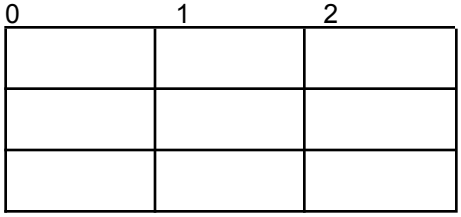
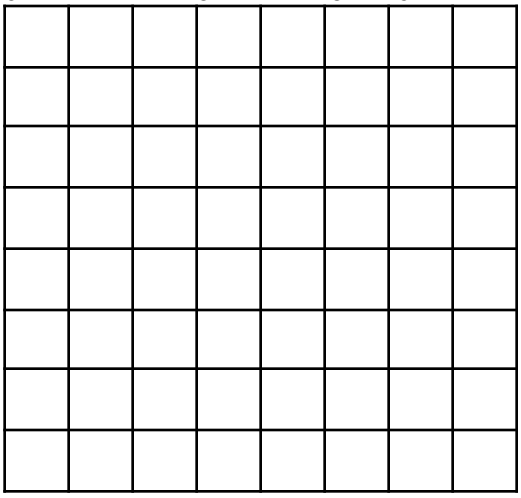
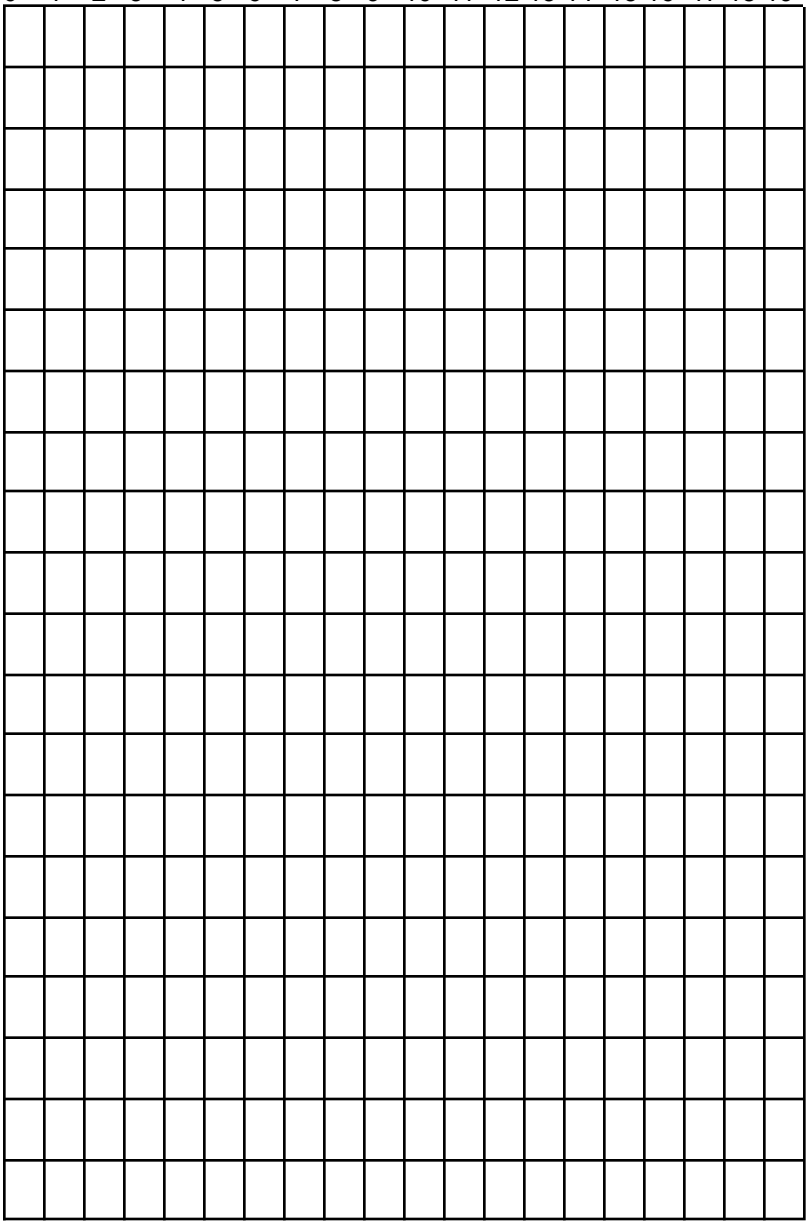
GameBoard(int entRow, int entColumn, int amtToWin)

Input:	Output:	Reason:
State: (no board has been created) entRow = 3 entColumn = 3 amtToWin = 3	State: (board of size was created) enteredRow = 3 enteredColumn = 3 enteredATW = 3 0 1 2 	This test was made to assess the ability of the constructor to make a board of various single digit size. Function Name: construct_Small_Board_Test

GameBoard(int entRow, int entColumn, int amtToWin)

Input:	Output:	Reason:
State: (no board has been created) entRow = 8 entColumn = 8 amtToWin = 5	State: (board of size was created) enteredRow = 8 enteredColumn = 8 enteredATW = 5 0 1 2 3 4 5 6 7 	This test was made to assess the ability of the constructor to make a board of equal size. Function Name: construct_Square_Board_Test

GameBoard(int entRow, int entColumn, int amtToWin)

Input:	Output:	Reason:
State: (no board has been created) entRow = 100 entColumn = 100 amntToWin = 11	State: (board of size was created) enteredRow = 100 enteredColumn = 100 enteredATW = 11 (20 by 20 table times 5, can not display a table of size 100) 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 	This test was made to assess the ability of the constructor to make a board the test the max possible size. Function Name: construct_Large_Board_Test

boolean checkIfFree(int col)

Input:	Output:	Reason:
‘col’ = 2 State: 0 1 2 3 O X O X	false State: Unchanged	This function tests that column is not free when it is completely full. Function Name: checkIfFree_Full_Test

boolean checkIfFree(int col)

Input:	Output:	Reason:
‘col’ = 1 State: 0 1 2 3	true State: Unchanged	This function tests that it accurately assess that there is a free spot in a column when it has not been populated. Function Name: checkIfFree_Empty_Test

boolean checkIfFree(int col)

Input:	Output:	Reason:
‘col’ =1 State: 0 1 2 3 X	true State: Unchanged	This function tests that it accurately assess that there is a free spot in a column when it has been populated but not fully. Function Name: checkIfFree_Partial_Test

boolean checkHorizWin(BoardPosition pos, char token)

Input:	Output:	Reason:
pos.getRow = 0 pos.getColumn = 3 'token' = X State:	True State: Unchanged	This function tests when the winning piece is placed on the lowest row to make sure it results correctly. Function Name: checkHorizWin_Low_Test

0	1	2	3
O	O	O	
X	X	X	X

boolean checkHorizWin(BoardPosition pos, char token)

Input:	Output:	Reason:
pos.getRow = 3 pos.getColumn = 0 'token' = X State:	True State: Unchanged	This function tests when the winning piece is placed on the highest row to make sure it results correctly. Function Name: checkHorizWin_High_Test

0	1	2	3
X	X	X	X
O	O	O	X
X	O	X	O
O	O	O	X

boolean checkHorizWin(BoardPosition pos, char token)

Input:	Output:	Reason:
pos.getRow = 1 pos.getColumn = 2 'token' = X State:	True State: Unchanged	This function tests when the winning piece is placed in the middle of other like pieces to ensure that it checks both directions for a win. Function Name: checkHorizWin_Middle_Test

0	1	2	3
	O		O
X	X	X	X
O	O	X	O

boolean checkHorizWin(BoardPosition pos, char token)

Input:	Output:	Reason:
pos.getRow = 2 pos.getColumn = 2 'token' = 0 State:	false State: Unchanged	This function tests when a horizontal win condition is not fulfilled Function Name: checkHorizWin_False_Test

0 1 2 3

X	O	O	
X	X	X	O
O	O	O	X

boolean checkVertWin(BoardPosition pos, char token)

Input:	Output:	Reason:
pos.getRow = 3 pos.getColumn = 0 'token' = X State:	True State: Unchanged	This function tests when the winning piece is placed on the top of the furthest left column to make sure it results correctly. Function Name: checkVertWin_Left_Test

0 1 2 3

X			
X	O		
X	O		
X	O		

boolean checkVertWin(BoardPosition pos, char token)

Input:	Output:	Reason:
pos.getRow = 3 pos.getColumn = 3 'token' = X State:	True State: Unchanged	This function tests when the winning piece is placed on the top of the furthest right column to make sure it results correctly. Function Name: checkVertWin_Right_Test

0 1 2 3

			X
		O	X
		O	X
		O	X

boolean checkVertWin(BoardPosition pos, char token)

Input:	Output:	Reason:
pos.getRow = 2 pos.getColumn = 2 'token' = X State:	True State: Unchanged	This function tests when the winning piece is not placed on the top row to make sure it results correctly. Function Name: checkVertWin_Mid_Test

0	1	2	3
		X	
	O	X	
	O	X	

boolean checkVertWin(BoardPosition pos, char token)

Input:	Output:	Reason:
pos.getRow = 2 pos.getColumn = 2 'token' = O State:	False State: Unchanged	This function tests when the winning number of pieces are placed in a column but are different tokens therefore not winning to make sure it results correctly. Function Name: checkVertWin_False_Test

0	1	2	3
	X	O	
	O	X	
	O	X	

boolean checkDiagWin(BoardPosition pos, char token)

Input:	Output:	Reason:
pos.getRow = 0 pos.getColumn = 0 'token' = O State:	True State: Unchanged	This function tests when the winning piece is placed on at the bottom left end of a diagonal. Function Name: checkDiagWin_BottomL_Test

0	1	2	3
	X	O	
	O	X	O
O	X	X	O

boolean checkDiagWin(BoardPosition pos, char token)

Input:	Output:	Reason:
pos.getRow = 3 pos.getColumn = 3 'token' = O State:	True State: Unchanged	This function tests when the winning piece is placed on at the top right end of a diagonal. Function Name: checkDiagWin_TopR_Test

0	1	2	3
			O
	X	O	X
	O	X	X
	O	X	O

boolean checkDiagWin(BoardPosition pos, char token)

Input:	Output:	Reason:
pos.getRow = 3 pos.getColumn = 0 'token' = X State:	True State: Unchanged	This function tests when the winning piece is placed on at the top left end of a diagonal. Function Name: checkDiagWin_TopL_Test

0	1	2	3
X			
O	X		
X	O	X	
O	O	X	

boolean checkDiagWin(BoardPosition pos, char token)

Input:	Output:	Reason:
pos.getRow = 0 pos.getColumn = 3 'token' = X State:	True State: Unchanged	This function tests when the winning piece is placed on at the bottom right end of a diagonal. Function Name: checkDiagWin_BottomR_Test

0	1	2	3
	X	O	
	O	X	
	O	X	X

boolean checkDiagWin(BoardPosition pos, char token)

Input:	Output:	Reason:
pos.getRow = 1 pos.getColumn = 1 'token' = O State:	True State: Unchanged	This function tests when the winning piece is placed in the middle of a right diagonal to make sure it results correctly. Function Name: checkDiagWin_BottomLToTopR_MiddleLast_Test

0	1	2	3
		O	
	O	X	
O	O	X	X

boolean checkDiagWin(BoardPosition pos, char token)

Input:	Output:	Reason:
pos.getRow = 1 pos.getColumn = 2 'token' = X State:	True State: Unchanged	This function tests when the winning piece is placed in the middle of a left diagonal to make sure it results correctly. Function Name: checkDiagWin_BottomRToTopL_MiddleLast_Test

0	1	2	3
			O
	X		O
O	O	X	X
X	O	X	X

boolean checkDiagWin(BoardPosition pos, char token)

Input:	Output:	Reason:
pos.getRow = 0 pos.getColumn = 0 'token' = O State:	False State: Unchanged	This function tests when the board is empty and checks for a diagonal win to ensure it doesn't mark the empty characters as winning. Function Name: checkDiagWin_Empty_Test

0	1	2	3

boolean checkTie()

Input:	Output:	Reason:
State: 0 1 2 3 O X O O X O X X O X O O X O X X	True State: Unchanged	This function tests if it accurately states that the board is a tie board when full of pieces Function Name: checkTieWin_Full_Test

boolean checkTie()

Input:	Output:	Reason:
State: 0 1 2 3 (blank)	False State: Unchanged	This function tests when the board is only populated with blank characters, so it should not result in a tie. Function Name: checkTieWin_Empty_Test

boolean checkTie()

Input:	Output:	Reason:
State: 0 1 2 3 (blank) (blank) (blank) X	False State: Unchanged	This function tests when only one piece is placed in the board and then checks for tie, making sure it doesn't see a piece and automatically declare tie without checking. Function Name: checkTieWin_OnlyOne_Test

boolean checkTie()

Input:	Output:	Reason:
State: 0 1 2 3 O X X O X O X X O X O O X O X	False State: 0 1 2 3 O X X O O X O X X O X O O X O X	This function tests when all pieces but one slot are filled, and the last spot on the board fulfills another win condition. Function Name: checkTieWin_finalTokenWin_Test

char whatsAtPos(BoardPosition pos)

Input:	Output:	Reason:
pos.getRow = 0 pos.getColumn = 0 Token = ‘ ‘ State: 0 1 2 3	‘ ‘ State: Unchanged	This function tests checking a position when nothing has been populated Function Name: whatsAtPos_Empty_Test

char whatsAtPos(BoardPosition pos)

Input:	Output:	Reason:
pos.getRow = 2 pos.getColumn = 2 Token = X State: 0 1 2 3 O X O O X O X X O X O O X O X X	‘X’ State: Unchanged	This function tests checking a position when all other positions have been populated Function Name: whatsAtPos_Full_Test

char whatsAtPos(BoardPosition pos)

Input: pos.getRow = 3 pos.getColumn = 0 Token = ‘ ‘ State: 0 1 2 3 [ ] [ ] [ ] [ ] [X] [ ] [ ] [ ] [O] [ ] [O] [X] [X] [O] [X] [O]	Output: ‘ ‘ State: Unchanged	Reason: This function tests when the board is partially populated and then testing an empty position to show that it is still empty Function Name: whatsAtPos_SemiFull_NoPlayer_Test

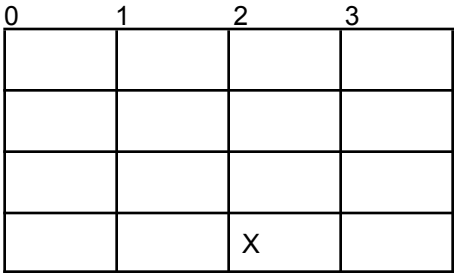
char whatsAtPos(BoardPosition pos)

Input: pos.getRow = 1 pos.getColumn = 2 Token = ‘X’ State: 0 1 2 3 [ ] [ ] [ ] [ ] [ ] [ ] [ ] [O] [O] [ ] [X] [X] [X] [O] [X] [O]	Output: ‘X’ State: Unchanged	Reason: This function tests when the board is partially populated and then test a full position to show that it has the correct character Function Name: whatsAtPos_SemiFull_Occupied_Test

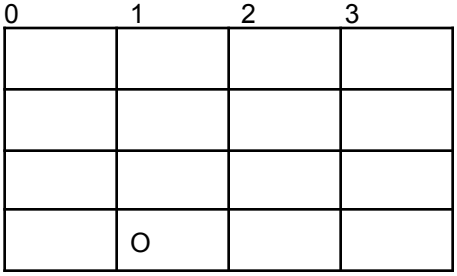
char whatsAtPos(BoardPosition pos)

Input: pos.getRow = 3 pos.getColumn = 1 Token = ‘ ‘ State: 0 1 2 3 [O] [ ] [O] [X] [X] [X] [O] [O] [O] [O] [X] [X] [X] [O] [X] [O]	Output: ‘ ‘ State: Unchanged	Reason: This function tests when the board is partially populated and then testing an empty position to show that it has the no character Function Name: whatsAtPos_SemiFull_Surrounded_Test

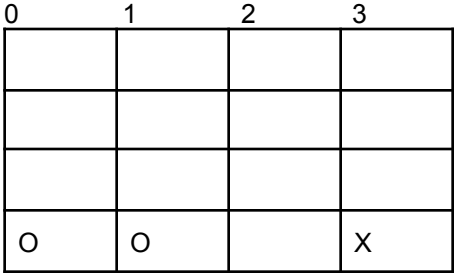
boolean isPlayerAtPos(BoardPosition pos, char player)

Input:	Output:	Reason:
pos.getRow = 0 pos.getColumn = 2 player = X State: 	True State: Unchanged	This function tests when the selected character is at the position. Function Name: isPlayerAtPos_IsAt_Test

boolean isPlayerAtPos(BoardPosition pos, char player)

Input:	Output:	Reason:
pos.getRow = 0 pos.getColumn = 1 player = X State: 	False State: Unchanged	This function tests when the selected character is not at the position, but another one is. Function Name: isPlayerAtPos_IsNotAt_Test

boolean isPlayerAtPos(BoardPosition pos, char player)

Input:	Output:	Reason:
pos.getRow = 0 pos.getColumn = 2 player = X State: 	False State: Unchanged	This function tests when a location that has no piece is checked, but there are other pieces around it. Function Name: isPlayerAtPos_NoPlayer_Test

boolean isPlayerAtPos(BoardPosition pos, char player)

Input:	Output:	Reason:
pos.getRow = 3 pos.getColumn = 3 player = O State:	True State: Unchanged	This function tests isPlayerAtPos when a token is placed at the top right of the board Function Name: isPlayerAtPos_topRight_Test

0	1	2	3
			O
			X
			O
			X

boolean isPlayerAtPos(BoardPosition pos, char player)

Input:	Output:	Reason:
pos.getRow = 3 pos.getColumn = 1 player = X State:	False State: Unchanged	This function tests when the board is full and a player is searched for in a position that they are not in. Function Name: isPlayerAtPos_Full_Test

0	1	2	3
O	O	O	X
X	X	X	O
O	O	O	X
X	X	X	O

void dropToken(char token, int column)

Input:	Output:	Reason:
pos.getColumn = 0 ‘token’ = X State:	True State: Unchanged	This function tests that after a token is dropped in the most left column of an empty board, the left bottom position is the correct token. Function Name: dropToken_Left_Bottom_Test

0	1	2	3
X			

void dropToken(char token, int column)

Input:	Output:	Reason:
pos.getColumn = 0 'token' = O State: 0 1 2 3 O X O X	True State: Unchanged	This function tests that dropToken can be used to fill the top left of the board and have the correct token. Function Name: dropToken_Left_Top_Test

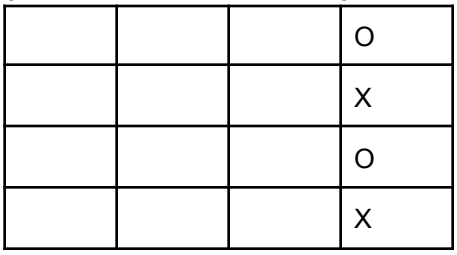
void dropToken(char token, int column)

Input:	Output:	Reason:
pos.getColumn = 1 'token' = X State: 0 1 2 3 X X X O O	True State: Unchanged	This function tests to make sure another piece is added on top when the piece is the same rather than forgoing it in the middle of the board. Function Name: dropToken_Middle_of_Board_Test

void dropToken(char token, int column)

Input:	Output:	Reason:
pos.getColumn = 3 'token' = X State: 0 1 2 3 X	True State: Unchanged	This function tests to make sure the most right column drops a token in the bottom right position of the empty board Function Name: dropToken_Right_Bottom_Test

void dropToken(char token, int column)

Input:	Output:	Reason:
pos.getColumn = 3 ‘token’ = O State: 0 1 2 3 	True State: Unchanged	This function tests that dropToken can be used to fill the top right of the board and have the correct token. Function Name: dropToken_Right_Top_Test