Ashlyn Cooper
CPSC 2150
Project 2
Project Report

Requirements Analysis

Functional Requirements

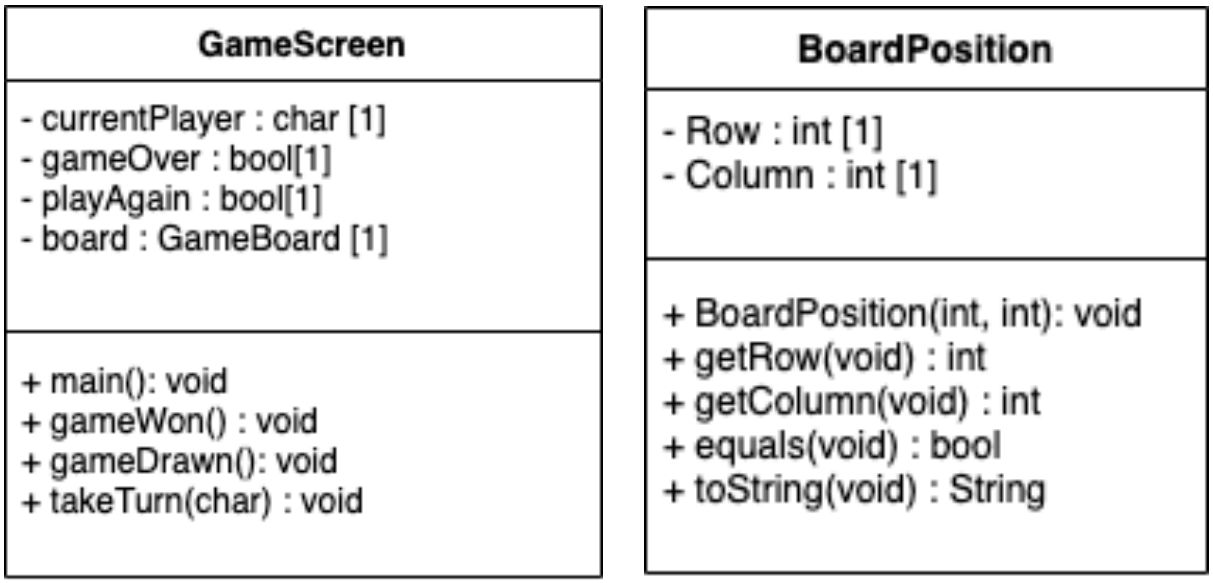
1. As a player, I can input a position so that I can play my token.
2. As a player, I can choose to play again so that I can replay tic tac toe.
3. As a player, I can have my input validated by the program so that I can place my token in a valid position.
4. As a player, I can view output prompting for an input so that I can see when I should make an input.
5. As a player, I can view an outputted message at the end of the game, so that I know how the game ended.
6. As a player, I can win horizontally, as that I can win the game.
7. As a player, I can win vertically, so that I can win the game.
8. As a player, I can win diagonally so that I can win the game.
9. As a player, I can make a move after my opponent (if they have not one), so that I can take my turn.
10. The program can end the game without either player winning, so that the game can end in a tie.

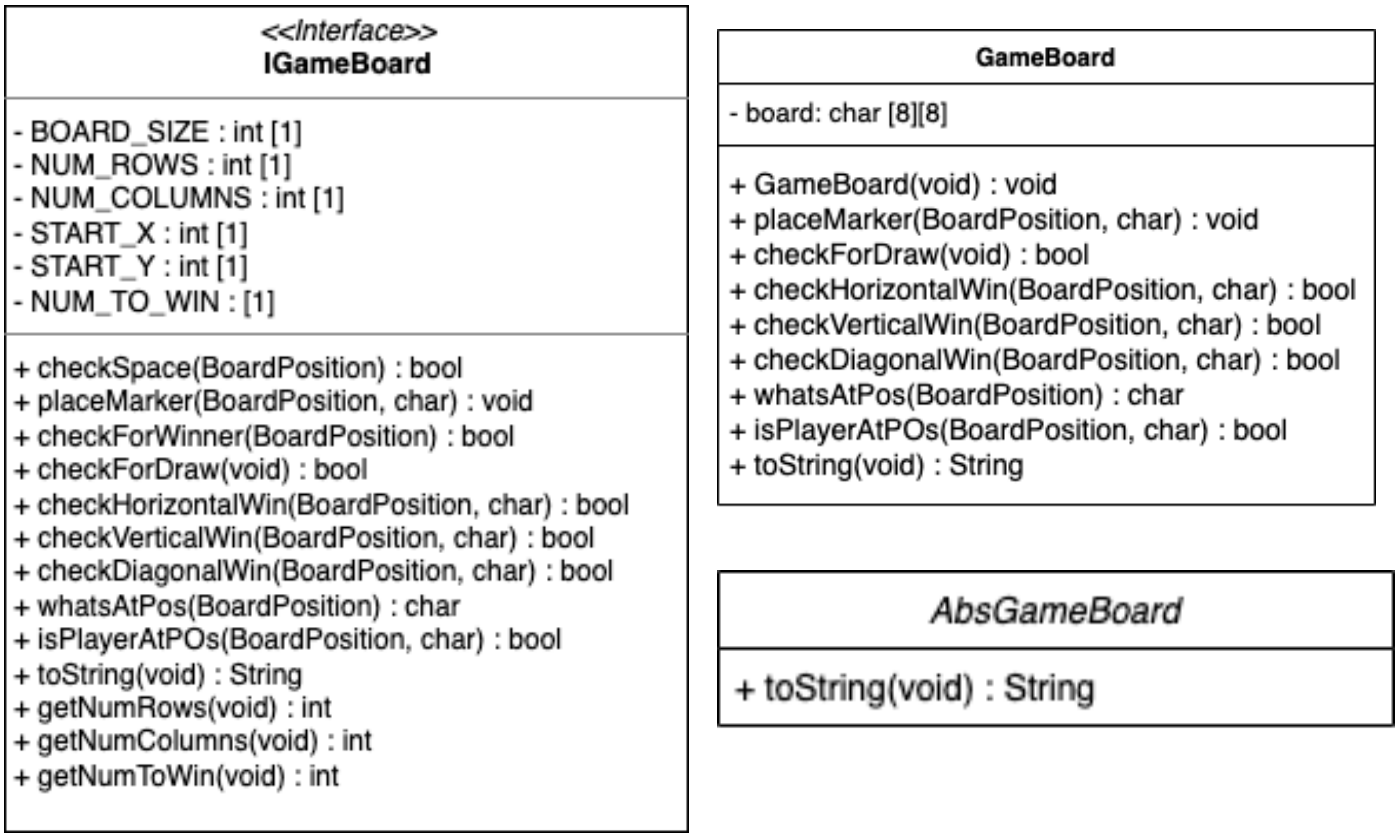
Non-functional Requirements

1. The system must be written in Java.
2. The system must run on a Unix machine.
3. The program can print the game board to the screen so that the player can see the board.
4. The program can alternate between players so that the program can be played by two players.
5. The program can be repeated so that the program has the ability to let the players play again.
6. The game board is of size 8x8.
7. Player X will always go first.
8. Coordinate (0,0) represents the top left corner of the game board.

Design

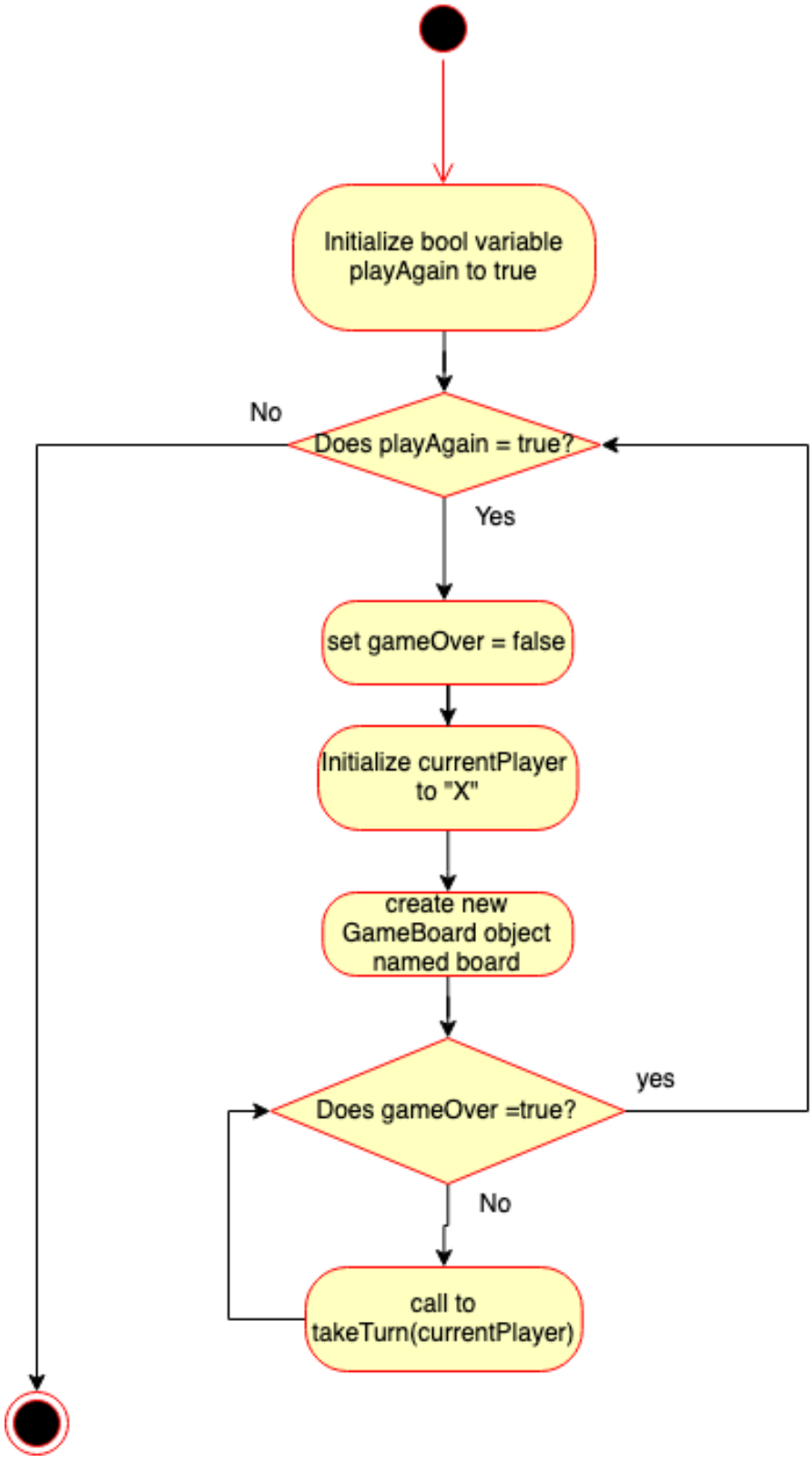
Class Diagrams



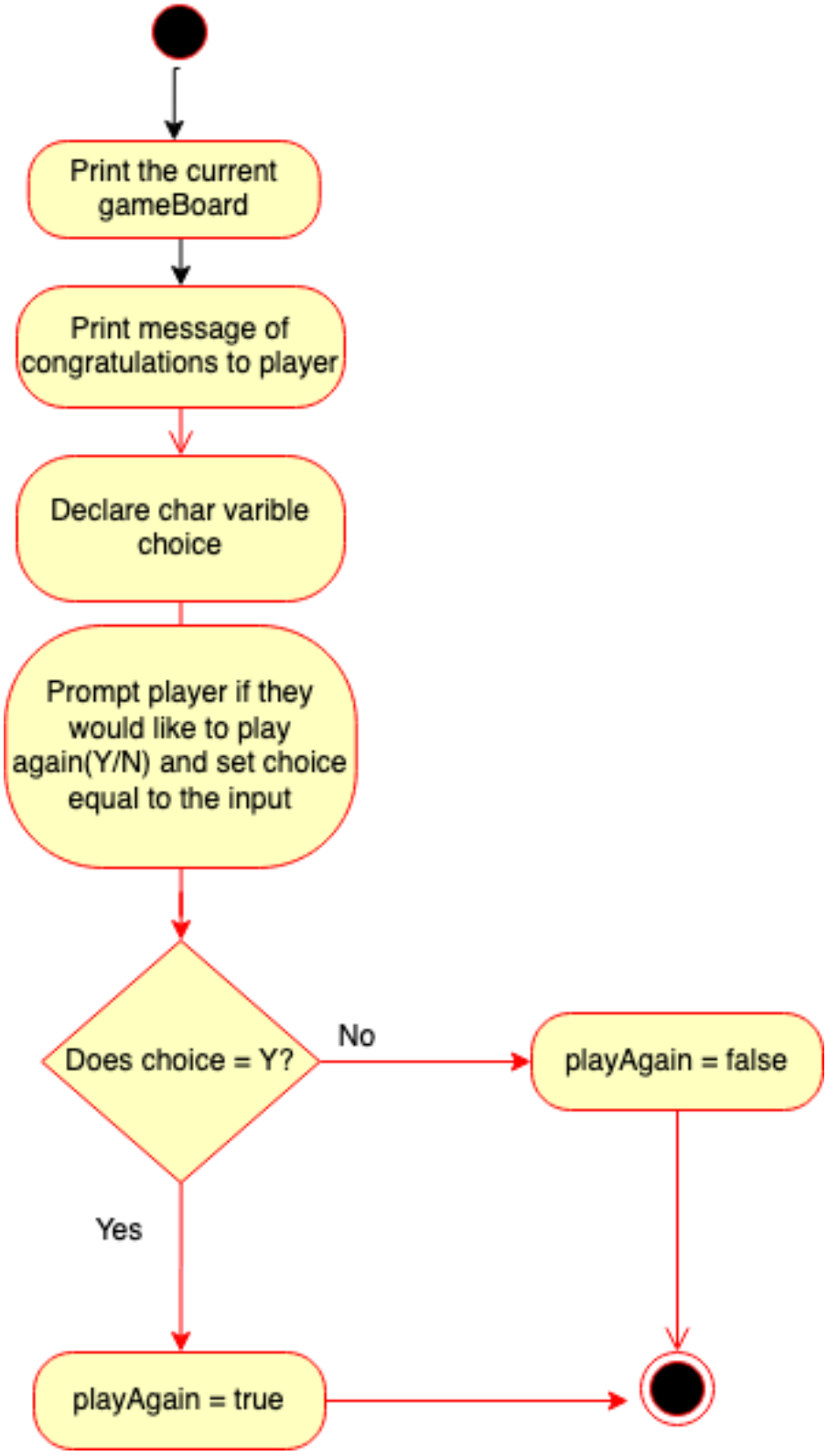


GameScreeN Activity Diagrams

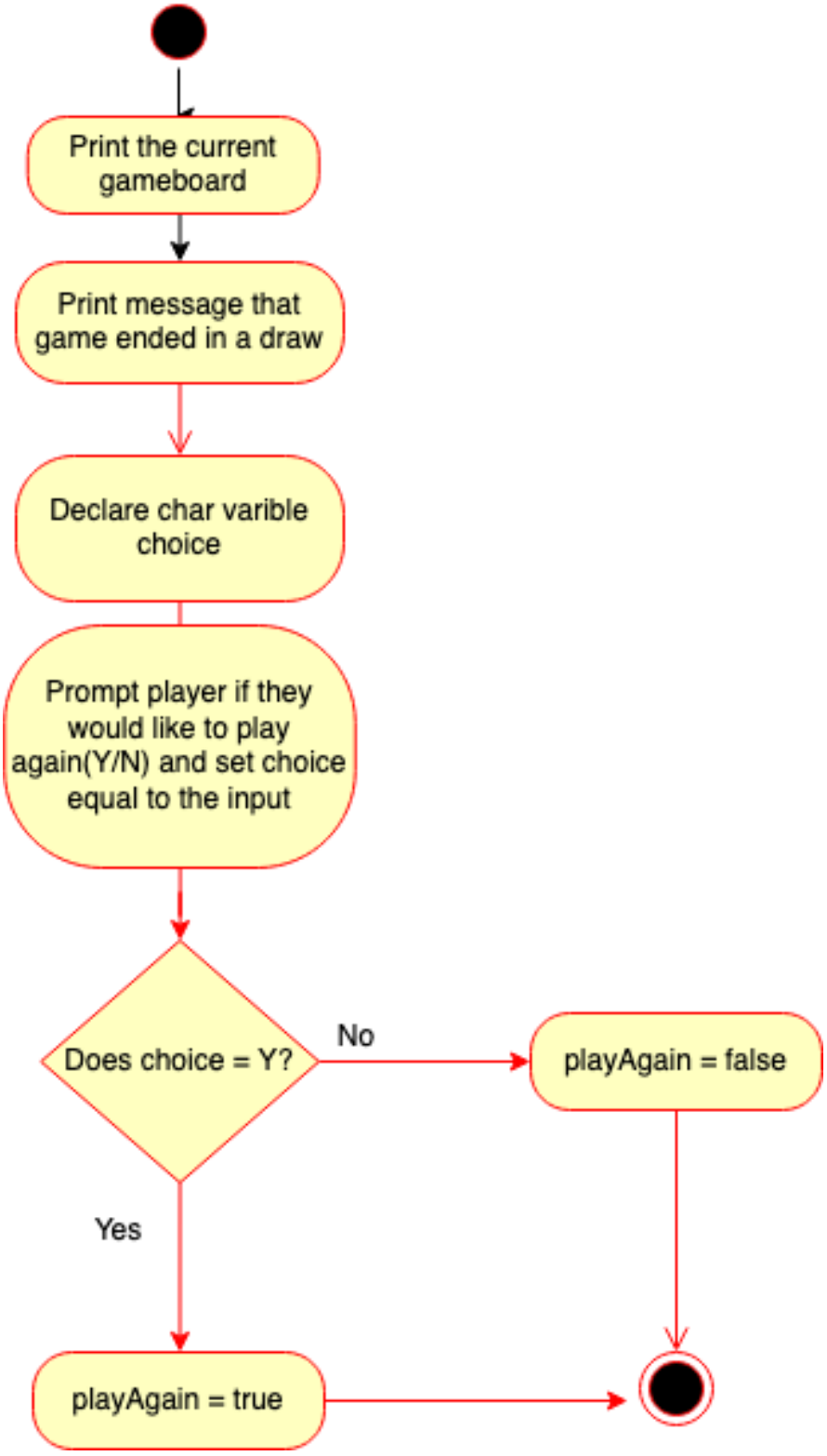
main()



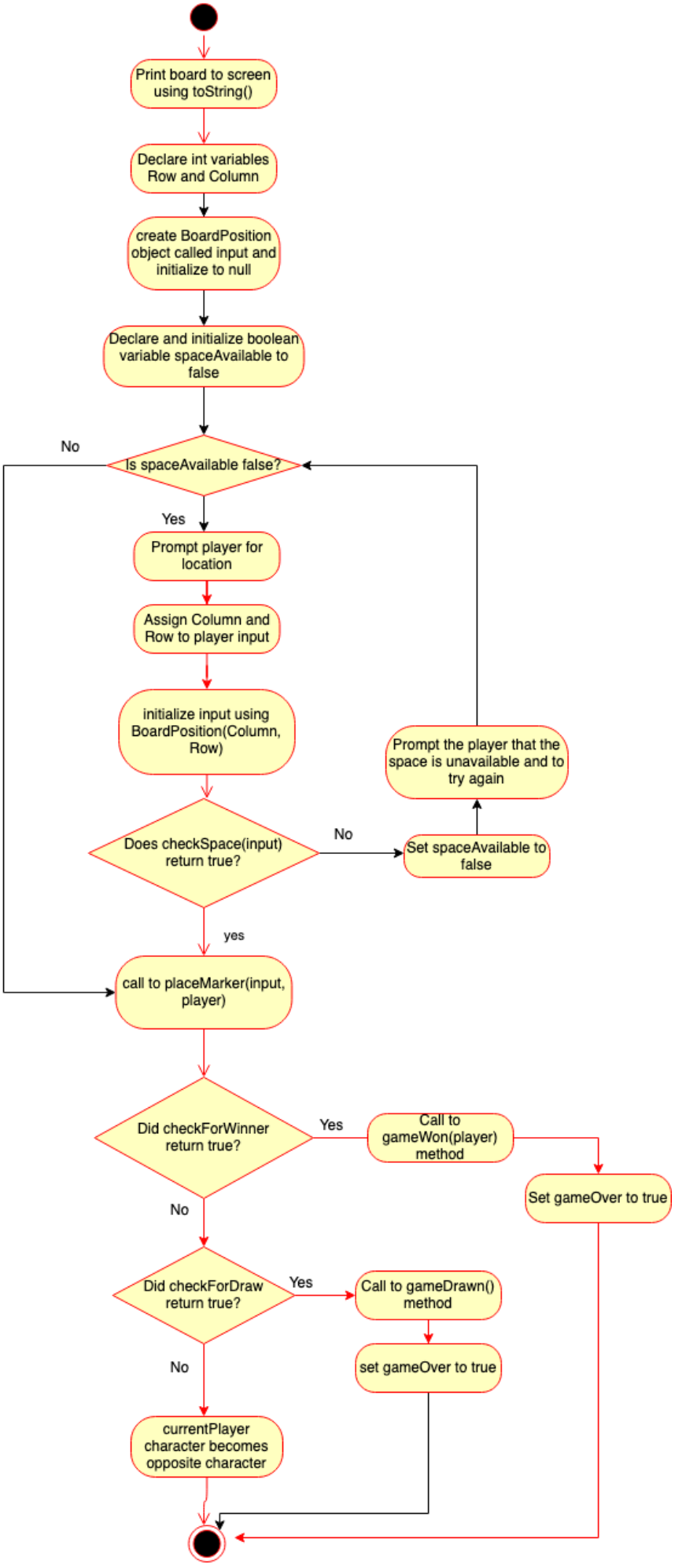
gameWon()


Print the current gameBoard
Print message of congratulations to player
Declare char varible choice
Prompt player if they would like to play again(Y/N) and set choice equal to the input
Does choice = Y?
No
playAgain = false
Yes
playAgain = true

gameDrawn()

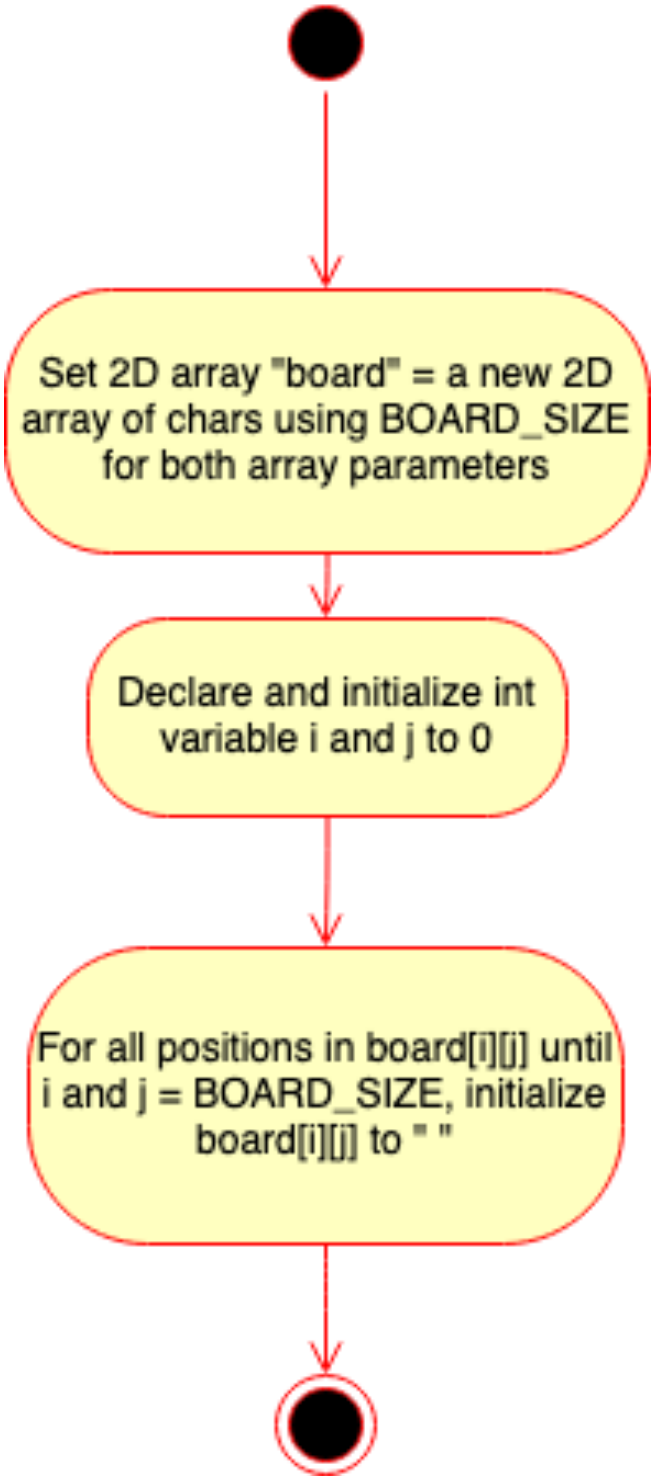

Print the current gameboard
Print message that game ended in a draw
Declare char varible choice
Prompt player if they would like to play again(Y/N) and set choice equal to the input
Does choice = Y?
No
playAgain = false
Yes
playAgain = true

takeTurn()

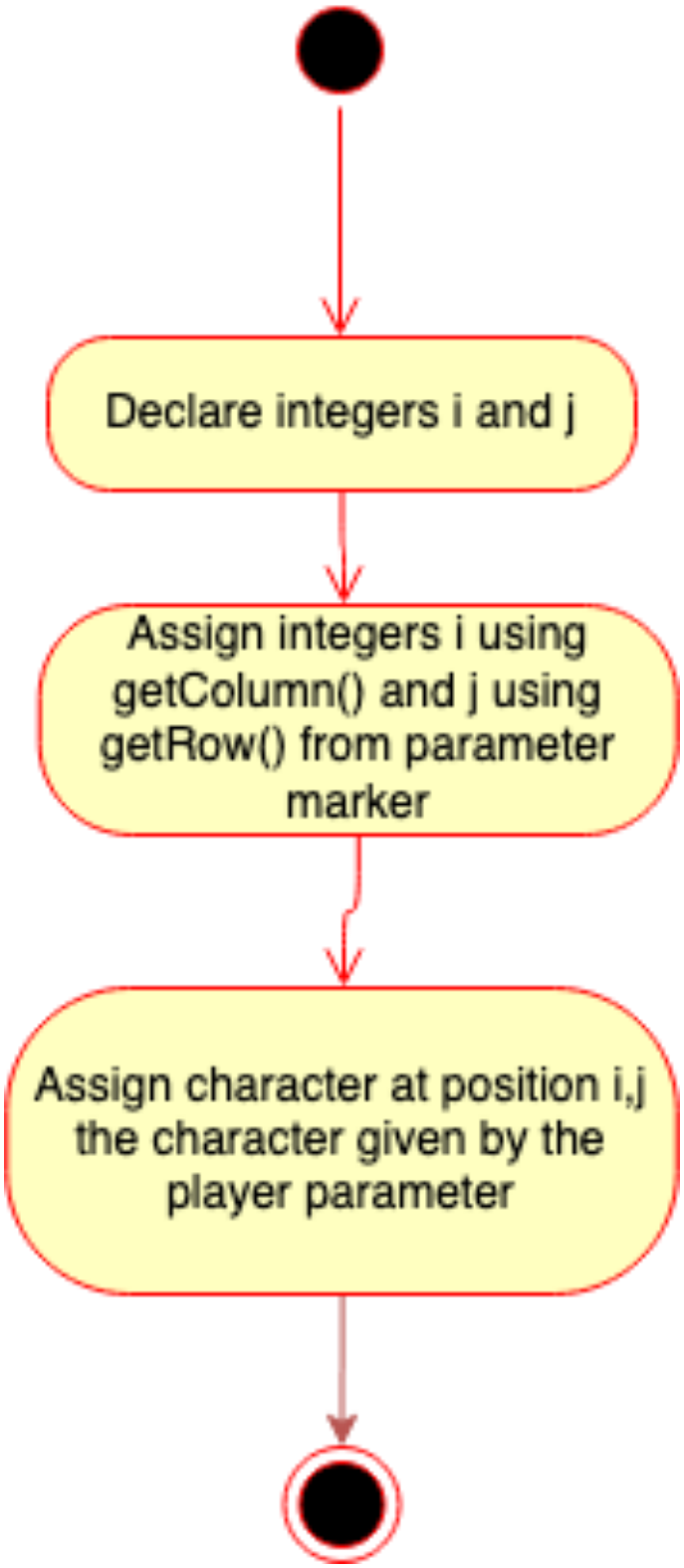

Print board to screen using toString()
Declare int variables Row and Column
create BoardPosition object called input and initialize to null
Declare and initialize boolean variable spaceAvailable to false
No
Is spaceAvailable false?
Yes
Prompt player for location
Assign Column and Row to player input
initialize input using BoardPosition(Column, Row)
Prompt the player that the space is unavailable and to try again
Does checkSpace(input) return true?
No
Set spaceAvailable to false
yes
call to placeMarker(input, player)
Did checkForWinner return true?
Yes
Call to gameWon(player) method
Set gameOver to true
No
Did checkForDraw return true?
Yes
Call to gameDrawn() method
set gameOver to true
No
currentPlayer character becomes opposite character

GameBoard and IGameBoard Activity Diagrams

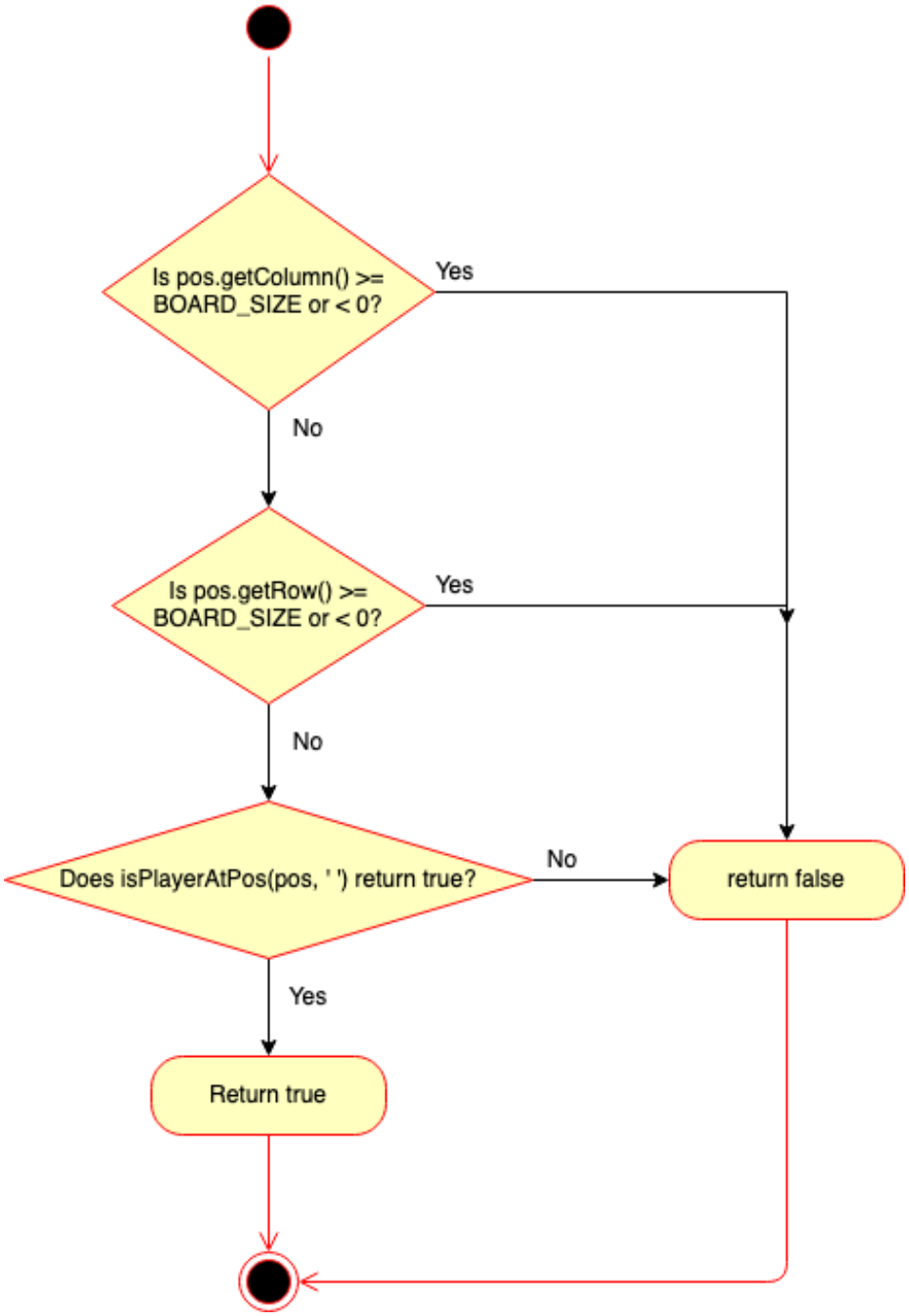
GameBoard Constructor



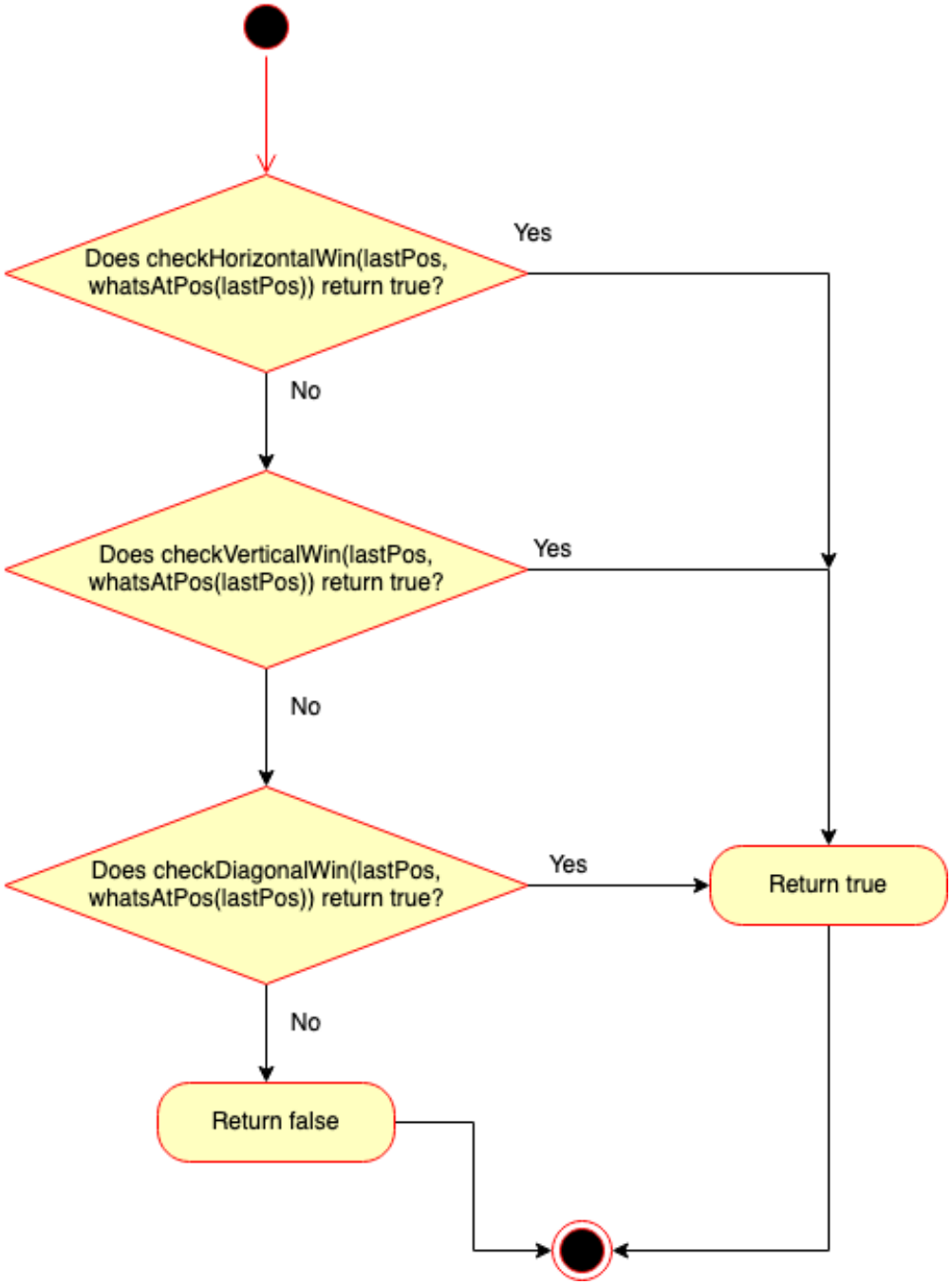
placeMarker(BoardPosition marker, char player)



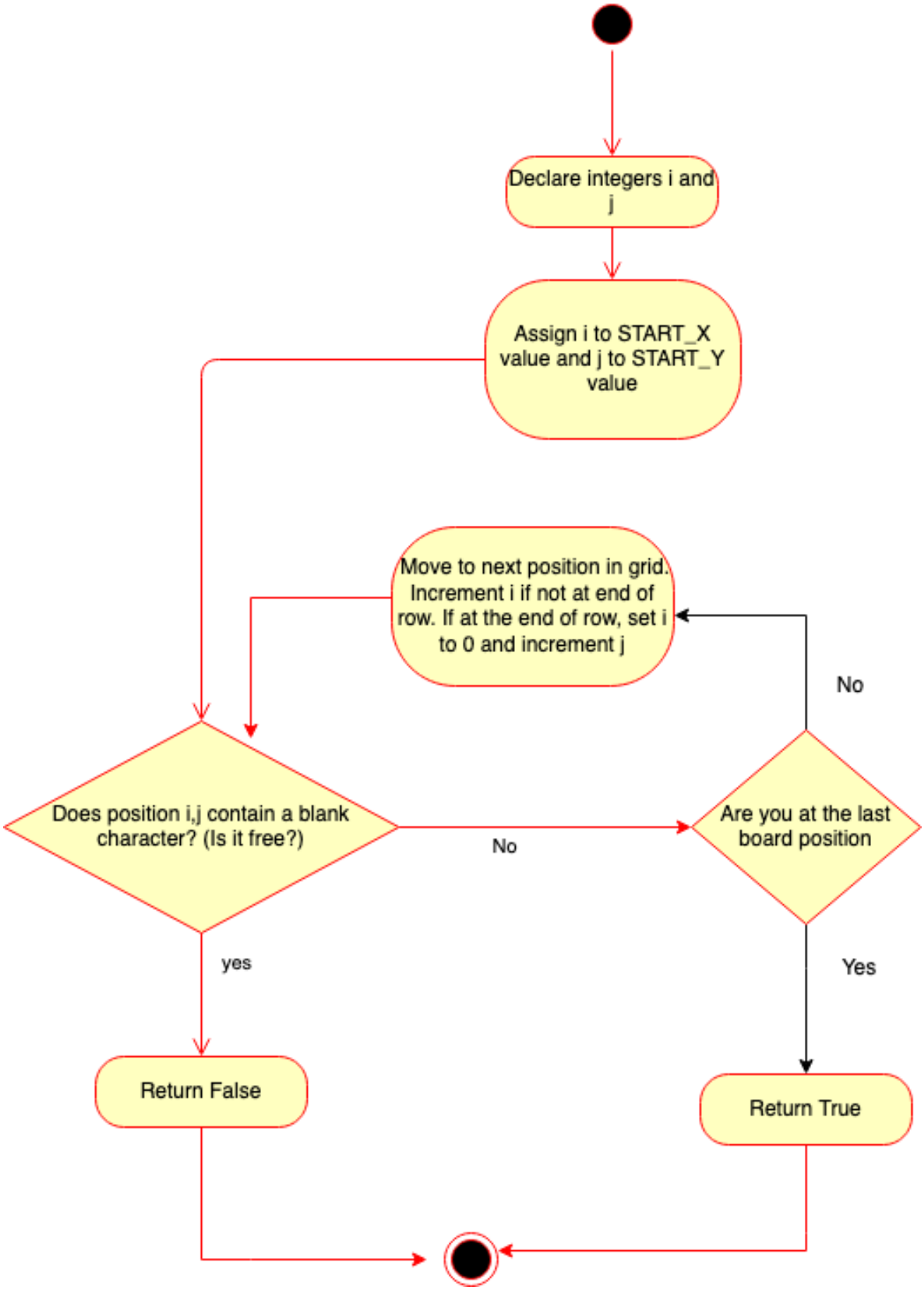
Default checkSpace(BoardPosition pos)



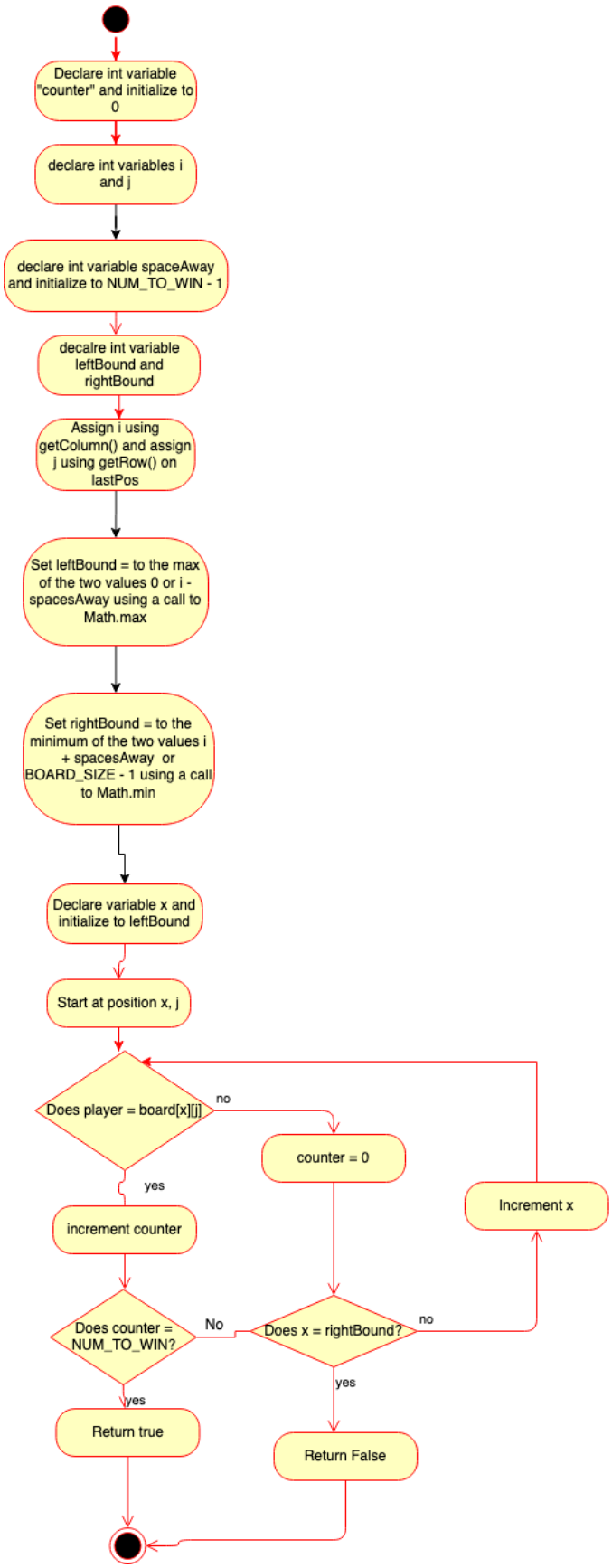
Default checkForWInner(BoardPosition lastPos)



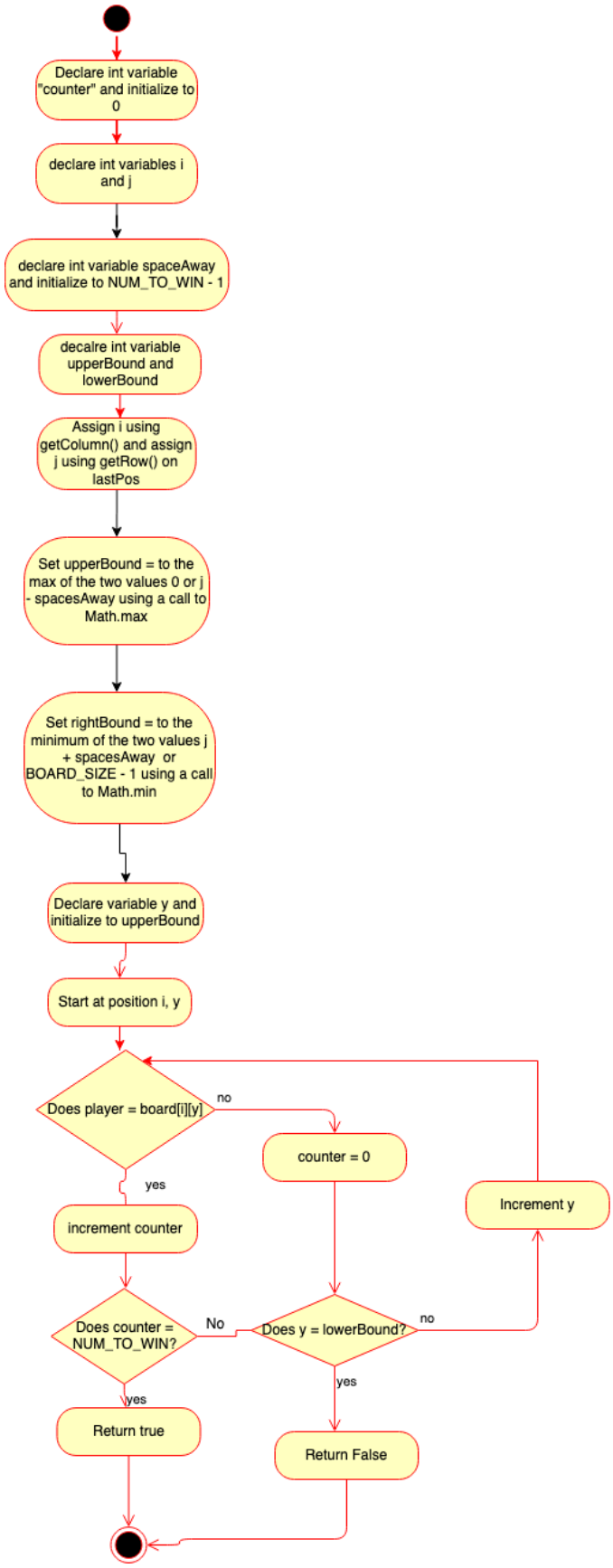
checkForDraw()



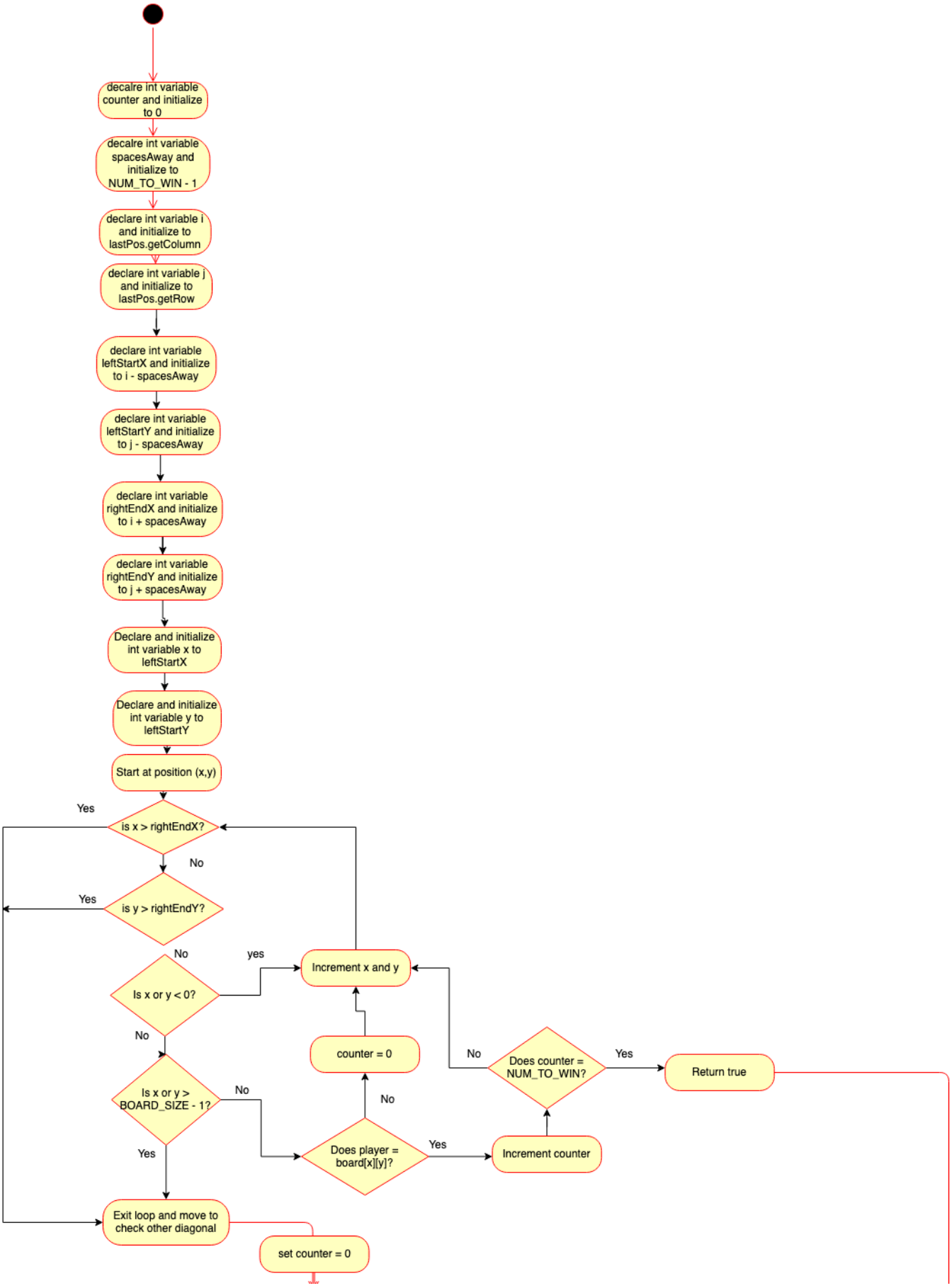
checkHorizontalWin(BoardPosition lastPos, char player)

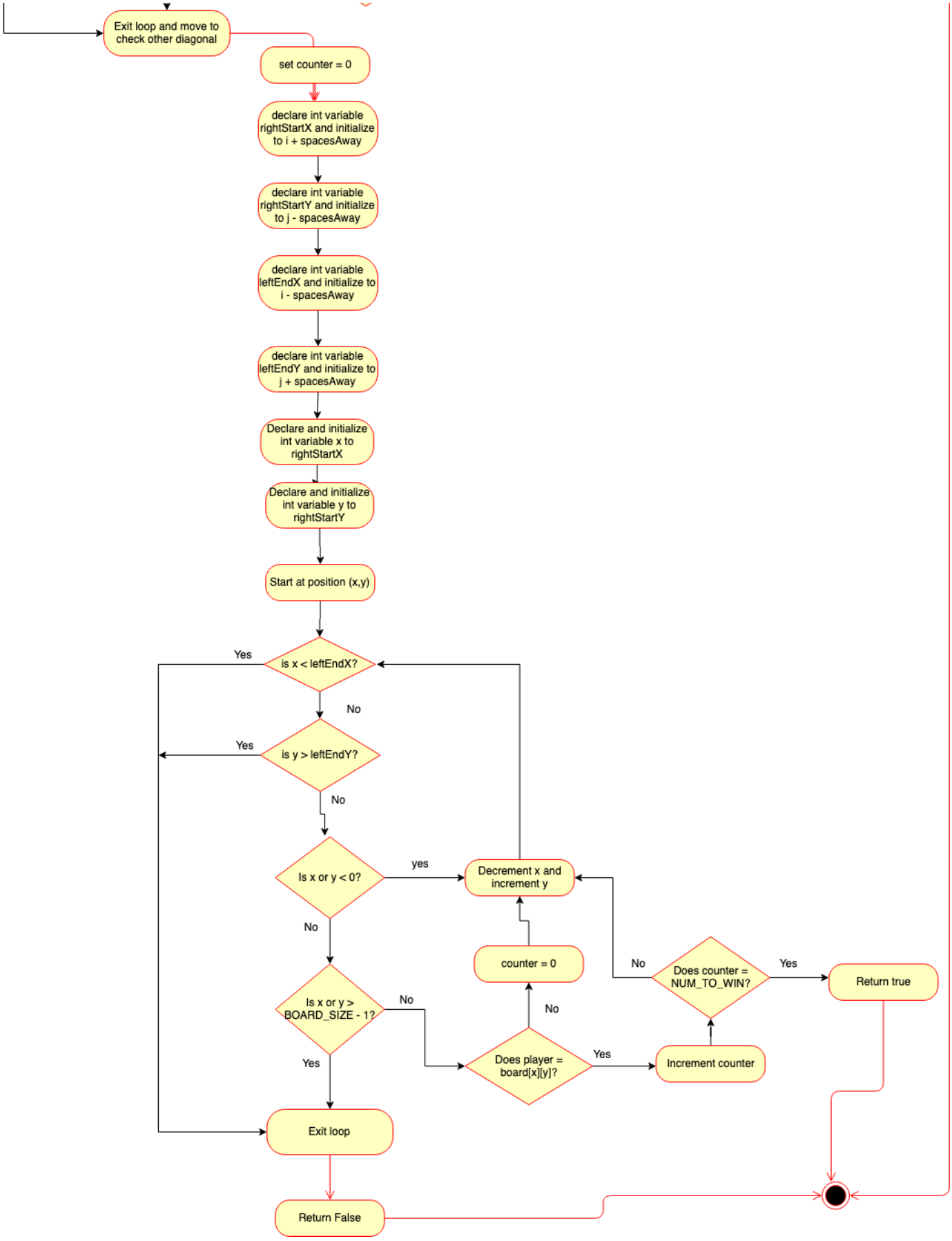


checkVerticalWin(BoardPosition lastPos, char player)

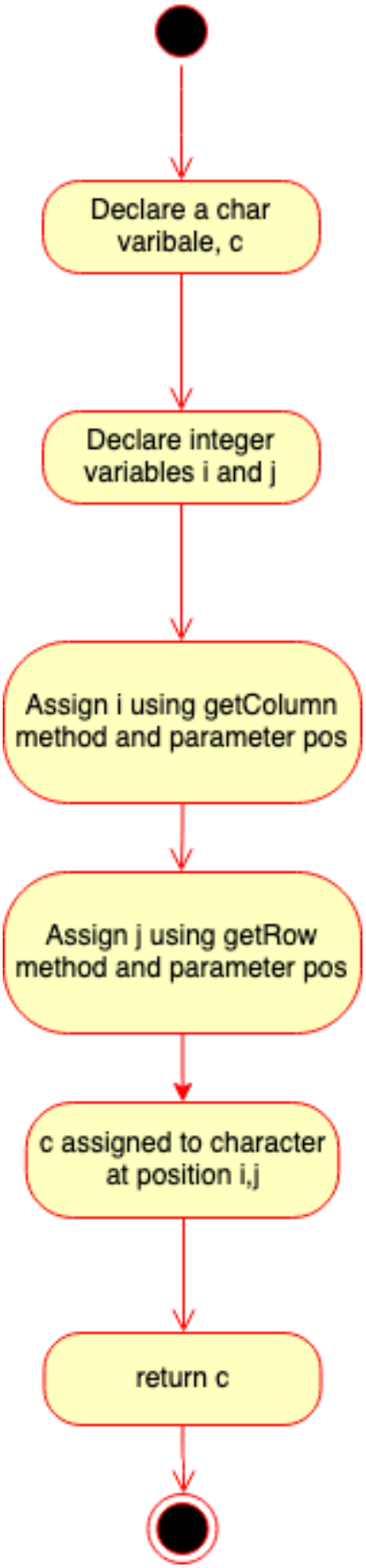


checkDiagonalWin(BoardPosition lastPost, char player)

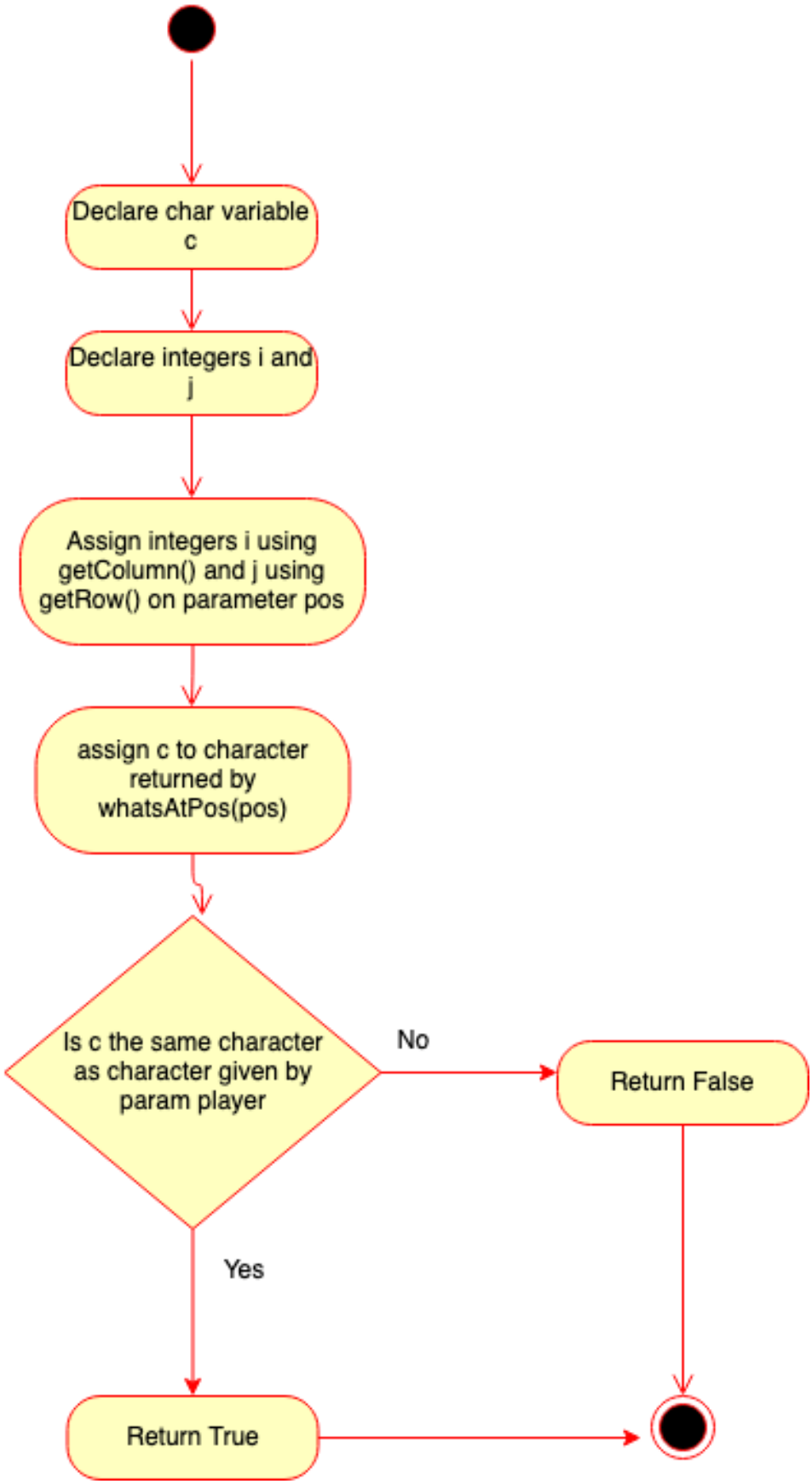



Exit loop and move to check other diagonal
set counter = 0
declare int variable rightStartX and initialize to i + spacesAway
declare int variable rightStartY and initialize to j - spacesAway
declare int variable leftEndX and initialize to i - spacesAway
declare int variable leftEndY and initialize to j + spacesAway
Declare and initialize int variable x to rightStartX
Declare and initialize int variable y to rightStartY
Start at position (x,y)
is x < leftEndX?
Yes
No
is y > leftEndY?
Yes
No
Is x or y < 0?
yes
Decrement x and increment y
No
Is x or y > BOARD_SIZE - 1?
No
Yes
Exit loop
Return False
counter = 0
Does player = board[x][y]?
Yes
No
Increment counter
Does counter = NUM_TO_WIN?
No
Yes
Return true

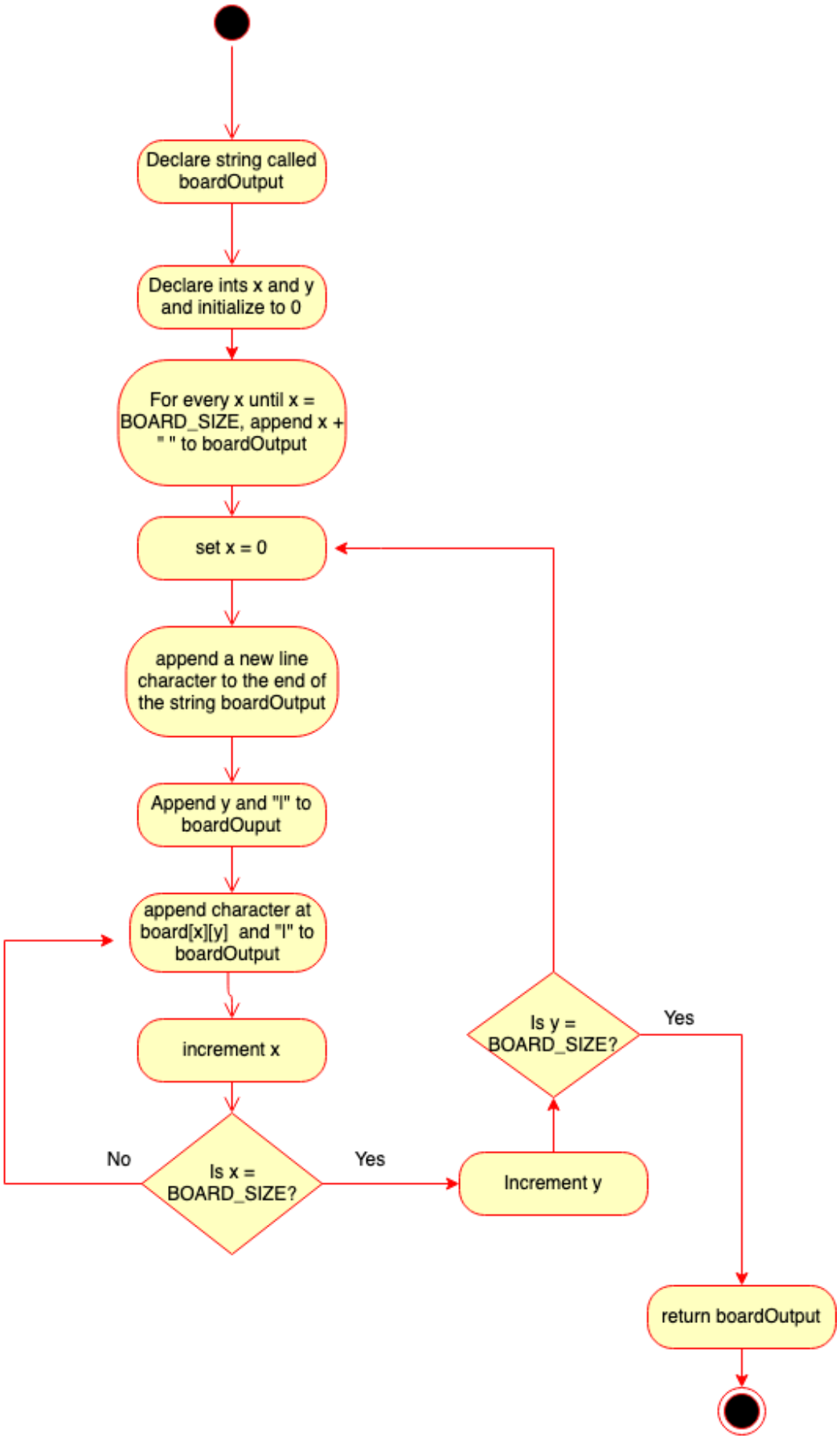
whatsAtPos(BoardPosition pos)



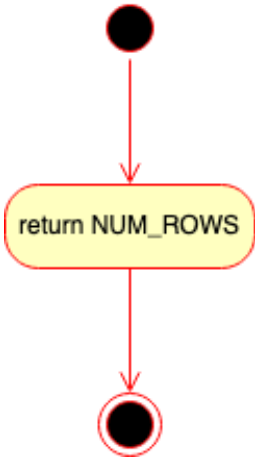
isPlayerAtPos(BoardPosition pos, char player)



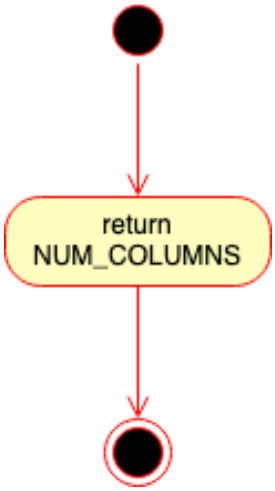
toString()


Declare string called boardOutput
Declare ints x and y and initialize to 0
For every x until x = BOARD_SIZE, append x + " " to boardOutput
set x = 0
append a new line character to the end of the string boardOutput
Append y and "|" to boardOuput
append character at board[x][y] and "|" to boardOutput
increment x
Is x = BOARD_SIZE?
No
Yes
Increment y
Is y = BOARD_SIZE?
Yes
return boardOutput

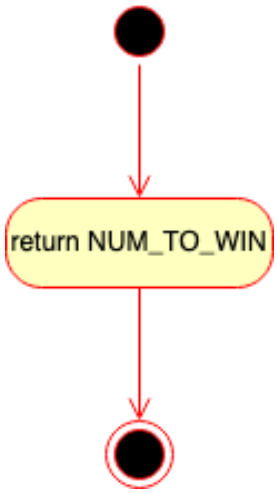
getNumRows()



getNumColumns()

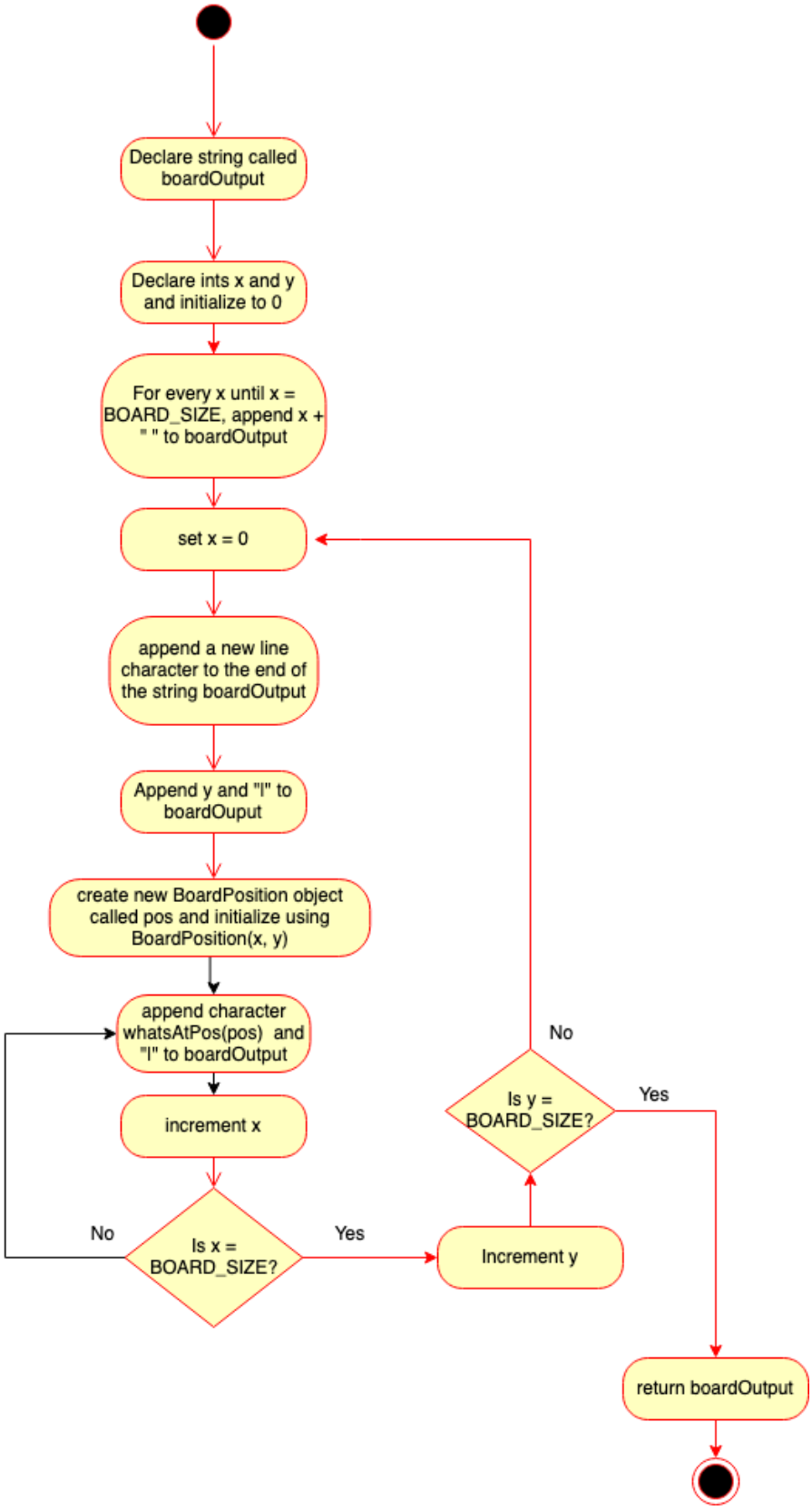


getNumToWin()



AbsGameBoard

toString()



Deployment

Running the extended Tic-Tac-Toe is based off the makefile. To use the makefile navigate to the project directory in the terminal window and type the following commands:

- To compile the code type `make`
- To run the program and play the game type `make run`
- To remove the compiled (.class) files type `make clean`