Test Cases for Board Class

setUpPits
public void setUpPits()
Establishes 12 empty Pits in the board

Category of Test	Test Case	Expected Result
setUpPitsTest	setUpPits()	Each pit has 4 stones

setUpStores
public void setUpStores()
Establishes 2 empty Stores in the board

Category of Test	Test Case	Expected Result
setUpStoresTest	setUpStores()	Each store has 0 stones

resetBoard
public void resetBoard()
Resets the board by redistributing stones but retains the players.

Category of Test	Test Case	Expected Result
resetBoardTest	resetBoard()	All pits have 4 stones and stores have 0

registerPlayers
public void registerPlayers(Player one, Player two)
Connects Players to their Stores. Will need to call methods in store and in player to ensure a two-way connection
Parameters: one - Player one, two - Player two

Category of Test	Test Case	Expected Result
registerPlayersTest	registerPlayers(playerOne, playerTwo);	store.get(0).getOwner = playerOne store.get(1).getOwner = playerTwo one.getStoreCount() = 0 two.getStoreCount() = 0

distributeStones

public int distributeStones(int startingPoint) throws PitNotFoundException

Helper method that distributes stones into pits and stores, skipping the opponent's store.

Parameters: startingPoint - The starting pit

Returns: The total number of stones added to pits and stores

Throws: PitNotFoundException - If the pit number is invalid

Category of Test	**Test Case**	**Expected Result**
distributeStonesP1ToP2SideTest	distributeStones(4) Pit 4 has 5 stones Pits start off w zero	Pits: 1-0, 2-0, 3-0, 4-0, 5-1, 6-1, 7-1, 8-1 9-0, 10-0, 11-0, 12-0 Store(0): 1 Store(1): 0 Return 5
distributeStonesP2ToP1SideTest	distributeStones(9) Pit 11 has 3 stones	Pits: 1-1, 2-0, 3-0, 4-0, 5-0, 6-0, 7-0, 8-0 9-0, 10-0, 11-0, 12-1 Store(0): 0 Store(1): 1 Return 3
distributeStonesSameSideTest	distributeStones(2) Pit 2 has 4 stones //captures pit 6 stone bc on empty pit	Pits: 1-0, 2-0, 3-1, 4-1, 5-1, 6-0, 7-0, 8-0 9-0, 10-0, 11-0, 12-0 Store(0): 1 Return 4
distributeStonesOutOfBoundsTest	distributeStones(0) distributeStones(13)	PitNotFoundException
distributeStonesSkipP2StoreTest	distributeStones(6) Pit 6 has 9 stones	Pits: 1-1, 2-1, 3-0, 4-0, 5-0, 6-0, 7-1, 8-1 9-1, 10-1, 11-1, 12-1 Store(0): 1 Store(1): 0 Return 9
distributeStonesSkipP1StoreTest	distributeStones(12) Pit 12 has 10 stones //captures stones	Pits: 1-1, 2-1, 3-1, 4-0, 5-1, 6-1, 7-1, 8-1 9-0, 10-0, 11-0, 12-0 Store(0): 0 Store(1): 3 Return 11

captureStones
public int captureStones(int stoppingPoint) throws PitNotFoundException
Captures stones from the opponent's pits.
Parameters: stoppingPoint - The stopping pit
Returns: The number of stones captured, if any
Throws: PitNotFoundException - If the pit number is invalid

Category of Test	Test Case	Expected Result
captureStonesP1Test	captureStones(3) Pit 3 is empty (with one just distributed) and opp (10 pit) side has 4 stones	Pits: 1-0, 2-0, 3-1, 4-0, 5-0, 6-0, 7-0, 8-0, 9-0, 10-0, 11-0, 12-0 Store(0): 5 Store(1): 0 Return 5
captureStonesP2Test	captureStones() Pit 9 is empty (with one just distributed) and opp (pit 4) side has 2 stones	Pits: 1-0, 2-0, 3-0, 4-2, 5-0, 6-0, 7-0, 8-0, 9-0, 10-0, 11-0, 12-0 Store(0): 0 Store(1): 3 Return 3
captureStonesOutOfBoundsTest	captureStones(13)	PitNotFoundException

getNumStones
public int getNumStones(int pitNum) throws PitNotFoundException
Gets the number of stones in a specific pit.
Parameters: pitNum - The pit number
Returns: The number of stones in the pit
Throws: PitNotFoundException - If the pit number is invalid

Category of Test	Test Case	Expected Result
getNumStonesEmptyTest	getNumStones(0)	0
getNumStonesP1StonesTest	getNumStones(3) Pit 3 has 5 stones	5
getNumStonesP2StonesTest	getNumStones(7) Pit 7 has 20 stones Pits start with 0	20
getNumStonesOutOfBoundsTest	getNumStones(13)	PitNotFoundException

isSideEmpty

public boolean isSideEmpty(int pitNum) throws PitNotFoundException

Indicates whether one side of the board is empty. An empty side indicates the end of the game.

Parameters: pitNum - The pit number

Returns: true if the side of the board that includes the parameter pit number is empty

Throws: PitNotFoundException

Category of Test	Test Case	Expected Result
isSideEmptySideNotEmptyTest	isSideEmpty(1) Is not empty	false
isSideEmptyEmptyTest	isSideEmpty(7) Is empty	true
isSideEmptyOutOfBoundsTest	isSideEmpty(0) isSideEmpty(13)	PitNotFoundException
isSideEmptyStartPit	isSideEmpty(4)	true

MoveStones

public int moveStones(int startPit, Player player) throws InvalidMoveException

Moves stones for the player starting from a specific pit.

Parameters: `startPit` - The starting pit, `player` - The player making the move

Returns: The total number of stones added to the corresponding store

Throws: `InvalidMoveException` - If the move is invalid

Category of Test	**Test Case**	**Expected Result**
moveStonesCaptureStonesTest	moveStones(2,player1) Remove stones from pit 7 Move 4 stones from pit 2 Capture stones in pits 6 and 7	Pits: 1-0, 2-0, 3-5, 4-5, 5-5, 6-0, 7-0, 8-0, 9-0, 10-0, 11-0, 12-0 Store(0): 5 Store(1): 0 Return 5
moveStonesCaptureValidMoveTest	captureStones() Move 4 stones from pit 3	Pits: 1-0, 2-0, 3-0, 4-5, 5-5, 6-5, 7-0, 8-0, 9-0, 10-0, 11-0, 12-0 Store(0): 1 Store(1): 0 Return 1
moveStonesCaptureInvalidMoveTest	moveStones(14,player1)	InvalidMoveException