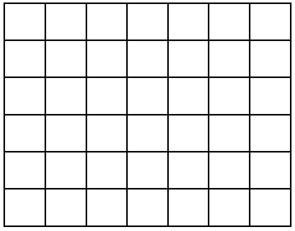
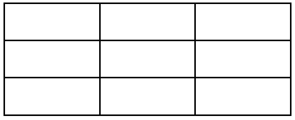
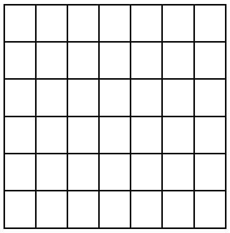
GameBoardMem(int rows, int cols, int win) & GameBoard(int rows, int cols, int win)

Input:
Rows=6
Cols=7
Win=4
	Output:
	
		Reason:This test case is unique because it tests the default values for gameboard construction
		Function name:
		testGameBoardConstructorStandardVals()
		testGameBoardMemConstructorStandardVals()

Input:
Rows=3
Cols=3
Win=3
	Output:
	
		Reason: this test case tests the minimum allowable dimensions for gameboard creation
		Function name:
		testGameBoardConstructorMinVals()
		testGameBoardMemConstructorMinVals()

Input:
Rows=100
Cols=100
Win=25
	Output:
	100x100 gameboard
		Reason: this test case tests the maximum allowable dimensions for gameboard creation
		Function name:
		testGameBoardConstructorMaxVals()
		testGameBoardMemConstructorMaxVals()

Boolean checkIfFree(int c)

Input:
State

C=0
	Output:
	Checkiffree=true
	State of board unchanged
		Reason: this test case tests checkiffree on an empty column
		Function name:
		testGameBoardCheckIfFreeEmptyColumn()
		testGameBoardMemCheckIfFreeEmptyColumn()

<table>
<tr>
<td>Input:
State
<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>
<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>X</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>C=0</td>
<td>Output:
Checkiffree=true

State of board unchanged</td>
<td>Reason: this test case tests checkiffree on a partially filled column
Function name:
testGameBoardCheckIfFreePartiallyFilledColumn()
testGameBoardMemCheckIfFreePartiallyFilledColumn()</td>
</tr>
<tr>
<td>Input:
State
<table>
<tr><td>X</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></tr>
<tr><td>X</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></tr>
<tr><td>X</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></tr>
</table>C=0</td>
<td>Output:
Checkiffree=false

State of board unchanged</td>
<td>Reason: this test case tests checkiffree on a full column
Function name:
testGameBoardCheckIfFreeFilledColumn()
testGameBoardMemCheckIfFreeFilledColumn()</td>
</tr>
</table>

```
boolean checkHorizWin(BoardPosition pos, char p)
```

<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>
<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>X</td><td>X</td><td>X</td><td>X</td><td></td><td></td><td></td></tr>
</table>Number to win = 4
Pos.getRow=0
Pos.getColumn=0;
P='X'</td>
<td>Output:
checkHorizWin=true
State of board unchanged</td>
<td>Reason: this test case test checkHorizwin if the winning tile is placed on the lefthand side
Function name:
testGameBoardCheckHorizWinBottomLeft()
testGameBoardMemCheckHorizWinBottomLeft()</td>
</tr>
<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>
<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>X</td><td>X</td><td>X</td><td>X</td><td></td><td></td><td></td></tr>
</table>Number to win = 4
Pos.getRow=0
Pos.getColumn=3;
P='X'</td>
<td>Output:
checkHorizWin=true
State of board unchanged</td>
<td>Reason: this test case test checkHorizwin if the winning tile is placed on the righthand side
Function name:
testGameBoardCheckHorizWinBottomRight()
testGameBoardMemCheckHorizWinBottomRight()</td>
</tr>
</table>

<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>
<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>X</td><td>X</td><td>X</td><td>X</td><td></td><td></td><td></td></tr>
</table>Number to win = 4
Pos.getRow=0
Pos.getColumn=1;
P='X'</td><td>Output:
checkHorizWin=true
State of board unchanged</td><td>Reason: this test case test checkHorizwin if the winning tile is placed in the middle
Function name:
<code>testGameBoardCheckHorizWinDropInMiddle()</code>
<code>testGameBoardMemCheckHorizWinDropInMiddle()</code></td></tr>
<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>
<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>X</td><td>X</td><td>O</td><td>X</td><td>X</td><td></td><td></td></tr>
</table>Number to win = 4
Pos.getRow=0
Pos.getColumn=4;
P='X'</td><td>Output:
checkHorizWin=false
State of board unchanged</td><td>Reason: this test case test checkHorizwin if the win condition is not met
Function name:
<code>testGameBoardCheckHorizWinWithBrokenLine()</code>
<code>testGameBoardMemCheckHorizWinWithBrokenLine()</code></td></tr>
</table>

```
boolean checkVertWin(BoardPosition pos, char p)
```

<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>X</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td>X</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td>X</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td>X</td><td></td><td></td><td></td></tr>
</table>Number to win = 4
Pos.getRow=3
Pos.getColumn=3
P='X'</td><td>Output:
checkVertWin=true
state of board unchanged</td><td>Reason: this tests checkvertwin if the winning vertical line is on an empty column
Function name:
<code>testGameBoardCheckVertWinEmptyColumn()</code>
<code>testGameBoardCheckVertWinMemEmptyColumn()</code></td></tr>
<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>X</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td>X</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td>X</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td>X</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td>O</td><td></td><td></td><td></td><td></td></tr>
</table>Number to win = 4
Pos.getRow=4
Pos.getColumn=2
P='X'</td><td>Output:
checkVertWin=true
state of board unchanged</td><td>Reason: this tests checkvertwin if the winning vertical line is on top of an opponent's piece with room remaining above
Function name:
<code>testGameBoardCheckVertWinMiddleColumn()</code>
<code>testGameBoardMemCheckVertWinMiddleColumn()</code></td></tr>
</table>

<table>
<tr>
<td><table>
<tr><td></td><td></td><td>X</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td>X</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td>X</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td>X</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td>O</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td>O</td><td></td><td></td><td></td><td></td></tr>
</table>Number to win = 4
Pos.getRow=5
Pos.getColumn=2
P=’X’</td>
<td>Output:
checkVertWin=true
state of board unchanged</td>
<td>Reason: this tests checkvertwin if the winning line is on top of an opponent’s piece with no free spaces above
Function name:
<code>testGameBoardCheckVertWinTopOfColumn()</code>
<code>testGameBoardMemCheckVertWinTopOfColumn()</code></td>
</tr>
<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>X</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></tr>
<tr><td></td><td>O</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></tr>
</table>Number to win = 4
Pos.getRow=3
Pos.getColumn=1
P=’X’</td>
<td>Output:
checkVertWin=false
state of board unchanged</td>
<td>Reason: this tests checkvertwin when the win conditions have not been met
Function name:
<code>testGameBoardCheckVertWinWithBrokenLine()</code>
<code>testGameBoardMemCheckVertWinWithBrokenLine()</code></td>
</tr>
</table>

```
boolean checkDiagWin(BoardPosition pos, char p)
```

<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>X</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td>X</td><td>O</td><td></td><td></td><td></td></tr>
<tr><td></td><td>X</td><td>O</td><td>O</td><td></td><td></td><td></td></tr>
<tr><td>X</td><td>O</td><td>O</td><td>O</td><td></td><td></td><td></td></tr>
</table>Number to win = 4
Pos.getRow=3
Pos.getColumn=3
P=’X’</td>
<td>Output:
checkDiagWin= true
state of board unchanged</td>
<td>Reason: this test checkdiagwin with an ascending line where the winning token is placed in the tip right
Function name
<code>testGameBoardCheckDiagWinAscendingLastTopRight()</code>
<code>testGameBoardMemCheckDiagWinAscendingLastTokenTopRight()</code></td>
</tr>
<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>X</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td>X</td><td>O</td><td></td><td></td><td></td></tr>
<tr><td></td><td>X</td><td>O</td><td>O</td><td></td><td></td><td></td></tr>
<tr><td>X</td><td>O</td><td>O</td><td>O</td><td></td><td></td><td></td></tr>
</table>Number to win = 4
Pos.getRow=0
Pos.getColumn=0
P=’X’</td>
<td>Output:
checkDiagWin= true
state of board unchanged</td>
<td>Reason: this test checkdiagwin with an ascending line where the winning token is placed in the bottom left
Function name
<code>testGameBoardCheckDiagWinAscendingLastTokenBottomLeft()</code>
<code>testGameBoardMemCheckDiagWinAscendingLastTokenBottomLeft()</code></td>
</tr>
</table>

<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>X</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td>X</td><td>O</td><td></td><td></td><td></td></tr>
<tr><td></td><td>X</td><td>O</td><td>O</td><td></td><td></td><td></td></tr>
<tr><td>X</td><td>O</td><td>O</td><td>O</td><td></td><td></td><td></td></tr>
</table>
Number to win = 4
Pos.getRow=1
Pos.getColumn=1
P=’X’
</td>
<td>Output:
checkDiagWin= true
state of board unchanged</td>
<td>Reason: this test checkdiagwin with an ascending line where the winning token is placed in the middle
Function name
<code>testGameBoardCheckDiagWinAscendingLastTokenMiddle()</code>
<code>testGameBoardMemCheckDiagWinAscendingLastTokenMiddle()</code></td>
</tr>
<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>X</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td>O</td><td>O</td><td></td><td></td><td></td></tr>
<tr><td></td><td>X</td><td>O</td><td>O</td><td></td><td></td><td></td></tr>
<tr><td>X</td><td>O</td><td>O</td><td>O</td><td></td><td></td><td></td></tr>
</table>
Number to win = 4
Pos.getRow=3
Pos.getColumn=3
P=’X’
</td>
<td>Output:
checkDiagWin= false
state of board unchanged</td>
<td>Reason: this test checkdiagwin with an ascending line where the win conditions are not met
Function name
<code>testGameBoardCheckDiagWinAscendingBrokenLine()</code>
<code>testGameBoardMemCheckDiagWinAscendingBrokenLine()</code></td>
</tr>
<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>X</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>O</td><td>X</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>O</td><td>O</td><td>X</td><td></td><td></td><td></td><td></td></tr>
<tr><td>O</td><td>O</td><td>O</td><td>X</td><td></td><td></td><td></td></tr>
</table>
Number to win = 4
Pos.getRow=3
Pos.getColumn=1
P=’X’
</td>
<td>Output:
checkDiagWin= true
state of board unchanged</td>
<td>Reason: this test checkdiagwin with a descending line where the winning token is placed top left
Function name
<code>testGameBoardCheckDiagWinDescendingLastTokenTopLeft()</code>
<code>testGameBoardMemCheckDiagWinDescendingLastTokenTopLeft()</code></td>
</tr>
<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>X</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>O</td><td>X</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>O</td><td>O</td><td>X</td><td></td><td></td><td></td><td></td></tr>
<tr><td>O</td><td>O</td><td>O</td><td>X</td><td></td><td></td><td></td></tr>
</table>
Number to win = 4
Pos.getRow=3
Pos.getColumn=0
P=’X’
</td>
<td>Output:
checkDiagWin= true
state of board unchanged</td>
<td>Reason: this test checkdiagwin with a descending line where the winning token is placed in the bottom right
Function name
<code>testGameBoardCheckDiagWinDescendingLastTokenBottomRight()</code>
<code>testGameBoardMemCheckDiagWinDescendingLastTokenBottomRight()</code></td>
</tr>
</table>

<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>X</td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>O</td><td>X</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>O</td><td>O</td><td>X</td><td></td><td></td><td></td><td></td></tr><tr><td>O</td><td>O</td><td>O</td><td>X</td><td></td><td></td><td></td></tr></table>Number to win = 4
Pos.getRow=0
Pos.getColumn=3
P='X'</td><td>Output:
checkDiagWin= true
state of board unchanged</td><td>Reason: this test checkdiagwin with a descending line where the winning token is placed in the middle
Function name
testGameBoardCheckDiagWinDescendingLastTokenMiddle()
testGameBoardMemCheckDiagWinDescendingLastTokenMiddle()</td></tr>
</table>

Boolean checkTie()

<table>
<tr><td><table><tr><td>O</td><td>X</td><td>O</td></tr><tr><td>X</td><td>O</td><td>X</td></tr><tr><td>X</td><td>O</td><td>X</td></tr></table></td><td>Output:
checkTie()=true
state of board unchanged</td><td>Reason: this tests checkTie when the board is full
Function name
testGameBoardCheckTieFullBoard()
testGameBoardMemCheckTieFullBoard()</td></tr>
<tr><td><table><tr><td>O</td><td>X</td><td></td></tr><tr><td>X</td><td>O</td><td>X</td></tr><tr><td>X</td><td>O</td><td>X</td></tr></table></td><td>Output:
checkTie()=false
state of board unchanged</td><td>Reason: this tests checkTie when there is one free space remaining
Function name
testGameBoardCheckTieOneFreeSpace()
testGameBoardMemCheckTieOneFreeSpace()</td></tr>
<tr><td><table><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr></table></td><td>Output:
checkTie()=false
state of board unchanged</td><td>Reason: this tests checkTie when the board is empty
Function name
testGameBoardCheckTieEmptyBoard()
testGameBoardMemCheckTieEmptyBoard()</td></tr>
<tr><td><table><tr><td>O</td><td>X</td><td></td></tr><tr><td>X</td><td>O</td><td></td></tr><tr><td>X</td><td>O</td><td></td></tr></table></td><td>Output:
checkTie()=false
state of board unchanged</td><td>Reason: this tests checkTie when there is one free column remaining
Function name
testGameBoardCheckTieOneFreeColumn()
testGameBoardMemCheckTieOneFreeColumn()</td></tr>
</table>

char whatsAtPos(BoardPosition pos)

<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><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>Pos.getRow=0
Pos.getColumn=0</td><td>Output:
whatsAtPos(pos) =' '
State of board unchanged</td><td>Reason: this tests whatsatpos on an empty board
Function name
testGameBoardWhatsAtPosEmptyBoard()
testGameBoardMemWhatsAtPosEmptyBoard()</td></tr>
</table>

<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>
<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>X</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
Pos.getRow=0
Pos.getColumn=0</td>
<td>Output:
whatsAtPos(pos) =’X’
State of board unchanged</td>
<td>Reason: this tests whatsatpos on the bottom of a column with one token played
Function name
<code>testGameBoardWhatsAtPosBottomOfColumnOneTokenPlayed()</code>
<code>testGameBoardMemWhatsAtPosBottomOfColumnOneTokenPlayed()</code></td>
</tr>
<tr>
<td><table>
<tr><td>O</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></tr>
<tr><td>O</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></tr>
<tr><td>O</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></tr>
</table>
Pos.getRow=5
Pos.getColumn=0</td>
<td>Output:
whatsAtPos(pos) =’O’
State of board unchanged</td>
<td>Reason: this tests whatsatpos on the top of a full column
Function name
<code>testGameBoardWhatsAtPosTopOfColumnFullColumn()</code>
<code>testGameBoardMemWhatsAtPosTopOfColumnFullColumn()</code></td>
</tr>
<tr>
<td><table>
<tr><td>O</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></tr>
<tr><td>O</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></tr>
<tr><td>O</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></tr>
</table>
Pos.getRow=2
Pos.getColumn=0</td>
<td>Output:
whatsAtPos(pos) =’X’
State of board unchanged</td>
<td>Reason: this tests whatsatpos on the middle of a full column
Function name
<code>testGameBoardWhatsAtPosMiddleOfColumnFullColumn()</code>
<code>testGameBoardMemWhatsAtPosMiddleOfColumnFullColumn()</code></td>
</tr>
<tr>
<td><table>
<tr><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></tr>
<tr><td>O</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></tr>
<tr><td>O</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></tr>
</table>
Pos.getRow=4
Pos.getColumn=0</td>
<td>Output:
whatsAtPos(pos) =’X’
State of board unchanged</td>
<td>Reason: this tests whatsatpos on the highest occupied space of a partially filled column
Function name
<code>testGameBoardWhatsAtPosTopOfPartiallyFullColumn()</code>
<code>testGameBoardMemWhatsAtPosTopOfPartiallyFullColumn()</code></td>
</tr>
</table>

Boolean isPlayerAtPos(BoardPosition pos, char player)

<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>
<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>
Pos.getRow=0
Pos.getColumn=0
Player=’X’</td>
<td>Output:
isPlayerAtPos(pos,’X’)=false
state of board unchanged</td>
<td>Reason: this tests if player is at position for an unoccupied space at the bottom of a column
Function name
<code>testGameBoardIsPlayerAtPosEmptySpace()</code>
<code>testGameBoardMemIsPlayerAtPosEmptySpace()</code></td>
</tr>
</table>

<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>
<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>X</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>Pos.getRow=0
Pos.getColumn=0
Player=’X’</td>
<td>Output:
isPlayerAtPos(pos,’X’)=true
state of board unchanged</td>
<td>Reason: this tests if the correct player is at position for an occupied space at the bottom of a column
Function name
<code>testGameBoardIsPlayerAtPosOccupiedSpaceCorrectPlayer()</code>
<code>testGameBoardMemIsPlayerAtPosOccupiedSpaceCorrectPlayer()</code></td>
</tr>
<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>
<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>X</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>Pos.getRow=0
Pos.getColumn=0
Player=’O’</td>
<td>Output:
isPlayerAtPos(pos,’O’)=false
state of board unchanged</td>
<td>Reason: this tests if the incorrect player is at the position of the occupied space at the bottom of a column
Function name
<code>testGameBoardIsPlayerAtPosOccupiedSpaceIncorrectPlayerBottomOfColumn()</code>
<code>testGameBoardMemIsPlayerAtPosOccupiedSpaceIncorrectPlayerBottomOfColumn()</code></td>
</tr>
<tr>
<td><table>
<tr><td>O</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></tr>
<tr><td>O</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></tr>
<tr><td>O</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></tr>
</table>Pos.getRow=5
Pos.getColumn=0
Player=’O’</td>
<td>Output:
isPlayerAtPos(pos,’O’)=true
state of board unchanged</td>
<td>Reason: this tests if the correct player is at the position of the occupied space at the top of a column
Function name
<code>testGameBoardIsPlayerAtPosOccupiedSpaceTopOfColumn()</code>
<code>testGameBoardMemIsPlayerAtPosOccupiedSpaceTopOfColumn()</code></td>
</tr>
<tr>
<td><table>
<tr><td>O</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></tr>
<tr><td>O</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></tr>
<tr><td>O</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></tr>
</table>Pos.getRow=2
Pos.getColumn=0
Player=’X’</td>
<td>Output:
isPlayerAtPos(pos,’X’)=false
state of board unchanged</td>
<td>Reason: this tests if the correct player is at the position of the occupied space at the middle of a column
Function name
<code>testGameBoardIsPlayerAtPosOccupiedSpaceMiddleOfColumn()</code>
<code>testGameBoardMemIsPlayerAtPosOccupiedSpaceMiddleOfColumn()</code></td>
</tr>
</table>

void dropToken(char p, int c)

<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><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>P = ‘X’
C = 0</td>
<td>Output<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><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>X</td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table>State</td>
<td>Reason: this tests using droptoken on an empty column
Function name
testGameBoardDroptokenEmptyColumn()
testGameBoardMemDroptokenEmptyColumn()</td>
</tr>
<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><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>P = ‘X’,’O’,’X’
C = 0,0,0</td>
<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><tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>O</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></tr></table>State</td>
<td>Reason: this tests using droptoken to partially fill up a column
Function name
testGameBoardDroptokenPartialColumn()
testGameBoardMemDroptokenPartialColumn()</td>
</tr>
<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><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>P = ‘X’, ‘X’, ‘X’, ‘X’, ‘X’, ‘X’
C = 0,0,0,0,0,0</td>
<td><table><tr><td>X</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></tr><tr><td>X</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></tr><tr><td>X</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></tr></table>State</td>
<td>Reason: this tests using droptoken to completely fill up a column
Function name
testGameBoardDroptokenFullColumn()
testGameBoardMemDroptokenFullColumn()</td>
</tr>
<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><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>P = ‘X’, ‘X’, ‘X’, ‘X’, ‘X’, ‘X’, ‘X’
C = 0,0,0,0,0,0,0</td>
<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><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>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr></table>State</td>
<td>Reason: this tests using droptoken to completely fill up a row
Function name
testGameBoardDroptokenFullRow()
testGameBoardMemDroptokenFullRow()</td>
</tr>
<tr>
<td><table><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr></table>P = ‘X’, ‘X’, ‘O’, ‘O’, ‘O’, ‘X’, ‘X’, ‘X’, ‘O’
C = 0, 0, 0, 1, 1, 1, 2, 2, 2</td>
<td><table><tr><td>O</td><td>X</td><td>O</td></tr><tr><td>X</td><td>O</td><td>X</td></tr><tr><td>X</td><td>O</td><td>X</td></tr></table>State</td>
<td>Reason: this tests using droptoken to completely fill up a board
Function name
testGameBoardDroptokenFillEntireBoard()
testGameBoardMemDroptokenFillEntireBoard( )</td>
</tr>
</table>