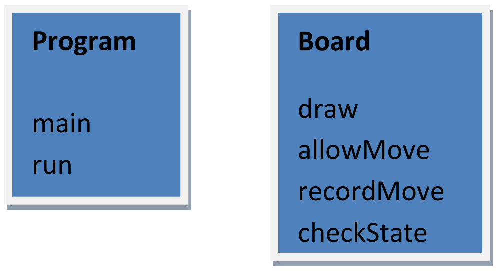
EH102 Tic-Tac-Toe Project

Specification

You are to develop a console based Tic-Tac-Toe game. Tic-Tac-Toe is a simple board game involving a 3x3 grid where each player places one of their game pieces until one player wins by forming a complete row, column or diagonal of their pieces. The first player is called X and the second player is O.



Your project should contain at least the following two classes: Program and Board which are shown in a diagram. Each of the methods of these classes is described in the *Functional Specification* section, and you should implement all of these methods as described.

You provided with an example of this game, which has been packaged into a JAR file. You can run it by typing:

```
java -jar tictactoe.jar
```

You have also been provided with skeleton versions of the `Board` and `Program` java classes to help get you started.

Additional Requirements

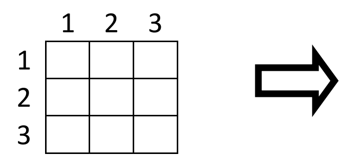
1. If the user types `Q` instead of a valid number then the program should quit immediately.
2. Javadoc all of the methods in both classes.

Extra Credit

1. Check for a situation where nobody has won, but the board is full.
2. Ask the user if they want to play again when the game is finished.

Functional Specification

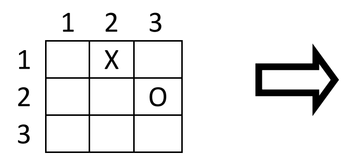
The easiest way to implement the Board class is using a two-dimensional array of integers, i.e.



```
int[][] board = new int[3][3];
```

Each position on the board is thus represented as an integer where:

- 0 (the default value) means that there is no piece at the position
- 1 means that player X has a piece at the position
- 2 means that player O has a piece at the position



```
board[0][1] = 1;
board[1][2] = 2;
```

Methods for Board class:

1. `public void draw()` – this will draw the board to the console
2. `public boolean allowMove(int row, int col)` - this will check to see if the proposed move should be allowed
3. `public void recordMove(int row, int col, boolean xTurn)` - records the move at the given position. E.g. `recordMove(1, 2, true)` records a move by player X at position (1,1) whereas `recordMove(2,3,false)` records a move by player O at position (2,3).
4. `public int checkState()` – this checks the state of the board and returns:
    a. 0 if no player has won
    b. 1 if player X has won
    c. 2 if player O has won

Methods for Program class:

1. `public static void main(String[] args)` – the only thing this method should do is create an instance of `Program` and call its `run` method.
2. `public void run()` – this will contain the main game loop which will continually
    a. Prompt the user to make a move
    b. Check if that move is valid (i.e. call the `allowMove` method of `Board`)
    c. Record that move (i.e. call the `recordMove` method of `Board`)
    d. Check the state of the board (i.e. call the `checkState` method) and if player has won, display a message saying so and quit the program