CPSC 2150 Project Report

Joel Miller

Requirements Analysis

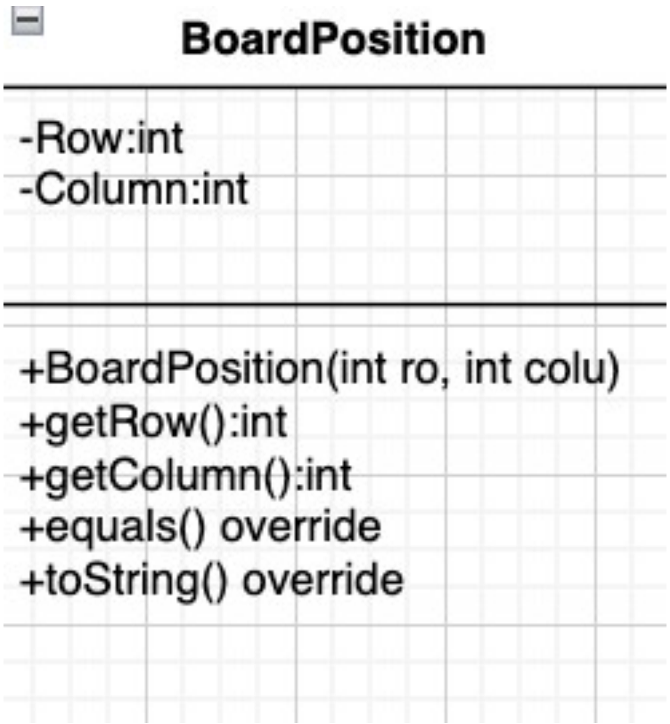
Functional Requirements:

1. Enumerated list of functional requirements
2. **PLAYER**

    ○ As a game player, I want to be able to play against an opponent {O} and I will be {X} ○ As a game player, I want to be able to win if I have 5 {O} in a row[Diagnol,vertical,horizontal]
    ○ As a game player, I want to be able to have the choice where to play my {O or X} ○ As a game player, I want the game board to be a 9X7 ○ As a game player, I want a column to be full once it has 9 tokens in it ○ As a game player, I want the game to inform the player who won or tied ○ As a game player, I want the game to ask if we want to play again at the end of the game

Non-Functional Requirements

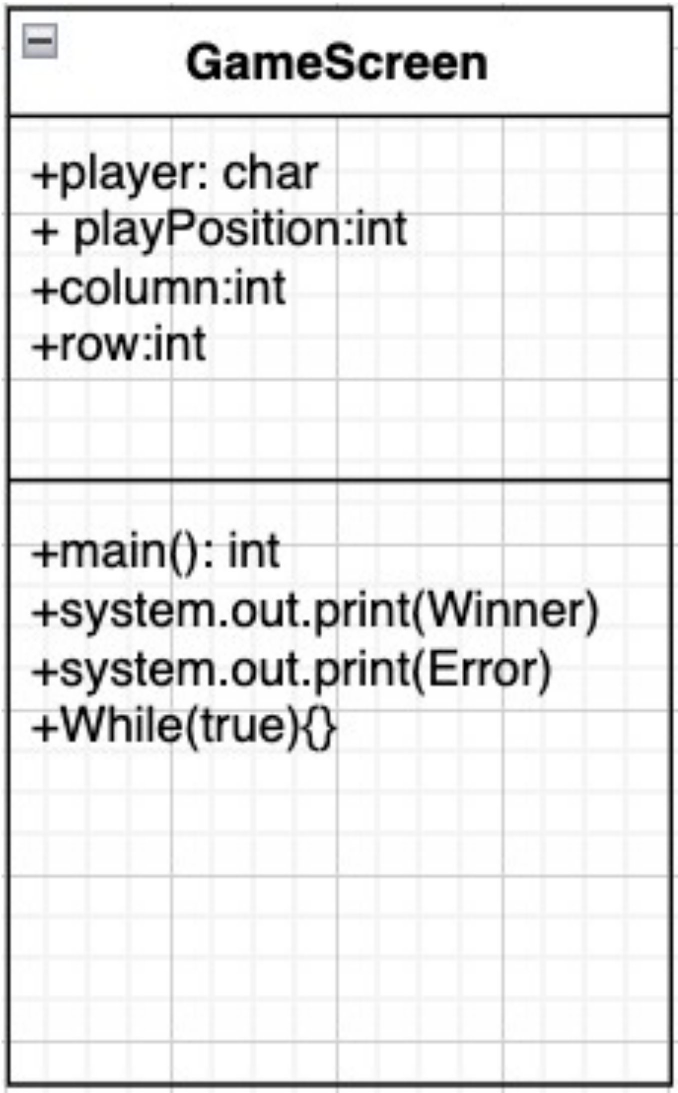
1. Enumerated list of Non-functional requirements ○ The game Code must be written in Java ○ The game must be compatible with a command line interface ○ The game must have three classes GameScreen.java/BoardPosition.java/GameBoard.java
    - **Gamescreen.java**
        - Must contain the main method
        - This class will interface with the player{It will alternate the game between players, say whose turn it is, get the column they would like, and place their marker}
    - **BoardPosition.java**
        - Will keep track of the row position and column position
        - Only 1 constructor which takes in an into for row and an int for column
        - Must also have some getter functions getcolumn and getrow
    - **GameBoard.java**
        - All attributes must be private
        - Functions must be made public
        - Each position will have a blank character



BoardPosition

-Row:int
-Column:int

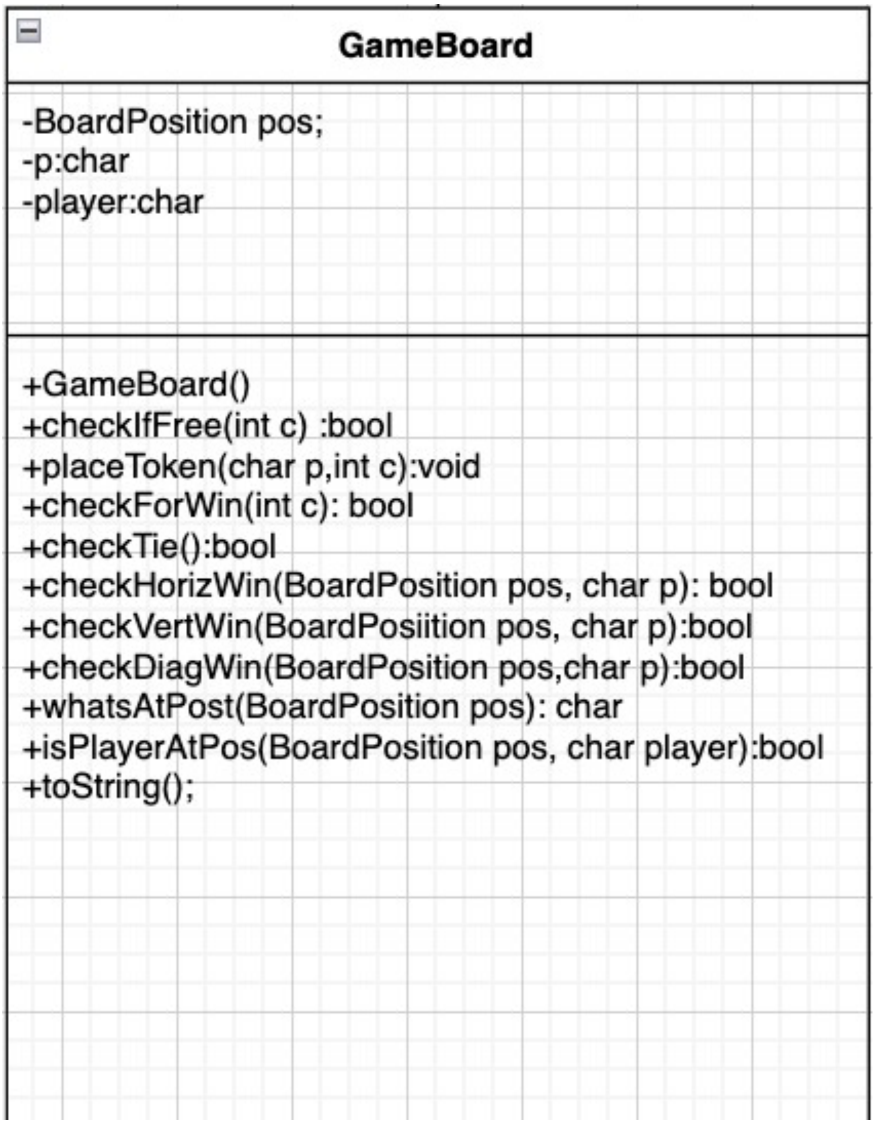
+BoardPosition(int ro, int colu)
+getRow():int
+getColumn():int
+equals() override
+toString() override



GameScreen

+player: char
+ playPosition:int
+column:int
+row:int

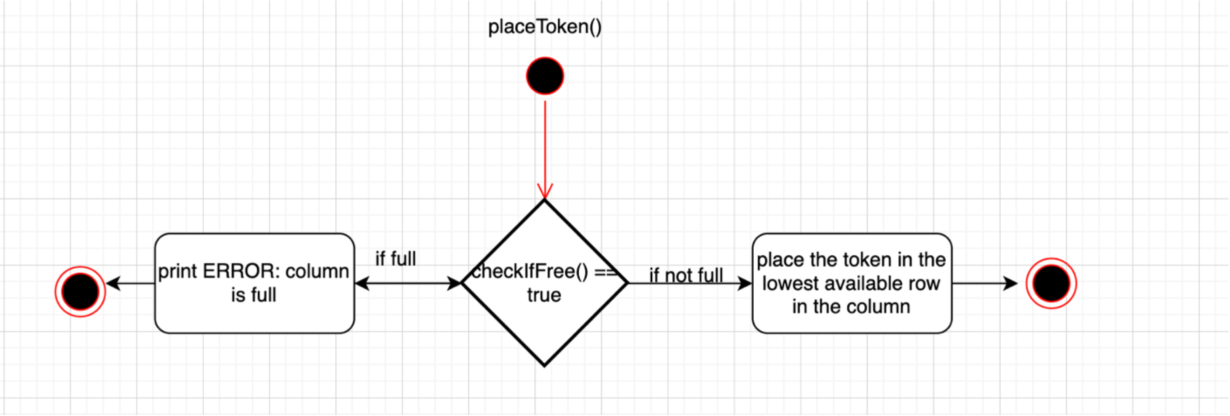
+main(): int
+system.out.print(Winner)
+system.out.print(Error)
+While(true){}

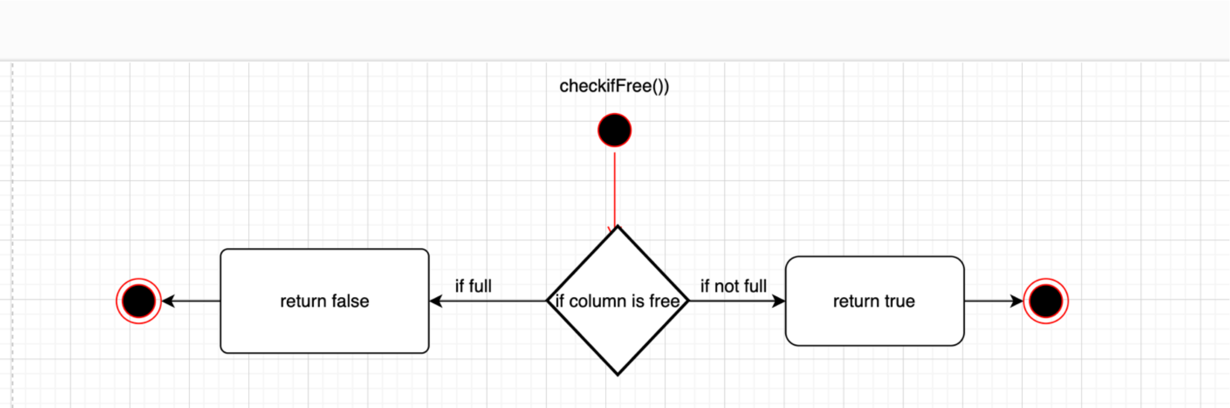


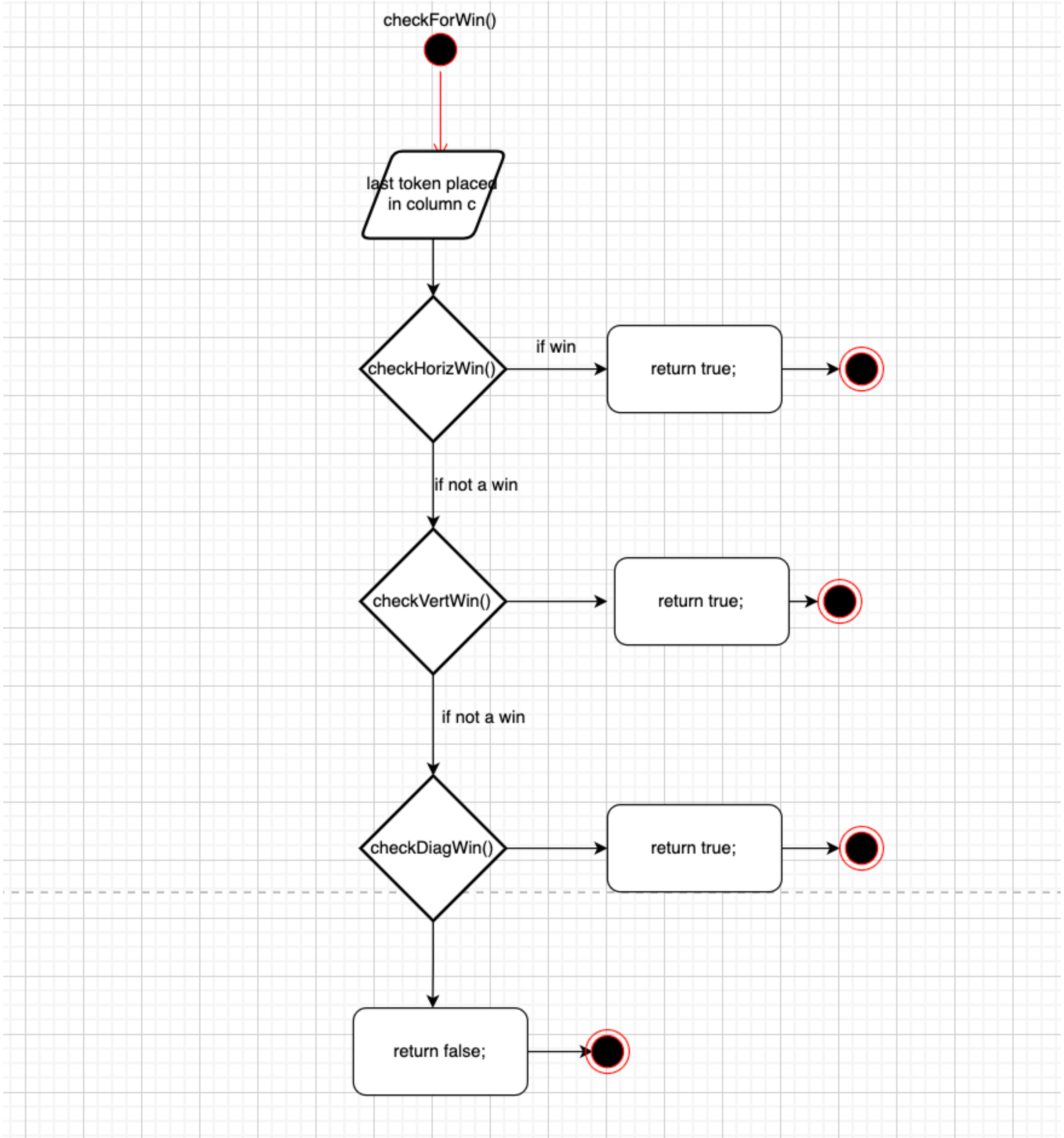
GameBoard

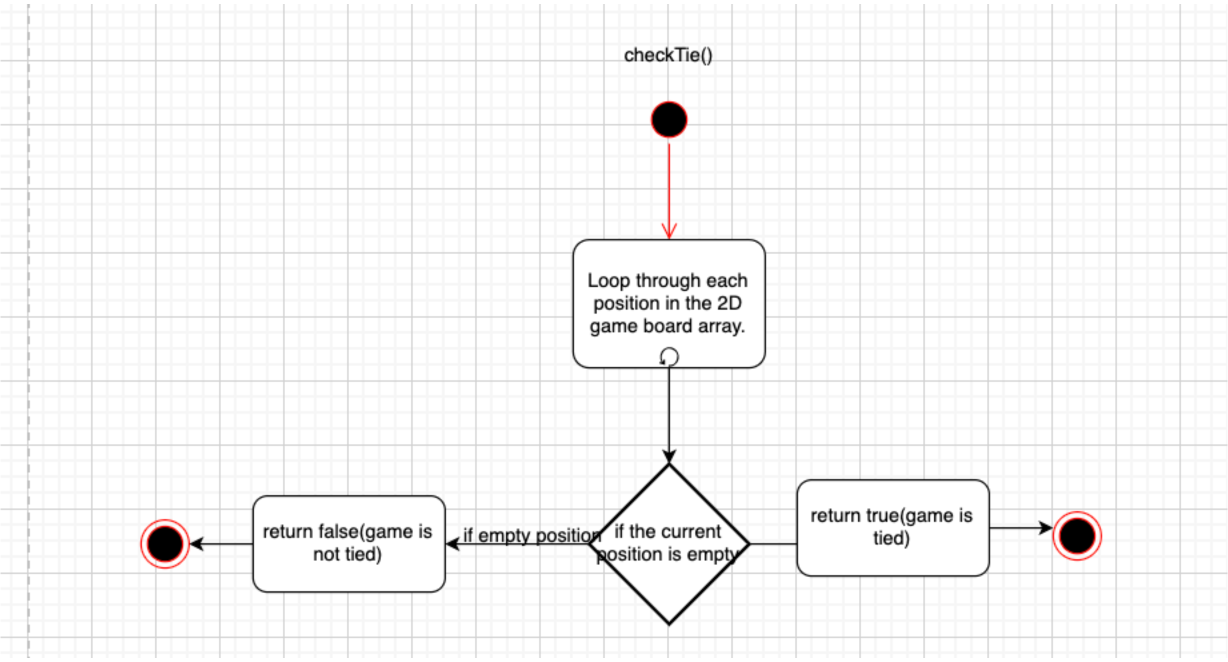
-BoardPosition pos;
-p:char
-player:char

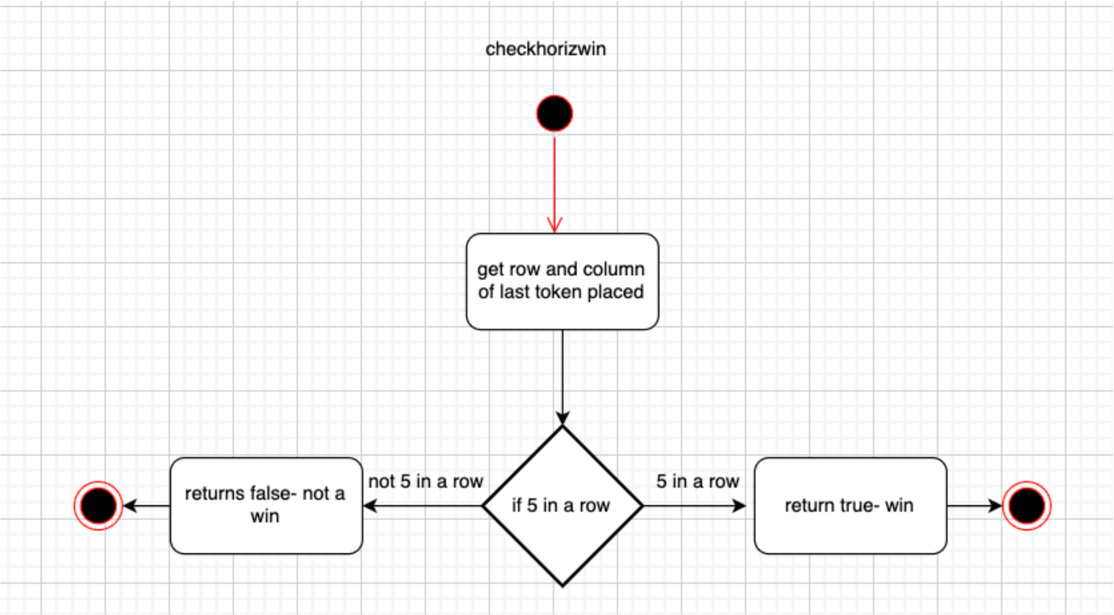
+GameBoard()
+checkIfFree(int c) :bool
+placeToken(char p,int c):void
+checkForWin(int c): bool
+checkTie():bool
+checkHorizWin(BoardPosition pos, char p): bool
+checkVertWin(BoardPosiition pos, char p):bool
+checkDiagWin(BoardPosition pos,char p):bool
+whatsAtPost(BoardPosition pos): char
+isPlayerAtPos(BoardPosition pos, char player):bool
+toString();

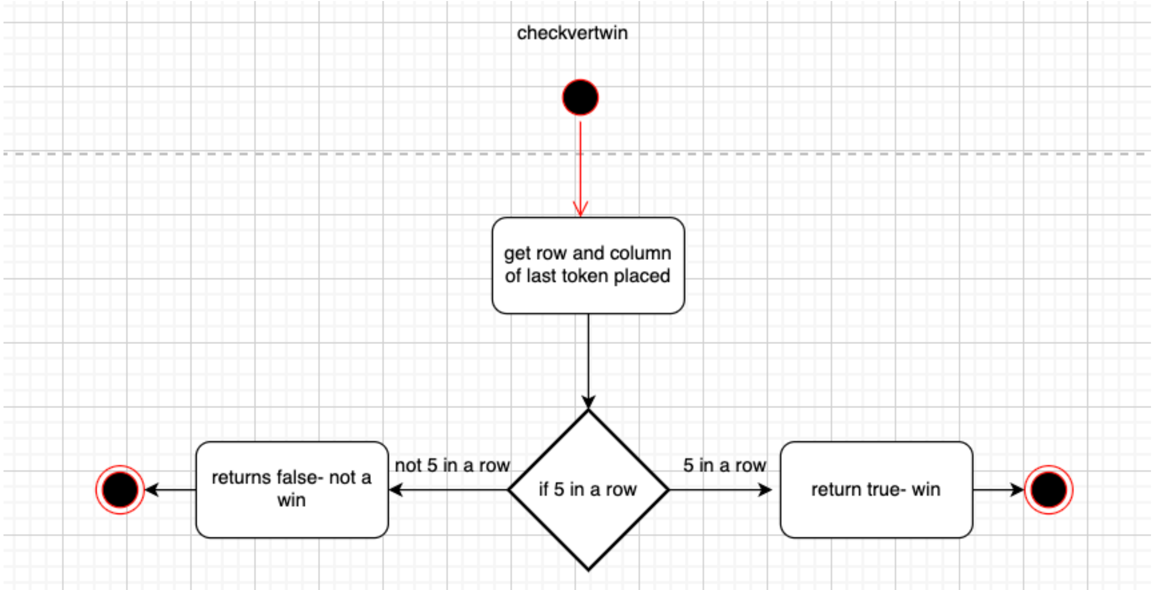

placeToken()
print ERROR: column is full
if full
checkIfFree() == true
if not full
place the token in the lowest available row in the column

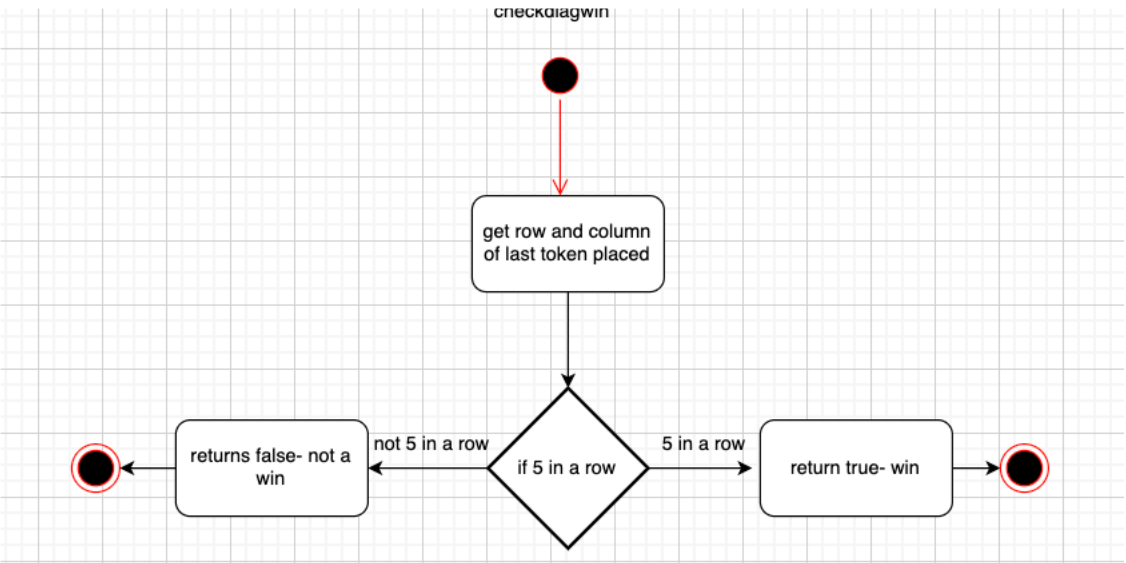

checkifFree())
return false
if full
if column is free
if not full
return true

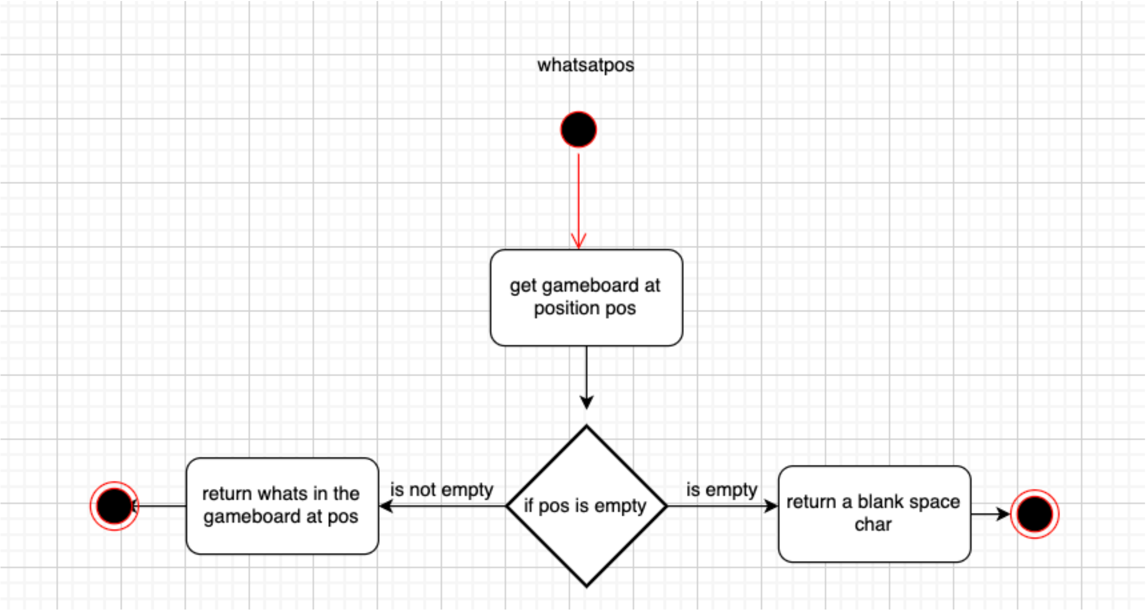

checkForWin()
last token placed in column c
checkHorizWin()
if win
return true;
if not a win
checkVertWin()
return true;
if not a win
checkDiagWin()
return true;
return false;

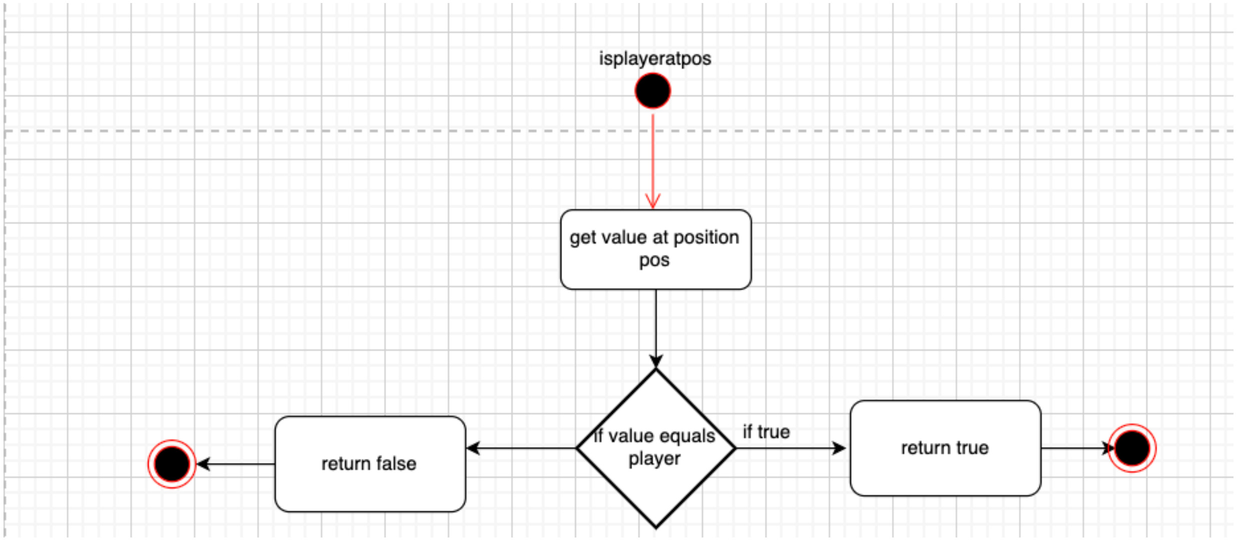

checkTie()
Loop through each position in the 2D game board array.
return false(game is not tied)
if empty position
if the current position is empty
return true(game is tied)

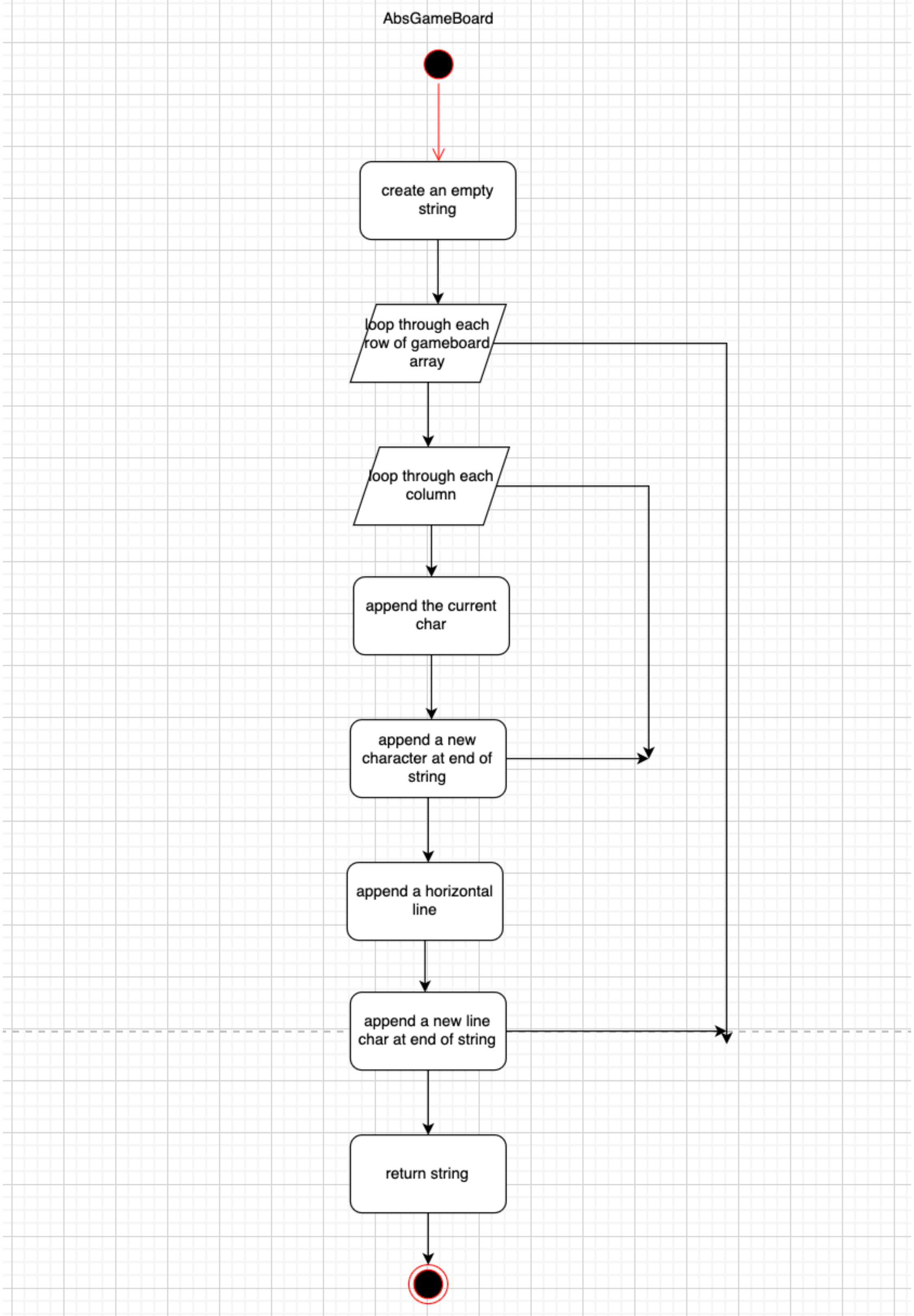

checkhorizwin
get row and column of last token placed
if 5 in a row
not 5 in a row
returns false- not a win
5 in a row
return true- win

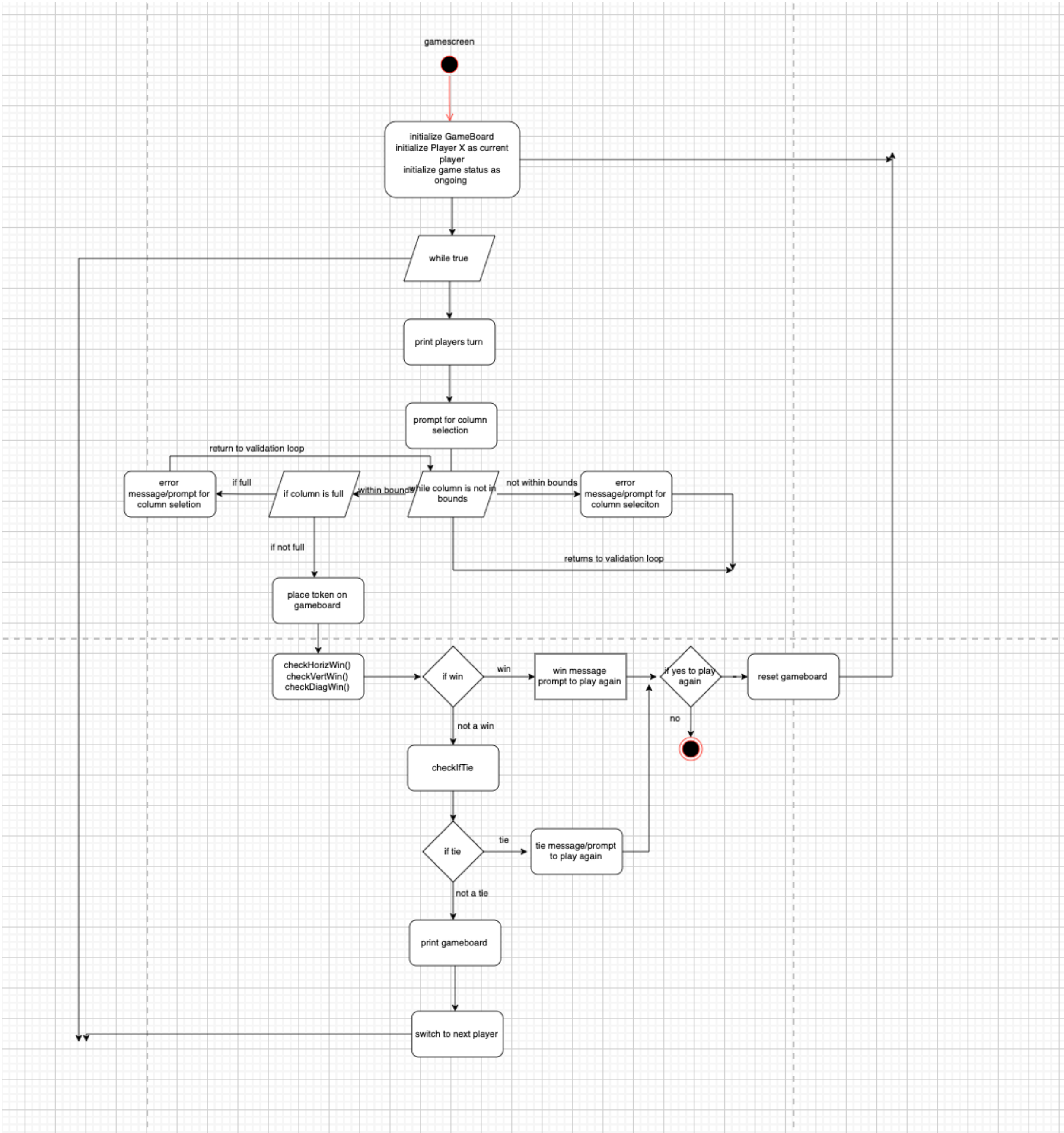

checkvertwin
get row and column of last token placed
if 5 in a row
not 5 in a row
returns false- not a win
5 in a row
return true- win

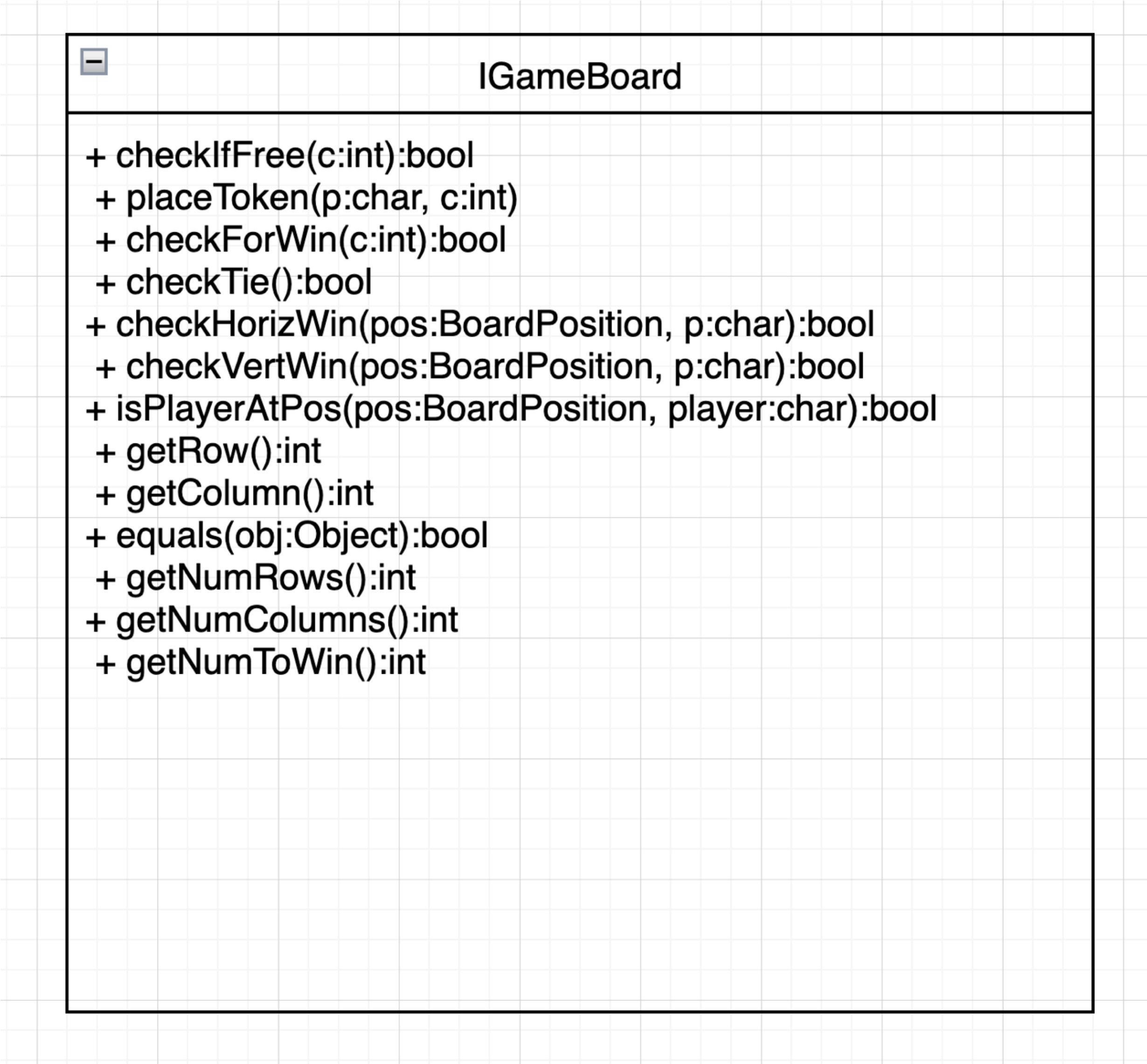

checkdiagwin
get row and column of last token placed
if 5 in a row
not 5 in a row
returns false- not a win
5 in a row
return true- win


whatsatpos
get gameboard at position pos
if pos is empty
is not empty
return whats in the gameboard at pos
is empty
return a blank space char


isplayeratpos
get value at position pos
if value equals player
if true
return true
return false

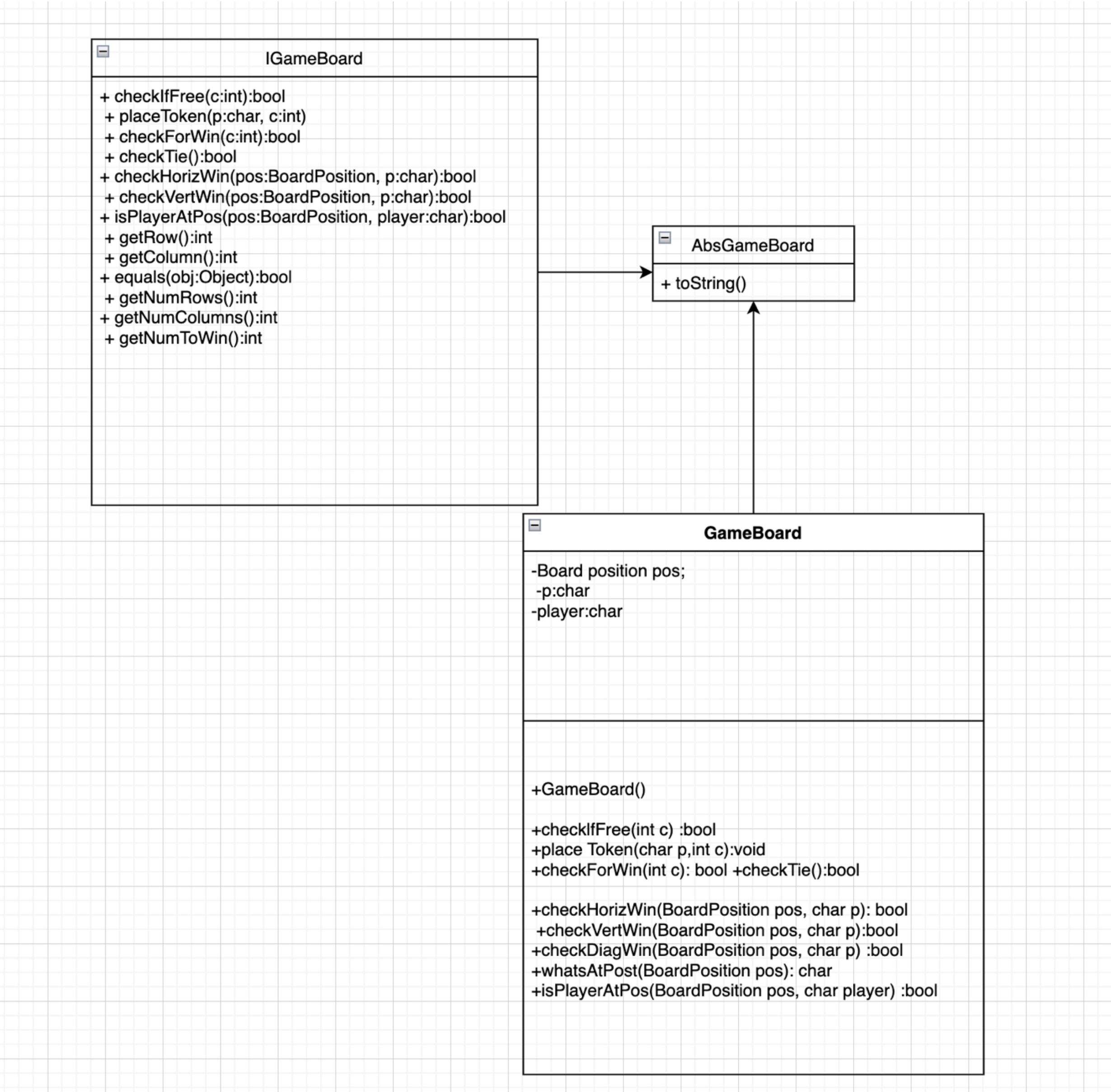

AbsGameBoard
create an empty string
loop through each row of gameboard array
loop through each column
append the current char
append a new character at end of string
append a horizontal line
append a new line char at end of string
return string


gamescreen
initialize GameBoard
initialize Player X as current player
initialize game status as ongoing
while true
print players turn
prompt for column selection
return to validation loop
error message/prompt for column seletion
if full
if column is full
within bounds
while column is not in bounds
not within bounds
error message/prompt for column seleciton
if not full
returns to validation loop
place token on gameboard
checkHorizWin()
checkVertWin()
checkDiagWin()
if win
win
win message
prompt to play again
if yes to play again
reset gameboard
no
not a win
checkIfTie
if tie
tie
tie message/prompt to play again
not a tie
print gameboard
switch to next player



IGameBoard

+ checkIfFree(c:int):bool
+ placeToken(p:char, c:int)
+ checkForWin(c:int):bool
+ checkTie():bool
+ checkHorizWin(pos:BoardPosition, p:char):bool
+ checkVertWin(pos:BoardPosition, p:char):bool
+ isPlayerAtPos(pos:BoardPosition, player:char):bool
+ getRow():int
+ getColumn():int
+ equals(obj:Object):bool
+ getNumRows():int
+ getNumColumns():int
+ getNumToWin():int



IGameBoard

+ checkIfFree(c:int):bool
+ placeToken(p:char, c:int)
+ checkForWin(c:int):bool
+ checkTie():bool
+ checkHorizWin(pos:BoardPosition, p:char):bool
+ checkVertWin(pos:BoardPosition, p:char):bool
+ isPlayerAtPos(pos:BoardPosition, player:char):bool
+ getRow():int
+ getColumn():int
+ equals(obj:Object):bool
+ getNumRows():int
+ getNumColumns():int
+ getNumToWin():int

AbsGameBoard

+ toString()

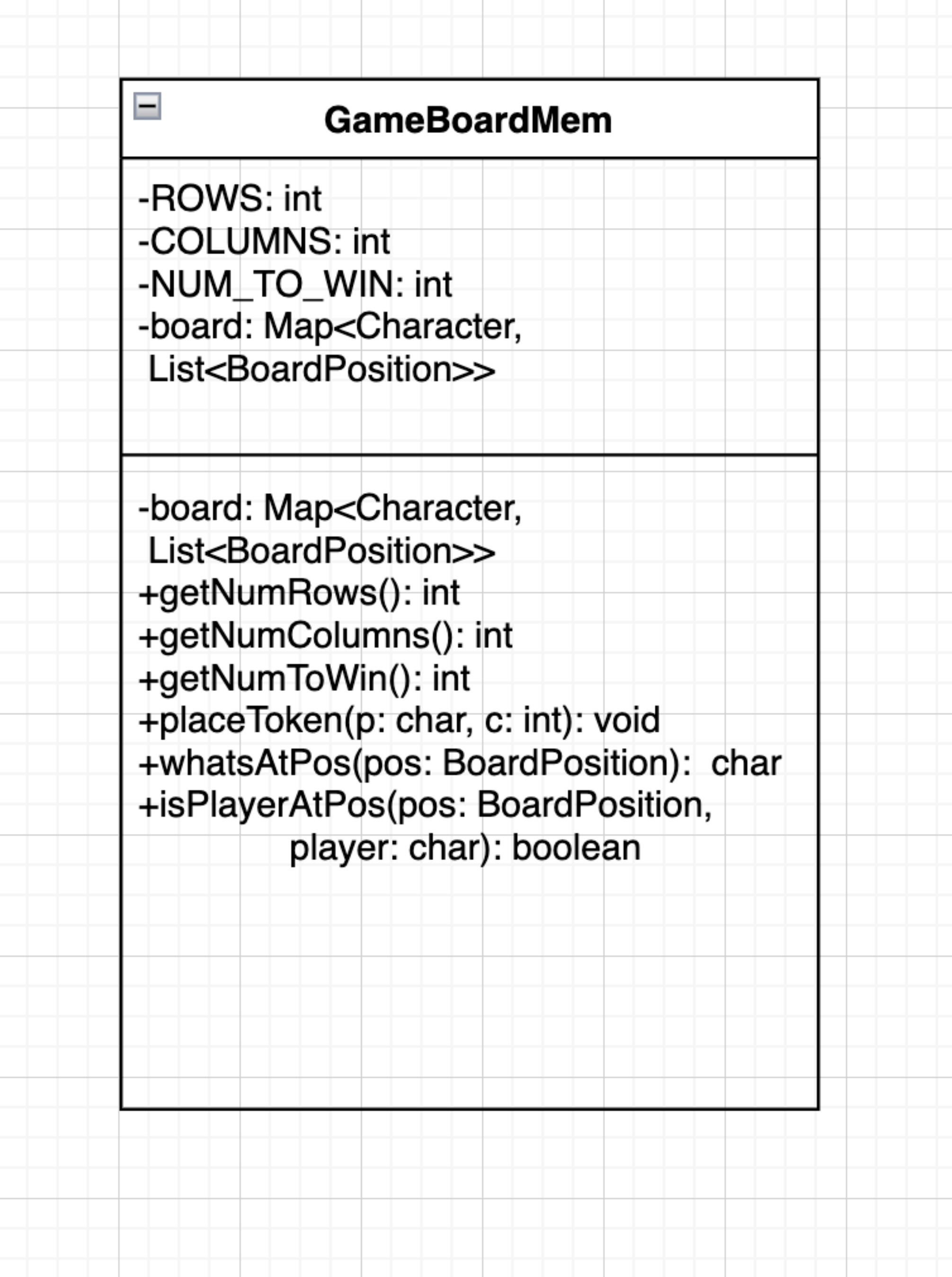
GameBoard

-Board position pos;
-p:char
-player:char

+GameBoard()

+checkIfFree(int c) :bool
+place Token(char p,int c):void
+checkForWin(int c): bool +checkTie():bool

+checkHorizWin(BoardPosition pos, char p): bool
+checkVertWin(BoardPosition pos, char p):bool
+checkDiagWin(BoardPosition pos, char p) :bool
+whatsAtPost(BoardPosition pos): char
+isPlayerAtPos(BoardPosition pos, char player) :bool



GameBoardMem

-ROWS: int
-COLUMNS: int
-NUM_TO_WIN: int
-board: Map<Character,
 List<BoardPosition>>

-board: Map<Character,
 List<BoardPosition>>
+getNumRows(): int
+getNumColumns(): int
+getNumToWin(): int
+placeToken(p: char, c: int): void
+whatsAtPos(pos: BoardPosition): char
+isPlayerAtPos(pos: BoardPosition,
               player: char): boolean