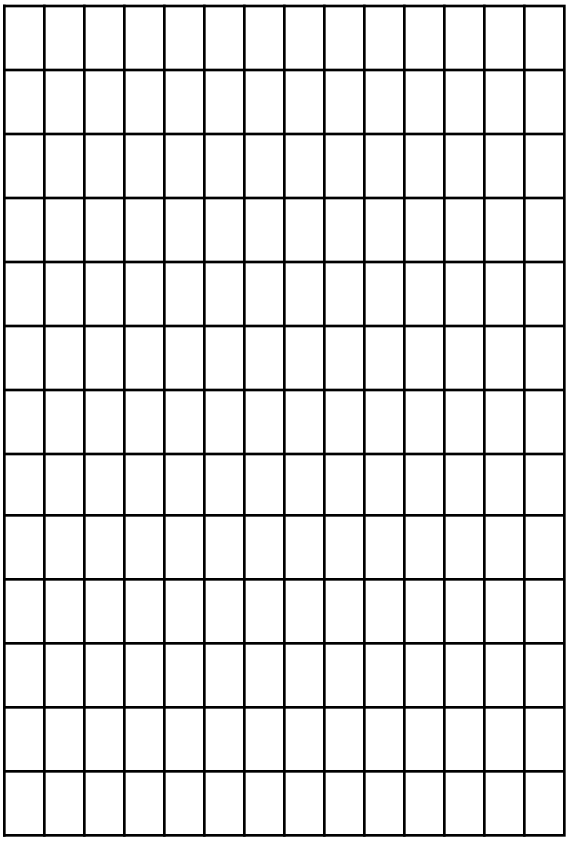
Player = ‘X’</td>
<td>Output:

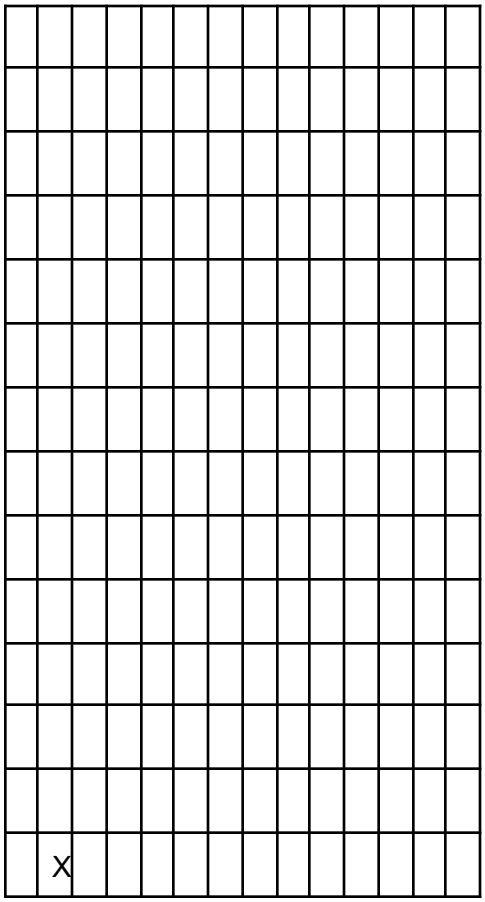
isPlayerAtPos = false

State of the game board remains unchanged</td>
<td>Reason:
The reason this test is unique is because it tests to make sure the function when taking in a null position will still return false.

Function Name:
testIsPlayerAtPos_when_given_null_character</td>
</tr>
<tr>
<td>void dropToken(char p, int c)

Input:
Char p = ‘X’
Int c = 1

  </td>
<td>Output:

 
</td>
<td>Reason:

This test is unique because it places a token in an empty board

Function Name:

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

Void dropToken(char p, int c)

Input:
Char p = ‘X’
Int c = 0

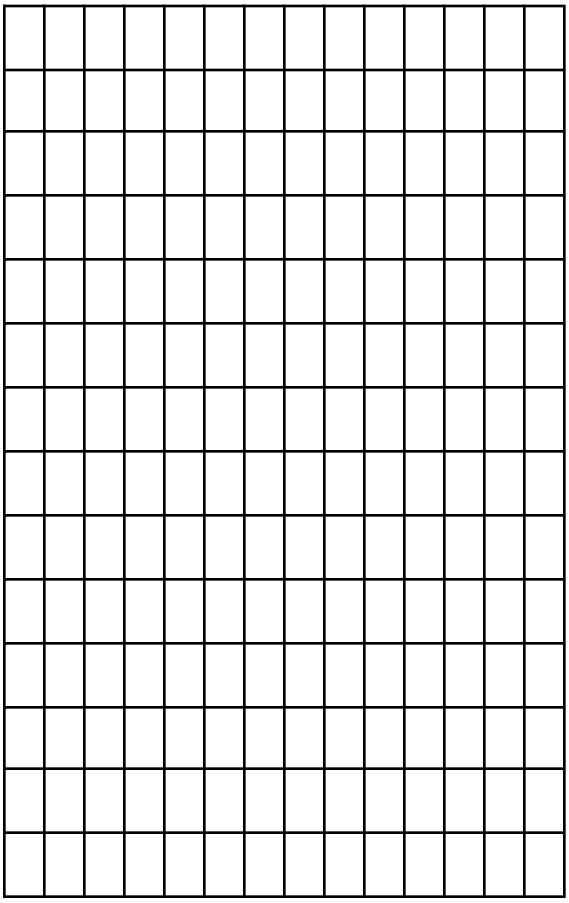
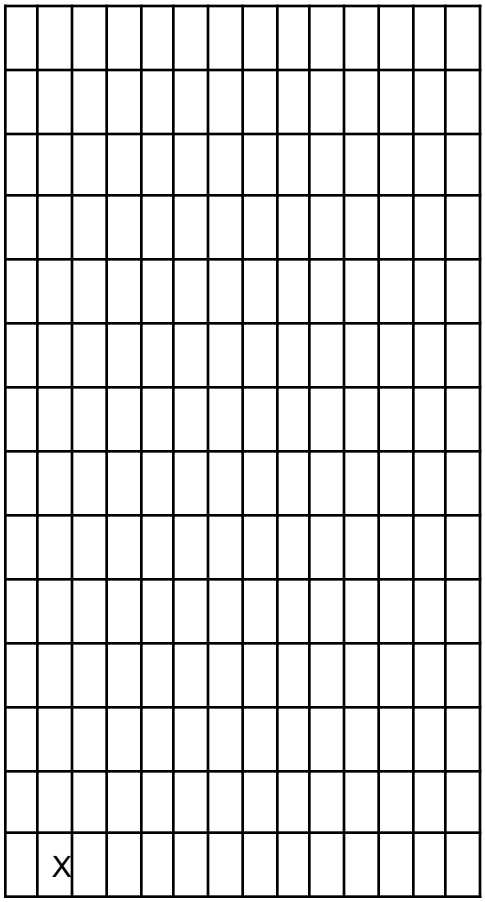
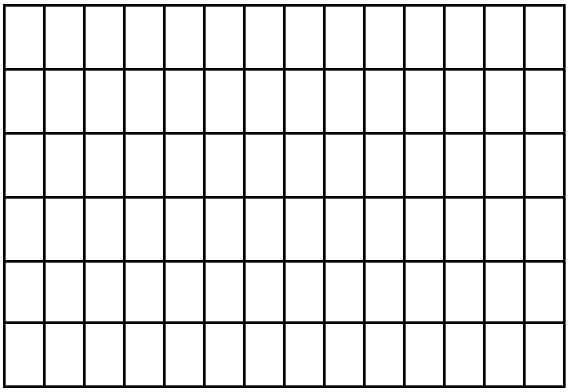
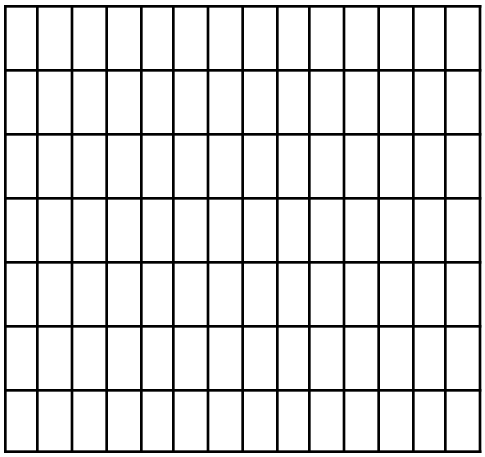
X													

Output

X													
X													
X													

Reason:
The reason this test is unique is because it tests tokens dropped in the same column

Function Name:
testDropToken_On_Partial_Filled_Column

Void dropToken(char p, int c) Input: 	Output  	Reason: The reason this test is unique is because it tests that the token was dropped in the proper column Function Name: testDropToken_Check_Only_One_Token_Placed
Void dropToken(char p, int c) Input:(0,1)(0,2)(0,3) 	Output 	Reason: The reason this test is unique is because it test that multiple columns take tokens Function Name: testDropToken_Different_Column_Selection

	X	X	X										

Void dropToken(char p, int c)

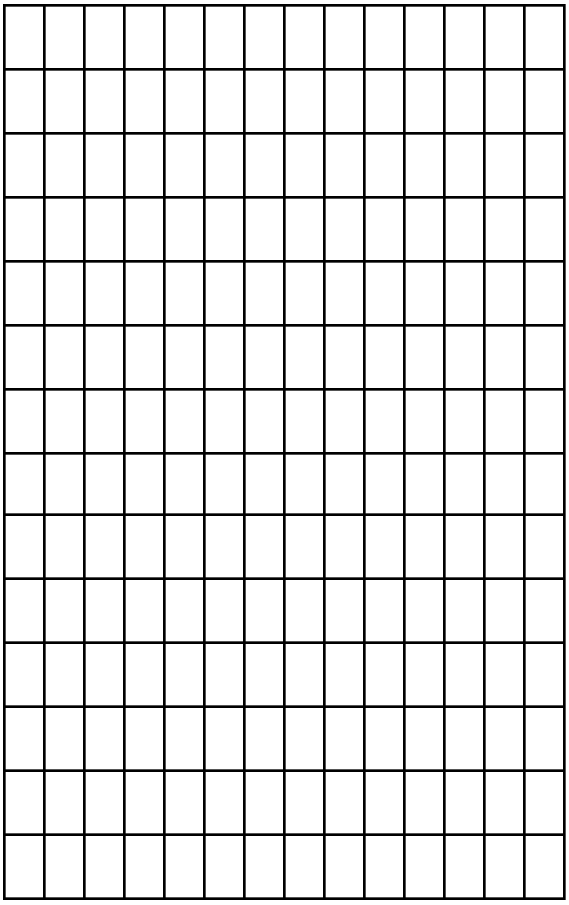
Input:

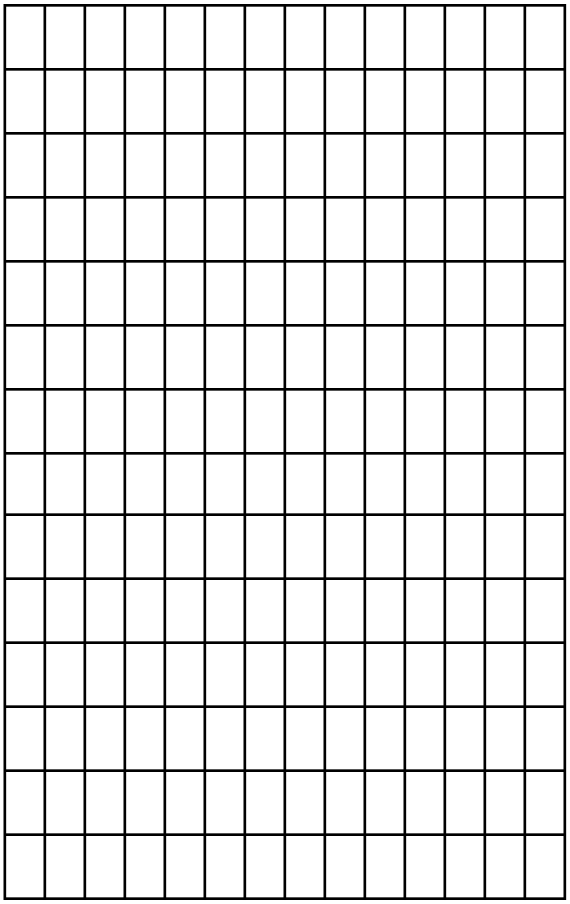
	X												
	X												
	X												
	X												
	X												
	X												
	X												
	X												
	X												
	X												
	X												
	X												
	X												
	X												

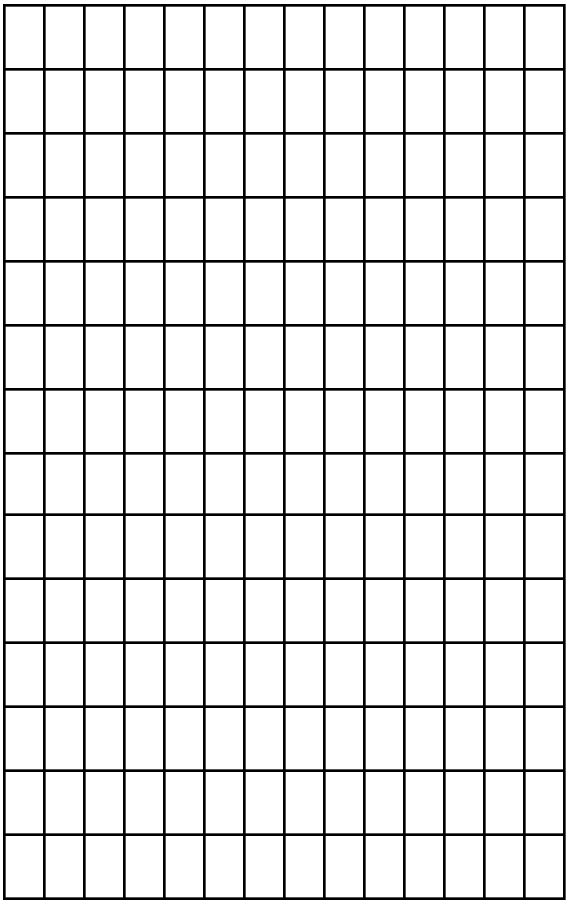
Output
No Change to state

Reason:
The reason this test is unique is because it tests a token isnt added to a full column

Function Name:
testDropToken_Dont_Drop_In_Filled

TestGameBoardMem Constructor Input: 	Output: Empty game board	Reason: This test case is unique because It checks the game board is constructed with the proper number of rows and columns Function Name: testConstructor_GameBoard_Correct_Size()

TestGameBoardMem Constructor Input: 	Output: Empty game board of blank characters	Reason: This test case is unique because it checks that the characters making up the board are blank. Function Name: testConstructor_Blank_Character_GameBoardMem()

TestGameBoardMem Constructor Input: 	Output: Empty Board that has been sized down to 7x7 with a Num To Win of 3	Reason: This test case is unique because it shows that the game board can become a different size according to user preference Function Name: testConstructor_Different_Sizes
