Parthiv:

Test Cases:

Constructor: GameBoard(int row, int column, int numTokens) Test Case 1:

Input: State: int row = 3 int column = 3 Int numTokens = 3	Output: getNumRows = row; getNumColumns = column; getNumToWin = numTokens;	Reason: The following test case is unique because the constructor is passed on with the minimum values for, number of rows, columns and tokens required for the game.
	Expected output of board is empty	. Function Name: testConstructor_min_values

Test Case 2:

Input: State: int row = 100 int column = 100 Int numTokens = 25	Output: getNumRows = row; getNumColumns = column; getNumToWin = numTokens; Expected output of board is empty	Reason: The following test case is unique because constructor is passed on with the maximum values for number of rows, columns and tokens required for the game. Function Name: testConstructor_max_values
---	---	--

Test Case 3:

Input: State: int row = 27 int column = 55 Int numTokens = 15	Output: getNumRows = row; getNumColumns = column; getNumToWin = numTokens;	Reason: The following test case is unique because the constructor is passed on with all the random values for setting up the game board.
	Expected output of board is empty	Function Name: testConstructor_random_valu es

CONNOR:

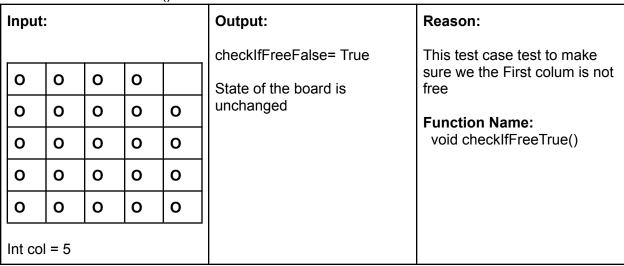
void checkIfFreeFalse()

Inpu	t:				Output:	Reason:		
	1	1		 1	checklfFreeFalse= false	This test case test to make		
X	X	X	X	X	State of the board is	sure we the third colum is not free Function Name:		
x	x	X	x	x	unchanged			
Х	Х	Х	Х	Х		void checkIfFreeFalse()		
X	х	х	Х	х				
О	0	0	0	0				
Int co	ol = 3							

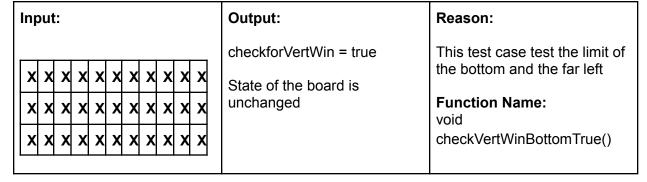
void checkIfFreeTrue()

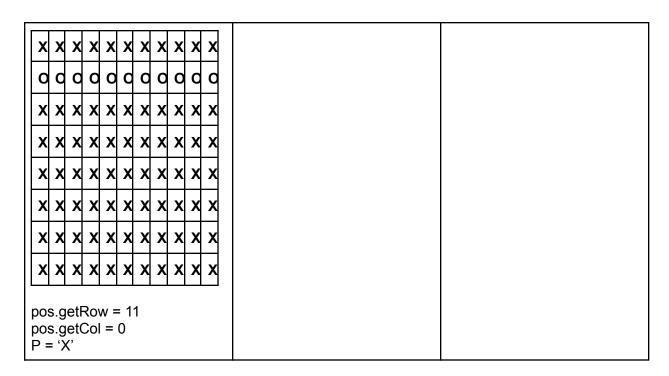
Input	:				Output:	Reason:		
	О	О	О	О	checkIfFreeFalse= True State of the board is	This test case test to make sure we the First colum is not free		
	О	0	o	О	unchanged	Function Name:		
	0	0	0	0		void checklfFreeTrue()		
	0	0	0	0				
	О	0	О	О				
Int co	l = 0							

void checkIfFree99True()



void checkVertWinBottomTrue()





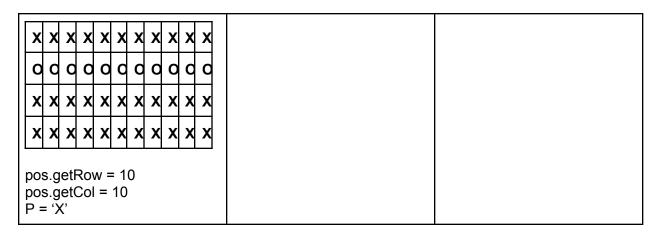
void checkVertWinBottomFlase()

Inp	Input:										Output:	Reason:
 									1	_	checkforVertWin = false	This test case test the limit of the bottom and the far right
X	X	X	X	X	X	X	X	X	X	X		the bottom and the far right
X	X	X	X	X	X	X	X	X	X	X	State of the board is unchanged	Function Name:
X	X	X	X	X	X	X	X	X	X	X		checkVertWinBottomFlase()
X	X	X	X	X	X	X	X	X	X	X		
x	X	Х	X	X	X	X	X	X	X	X		
x	Х	Х	Х	X	X	X	X	X	X	X		
X	Х	Х	Х	X	X	X	X	X	X	X		
x	Х	Х	X	X	X	X	X	X	X	X		
0	О	О	O	0	Q	q	O	0	q	d		
X	Х	Х	Х	X	X	X	X	X	X	X		
X	X	X	X	X	X	X	X	X	X	X		
po	pos.getRow = 0 pos.getCol = 10 P = 'X'											

void checkVertWinTopFalse()

void checkVertWinTopTrue()

Inp	nput:										Output:	Reason:
X	x x x x x x x x x x x								X	X	checkforVertWin = true State of the board is	This test case test the limit of the top and the far right
X	Х	X	Х	X	X	Х	X	X	X	X	unchanged	Function Name: void checkVertWinTopTrue()
X	Х	X	Х	Х	Х	Х	Х	Х	X	Х		()
X	Х	Х	Х	X	X	X	X	X	X	Х		
X	Х	X	Х	X	X	X	X	X	X	X		
X	х	X	Х	X	X	X	X	X	X	X		
X	X	X	X	X	X	X	X	X	X	X		



void checkHorzWinBotLeftTrue()

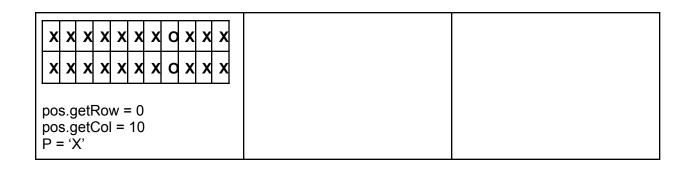
void checkHorzWinTopRightTrue()

Input:	C	Output:	Reason:
	C	checkforHoriztWin = True	This test case test the limit of the top and the far right

X	Х	х	х	O	X	X	X	X	Х	X	State of the board is unchanged	Function Name:
X	X	X	Х	O	Х	Х	X	X	X	Х	anonangou	void checkHorzWinTopRightTrue()
X	Х	Х	Х	O	X	Х	X	X	Х	X		checki lorzwii nopkigiit nue()
X	X	X	X	O	X	X	X	X	X	X		
X	X	X	X	0	X	X	X	X	X	X		
X	X	X	X	0	X	X	X	X	X	X		
X	X	X	X	O	X	X	X	X	X	X		
X	X	X	X	O	X	X	X	X	X	X		
X	X	X	X	O	X	X	X	X	X	X		
x	X	X	X	o	X	X	X	X	X	X		
X	X	X	X	O	X	X	X	X	X	X		
pos	pos.getRow = 0 pos.getCol = 10 P = 'X'											

void checkHorzWinBotRightFalse()

Inp	nput:										Output:	Reason:
									1	_	checkforHoriztWin = false	This test case test the limit of
X	X	X	X	X	X	X	Q	X	X	X	State of the board is	the Bottom and the far right
X	X	X	X	X	X	X	0	X	X	X	unchanged	Function Name:
X	X	X	X	X	X	X	O	X	X	X		checkHorzWinBotRightFalse(
Х	X	X	X	X	X	X	O	X	X	X)
Х	X	X	X	X	X	X	0	X	X	X		
Х	X	X	X	X	X	X	O	X	X	X		
Х	X	X	X	X	X	X	0	X	X	X		
Х	X	X	X	X	X	X	0	X	X	X		
х	X	X	Χ	Х	Х	X	Q	Х	Х	X		



void checkHorzWinTopLeftTrue()

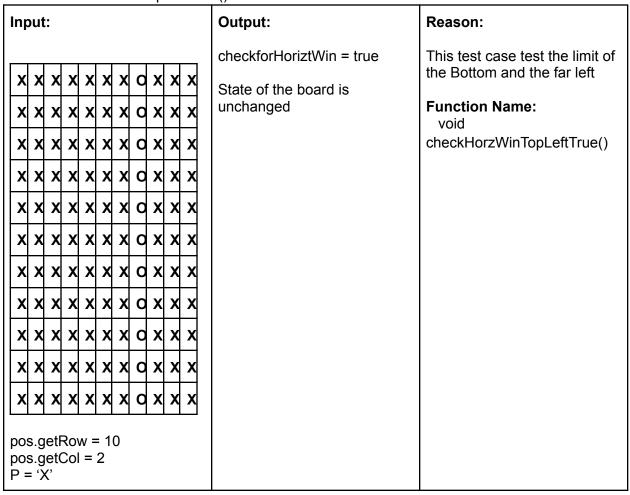
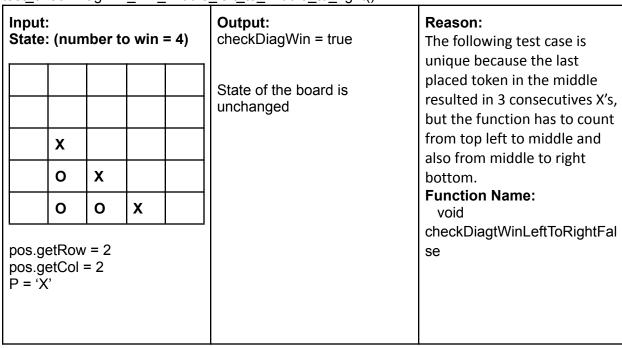


Table by Connor & Parthiv, Output and reason by Parthiv boolean checkDiagtWinLeftToRightBotTrue()

Input State		nber to	o win	= 4)	Output: checkDiagWin = true	Reason:
	•					This test case is distinct
o	0	O	0	0	State of the board is	because it checks if checkDiagWin is true when
	!		ļ.		Otate of the board is	Checkblagviii is true when

О	О	0	Х	О	unchanged	there is a diagonal chain from left bottom to top right
0	0	X	0	0		Function Name:
О	X	0	0	О		void
X	0	0	0	О		checkDiagtWinLeftToRightBot True
	jetRow jetCol K'		•			

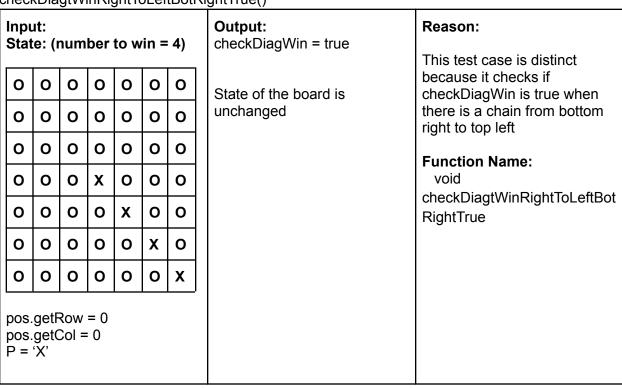
test_checkDiagWin_win_middle_left_to_middle_to_right()



checkDiagtWinLeftToRightTopRightTrue()

	ut: ite: (num	ber	to w	/in =	4)	Output: checkDiagWin = true	Reason: This test case is distinct
О	0	0 0 0 0 x					State of the board is	because it checks if checkDiagWin is true when
0	0	0	0	0	Х	0	unchanged	there is a chain from left to top right
0	0	0	0	X	0	0		Function Name:
0	0	0	Х	0	0	0		

checkDiagtWinRightToLeftBotRightTrue()



void checkDiagtWinRightToLeftTopLeftTrue()

Inpu Stat		num	ber	to w	/in =	4)	Output: checkDiagWin = true	Reason: This test case is distinct
X	0	0	O	0	0	0	State of the board is	because it checks if checkDiagWin is true when
0	x	0	0	0	0	0	unchanged	there is a chain from right to top left
0	0 X 0 0 0 0				0	0		Function Name:
0	0	0	X	0	0	0		

О	0	0	0	0	0	О	void checkDiagtWinRightToLeftTo
0	0	0	0	0	0	0	pLeftTrue
0	0	0	0	0	0	0	
pos.getRow = 5 pos.getCol = 1 P = 'X'							
void checkDiagtWinRightToLeftFalse()							

Input: State: (number to win = 4)						: 4)	Output: checkDiagWin = false	Reason: This test case is distinct
х о	о х о	о о х	0 0	0	0 0	0	State of the board is unchanged	because it checks if checkDiagWin is false when there is a chain from Left to right
0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0	0 0 0 0	0 0 0		Function Name: void checkDiagtWinRightToLeftFal se
	.getF .get('X'							

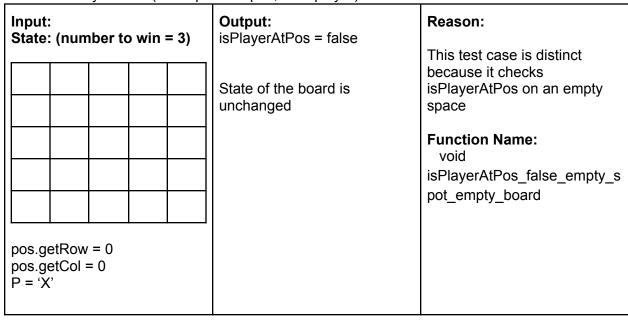
 $void\ check Diagt Win Right To Left Bot Left True ()$

Inpu Stat		num	ber	to w	/in =	· 4)	Output: checkDiagWin = true	Reason: This test case is distinct
x	0	0	o	0	0	0	State of the board is	because it checks if checkDiagWin is true when
0	Х	0	0	0	0	0	unchanged	there is a chain from Left to bottom right
0	0	X	0	0	0	0		Function Name:
0	0	0	X	0	0	0		

0	О	0	0	X	0	0
0	0	0	o	0	Х	0
0	0	0	0	0	0	X
pos. pos. P =	.getF .get('X'	Row Col :	= 0 = 6			

Prahalad:

boolean isPlayerAtPos(Boardposition pos, char player)



boolean isPlayerAtPos(Boardposition pos, char player)

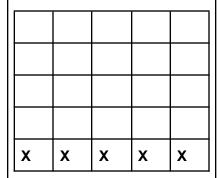
State: (number to win = 3)	isPlayerAtPos = true	This test case is distinct
	State of the board is unchanged	because it checks isPlayerAtPos on one spot with a token on a board that is empty on all other spots
0		Function Name: void isPlayerAtPos_one_char_on_ board



boolean isPlayerAtPos(Boardposition pos, char player)

Input:

State: (number to win = 3)



Output:

isPlayerAtPos = true

State of the board is unchanged

Reason:

This test case is distinct because it checks isPlayerAtPos on a character on a filled row

Function Name:

void isPlayerAtPos_one_filled_row

boolean isPlayerAtPos(Boardposition pos, char player)

Input:

State: (number to win = 3)

X	x	х	х	
Х	Х	Х	Х	х
Х	x	х	х	Х
Х	Х	х	х	Х
х	Х	х	х	Х

Output:

isPlayerAtPos = false

State of the board is unchanged

Reason:

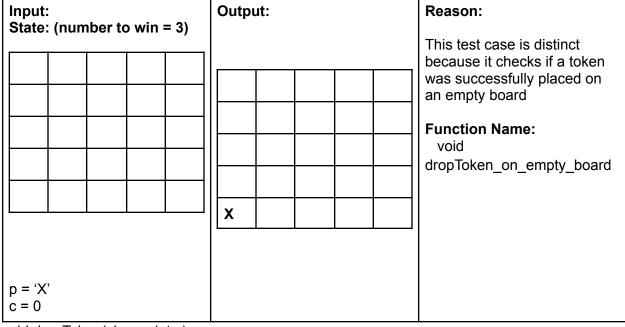
This test case is distinct because it checks isPlayerAtPos on an empty space with an otherwise full board

Function Name:

void isPlayerAtPos_false_almost_f ull_board_empty_space boolean isPlayerAtPos(Boardposition pos, char player)

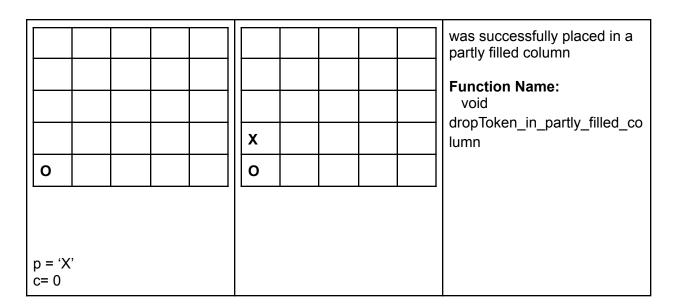
Inpu State		mber	to win	= 3)	Output: isPlayerAtPos = true	Reason: This test case is distinct
X	Х	Х	Х	Х	State of the board is	because it checks isPlayerAtPos on a character
X	X	X	X	x	unchanged	in a completely filled board
X	х	X	х	Х		Function Name: void
X	Х	Х	X	X		isPlayerAtPos_full_board
X	X	X	X	x		
	getRov getCol X'		•			

void dropToken(char p, int c)

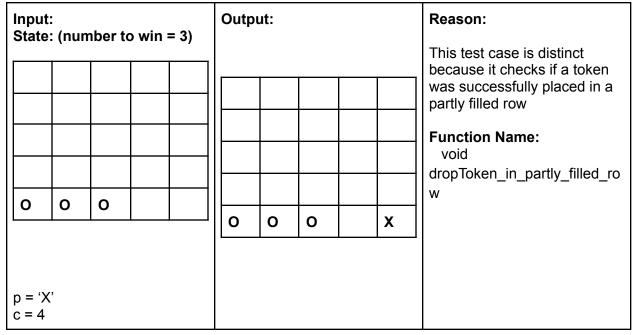


void dropToken(char p, int c)

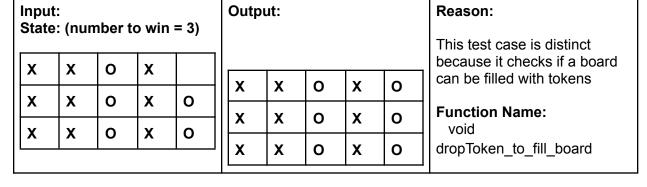
Input: State: (number to win = 3)	Output:	Reason:
		This test case is distinct because it checks if a token



void dropToken(char p, int c)



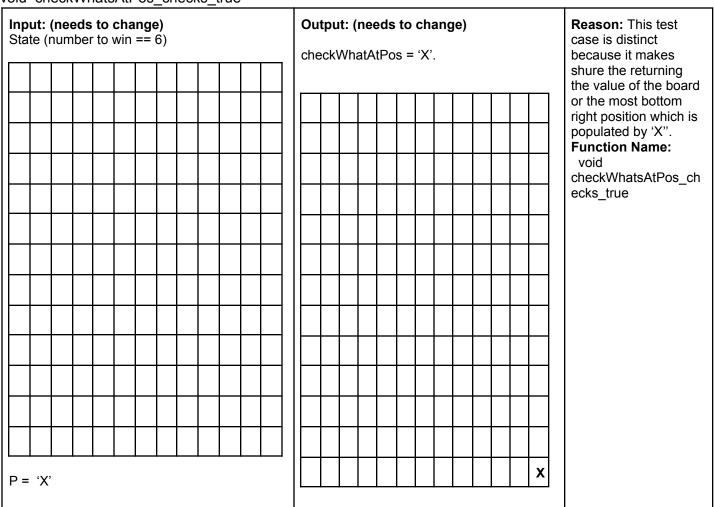
void dropToken(char p, int c)



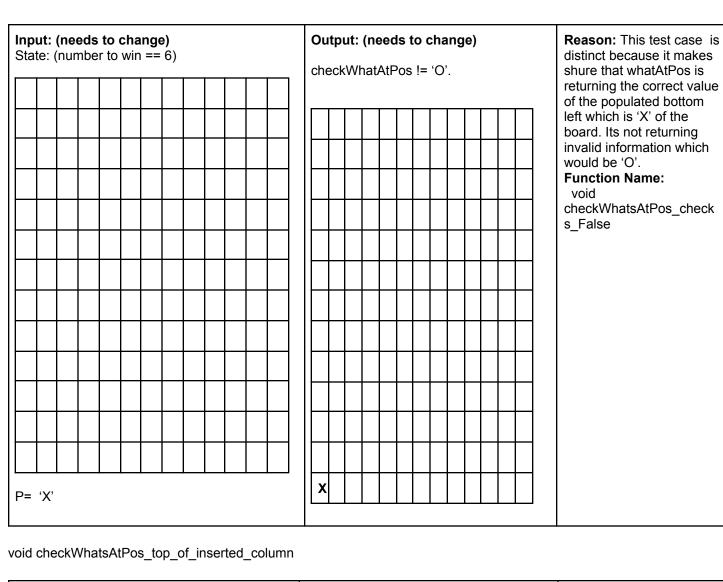
х	X	0	Х	О] [x	x	О	Х	О
X	х	0	х	0	X	Х	0	Х	0
						·			
p = 'C c = 4)'								

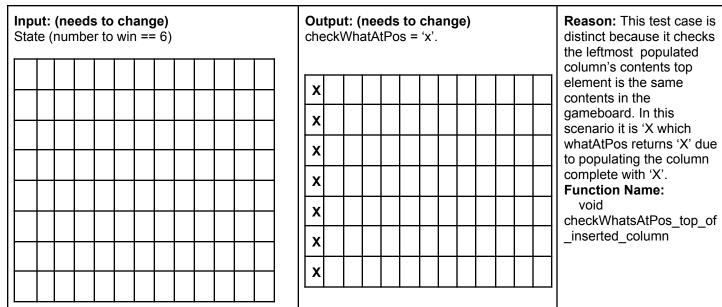
Sean Farrell Next 9 functions

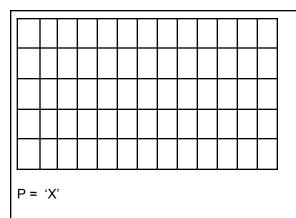
void checkWhatsAtPos_checks_true



void checkWhatsAtPos_checks_False



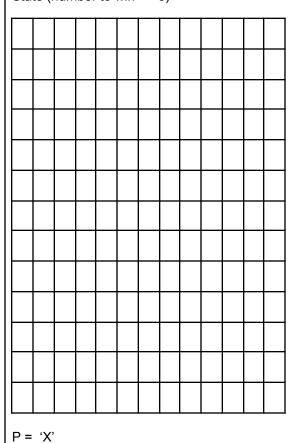




X						
x						
x						
Х						
x						
x						

void checkWhatsAtPos_empty_position

Input: (needs to change) State (number to win == 6)



Output: (needs to change)

checkWhatAtPos = ' '.

State of the gameboard is unchanged

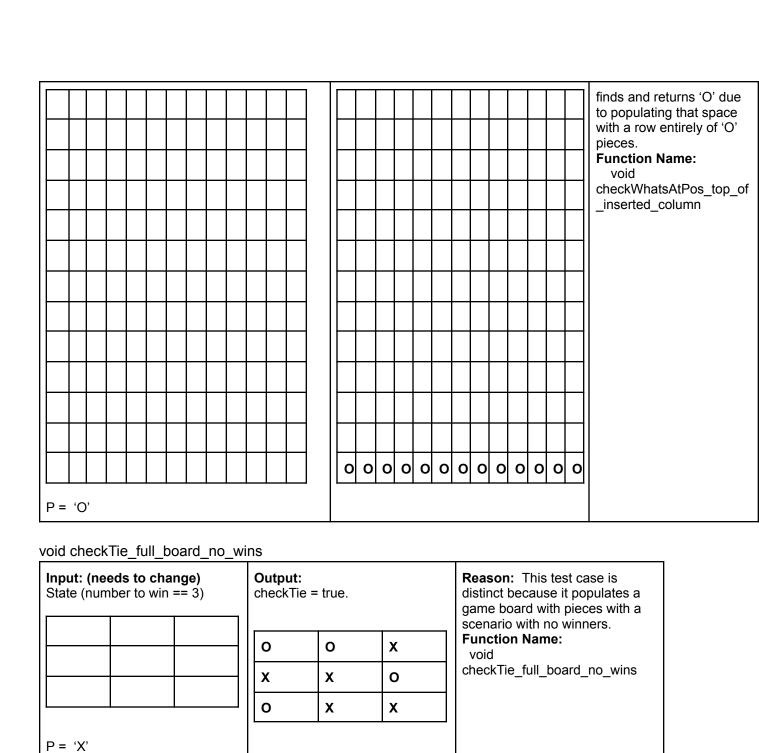
Reason: This test case is distinct it makes shure when a game board is constructed before the game starts, The top left conor will always be empty at that position. Checks to see if whatAPos returns ' ' which is empty for that space. **Function Name:** void checkWhatsAtPos e mpty_position

void checksWhatsAtPos_rightmost_position_inserted_row

Input: (needs to change)
State (number to win == 6)

Output: (needs to change) checkWhatAtPos = 'O'.

Reason: This test case is distinct because it checks the rightmost populated piece in the lowest row. It



void checkTie_partially_filled_board_with_win

Input: (needs to change) State (number to win == 3)	Output: checkTi	e = false.		Reason: This test case is distinct because it has a finished game where 'O' wins making shure the checktie method
			О	doesnt mistake a win as a tie in a not fully populated
	x	О	0	gameboard. Function Name:
		•	•	- 1

P = 'X' void checkTie_empty_board	О	х	х	Void checkTie_partially_filled_boar d_with_win		
Input: (needs to change) State (number to win == 3) P = 'X'	Output: checkTie = State of the unchanged	e gameboa	ard is	Reason: This test case is distinct because when a gameboard is constructed before the game has started. CheckTie should be false due to the game has not even started yet. You cant tie a game if you havent started the game. Function Name: void checkTie_empty_board		
checkTie_partaily_filled_board_v	with_no_wi	n				
Input: (needs to change) State (number to win == 3)	checkTie =	false.		Reason: This test case is distinct because when a game is		
	О			currently being played and the board is not yet completely full		
	х	0	О	the checktie function should return false.		
	О	х	х	Function Name: void checkTie_empty_board		
P = 'X'		•				