CheckerBoard(void) - [Test_Constructor_DefaultBoard_8x8]

Input: (8x8) Output: N/A State: N/A State: Χ Χ Х Χ Х Χ Х Х 0 0 0 0 0 * 0 0 0 0

 $whats At Pos (Board Position\ pos) - [Test_whats At Pos_Empty Square_row 3_column 1]$

Input: pos = (3, 1)

State:

State) :						
Х	*	Х	*	Х	*	Х	*
*	Х	*	Х	*	Х	*	Х
Х	*	Х	*	Х	*	Х	*
*		*		*		*	
	*		*		*		*
*	0	*	0	*	0	*	0
0	*	0	*	0	*	0	*
*	0	*	0	*	0	*	0

Output: ''

State:

[State of the board is unchanged]

whatsAtPos(BoardPosition pos) – [Test_whatsAtPos_ BlackPiece_row0_column0]

Input: pos = (0,0)

State:

otate.								
*	Х	*	Х	*	Х	*		
Х	*	Х	*	Х	*	Х		
*	Х	*	Х	*	Х	*		
	*		*		*			
*		*		*		*		
0	*	0	*	0	*	0		
*	0	*	0	*	0	*		
0	*	0	*	0	*	0		
	* X * * O *	* 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 * 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 * X * X * X * X		

Output: 'x'

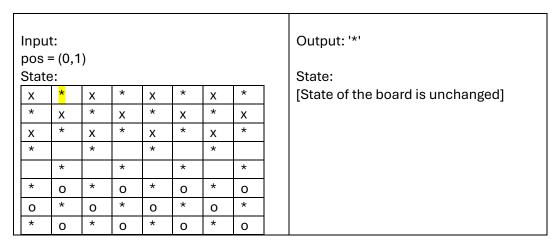
State:

[State of the board is unchanged]

whatsAtPos(BoardPosition pos) - [Test_whatsAtPos_WhitePiece_row5_column1]

Inpu pos Stat	= (5,1	1)						Output: 'o' State:
Х	*	Х	*	Х	*	Х	*	[State of the board is unchanged]
*	Х	*	Х	*	Х	*	Х	
Х	*	Х	*	Х	*	Х	*	
*		*		*		*		
	*		*		*		*	
*	O	*	0	*	0	*	0	
0	*	0	*	0	*	0	*	
*	0	*	0	*	0	*	0	

whatsAtPos(BoardPosition pos) – [Test_whatsAtPos_Black_Tile_row0_column1]



whatsAtPos(BoardPosition pos) - [Test_whatsAtPos_OutOfBounds_row8_column8]

Inpur pos : State	= (8,8	3)						Output: Throws an exception
Х	*	Х	*	Х	*	Х	*	State:
*	Х	*	Х	*	Х	*	Х	[State of the board is unchanged]
Х	*	Х	*	Х	*	Х	*	
*		*		*		*		
	*		*		*		*	
*	0	*	0	*	0	*	0	
0	*	0	*	0	*	0	*	
*	0	*	0	*	0	*	0	

 $place Piece (Board Position\ pos,\ char\ player)-[Test_place Piece_Valid Placement_row 3_column 1]$

Inpu pos play	= (3, er = '	1) x'							Out _l Stat	out: N e:	N/A					
Stat	<u>e:</u>			,				,		,		,		,	,	
	*		*		*		*			*		*		*		*
*		*		*		*			*		*		*		*	
	*		*		*		*			*		*		*		*
*		*		*		*			*	Х	*		*		*	
	*		*		*		*			*		*		*		*
*		*		*		*			*		*		*		*	
	*		*		*		*			*		*		*		*
*		*		*		*			*		*		*		*	

placePiece(BoardPosition pos, char player) [Test_placePiece_InvalidPlacement_Black_Tile_row0_column1]

playe	= (0, [·] er = '›	1) ‹'						Output: Exception Thrown
State) :							State:
	*		*		*		*	[State of the board is unchanged]
*		*		*		*		
	*		*		*		*	
*		*		*		*		
	*		*		*		*	
*		*		*		*		
	*		*		*		*	
*		*		*		*		

placePiece(BoardPosition pos, char player) – [Test_placePiece_OccupiedSquare_row3_column3]

pos :	Input: pos = (3,3) player = 'o'								
	*		*		*		*		
*		*		*		*			
	*		*		*		*		
*		*	Х	*		*			
	*		*		*		*		
*		*		*		*			
	*		*		*		*		
*		*		*		*			

Output:

Exception thrown

State:

[State of the board is unchanged]

placePiece(BoardPosition pos, char player) – [Test_placePiece_InvalidCharacter_row4_column4]

Input pos = play State	= (4,4 er = '2	•				
	*		*		*	
*		*		*		*

Sta	ate:						
	*		*		*		*
*		*		*		*	
	*		*		*		*
*		*		*		*	
	*		*		*		*
*		*		*		*	
	*		*		*		*
*		*		*		*	

Output:

Exception Thrown

State:

[State of the board is unchanged]

placePiece(BoardPosition pos, char player) – [Test_placePiece_CrownedPiece_row2_column0]

Inpu pos play	it: = (2,0 er = '))) X'						Outp		I/A					
State									*		*		*		*
	*		*		*		*	*		*		*		*	
*		*		*		*		Χ	*		*		*		*
	*		*		*		*	*		*		*		*	
*		*		*		*			*		*		*		*
	*		*		*		*	*		*		*		*	
*		*		*		*			*		*		*		*
	*		*		*		*	*		*		*		*	
*		*		*		*			1	1		1			

getPieceCounts(void) - [Test_getPieceCounts_AfterPlacement]

Inpu N/A	t:							Output: {12, 12} (12 black pieces, 12 white pieces)
State	e:							State:
Х	*	Х	*	Х	*	Х	*	[State of the board is unchanged]
*	Х	*	Х	*	Х	*	Х	
Х	*	Х	*	Х	*	Х	*	
*		*		*		*		
	*		*		*		*	
*	0	*	0	*	0	*	0	
0	*	0	*	0	*	0	*	
*	0	*	0	*	0	*	0	

getViableDirections(void) - [Test_getViableDirections_InitialSetup]

Input: N/A	Output:
State: N/A	'x' = [DirectionEnum.SE, DirectionEnum.SW] 'X' = [DirectionEnum.SE, DirectionEnum.SW,
	State: [State of the board is unchanged]

checkPlayerWin(Character player) - [Test_checkPlayerWin_PlayerWins]

Input: player = 'x' State: Χ * Χ *

Output: True

State:

[State of the board is unchanged]

checkPlayerWin(Character player) - [Test_checkPlayerWin_PlayerNotWin]

Input:	
player = 'x'	

State	:						
	*	Х	*		*		*
*		*		*		*	
	*		*	Х	*		*
*		*		*		*	
Х	*		*		*	0	*
*		*		*		*	
	*		*		*		*
*	0	*		*	0	*	

Output: False

State:

[State of the board is unchanged]

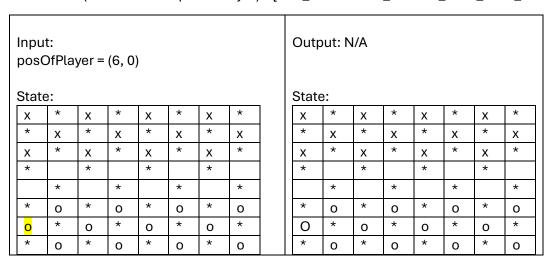
Input:

player = 'o' State:

crownPiece(BoardPosition posOfPlayer) - [Test_crownPiece_PLAYER_ONE_row2_column6]

Inpu pos		ıyer =	(2, 6)				Outp	out: N	I/A					
Stat	e:							State	e:						
Χ	*	Х	*	Х	*	Х	*	Х	*	Х	*	Х	*	Х	*
*	Х	*	Х	*	Х	*	Х	*	Х	*	Х	*	Х	*	Х
Х	*	Х	*	Х	*	X	*	Х	*	Х	*	Х	*	Χ	*
*		*		*		*		*		*		*		*	
	*		*		*		*		*		*		*		*
*	0	*	0	*	0	*	0	*	0	*	0	*	0	*	0
0	*	0	*	О	*	О	*	0	*	0	*	0	*	0	*
*	0	*	0	*	0	*	0	*	0	*	0	*	0	*	0

crownPiece(BoardPosition posOfPlayer) - [Test_crownPiece_PLAYER_TWO_row6_column0]



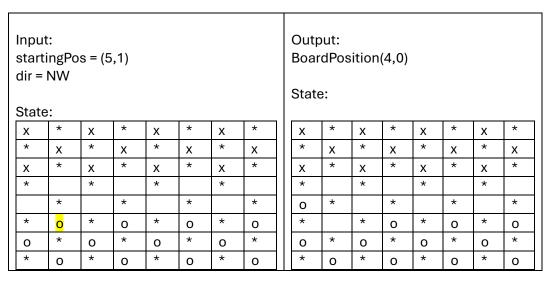
crownPiece(BoardPosition posOfPlayer) - [Test_crownPiece_NonPlayer_row0_column1]

Inpu pos(Stat	OfPla	yer =	(0, 1)				Output: Exception Thrown State:
Х	*	Х	*	Х	*	Х	*	[State of the board is unchanged]
*	Х	*	Х	*	Х	*	Х	
Х	*	Х	*	Х	*	Х	*	
*		*		*		*		
	*		*		*		*	
*	0	*	0	*	0	*	0	
0	*	0	*	0	*	0	*	
*	0	*	0	*	0	*	0	

movePiece(BoardPosition startingPos, DirectionEnum dir) – [Test_movePiece_ValidMove_SE_row2_column2]

Output: Input: startingPos = (2, 2)BoardPosition(3, 3) dir = SEState: State: Х Χ Х Х Χ * * Х Х Х Χ Χ Х Х Χ Χ Χ Х Х Х Χ 0

movePiece(BoardPosition startingPos, DirectionEnum dir) – [Test_movePiece_ValidMove_NW_row5_column1]



movePiece(BoardPosition startingPos, DirectionEnum dir) – [Test_movePiece_Crowned_SE_row0_column2]

Input start dir =	ingPo	s = (0	,2)					Outp Boar	out: dPos	ition	(1,3)				
								State	e:						
State	e:														
Х	*	O	*	Х	*	Х	*	Х	*		*	Х	*	Х	*
*	Х	*		*	Х	*	Х	*	Х	*	0	*	Х	*	Х
Х	*	Х	*		*	Х	*	Х	*	Х	*		*		*
*		*		*		*		*		*		*		*	
	*		*		*		*		*		*		*		*
*	0	*		*	0	*	0	*	0	*		*	0	*	0
0	*	0	*	0	*	0	*	0	*	0	*	0	*	0	*
*	0	*	0	*	0	*	0	*	0	*	0	*	0	*	0

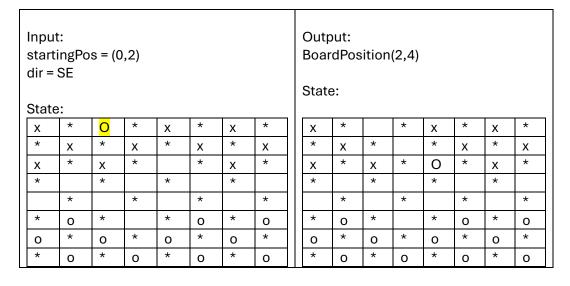
jumpPiece(BoardPosition startingPos, DirectionEnum dir) – [Test_jumpPiece_ValidJump_NW_row4_column2]

Input start dir =	ingPo	os = (4	,2)					Outp Boar		ition	(2,0)				
State	e:							State	е:						
Х	*	Х	*	Х	*	Х	*	Х	*	Х	*	Х	*	Х	*
*	Х	*	Х	*	Х	*	Х	*	Х	*	Х	*	Х	*	Х
	*	Х	*	Х	*	Х	*	0	*	Х	*	Х	*	Х	*
*	Х	*		*		*		*		*		*		*	
	*	O	*		*		*		*		*		*		*
*	0	*		*	0	*	0	*	0	*		*	0	*	0
0	*	0	*	0	*	0	*	0	*	0	*	0	*	0	*
*	0	*	0	*	0	*	0	*	0	*	0	*	0	*	0

jumpPiece(BoardPosition startingPos, DirectionEnum dir) – [Test_jumpPiece_ValidJump_SE_row2_column4]

dir=	ingPo SE	os = (2	2,4)					Outp Boar State	dPos	ition	(4,6)				
State				1		1		r		1	1	1		1	
Χ	*	Х	*	Х	*	Х	*	Х	*	Χ	*	Х	*	Х	*
*	Х	*	Х	*	Х	*	Х	*	Х	*	Х	*	Х	*	Х
Х	*	Х	*	X	*	Х	*	Х	*	Х	*		*	Х	*
*		*		*	0	*		*		*		*		*	0
	*		*		*		*		*		*		*	Х	*
*	0	*	0	*	0	*		*	0	*	0	*	0	*	
0	*	0	*	0	*	0	*	0	*	0	*	0	*	0	*
*	0	*	0	*	0	*	0	*	0	*	0	*	0	*	0

jumpPiece(BoardPosition startingPos, DirectionEnum dir) [Test_jumpPiece_Crowned_SE_row0_column2]



scanSurroundingPositions(BoardPosition startingPos) – [Test_scanSurroundingPositions_NonBoundary_row2_column2]

Input:

startingPos = (2,2)

State:

State	:						
Х	*	Х	*	Х	*	Х	*
*	Х	*	Х	*	Х	*	Х
Χ	*	X	*	Х	*	Х	*
*		*		*		*	
	*		*		*		*
*	0	*	0	*	0	*	0
0	*	0	*	0	*	0	*
*	0	*	0	*	0	*	0

Output:

DirectionEnum.NW: 'x'
DirectionEnum.NE: 'x'
DirectionEnum.SW: ''
DirectionEnum.SE: ''

State:

[State of the board is unchanged]

scanSurroundingPositions(BoardPosition startingPos) – [Test_scanSurrondingPositions_row6_column6]

Input:

startingPos = (6,6)

State:

Otato	•						
Х	*	Х	*	Х	*	Х	*
*	Х	*	Х	*	Х	*	Х
Х	*	Х	*	Х	*	Х	*
*		*		*		*	
	*		*		*		*
*	0	*	0	*	0	*	0
0	*	0	*	0	*	<mark>0</mark>	*
*	0	*	0	*	0	*	0

Output:

DirectionEnum.NW: 'o'
DirectionEnum.NE: 'o'
DirectionEnum.SW: 'o'
DirectionEnum.SE: 'o'

State:

[State of the board is unchanged]

scanSurroundingPositions(BoardPosition startingPos) –

 $[Test_scanSurrondingPositions_Crowned_row1_column3]$

Inpu start	t: :ingPo	s = (1	,3)					Output: DirectionEnum.NW = 'O' DirectionEnum.NE = 'x'
State	e:							DirectionEnum.SW = 'x'
Х	*	0	*	Х	*	Х	*	DirectionEnum.SE = ''
*	Х	*	X	*	Х	*	Х	
Х	*	Х	*		*	Х	*	
*		*		*		*		State:
	*		*		*		*	[State of the board is unchanged]
*	0	*		*	0	*	0	
0	*	0	*	0	*	0	*	
*	0	*	0	*	0	*	0	

$getDirection (DirectionEnum\ dir) - [Test_getDirection_ValidDirection_SE]$

Input:	Output:
dir = SE	new BoardPosition(1,1)
State:	State:
N/A	[State of the board is unchanged]

What tests did each team member write? Just tell me the names of the functions (unless for some reason multiple team members wrote functions for the same method. In that case, tell me which tests specifically by giving me the test names)

Nathan McMillan	movePiece(BoardPosition, DirectionEnum) jumpPiece(BoardPosition, DirectionEnum) scanSurroundingPositions(BoardPosition) getDirection(DirectionEnum)
Fahed-Dan Geravi	CheckerBoard(int) – 1 test whatsAtPos(BoardPosition) – 5 tests getPieceCounts(void) - 1 test checkPlayerWin(Character) – 2 test
Shaun Whitt	placePiece(BoardPosition, char) getViableDirections(void) crownPiece(BoardPosition)
[member 4]	