Requirement Analysis

1. As a user I can enter my desired row and column location so my mark can be placed on the board
2. As a user I can chose the row and column size of the board so that I can change the board to be whatever size I desire
3. As a user I can chose the number of consecutive placements that result in a win so that I can play short or long matches
4. As a user I can play with one up to 9 other people on the board so I can play with large groups of people
5. As a user I can win the game Vertically/Diagonally/Horizontally so I can maybe play again
6. As a user I can tie the game so I can maybe play again
7. As a user I end the game in a tie so that I can maybe restart
8. As a user I can end the game in a win or loss so that I can maybe restart
9. As a user I can lose the game so I can restart or play again
10. As a user after I either win lose or tie I can chose to play again or close the game so I can play or go do something else
11. As a user I can see the board printed out so I can determine what my next placement is going to be
12. As a user I can choose to be any character I desire so I can choose what I character I want
13. As a user if I mis enter a position I can re enter my position so that I can continue playing
14. As a user if I chose a already chosen position I can re enter my position so that I can continue playing
15. As a user I can chose a memory efficient or speed based implementation depending on what I chose to be necessary

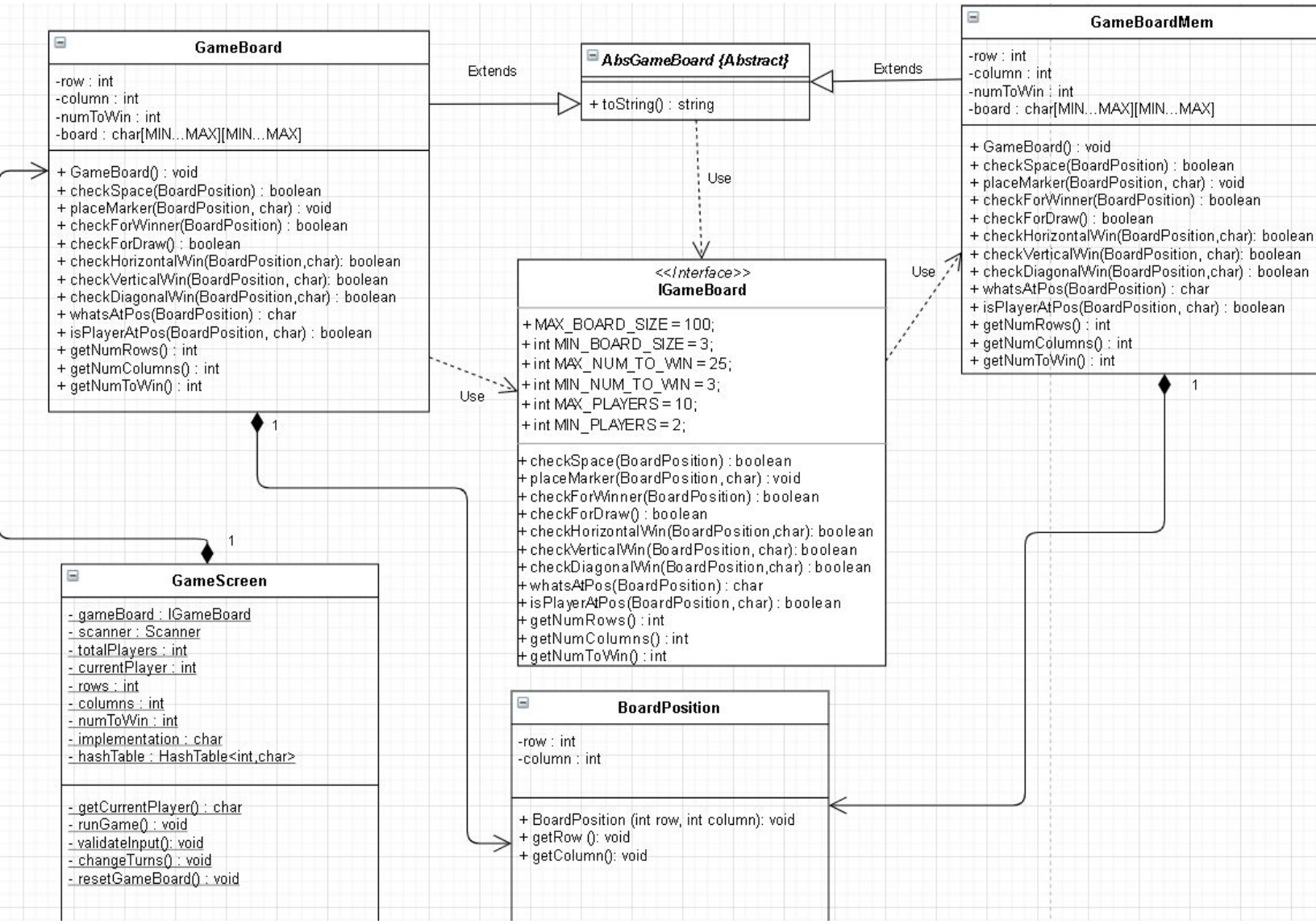
Nonfunctional Requirements

1. Must be written in java
2. Must run on Unix
3. Board is [3..100][3...100]
4. Who ever choses there character first always go first
5. (0,0) is top left
6. Must run work from makeFile

Note: Only methods that are overridden are shown. Defaults are under the interface IGameBoard. Ordering is as follows

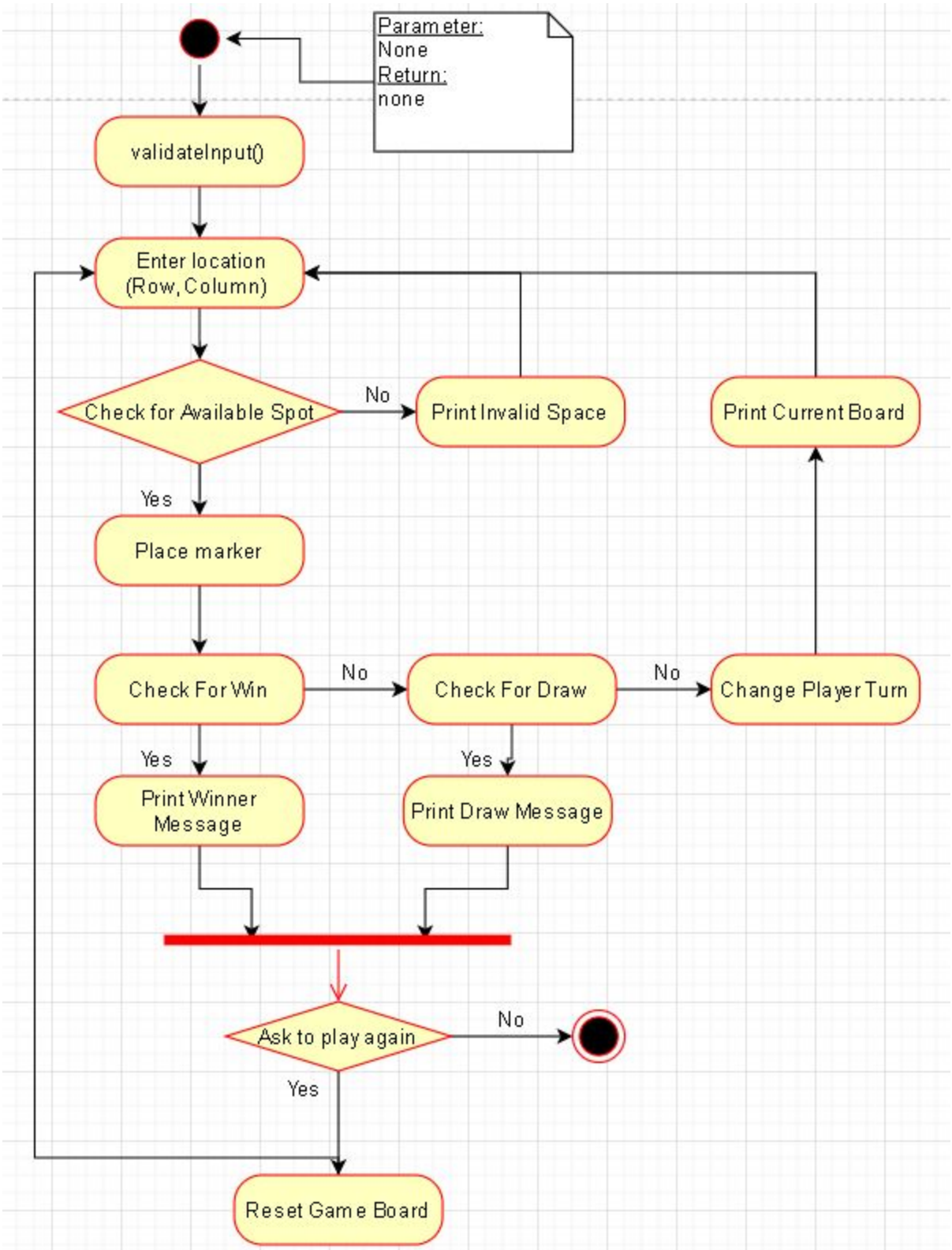
1. Design
2. GameScreen
3. BoardPosition
4. GameBoard
5. AbsGameBoard
6. GameBoardMem
7. IGameBoard

Design -

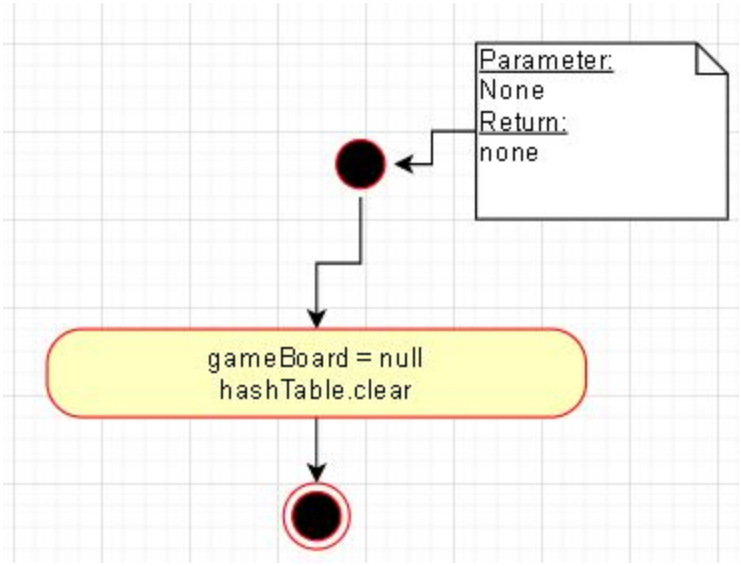

GameBoard
-row : int
-column : int
-numToWin : int
-board : char[MIN...MAX][MIN...MAX]
+ GameBoard() : void
+ checkSpace(BoardPosition) : boolean
+ placeMarker(BoardPosition, char) : void
+ checkForWinner(BoardPosition) : boolean
+ checkForDraw() : boolean
+ checkHorizontalWin(BoardPosition,char): boolean
+ checkVerticalWin(BoardPosition, char): boolean
+ checkDiagonalWin(BoardPosition,char) : boolean
+ whatsAtPos(BoardPosition) : char
+ isPlayerAtPos(BoardPosition, char) : boolean
+ getNumRows() : int
+ getNumColumns() : int
+ getNumToWin() : int
Extends
AbsGameBoard {Abstract}
+ toString() : string
Extends
GameBoardMem
-row : int
-column : int
-numToWin : int
-board : char[MIN...MAX][MIN...MAX]
+ GameBoard() : void
+ checkSpace(BoardPosition) : boolean
+ placeMarker(BoardPosition, char) : void
+ checkForWinner(BoardPosition) : boolean
+ checkForDraw() : boolean
+ checkHorizontalWin(BoardPosition,char): boolean
+ checkVerticalWin(BoardPosition, char): boolean
+ checkDiagonalWin(BoardPosition,char) : boolean
+ whatsAtPos(BoardPosition) : char
+ isPlayerAtPos(BoardPosition, char) : boolean
+ getNumRows() : int
+ getNumColumns() : int
+ getNumToWin() : int
Use
Use
Use
<<Interface>>
IGameBoard
+ MAX_BOARD_SIZE = 100;
+ int MIN_BOARD_SIZE = 3;
+ int MAX_NUM_TO_WIN = 25;
+ int MIN_NUM_TO_WIN = 3;
+ int MAX_PLAYERS = 10;
+ int MIN_PLAYERS = 2;
+ checkSpace(BoardPosition) : boolean
+ placeMarker(BoardPosition, char) : void
+ checkForWinner(BoardPosition) : boolean
+ checkForDraw() : boolean
+ checkHorizontalWin(BoardPosition,char): boolean
+ checkVerticalWin(BoardPosition, char): boolean
+ checkDiagonalWin(BoardPosition,char) : boolean
+ whatsAtPos(BoardPosition) : char
+ isPlayerAtPos(BoardPosition, char) : boolean
+ getNumRows() : int
+ getNumColumns() : int
+ getNumToWin() : int
1
1
1
GameScreen
- gameBoard : IGameBoard
- scanner : Scanner
- totalPlayers : int
- currentPlayer : int
- rows : int
- columns : int
- numToWin : int
- implementation : char
- hashTable : HashTable<int,char>
- getCurrentPlayer() : char
- runGame() : void
- validateInput(): void
- changeTurns() : void
- resetGameBoard() : void
BoardPosition
-row : int
-column : int
+ BoardPosition (int row, int column): void
+ getRow (): void
+ getColumn(): void

Class Diagram of Game Screen

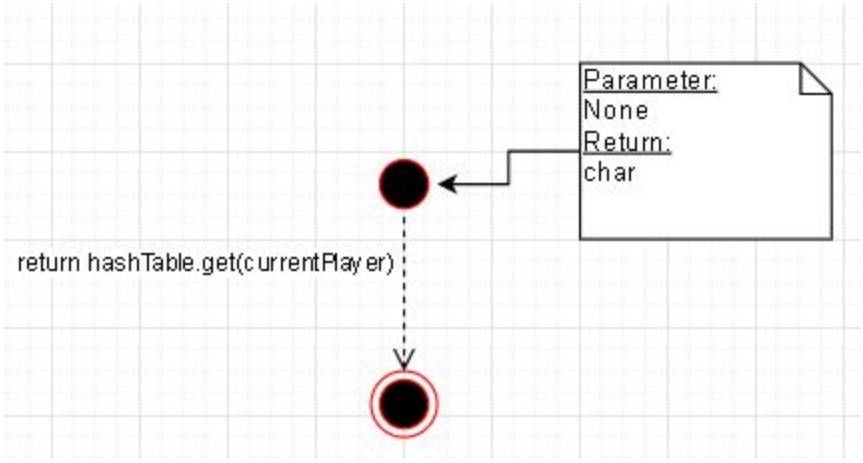
GameScreen - runGame()



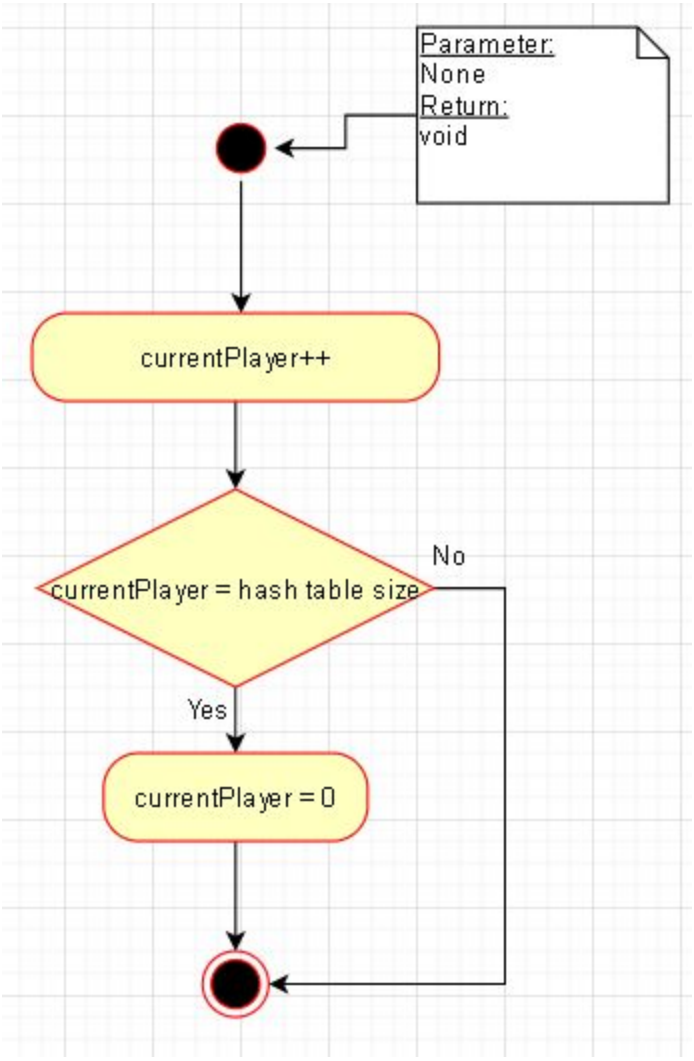
GameScreen - resetGameBoard()



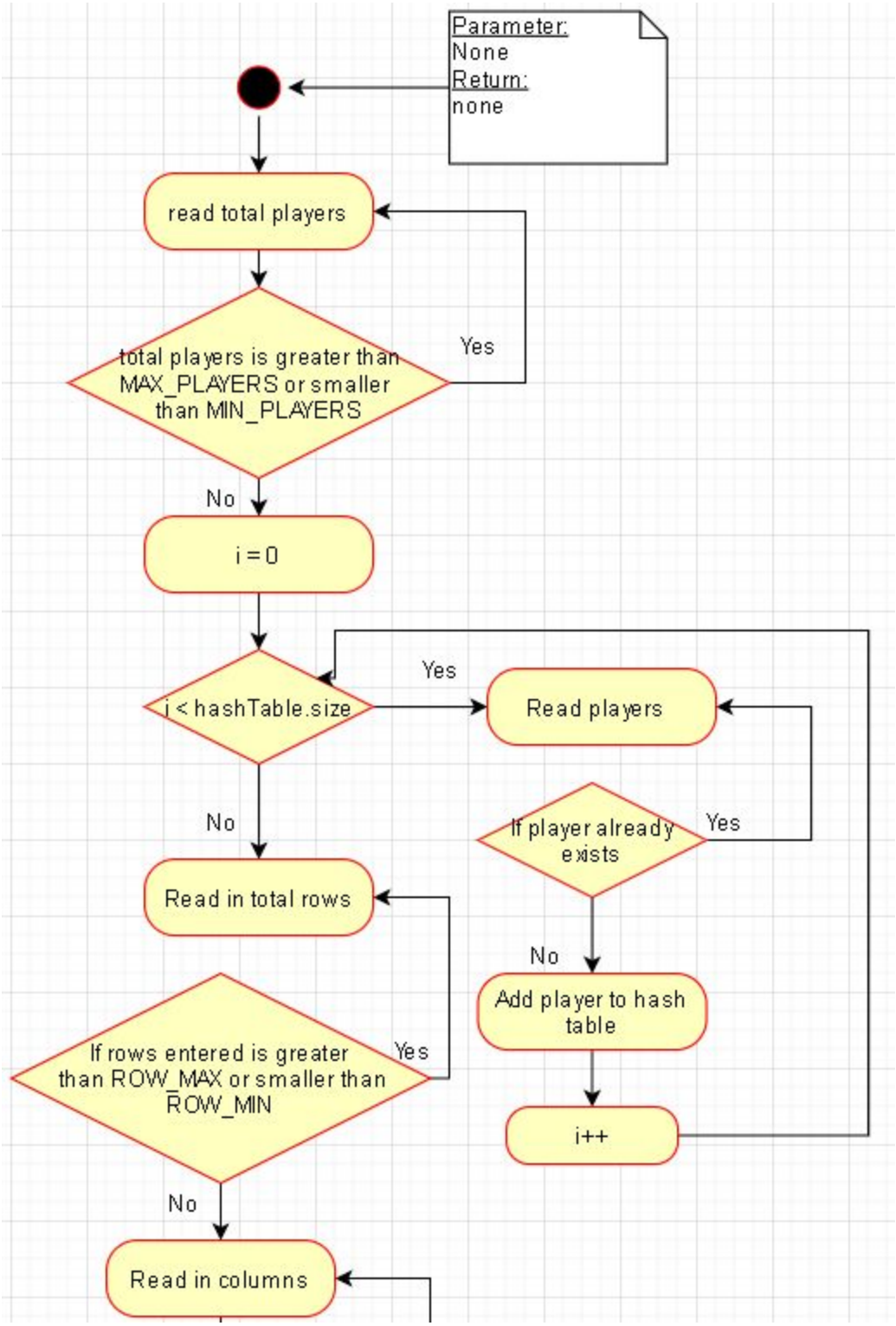
GameScreen - getCurrentPlayer()

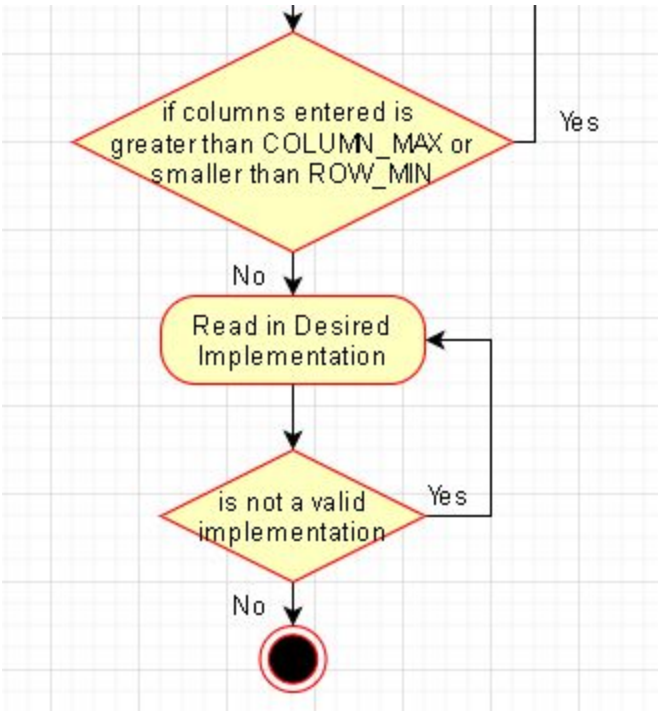


GameScreen - changeTurns()

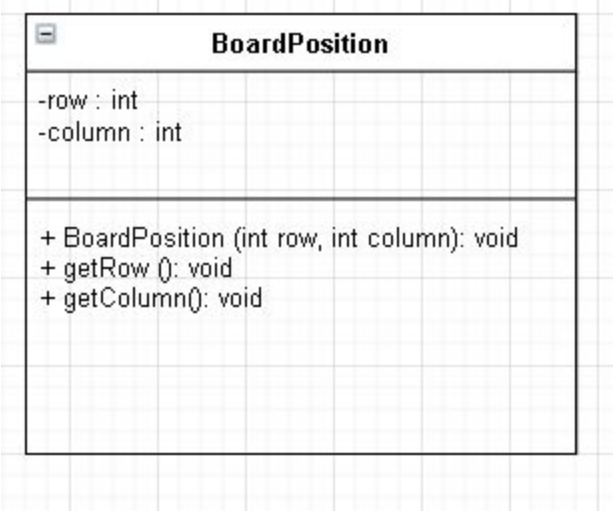


GameScreen - validateInput()

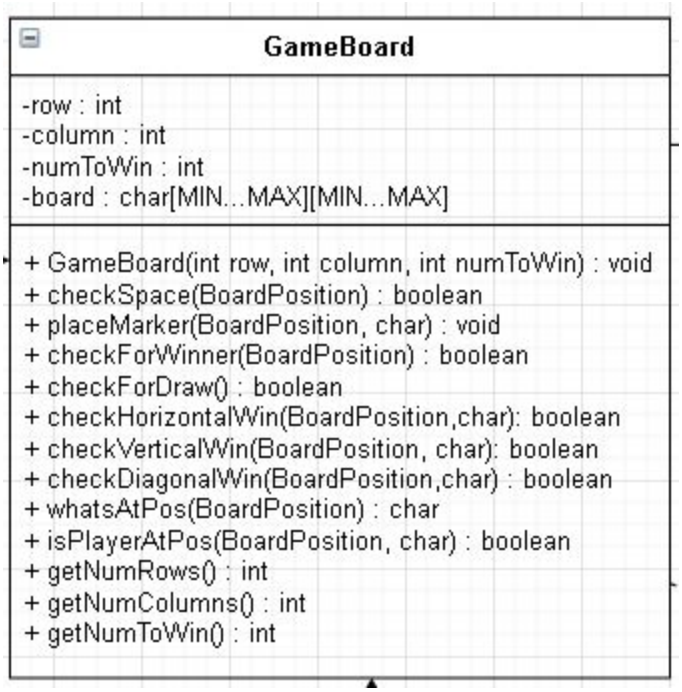




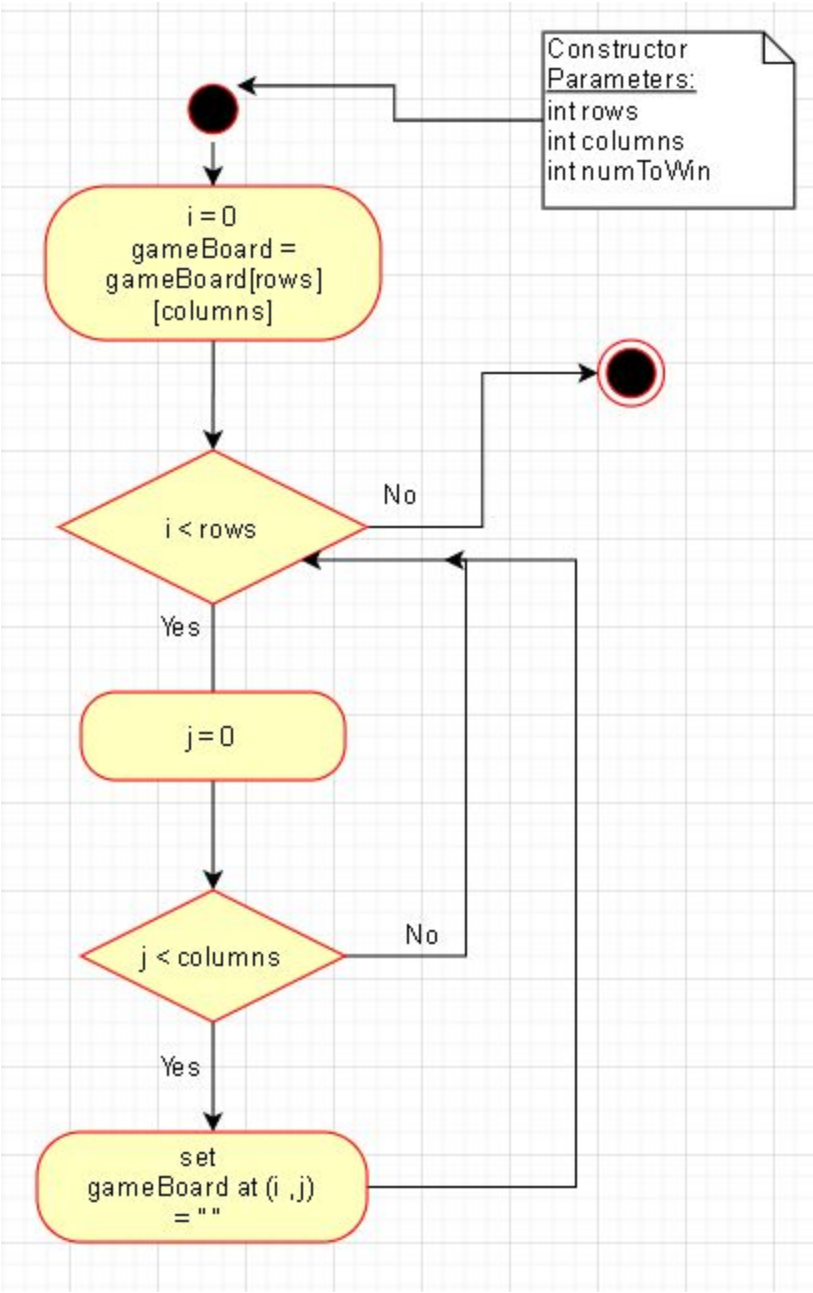
Class Diagram of Board Position



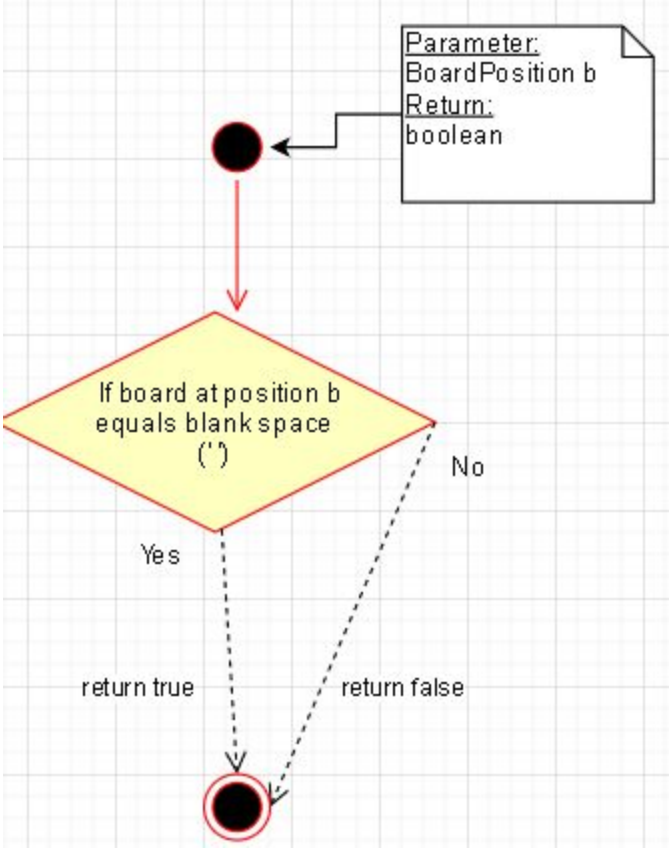
Class Diagram of GameBoard



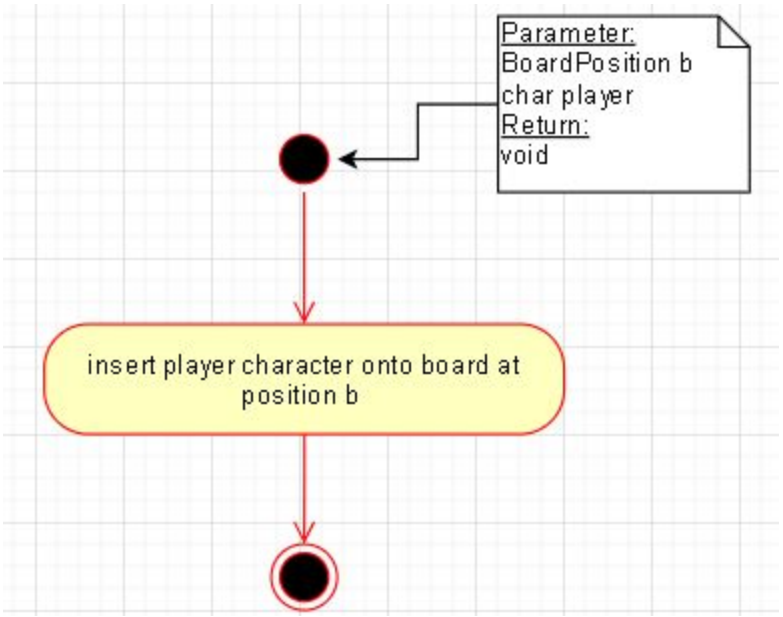
GameBoard - GameBoard()



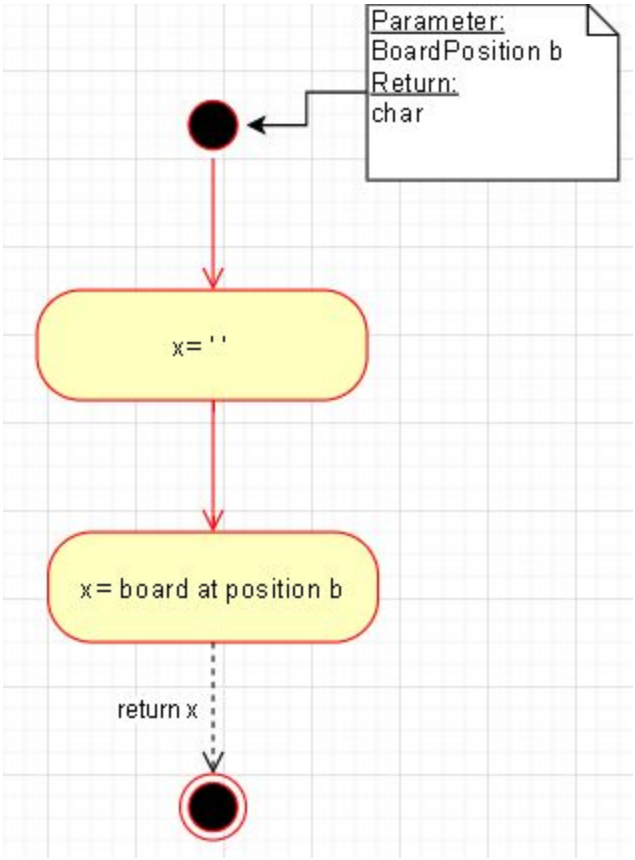
GameBoard - [override] checkSpace(BoardPosition b)



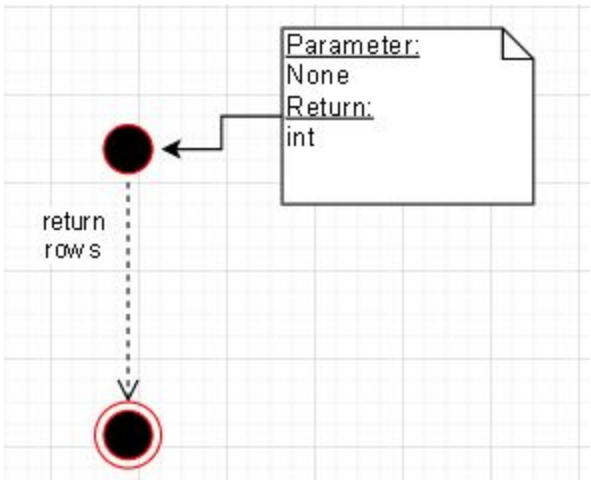
GameBoard - [override] placeMarker(BoardPosition b, char player)



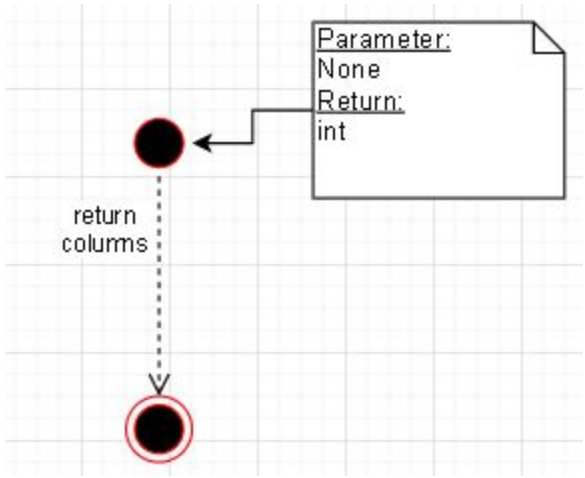
GameBoard - [override] whatsAtPos(BoardPosition b)



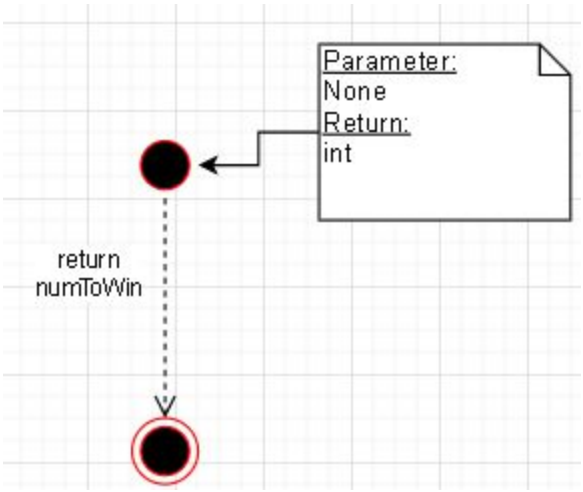
GameBoard - [override] getNumRows()



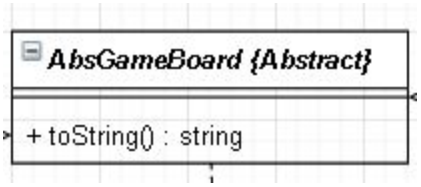
GameBoard - [override] getNumColumns()



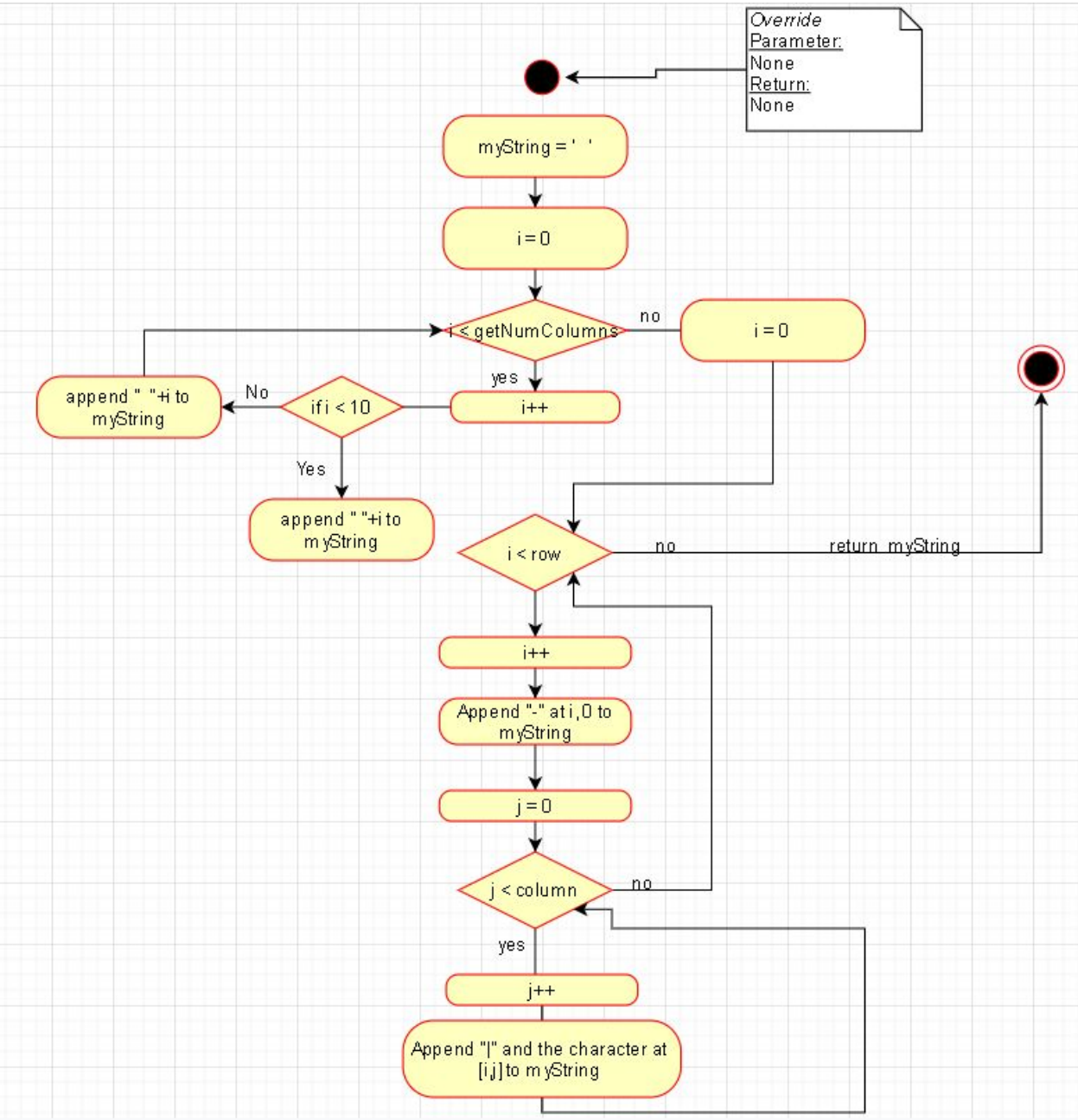
GameBoard - [override] getNumToWin()



Class Diagram of AbsGameBoard

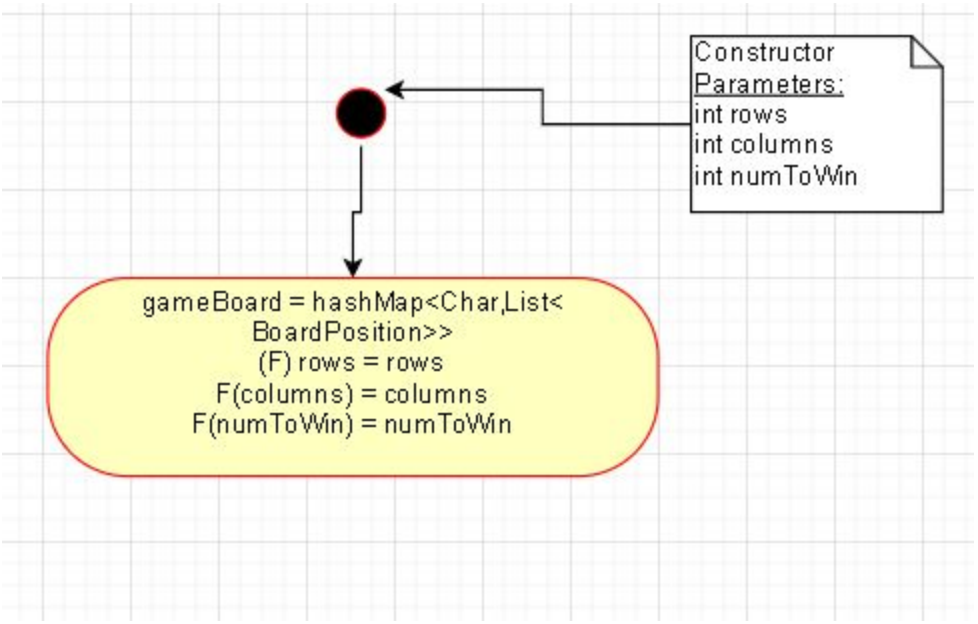


AbsGameBoard - [override] toString

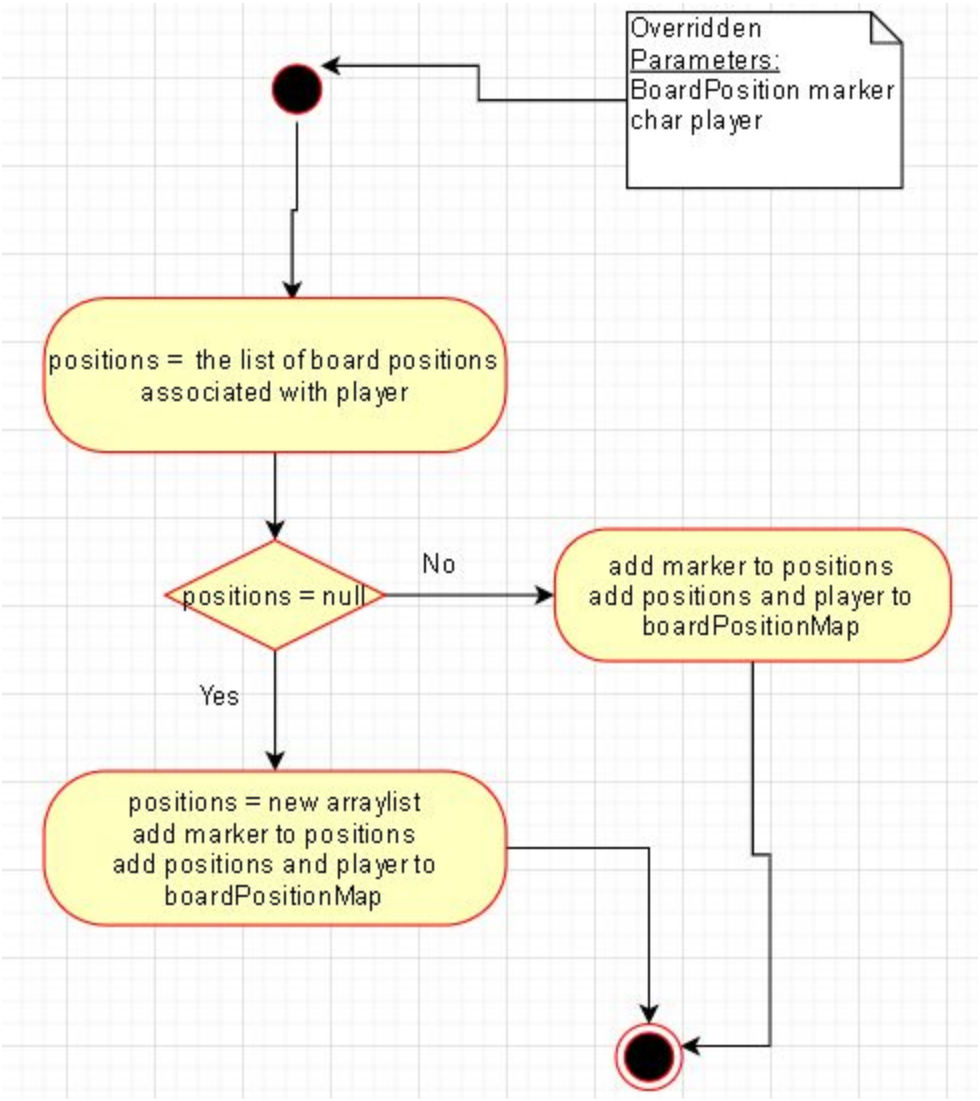


Class Diagram of GameBoardMem

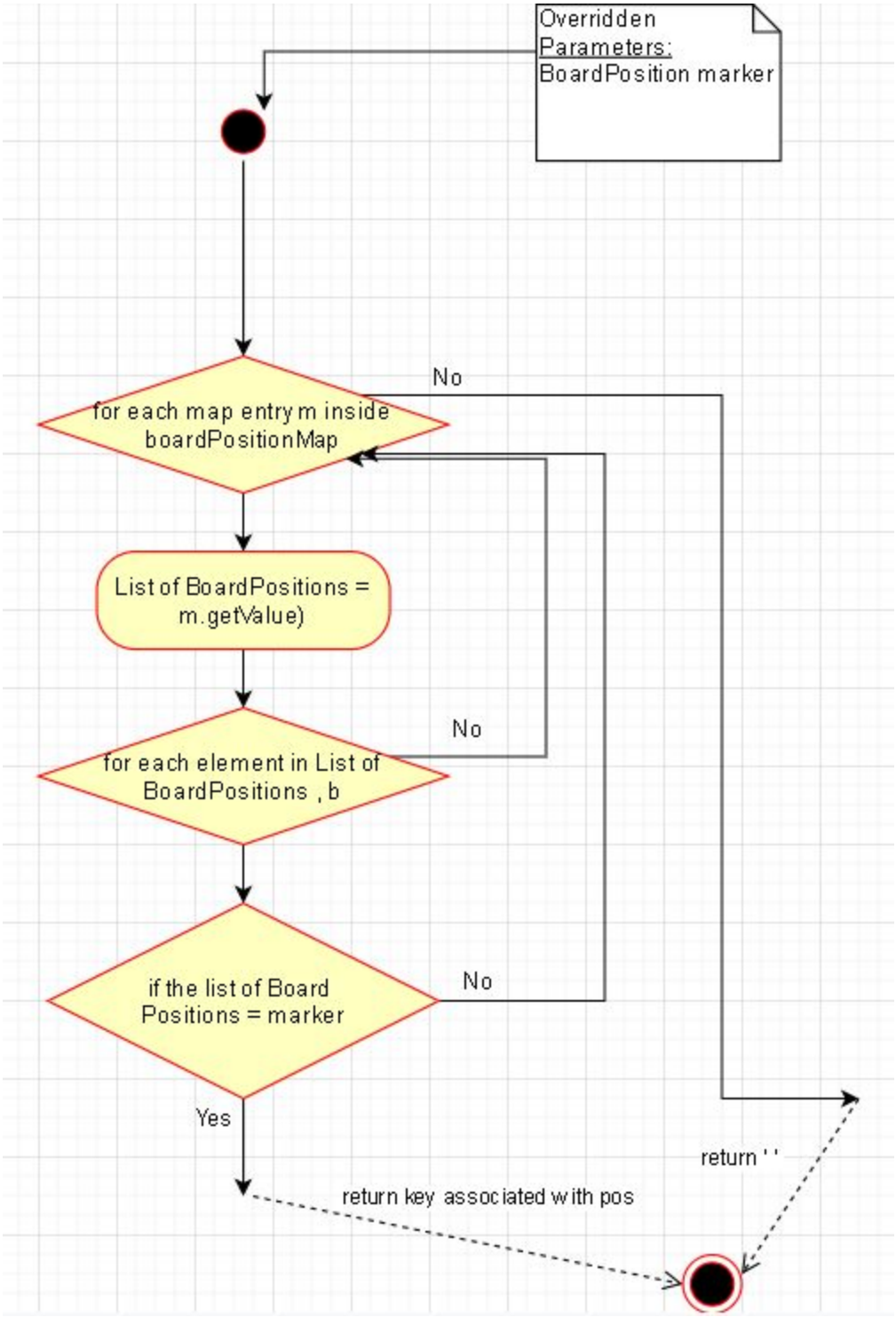
GameBoardMem - GameBoardMem()



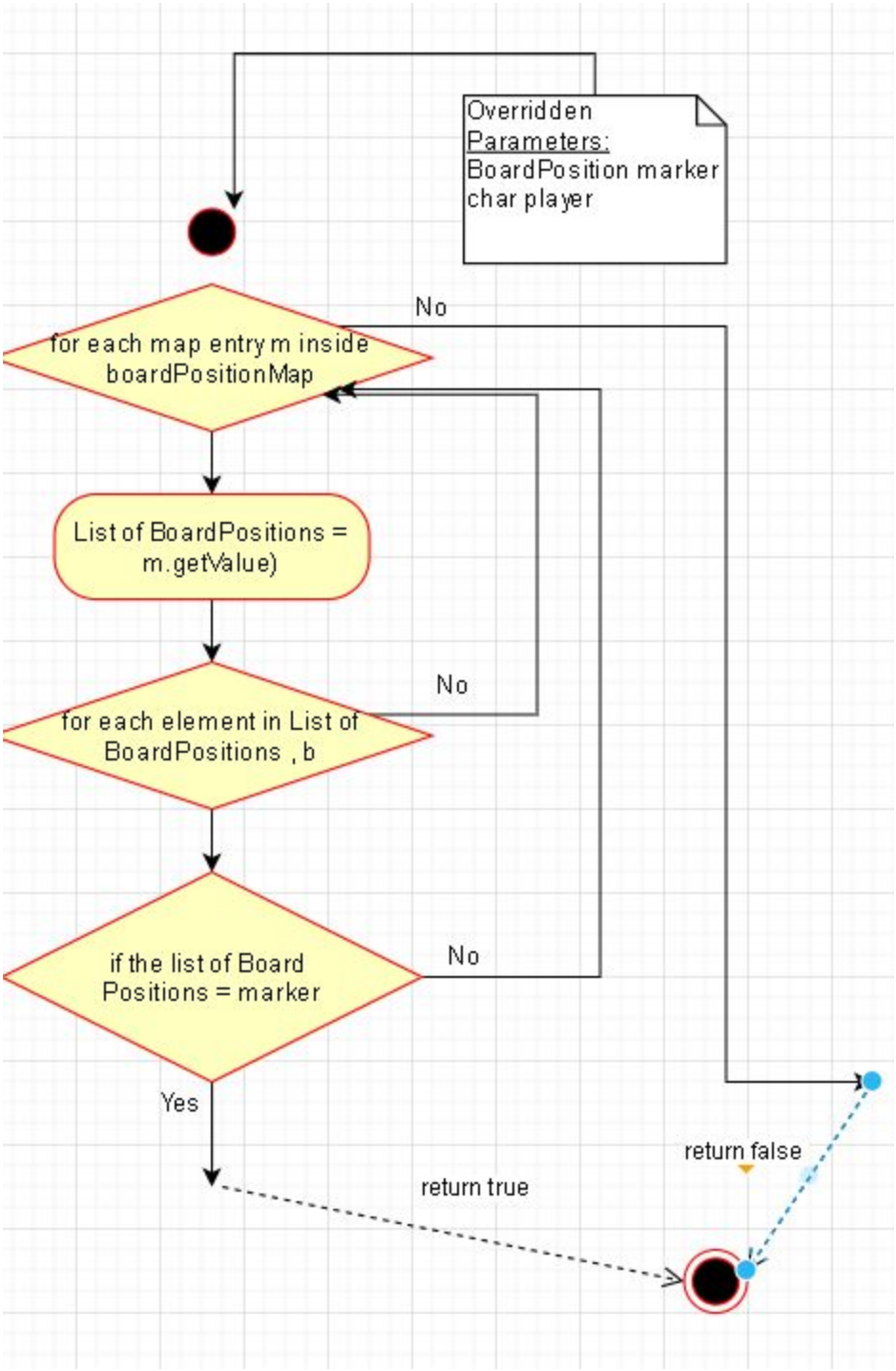
GameBoardMem - placeMarker(BoardPosition marker, char player)



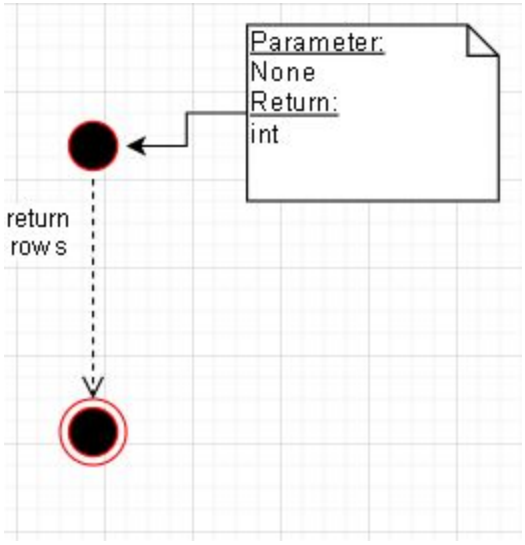
GameBoardMem - whatsAtPos(BoardPosition pos)



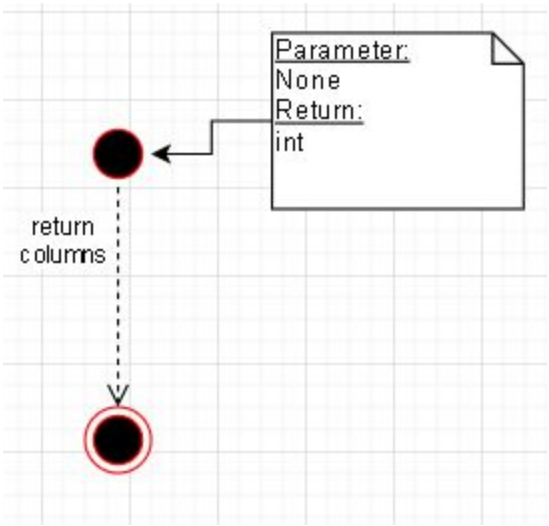
GameBoardMem - isPlayerAtPos(BoardPosition pos, char player)



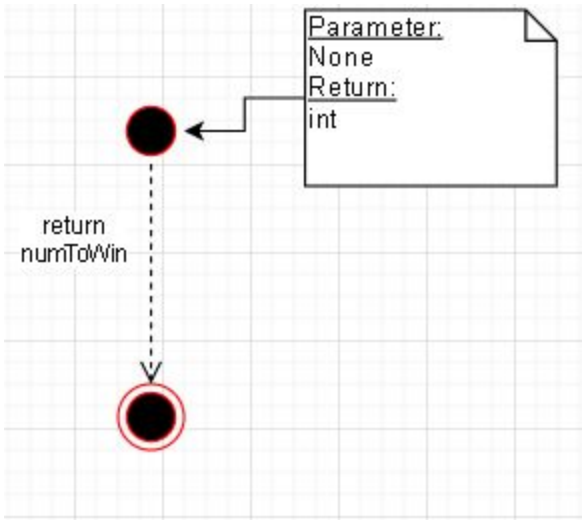
GameBoardMem - getNumRows()



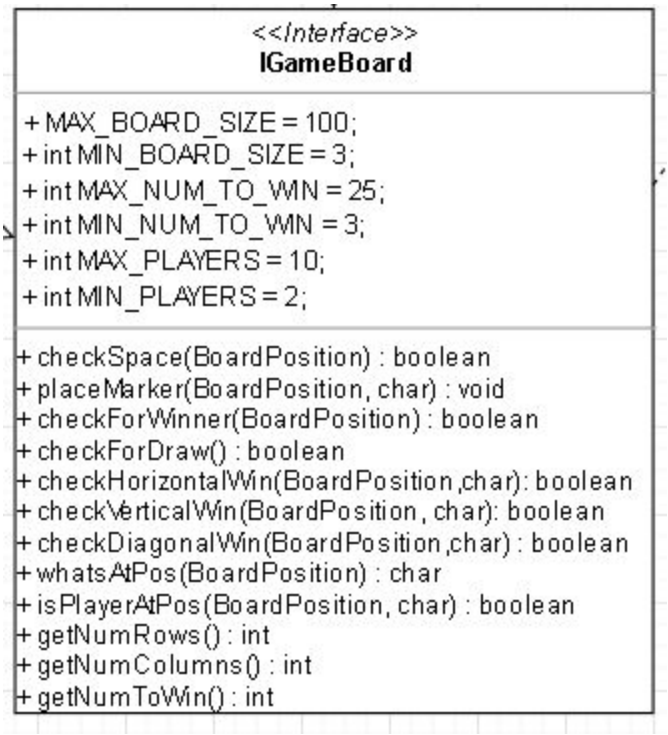
GameBoardMem - getNumColumns()



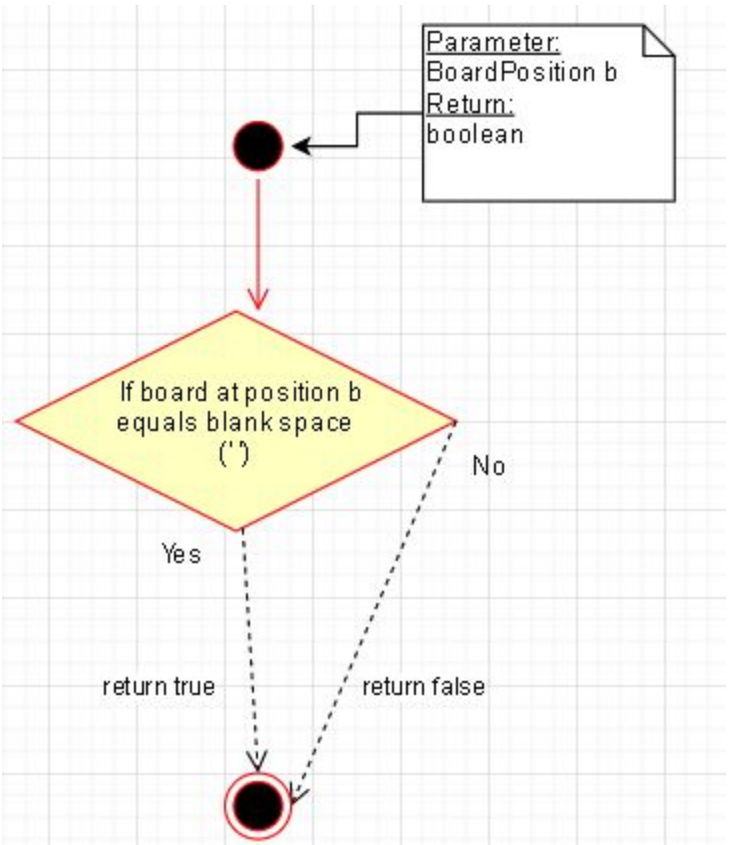
GameBoardMem getNumToWin()



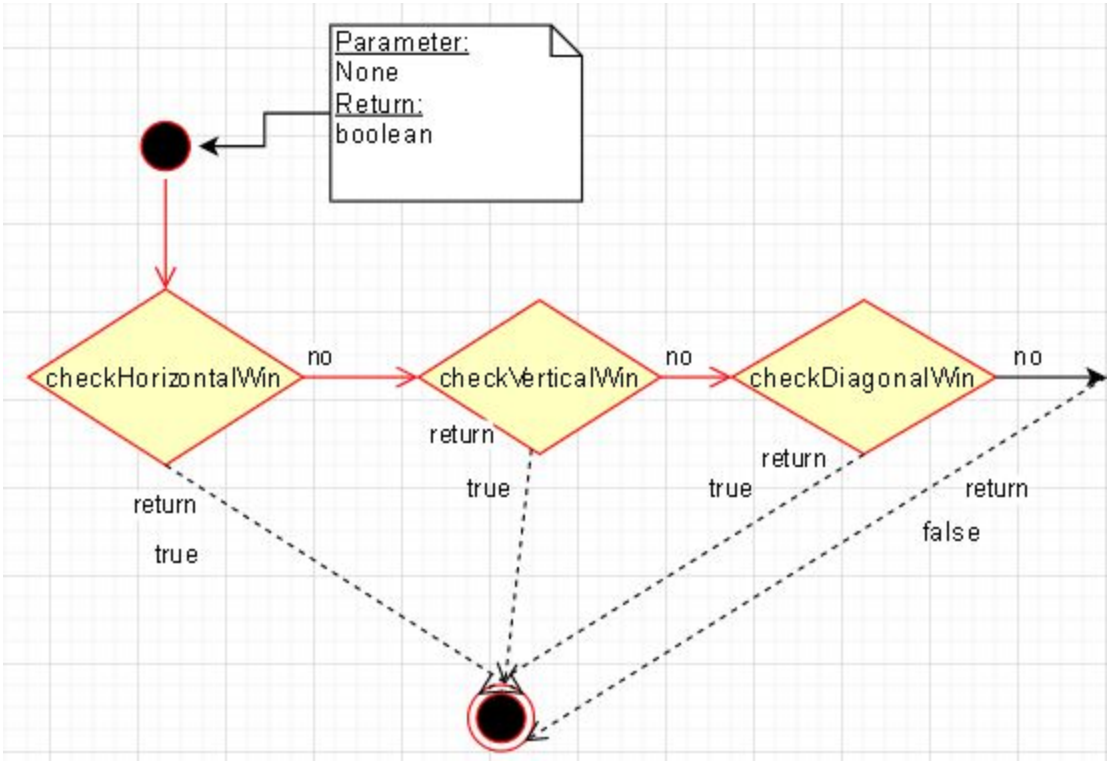
Class Diagram of IGameBoard



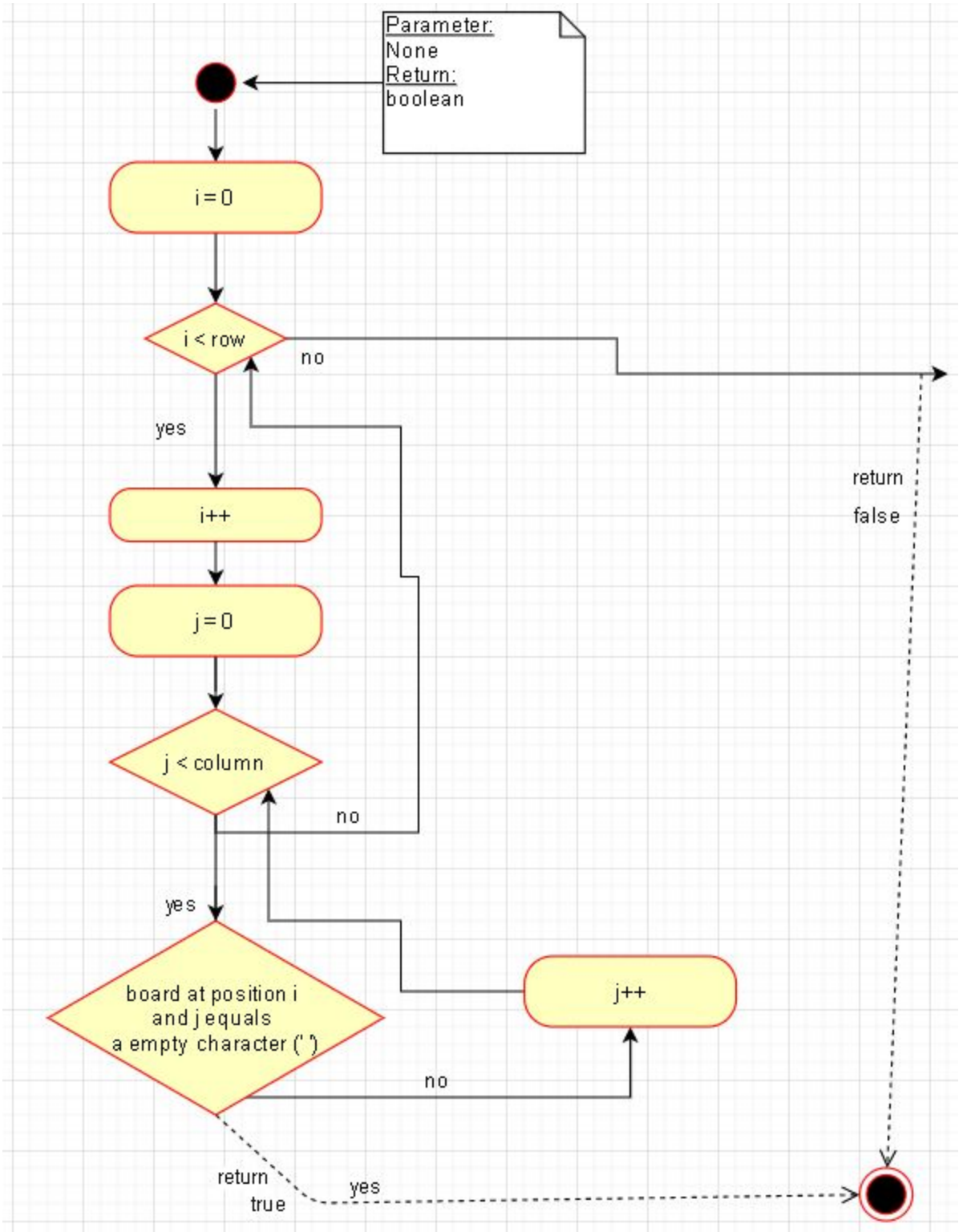
IGameBoard - checkSpace(BoardPosition pos)



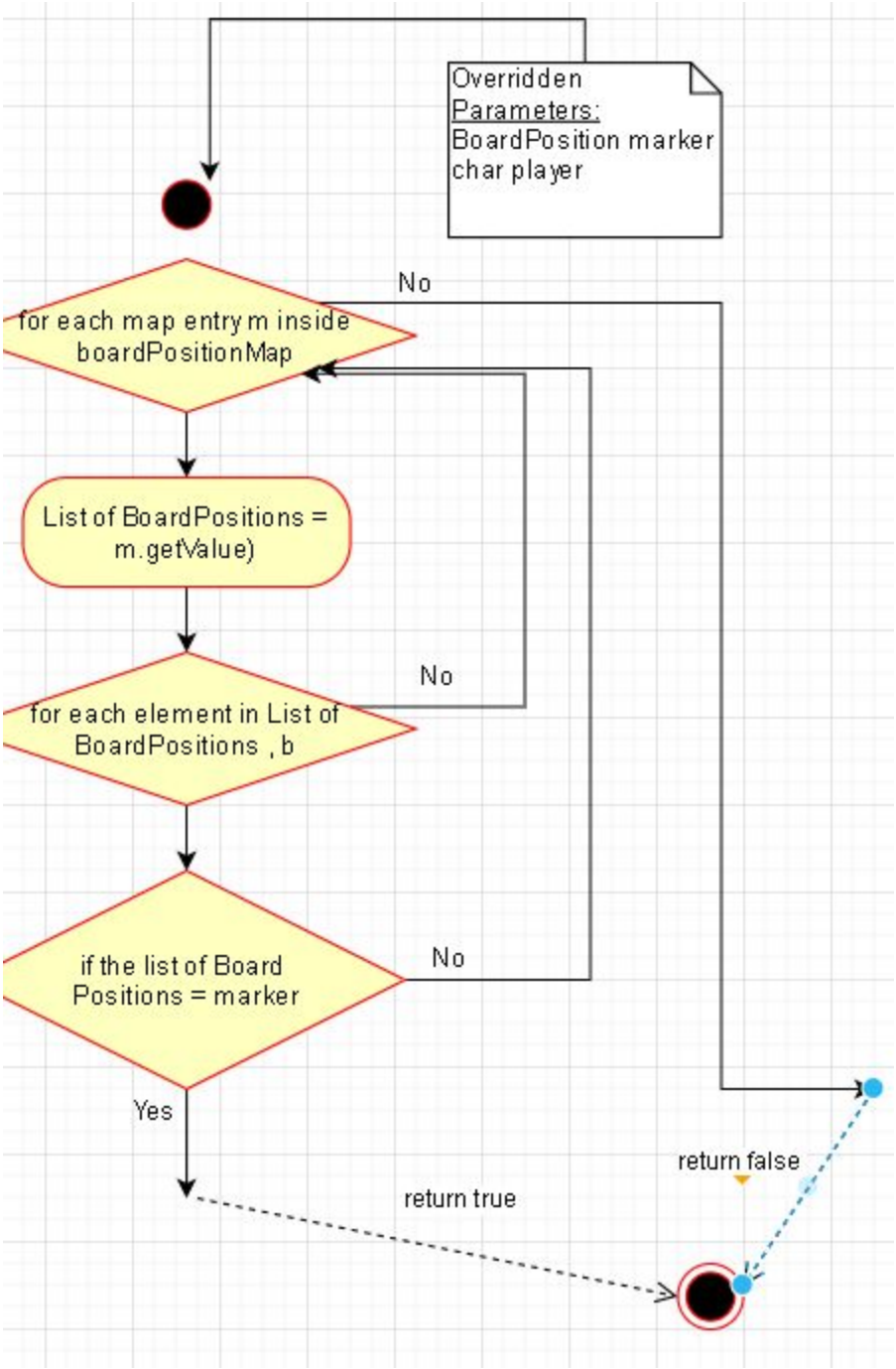
IGameBoard - checkForWinner(BoardPosition lastPos)



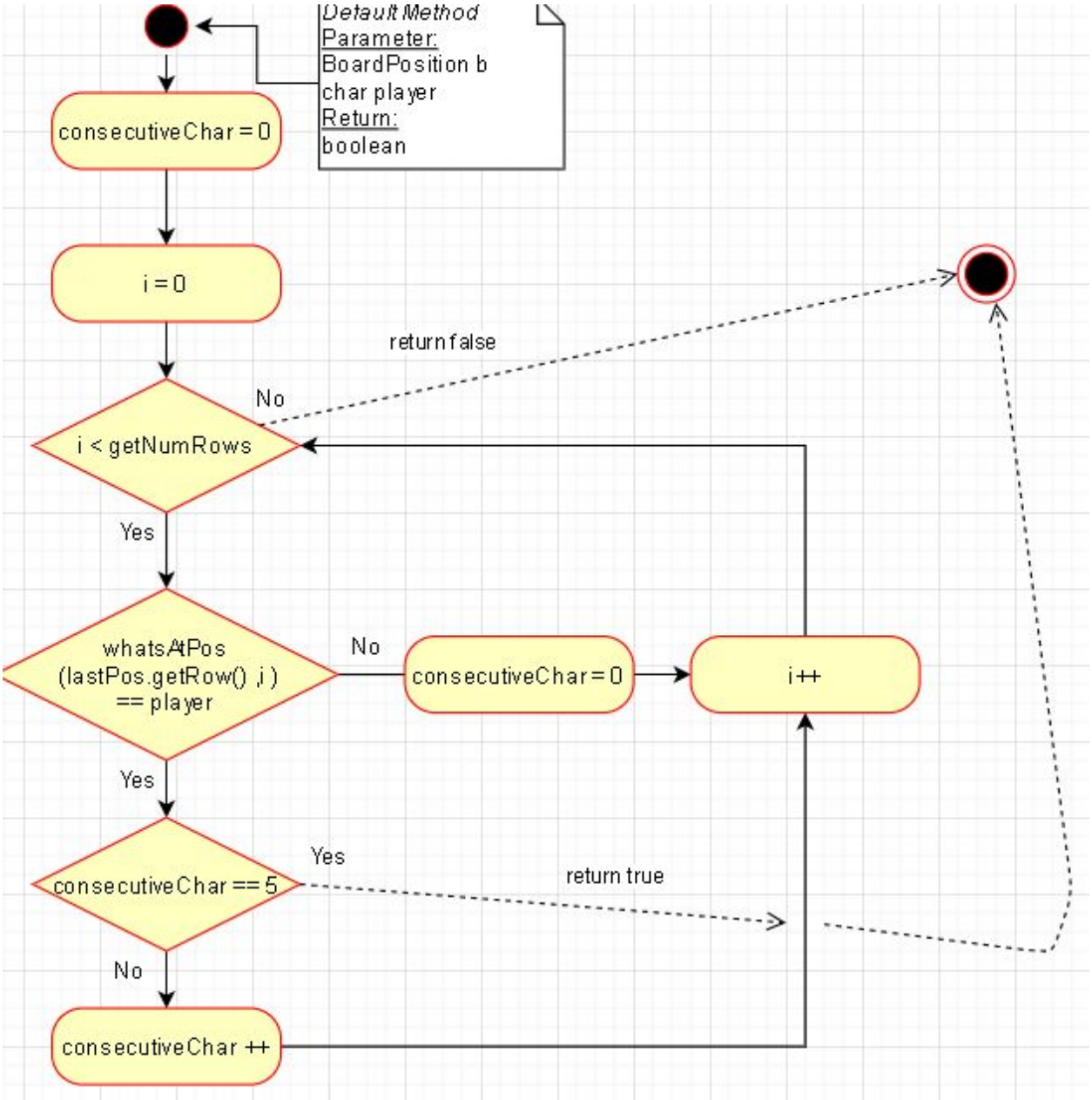
IGameBoard - checkForDraw()



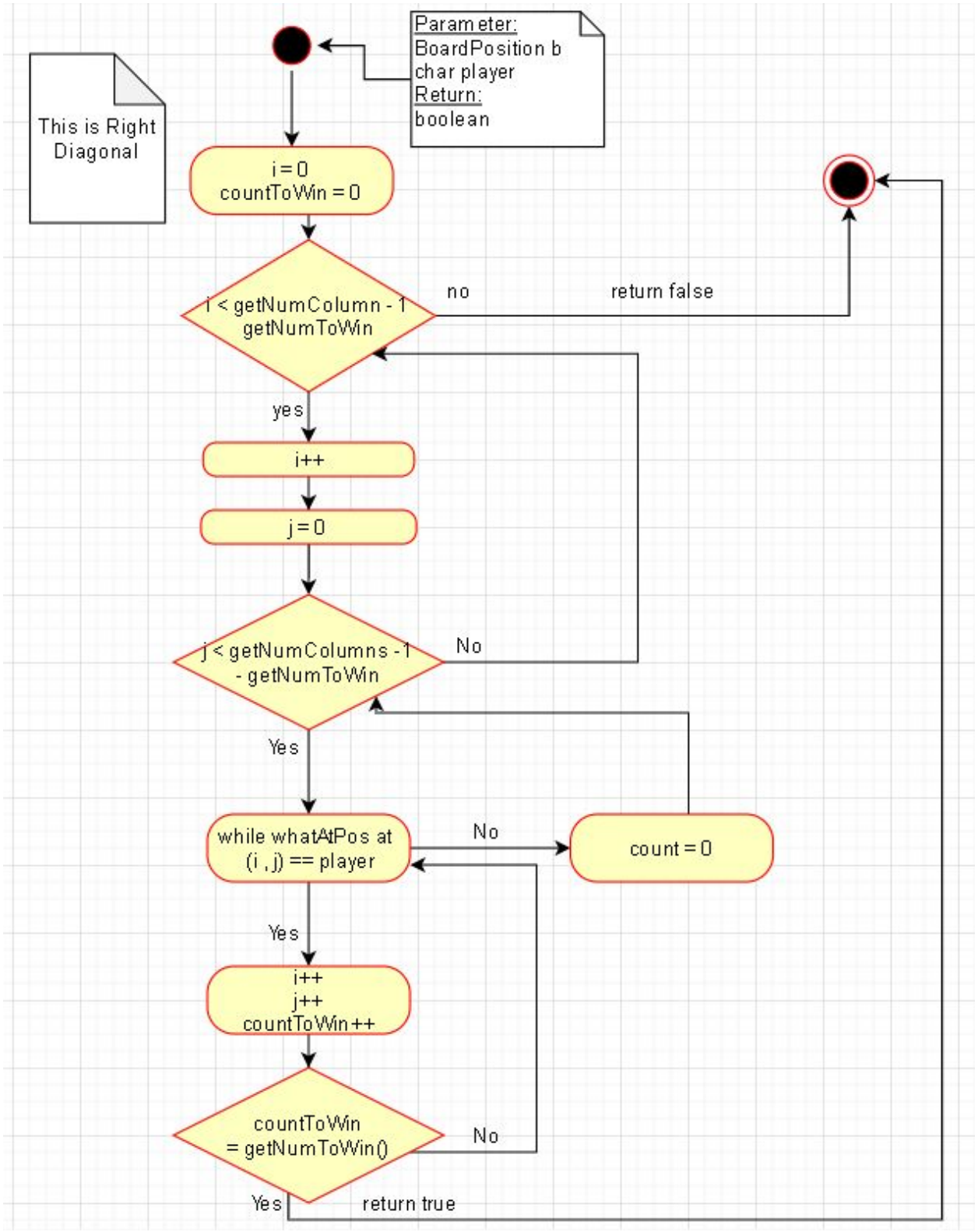
IGameBoard - checkHorizontalWin(BoardPosition lastPos, char player)

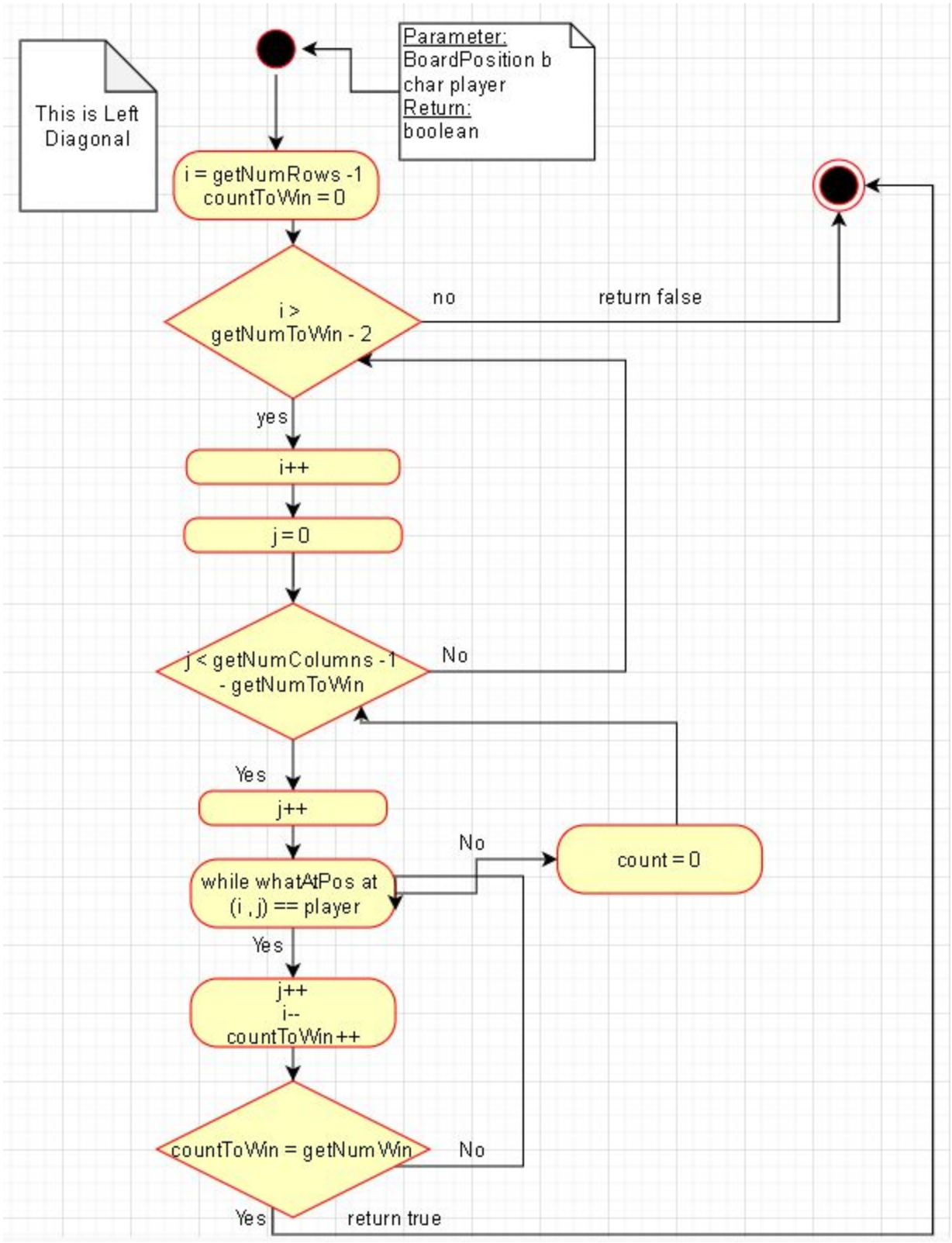


IGameBoard - checkVerticalWin(BoardPosition lastPos , char player)



IGameBoard - checkDiagonalWin(BoardPosition lastPos, char player)




This is Left Diagonal
Parameter:
BoardPosition b
char player
Return:
boolean
i = getNumRows -1
countToWin = 0
i > getNumToWin - 2
no
return false
yes
i++
j = 0
j < getNumColumns -1 - getNumToWin
No
Yes
j++
while whatAtPos at (i , j) == player
No
count = 0
Yes
j++
i--
countToWin ++
countToWin = getNumWin
No
Yes
return true

IGameBoard isPlayerAtPos(BoardPosition pos , char player)