GameBoard(int numRows, int numColumns, int numToWin) GameBoardMem(int numRows, int numColumns, int numToWin)

Input:	Output:	Reason:
numRows = 3 numColumns = 3 numToWin = 3	getNumRows = 3 getNumColumns = 3 getNumToWin = 3	This case is unique because it passes all of the minimum values that follow the constructor's contract as arguments
	Board contains only ' '	
		Function Name: testBoardConstructor_rows3_ cols3_toWin3

Input:	Output:	Reason:
numRows = 100 numColumns = 100 numToWin = 100	getNumRows = 100 getNumColumns = 100 getNumToWin = 100	This case is unique because it passes all of the maximum values that follow the constructor's contract as arguments
	Board contains only ' '	Function Name: testBoardConstructor_rows10 0_cols100_toWin25

Input:	Output:	Reason:
numRows = 23 numColumns = 74 numToWin = 13	getNumRows = 23 getNumColumns = 74 getNumToWin = 13 Board contains only ''	This case is unique because it passes values that are well within the minimum and maximum values provided in the constructor's contract and are not edge values
		Function Name: testBoardConstructor_rows23 _cols74_toWin13

boolean checklfFree(int c)

Input:	Output:	Reason: This case is unique because
State:	checkIfFree(0) = true State of the board is unchanged	This case is unique because the column being checked for a free space has no tokens at all Function Name: testCheckIfFree_emptyCol

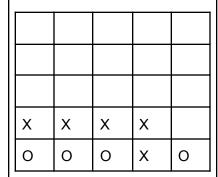
Input:	Output:	Reason:
State: X X X X X X X	checkIfFree(0) = false State of the board is unchanged	This case is unique because the column being checked for a free space is completely full Function Name: testCheckIfFree_fullCol

Input:	Output:	Reason:
State:	checkIfFree(0) = true	This case is unique because the column being checked for
	State of the board is	a free space is almost full but has a single remaining free
X	unchanged	space
x		Function Name: testCheckIfFree_almostFullC
X		ol
X		

boolean checkHorizWin(BoardPosition pos, char p)

Input:

State: (number to win = 4)



pos.getRow = 1 pos.getCol = 2 p = 'X'

Output:

checkHorizWin = true

State of the board is unchanged

Reason:

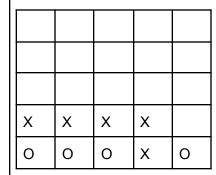
This case is unique because the last x was placed in the middle of the row of 4 consecutive X tokens instead of on the end, so tokens must be checked on both sides of it

Function Name:

testCheckHorizWin_win_last_ marker_middle

Input:

State: (number to win = 4)



pos.getRow = 1 pos.getCol = 0 p = 'X'

Output:

checkHorizWin = true

State of the board is unchanged

Reason:

This case is unique because the last x was placed on the left end of the row of 4 consecutive X tokens, so tokens must be checked only to the right side of it

Function Name:

testCheckHorizWin_win_last_ marker_left

pos.getRow = 1 pos.getCol = 4 p = 'X'

Output:

checkHorizWin = true

State of the board is unchanged

Reason:

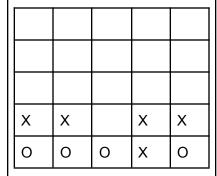
This case is unique because the last x was placed on the right end of the row of 4 consecutive X tokens, so tokens must be checked only to the left side of it

Function Name:

testCheckHorizWin_win_last_ marker_right

Input:

State: (number to win = 4)



pos.getRow = 1 pos.getCol = 4 p = 'X'

Output:

checkHorizWin = true

State of the board is unchanged

Reason:

This case is unique because the last X was placed on a row of the board that has 4 non-consecutive X tokens, so a win does not occur

Function Name:

testCheckHorizWin_noWin

boolean checkVertWin(BoardPosition pos, char p)

Input: State: (number to win = 4) O X O X O X

Output:

State of the board is unchanged

checkVertWin = false

Reason:

This case is unique because there are not enough of the player's tokens stacked on top of one another to result in a win

Function Name:

testCheckVertWin noWin

Input:

State: (number to win = 4)

١.	<u> </u>	1	 	* *	• /
		X			
	0	X			
	0	Х			
	0	Х			

pos.getRow = 3 pos.getCol = 1 p = 'X'

Output:

checkVertWin = true

State of the board is unchanged

Reason:

This case is unique because the player has placed the required number of tokens on top of one another and the bottom of the chain of tokens resulting in the win touches the bottom of the board

Function Name:

testCheckVertWin_win_token s_touch_bottom

Input: **Output:** State: (number to win = 4) checkVertWin = true State of the board is unchanged Χ Χ 0 Χ 0 Χ 0 0 Χ pos.getRow = 4

Reason:

This case is unique because the player has placed the required number of tokens on top of one another and there is a different player's token below the bottom token in the chain resulting in the win

Function Name:

testCheckVertWin_win_other _player_below

State: (number to win = 4)						
	Х					
	Х					
	Х					
0	Х					
0	0	Х				
0	0	Х				

pos.getRow = 5 pos.getCol = 1 p = 'X'

pos.getCol = 1

p = 'X'

Input:

Output:

State of the board is unchanged

checkVertWin = true

Reason:

This case is unique because the player has placed the required number of tokens on top of one another and the chain was completed with a token that resulted in the column becoming full

Function Name:

testCheckVertWin_win_full_c ol

boolean checkDiagWin(BoardPosition pos, char p)

Input: State: (number to win = 4) X O X O X O O X

pos.getRow = 0 pos.getCol = 4 p = 'X'

Output:

State of the board is unchanged

checkDiagWin = true

Reason:

This case is unique because the player has placed the required number of tokens on a left diagonal and the player placed the final token at the bottom right of the diagonal

Function Name:

testCheckDiagWin_win_leftDi ag_last_bottom

Input:

State: (number to win = 4)

Otate. (Humber to win = +)				
	Х			
	0	Х		
	0	Х	Х	
	0	0	0	Х

pos.getRow = 3 pos.getCol = 1 p = 'X'

Output:

checkDiagWin = true

State of the board is unchanged

Reason:

This case is unique because the player has placed the required number of tokens on a left diagonal and the player placed the final token at the top left of the diagonal

Function Name:

testCheckDiagWin_win_leftDiag_last_top

pos.getRow = 1 pos.getCol = 3 p = 'X'

Output:

State of the board is unchanged

checkDiagWin = true

Reason:

This case is unique because the player has placed the required number of tokens on a left diagonal and the player placed the final token in the middle of the diagonal

Function Name:

testCheckDiagWin_win_leftDi ag_last_mid

Input:

State: (number to win = 4)

Otate. (Harriber to Will - +)				
				X
			Х	0
		X	X	0
	Χ	0	0	0

pos.getRow = 0 pos.getCol = 1 p = 'X'

Output:

checkDiagWin = true

State of the board is unchanged

Reason:

This case is unique because the player has placed the required number of tokens on a right diagonal and the player placed the final token at the bottom left of the diagonal

Function Name:

testCheckDiagWin_win_right Diag_last_bottom

Input	•						
State:	State: (number to win = 4)						
				Х			
			Х	0			
		Х	Х	0			
	Х	0	0	0			
					١		

pos.getRow = 3 pos.getCol = 4 p = 'X'

Output:

checkDiagWin = true

State of the board is unchanged

Reason:

This case is unique because the player has placed the required number of tokens on a right diagonal and the player placed the final token at the top right of the diagonal

Function Name:

testCheckDiagWin_win_right Diag_last_top

Input:

State: (number to win = 4)

Otato.	Otate. (Harriber to Will - +)			
				X
			Х	0
		X	X	0
	Х	0	0	0

pos.getRow = 1 pos.getCol = 2 p = 'X'

Output:

checkDiagWin = true

State of the board is unchanged

Reason:

This case is unique because the player has placed the required number of tokens on a right diagonal and the player placed the final token in the middle of the diagonal

Function Name:

testCheckDiagWin_win_right Diag_last_mid

Input:					Output:	Reason: This case is unique because
State	: (num	ber to	win =	4)	checkDiagWin = false	the player has not placed the required number of tokens on
					State of the board is	either diagonal. There is one
		Х			unchanged	less token than necessary on both the left and right
0	Х	0	Х	0		diagonals formed by the last placed token.
x	0	0	X	x		Function Name:
0	Х	0	0	Х		testCheckDiagWin_loss
	jetRow jetCol ('					

boolean checkTie()

Input:	Output:	Reason: This case is unique because
State:	checkTie() = false	the board is completely empty. It couldn't be any
	State of the board is unchanged	further from a tie
		Function Name: testCheckTie_emptyBoard

Inpu	t:				Output:	Reason: This case is unique because
State:					checkTie() = true	the board is completely full. This is the only scenario
Х	Х	Х	Х	Х	State of the board is unchanged	where a tie occurs
Χ	Х	Х	Х	Х		Function Name: testCheckTie fullBoard
Х	Х	Х	Х	Х		1001011001110_10110
Χ	Х	Х	Х	Х		
Х	Х	Х	Х	Х		

Inpu	ıt:				Output:	Reason: This case is unique because
State	State:				checkTie() = false	the board has only 1 free space. It is 1 token away from
Х	x x x x				State of the board is a tie unchanged	
Х	Х	Х	Х	Х		Function Name: testCheckTie_almostFullBoar
Х	Х	Х	Х	Х		d
Х	Х	Х	Х	Х		
X	X	X	X	X		

Input	:			Output:	Reason:
State				checkTie() = false	This case is unique because the board is close to half full. A good many more tokens
x x				State of the board is unchanged	would be needed to cause a tie.
×	Х				Function Name:
X	Х				testCheckTie_halfFullBoard
X	Х				
X	Х				

char whatsAtPos(BoardPosition pos)

Input: State:					 Output: For all positions in board, whatsAtPos = ' '	Reason: This case is unique because all positions in the board are blank spaces.
					State of the board is unchanged	Function Name: testWhatsAtPos_emptyBoard

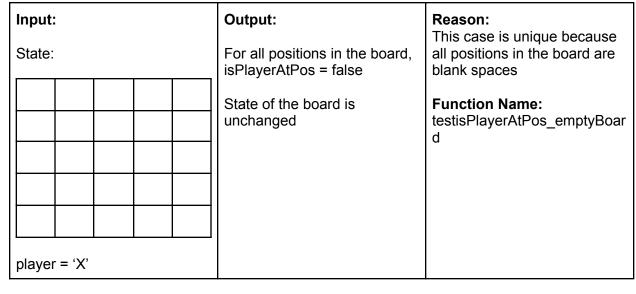
Input:	whatsAtPos = 'X' State of the board is unchanged the X at the speciate the only nonthe board Function Name	Reason: This case is unique because	
State:		the X at the specified position is the only non-blank space in	
		Function Name: testWhatsAtPos_nonEmptyB	
X			
pos.getRow = 0 pos.getColumn = 0			

Input:	Output:	Reason: This case is unique because
State:	whatsAtPos = ' '	the space being checked for a ' ' has a player token below
x pos.getRow = 1 pos.getColumn = 0	State of the board is unchanged	Function Name: testWhatsAtPos_blankSpace AboveToken

Input:			Output: For all positions in Column 0, whatsAtPos = 'X' State of the board is unchanged	Reason:
State:				This case is unique because the column whose positions are being checked is completely full of 'X' Function Name: testWhatsAtPos fullCol
Х				
Х				
Х				tostwindto/tti os_idilooi
х				
Х				
pos.getColum	nn = 0			

Inpu	t:				Output:	Reason:
State	State:				For all positions in board, whatsAtPos = 'X'	This case is unique because all positions in the board are 'X's
Х	X	X	Х	X	State of the board is	Function Name:
Х	Х	Х	Х	Х	unchanged	testWhatsAtPos_fullBoard
Х	Х	Х	Х	Х		
Х	Х	Х	Х	Х		
Х	Х	Х	Х	Х		
	1					

boolean isPlayerAtPos(BoardPosition pos, char player)



Input:	Output: Reason:	Reason: This case is unique because
State:	isPlayerAtPos = true	the X at the specified position is the only non-blank space in
	State of the board is unchanged	the board
	3	Function Name: testisPlayerAtPos_nonEmpty
		Board
X		
pos.getRow = 0 pos.getColumn = 0 player = 'X'		

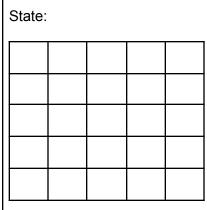
Input:	Output:	Reason:		
State:	isPlayerAtPos = false	This case is unique because it looks for the absence of a player token above a player		
	State of the board is unchanged	token Function Name: testIsPlayerAtPos_blankSpaceAboveToken		
pos.getRow = 1 pos.getColumn = 0 player = 'X'				

Input:				Output:	Reason: This case is unique because	
State:				For all positions in Column 0, isPlayeratPos = true	the column whose positions are being checked is	
x				State of the board is	completely full of 'X'	
x				unchanged	Function Name: testIsPlayerAtPos fullCol	
Х					tootion layer, ta es_iaireoi	
X						
X						
pos.getColumn = 0 player = 'X'						

Input:					Output:	Reason:
State:					For all positions in the board, isPlayeratPos = false	This case is unique because all positions in the board are 'X's
Χ	X	X	Х	X		Function Name:
Х	Х	Х	Х	Х		testIsPlayerAtPos_fullBoard
Х	Х	Х	Х	Х		
Х	Х	Х	Х	Х		
Χ	Х	Х	Х	Х		
Player = 'X'						

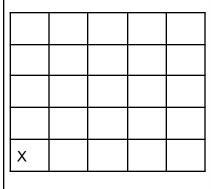
void dropToken(char p, int c)

Input:



Output:

State:



Reason:

This case is unique because the token being placed is the first player token in the board

Function Name:

testDropToken_emptyBoard

Input:

State:

Ι.					
	X	Х	Χ	Х	
	X	Х	Х	Х	
	Х	Х	Х	Х	
	Х	Х	Х	Х	
	Х	Х	Х	Х	

o	=	'X'
_	_	1

Output:

State:

X	(X	X	Χ	
×	(Х	Х	Х	
×	(Х	Х	Х	
×	(Х	Х	Х	
\backslash	(Х	Х	Х	Х

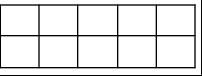
Reason:

This case is unique because the token is being placed in the last available column

Function Name:

 $testDropToken_lastRemainin\\gCol$

State:



Output:

State:

Γ			
L			
L			

Reason:

This case is unique because the token is being placed on top of another token

Function Name:

testDropToken_colNotEmpty

Х	0	Х	

		0	
Х	0	Х	

State:

		0	
		Х	
		0	
Х	0	Х	

Output:

State:

		Х	
		0	
		Х	
		0	
Х	0	Х	

Reason:

This case is unique because the token is being placed on top of another token to serve as the token that makes the column full

Function Name:

testDropToken_colAlmostFull

ın	n	
	~	

State:

Х	Х	Х	Х	
Х	Х	Х	Х	Х
Х	Х	Х	X	Х
Х	Х	Х	Х	Х
Х	Х	Х	Х	Х

p = 'X' c = 4

Output:

State:

_					
	X	X	X	X	X
	X	X	X	X	X
	X	X	Х	Х	X
Ī	Х	Х	Х	Х	Х
	Х	Х	Х	Х	Х

Reason:

This case is unique because the token is being placed on top of another token to serve as the token that makes the column and the entire board full

Function Name:

testDropToken_lastSpaceInB oard