

Paige Clemons (pclemon)
Lab Section 5
Homework 3
Due: 10/28/18

Requirement Analysis

Functional Requirements

- Player should be able to see the board in order to know where they can place their tokens.
- Players should be able to choose where to place their tokens in order to effectively play the game to beat their opponent.
- Players should be able to choose the size of the board so that they can play the game on whatever size board they want.
- Players should be able to choose how many tokens in a row they need to win so that they can play the game in the way they choose.
- Players should be able to choose if they want to play again in order to have a rematch with an opponent.
- Players should be able to choose how many players they want to play with in order to play with however many people they want to.
- Players should be able to choose the token they want in order to have a unique identifier for where they placed their token.
- Players should be able to choose between a fast or memory efficient implementation of the game so that they can play using whichever implementation they would like.

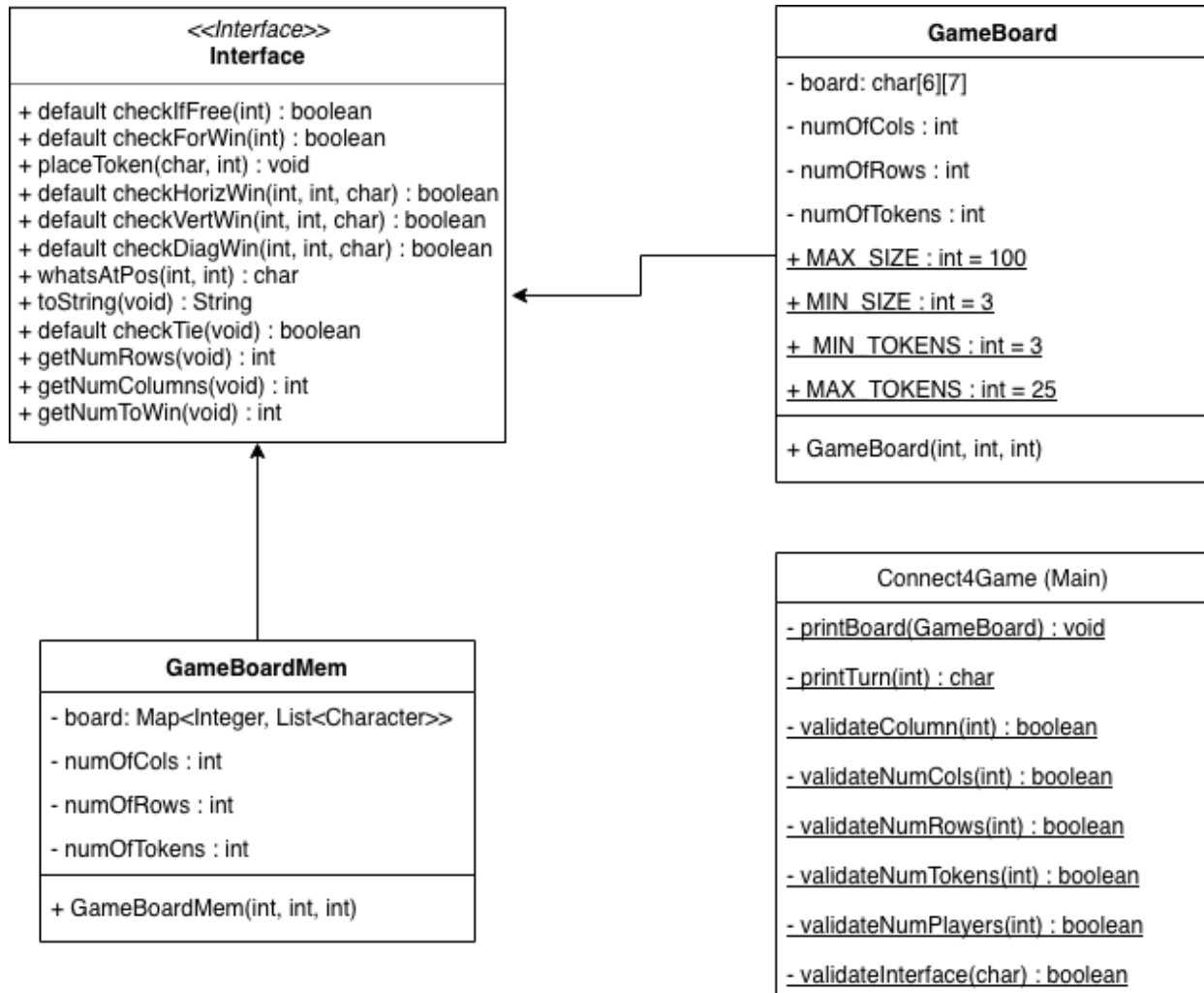
Non-functional Requirements

- Usability
 - An updated game board should be printed to the screen after every move.
 - A prompt should display which players turn it is, and it should ask where the player wants to move next.
 - The player should also be asked if he would like to play again.
- Reliability
 - A prompt should also display stating that a column is full if a player tries to put a token in that column.
 - A prompt should also display if the player tries to input an invalid column number.
 - A prompt should display if the player inputs an invalid board size or invalid winning token amount.
- Performance
 - The game should keep track of two players data.
 - The data should be handled in the most efficient way possible.
- Supportability
 - The game should be playable between two players.
 - The program can run on any computer with Java installed.
- Implementation
 - The program must be coded in Java.
 - The program must be able to run on Unix.
 - Comments should be added to better understand the code.

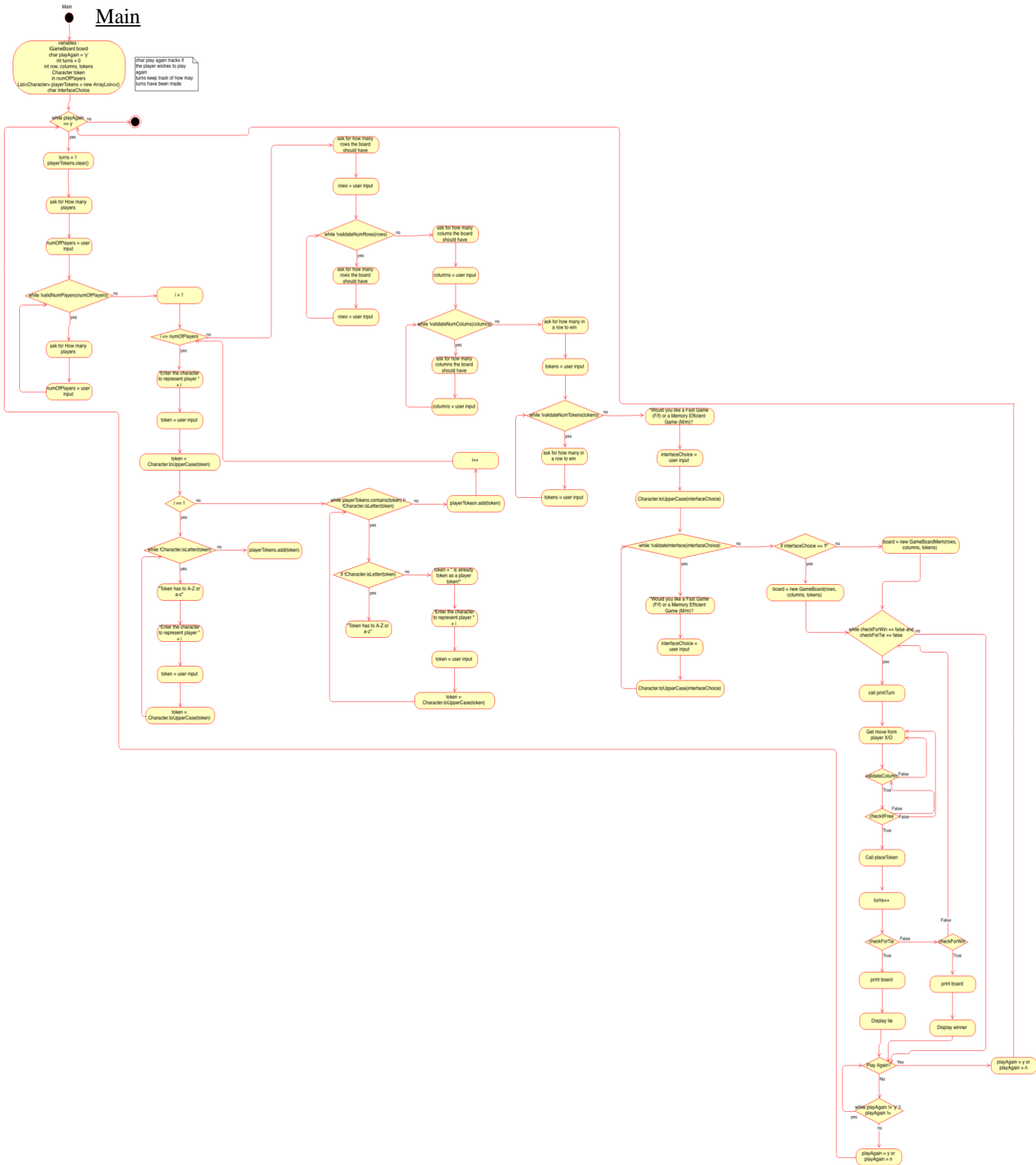
- Javadoc comments should also be added to know the requirements of the classes and functions.
- The program should also use an interface for the methods of the class and have an interface specification.
- The program should have two implementation of the game a fast implementation and a memory efficient implementation.

Diagrams

Class Diagrams

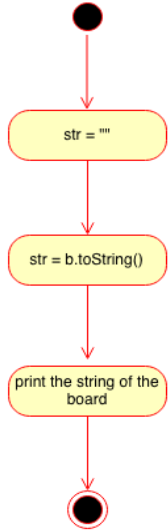


Main

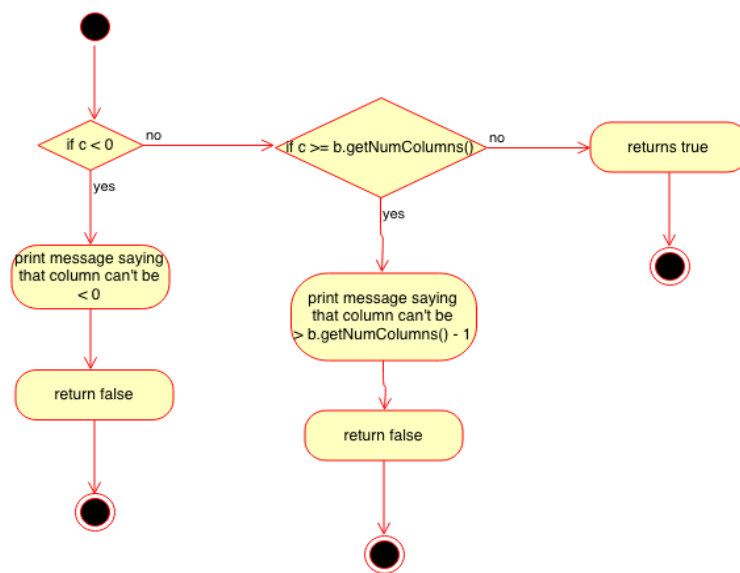


Private Helper Functions in Main

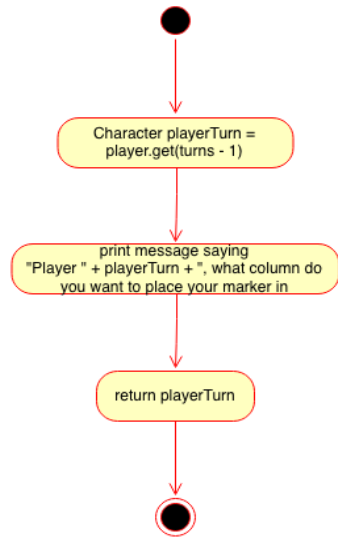
void printBoard(IGameBoard b)



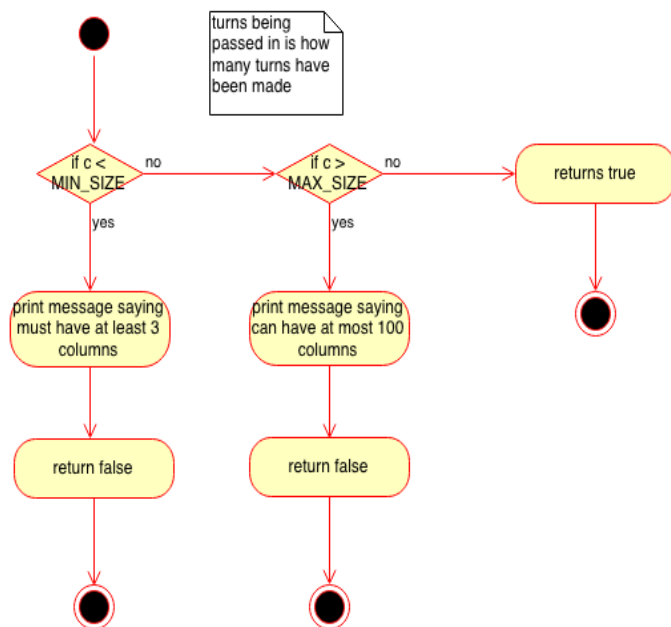
boolean validateColumn(int c, IGameBoard b)



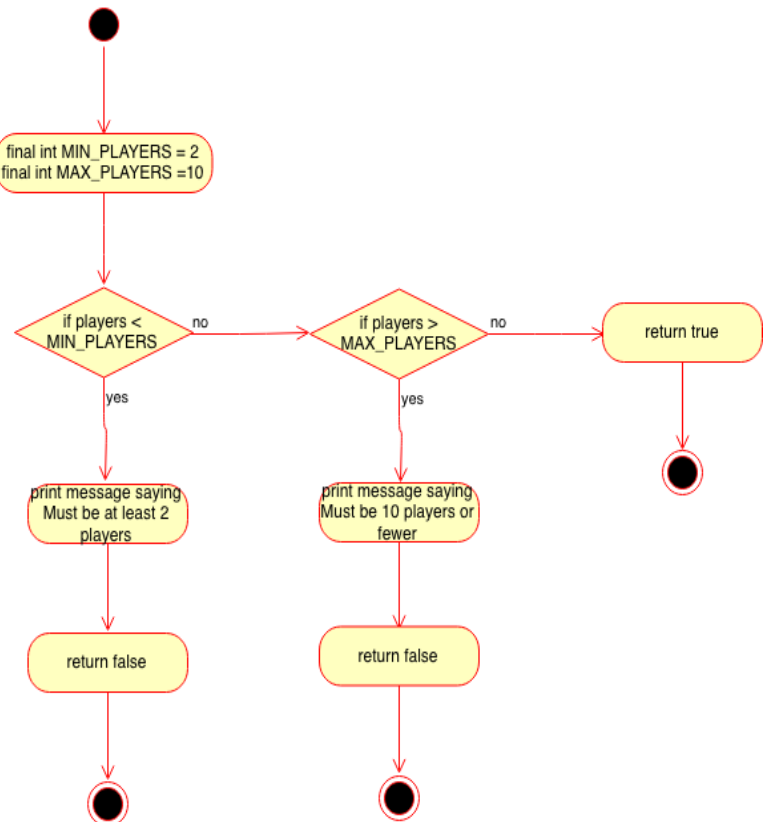
char printTurn(int turns, List<Character> players)



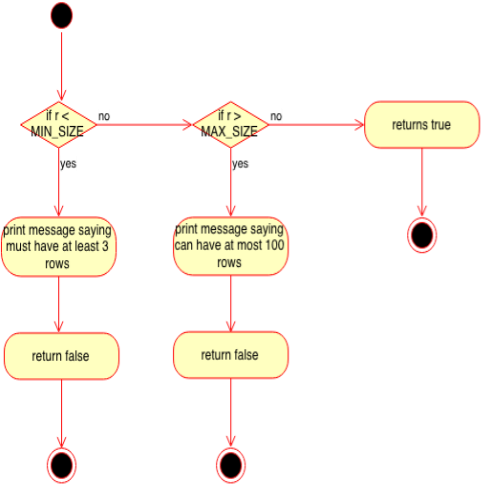
boolean validateNumCols(int c)



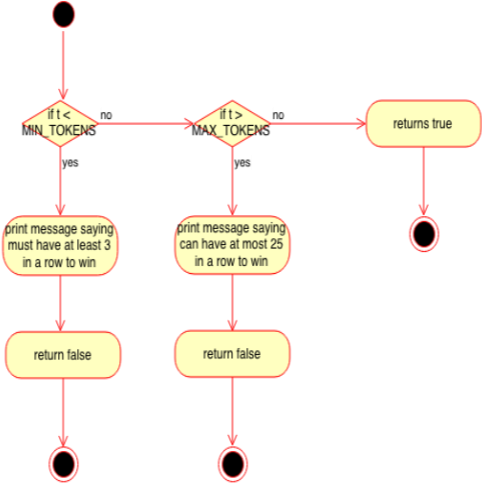
boolean validateNumPlayers(int players)



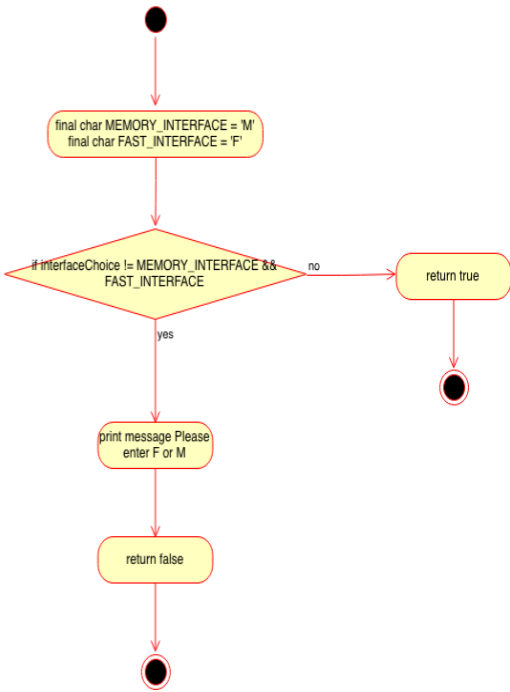
boolean validateNumRows(int r)



boolean validateNumTokens(int t)

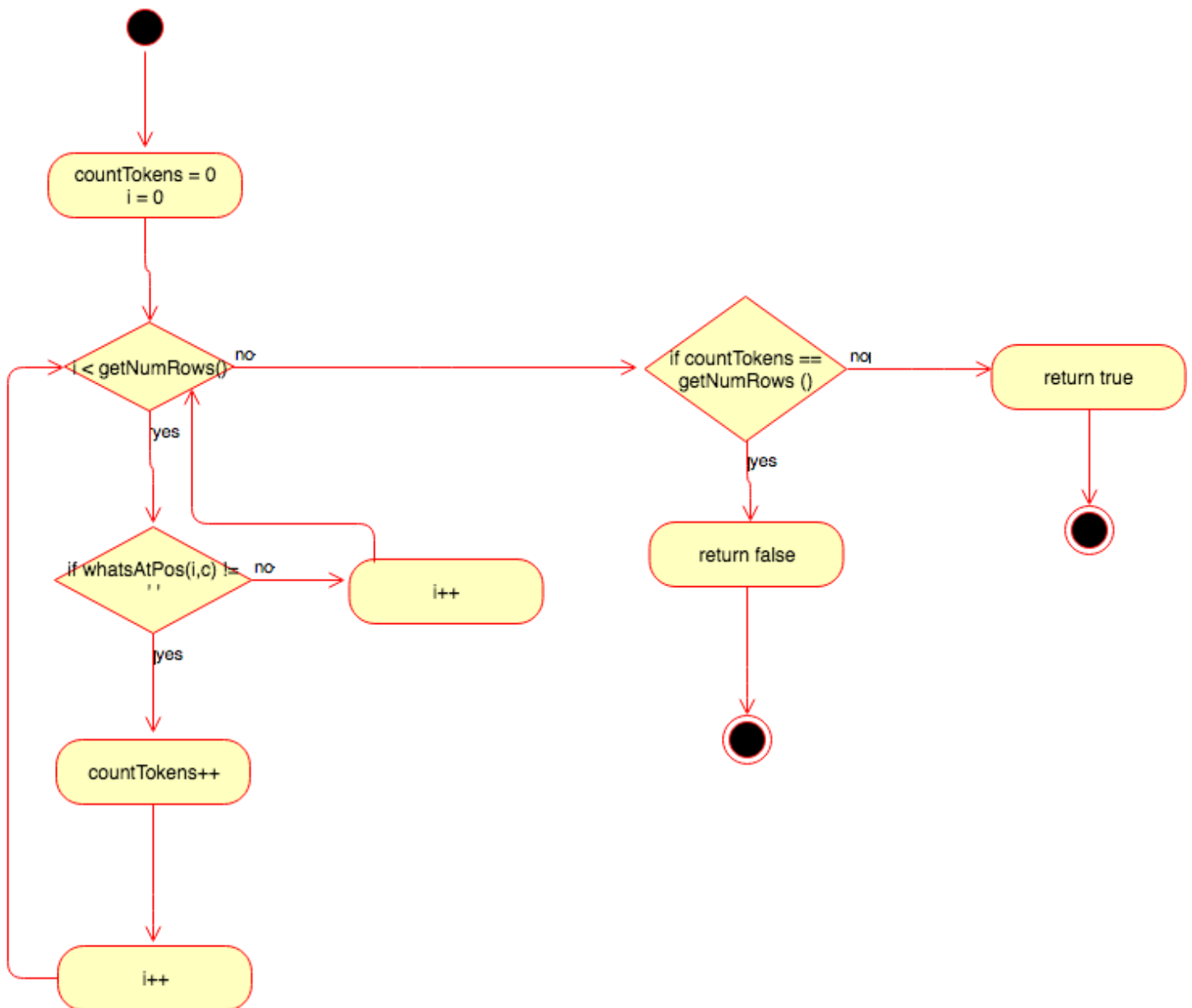


boolean validateInterface(char interfaceChoice)



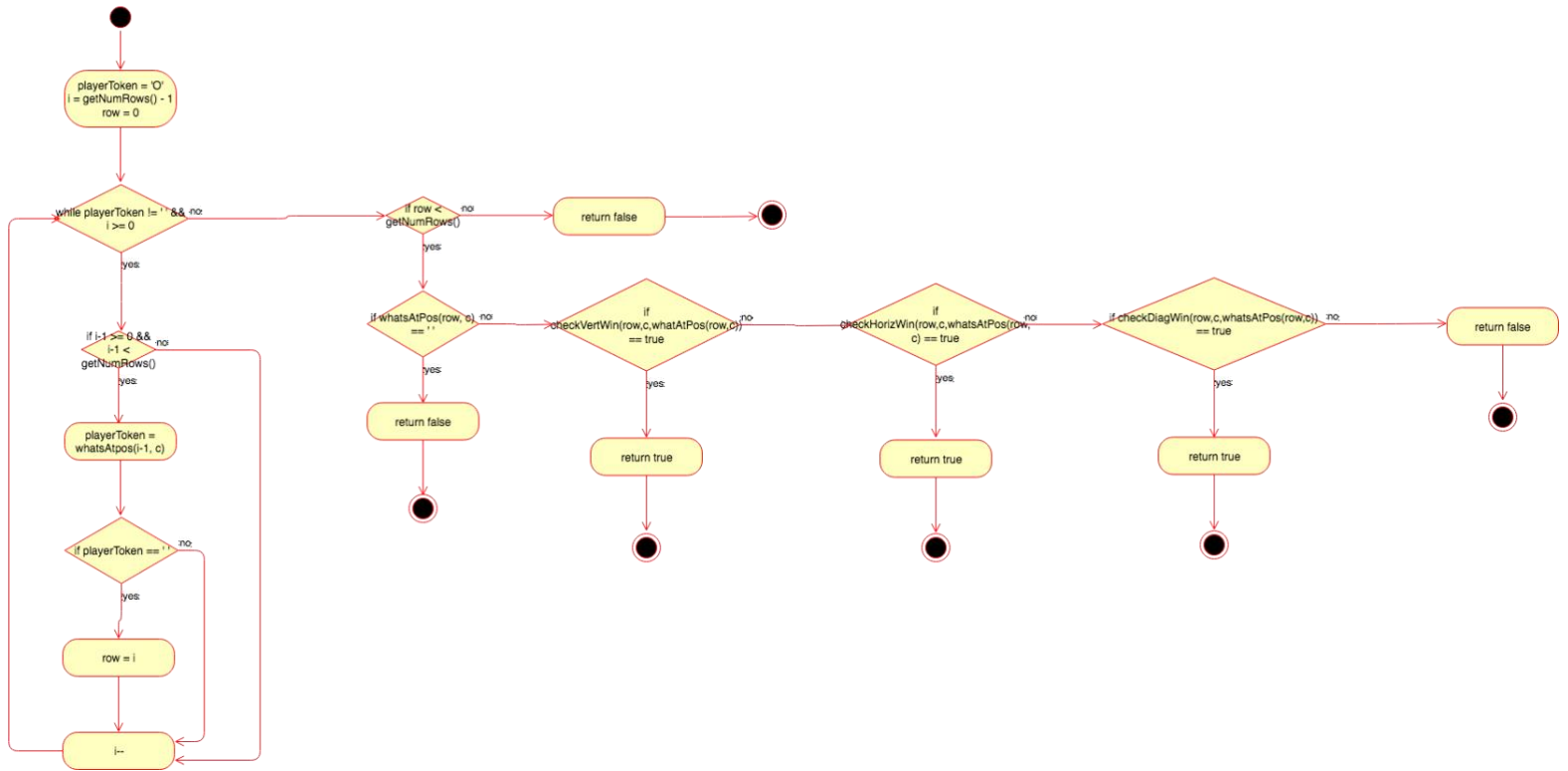
checkIfFree (default method)

boolean checkIfFree(int c)



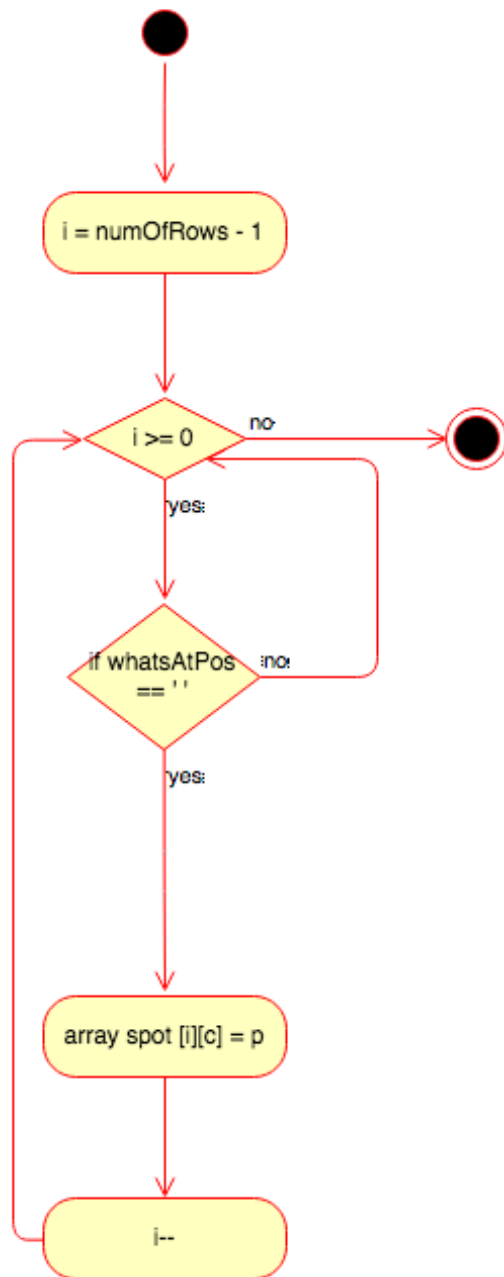
checkForWin (default method)

boolean checkForWin(int c)



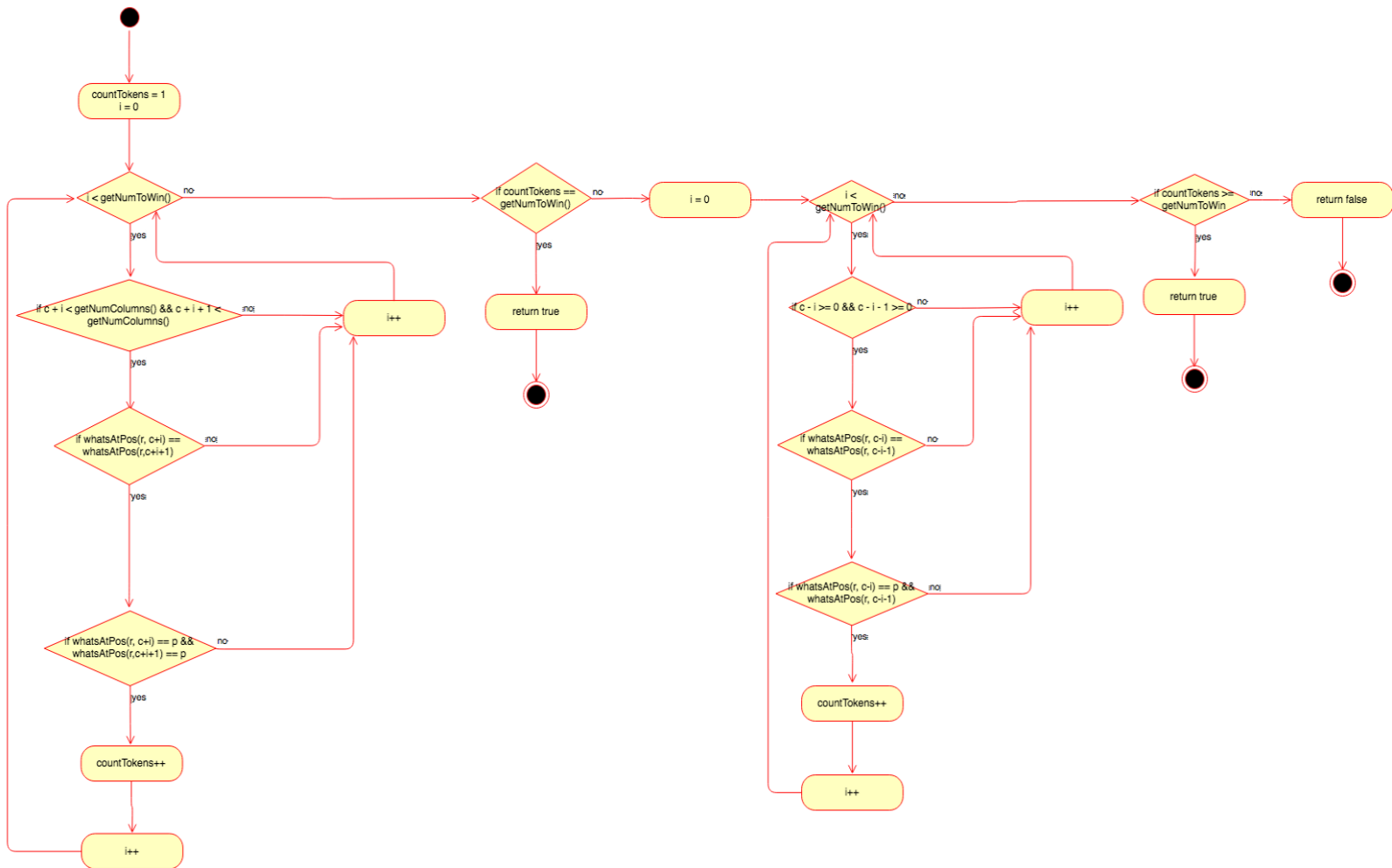
placeToken (in original GameBoard class)

void placeToken(char p, int c)



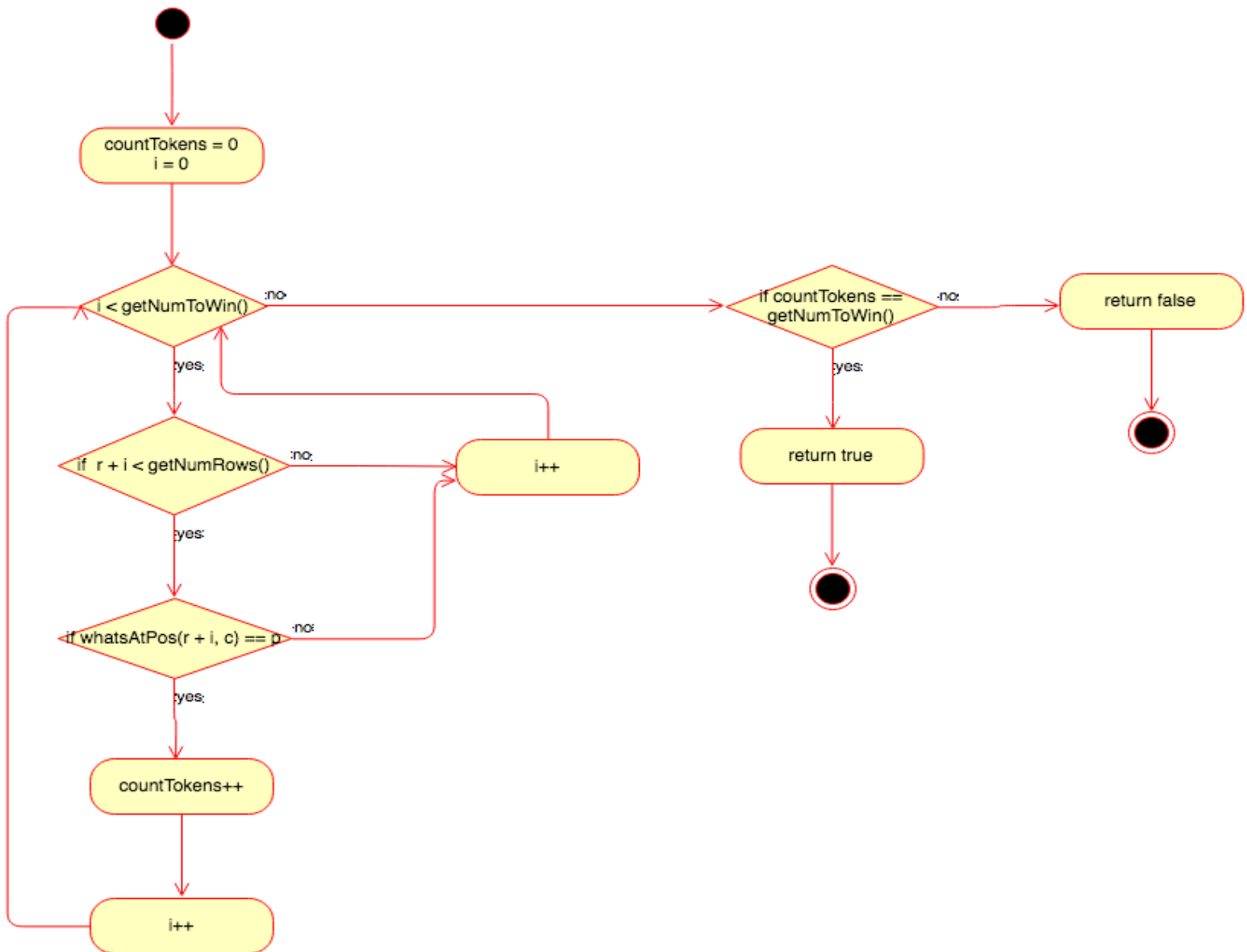
checkHorizWin (default method)

boolean checkHorizWin(int r, int c, char p)



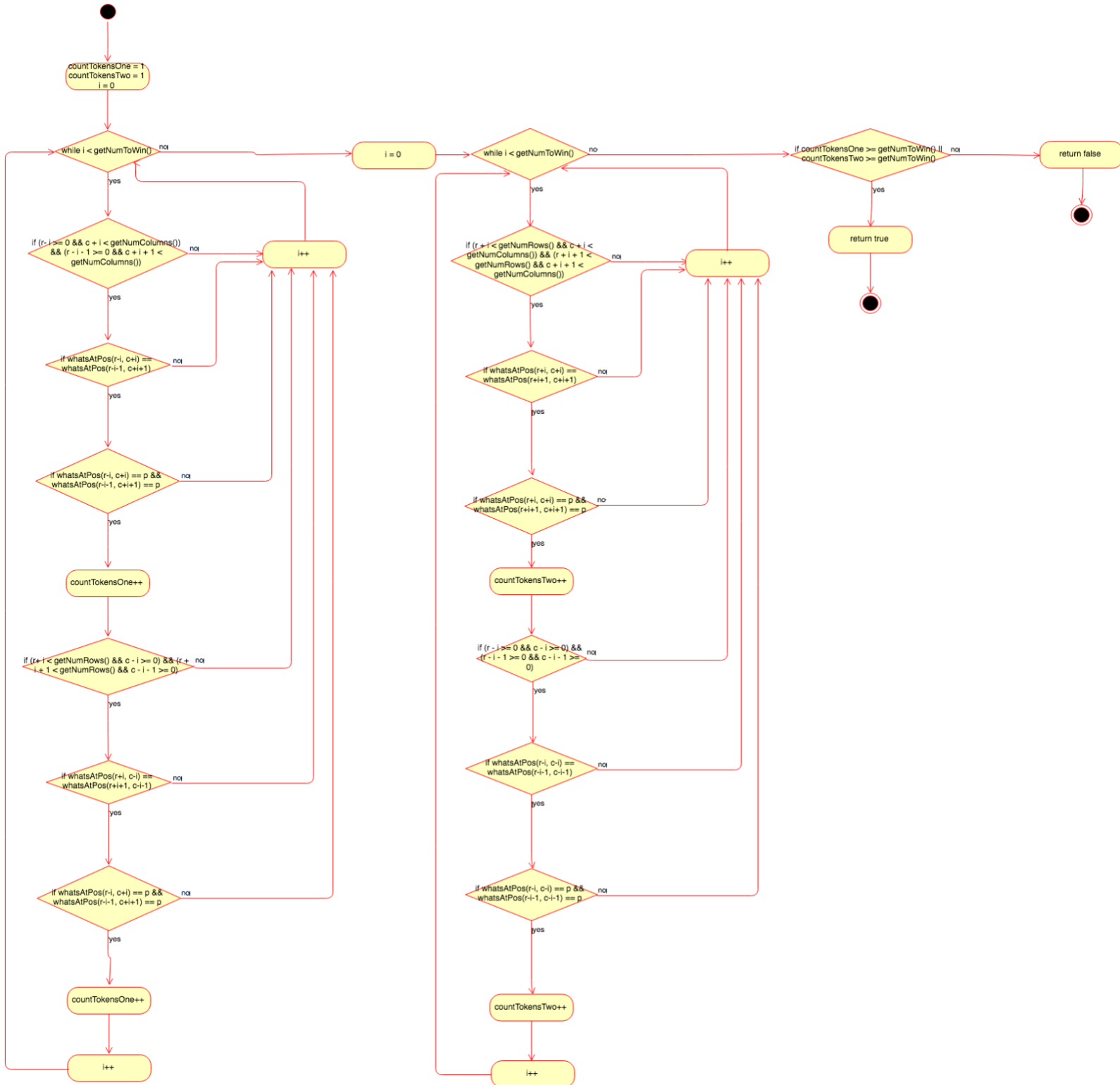
checkVertWin (default method)

boolean checkVertWin(int r, int c, char p)



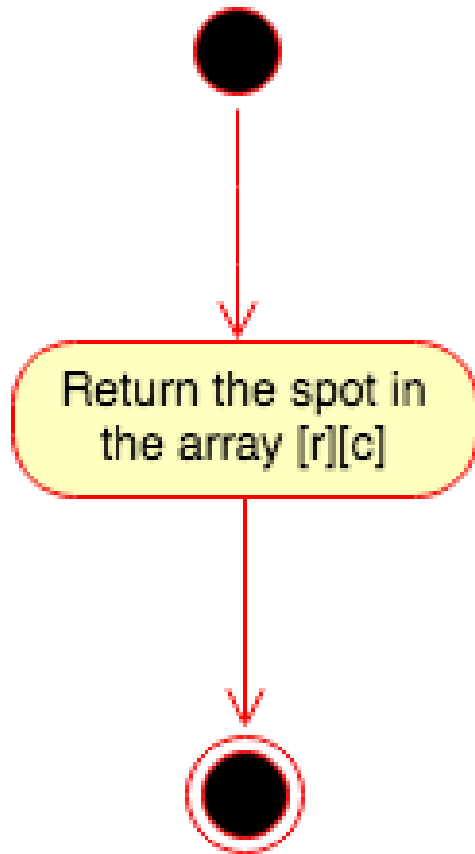
checkDiagWin (default method)

boolean checkDiagWin(int r, int c, char p)

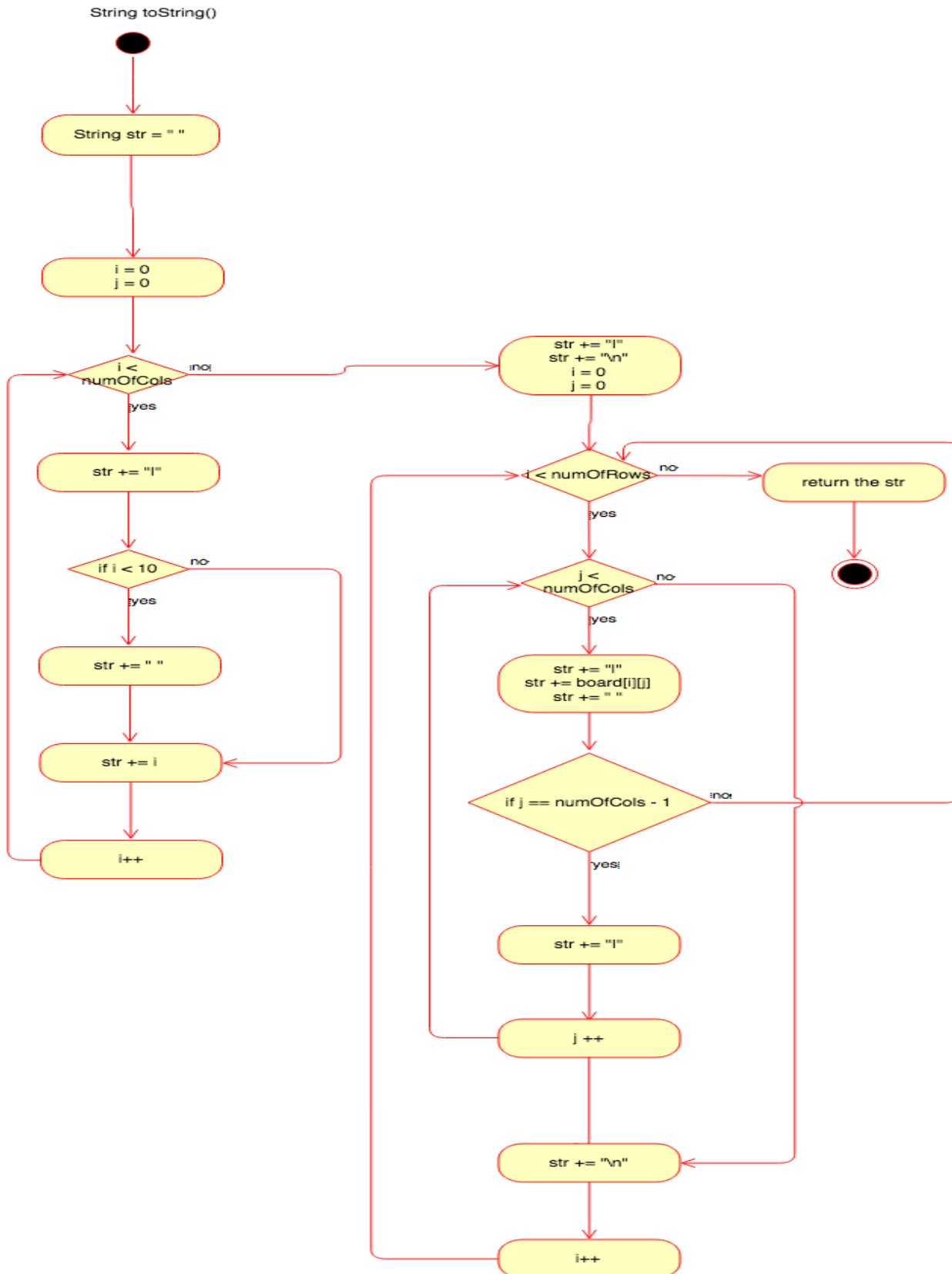


whatsAtPos (in original GameBoard class)

char whatsAtPos(int r, int c)

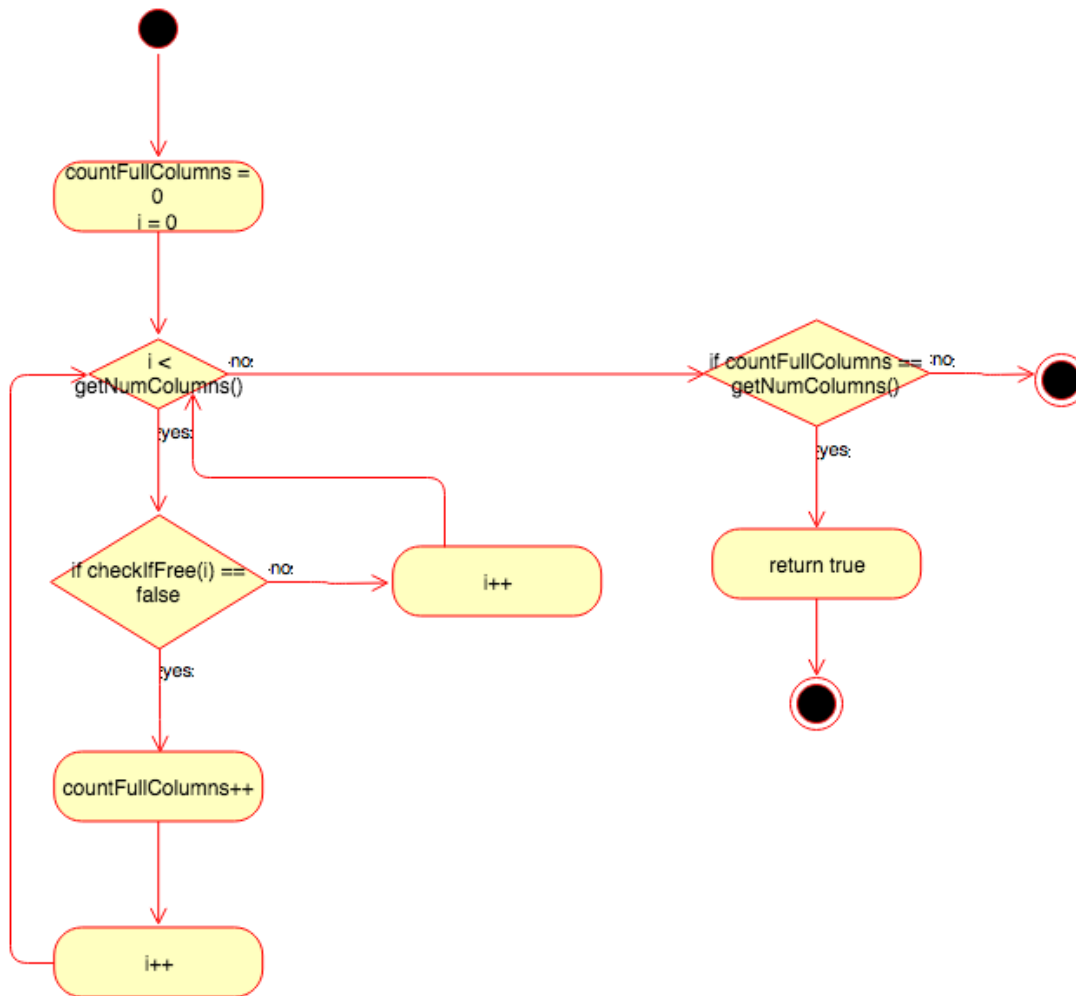


toString (in original GameBoard class)



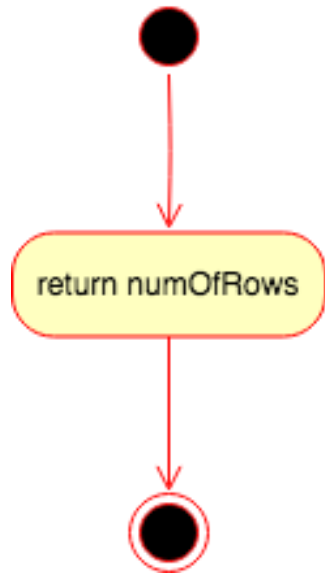
checkTie (default method)

boolean checkTie()

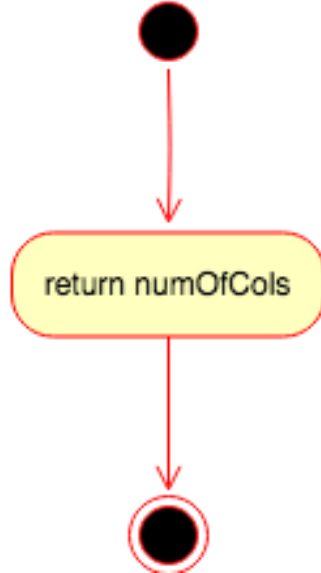


Getters (in original GameBoard class)

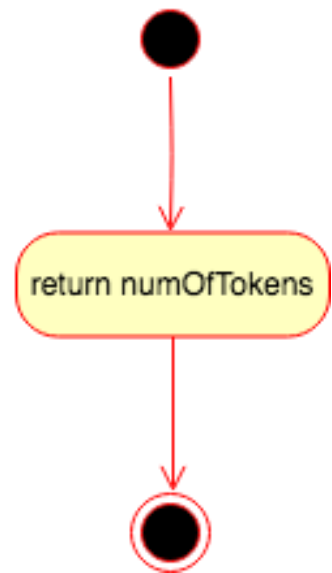
int getNumRows()



int getNumColumns()

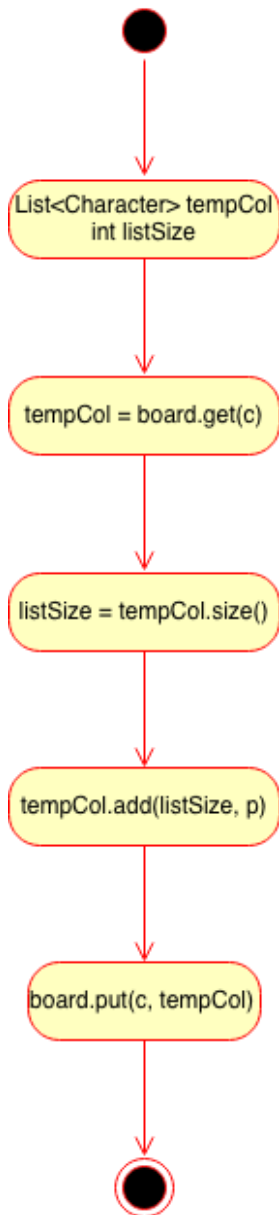


int getNumToWin()



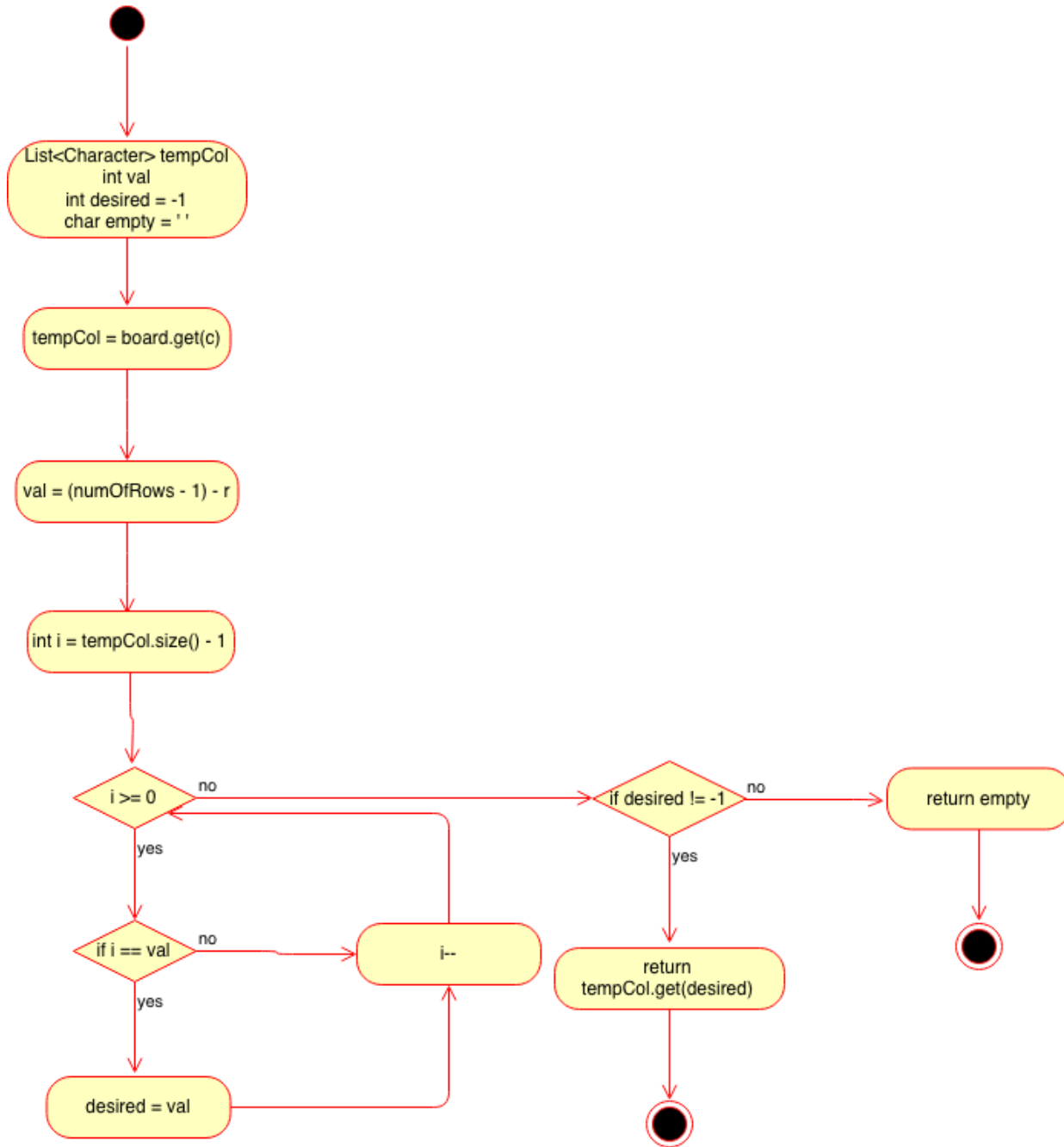
placeToken (GameBoardMem)

```
void placeToken(char p, int c)  
(GameBoardMem)
```



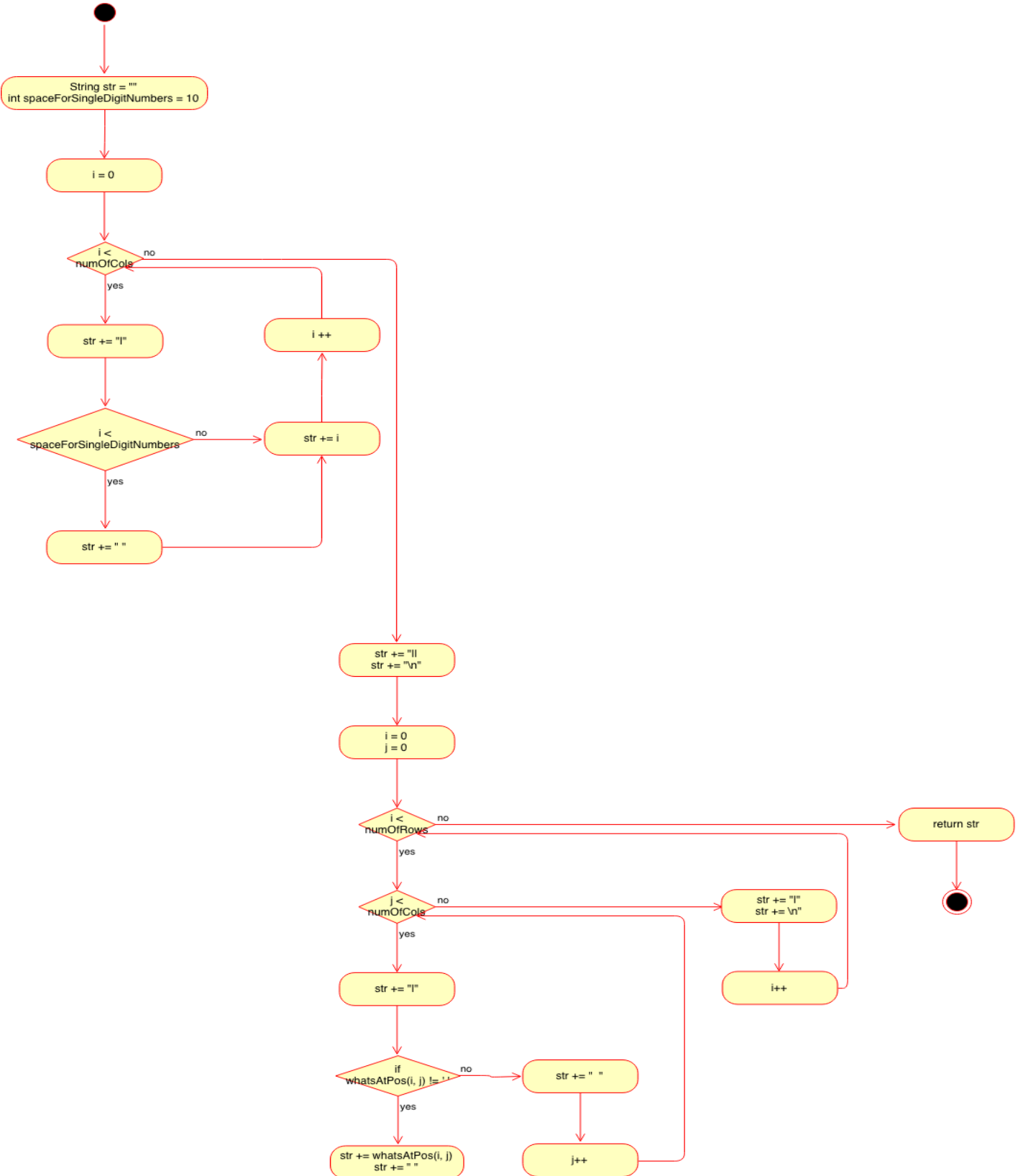
whatsAtPos (GameBoardMem)

whatsAtPos(int r, int c)
(GameBoardMem)



toString (GameBoardMem)

toString()
(GameBoardMem)



Deployment

In order to run my program, you first have to unzip the PaigeClemonsHW3 folder. Then navigate to that folder in your terminal. Once you in the PaigeClemonsHW3 folder, there should only be a cpsc2150 folder, the makefile, and my report in the folder. After you check that stay in that folder, you can compile my program by typing “make” into the terminal. Then to run the program all you have to do is type “make run” then you can play the Connect Four game as usual. If you wish to delete the .class files then all you have to do is type “make clean” into the terminal.