**Project: Tic-Tac-Toe**

**Collaboration:** Pair or solo. Everyone must sign up to receive 100% for Quiz 8. That means if you are solo, complete the Google form [http://goo.gl/forms/mRDRpaBTX8](http://goo.gl/forms/mRDRpaBTX8). If you have a partner, both you and your partner must complete the Google form entering your name before your partner's name.

**Goals**
- Use a two dimensional array instance variable
- Develop a well-tested type as a Java class
- Write your own tests to make sure the code works
- Write a program that plays a game at the console with a Scanner

1. **Implement class TicTacToe**

The characters of the tic-tac-toe board must always be one of these three:
- ' _ ' Not selected (available)
- 'X' Selected by the eX player
- 'O' Selected by the Oh player (this must an upper case letter O, not zero)

At construction, your board will look like this (you must use ' _ ' as an available location)

```
_ _ _
_ _ _
_ _ _
```

After the first move with the message `choose(1, 1)`, your board will look like this ('X' always goes first).

```
_ _ _
_ X _
_ _ _
```

Here is the beginning of class TicTacToe with a comment included to remind you that you must place your name and SOLO or both names if on a team of two at the beginning of the file.

```java
/**
 * The model for a Tic Tac Toe game that can be played the console or with
 * the GUI http://www.cs.arizona.edu/~mercer/Projects/TicTacToeGUI.java
 *
 * YOUR NAME
 * YOUR PARTNERS NAME or SOLO
 */
public class TicTacToe {

Here are the required methods for class TicTacToe

```java
public TicTacToe()

public boolean choose(int row, int column)
```
// Return the next player as 'X' or 'O'.
public char getNextPlayerChar()

// Return a textual version of the tic tac toe board like this:
//   O _ X
//   O X _
//   O _ X
//
@Override
public String toString()

// Get back the current state of the game with 'X' and 'O' if chosen.
// This is needed in the GUI to display a graphical view.
public char[][] getCharArray()

// Allow anyone to check if either the 'X' or 'O' player has won
public boolean didWin(char playerChar)

// Use this method to see if there is a tie or not.
// Return true if there are no places remaining and there is no win
public boolean didTie()

// Allow users to know if there are any more possible choices. Return true
// if neither player has won and there is at least one place to choose.
public boolean notDone()

The following code is included to show how to use all methods to generates the output in the box. You must write your own unit test with @Test methods that play entire games with these methods.

```
TicTacToe game = new TicTacToe();
System.out.println(game.getNextPlayerChar());
System.out.println(game.notDone());
System.out.println(game.choose(1, 1));
System.out.println(game.notDone());
System.out.println(game.choose(0, 0));
System.out.println(game.getCharArray()[2][2]);
System.out.println(game.didWin('X'));
System.out.println(game.didWin('Y'));
System.out.println(game.didTie());
System.out.println(game.toString());
```

2. Implement a game in TicTacToeMain
Complete a console based game--use a new Scanner(System.in)--of tic-tac-toe that plays a game of tic-tac-toe by asking the user to enter the row and column of their choosing. You game must be in a class named TicTacToeMain. This class will have method public static void main(String[] args) so it can be run as a Java application. For each turn, show the current state of the game. Report a win or tie as soon as possible. Tell the user there was an incorrect input if the row or column is out of range until they enter a correct choice. The dialog begins like this (user input for a row and column shows as 1 1):

```
Play a game of Tic Tac Toe

Row and column for X? 1 1
_ _ _
_ X _
_ _ _
```

```
Row and column for O? 2 2
_ _ _
_ X _
_ _ O

Row and column for X? 1 1
Row and column for X? 2 2
Row and column for X? 3 99
Row and column for X? 0 1
_ X _
_ X _
_ _ O

Row and column for O? 1 2
_ X _
_ X O
_ _ O

Row and column for X? 2 1
_ X _
_ X O
_ X O

========= Game Over
X won

Optional GUI: When completely done with the console game, see how your code runs as an event-driven program with a graphical user interface (GUI). Put this file into your project and run it as a Java Application:

http://www.cs.arizona.edu/~mercer/Projects/TicTacToeGUI.java

Grading Criteria  Maximums 90..95 points on WebCat or 100..105 after the console game is graded.

___/ 90pts  WebCat turnin of TicTacToe.java and TicTacToeTest.java: code and problem coverage
___/ 10pts  Console game, or a program with a main method in the same project
  +2  X goes first
  +2  Your dialog matches the sample given
  +2  Shows who wins or indicates a tie.
  +2  Reprompts if a choice is already taken
  +2  Handles out of range integers such as 3, -1, or 99