Practical 1: Programming Warm-Up Through the Tic-Tac-Toe Game

Instructor: Shailesh B. Pandey (Everest Engineering College) Course: Intelligent System (BEIT)
Course webpage: sites.google.com/view/eec-is

1. Overview

Tic-Tac-Toe is a two-player game that is usually played on a 3x3 board (grid). As you may know, if the game is played with an ideal strategy the game always ends in a draw. In our course we will study how we can create an "intelligent" Tic-Tac-Toe playing Bot (agent) that plays the game perfectly against a Human opponent i.e. it never loses. The task in this practical is rather non-challenging. We are going to code a console-based Tic-Tac-Toe game that is played between two Human players. The objective of this practical is for you to get your programming skills up to the level that is required in this course. Take this course as an opportunity to improve your knowledge on programming language basics, data structure (particularly, graphs and trees) and implementing AI algorithms.

2. Input and Output

There will be two players: PLAYER_1 ('X') and PLAYER_2 ('O'). You will ask the players to choose the location in the grid where they want to put their mark. Tic-Tac-Toe can board can be stored in a 2D array. So, you will need to get a row and column number from the standard input.

The output should look like this:

```
   |   |
-----------
   |   |
-----------
   |   |
X's turn to put a mark. Type row and col:
1
1
   |   |
-----------
   | X |
-----------
   |   |
O's turn to put a mark. Type row and col:
```

There are three results possible: X wins or O wins or there is a draw. At the end of the game a suitable message is displayed.

3. Tasks

I have provided a starter code to build the game. Study the `TTTGame` class. It is the class that you will run to play the game. Observe how the game starts, is played and ends i.e. program flow. Try to understand what each method is doing without worrying about how it is being done. There is more than one way to do the same thing. This class utilizes the `TicTacToe` class to handle the game board (grid). It declares a 2D char array (matrix) to store the board information. There are variables to determine the turn of the players and the symbol ('X' or 'O') the players will use.

Part A:

- Your main task is to complete the code in the TicTacToe.java file. The comments give you information on what these methods are supposed to do. After you have implemented all the empty methods, you can execute `TTTGame` class to play the game.
- Add code to check row and column are within the range [0,2]. If you run the current code and input a negative number, the program will crash. This is because an array does not take negative index.

Part B (Extra Marks):

- The code only works on a 3x3 grid. Modify the code to handle a n x n board. Modify starter code to handle any size of board.
- There are no standard rules to play n x n Tic-Tac-Toe. You can then implement the following rule: the game continues until all cells are filled. The winner is the one who has the greatest number of three-in-a-row lines (horizontal, vertical and diagonal) i.e. count how many three lines with three consecutive X's or O's each player has.

4. Submission

You can print or write the solution in A4 paper. Include a cover page with the usual: name, roll number, subject, practical title etc. Make sure the code you have added is well commented. There is marks for it. Good naming choice for variables, methods and class, and code quality also receives marks. Plagiarism is not tolerated. You can always ask for help from friends and me when you are stuck but you cannot ask for them to write your code or use their code. If you have not written the code yourself you will not get any marks. If you have copied from your friends both will lose marks. Submission only does not give you passing marks.

5. Starter Code

Create a Java Project in Eclipse or IDE of your choice. Add these two classes and complete the code.

`TTTGame.java`

```java
import java.util.Scanner;

public class TTTGame
{
    TicTacToe game = new TicTacToe();  // For all TTT board related tasks.
    char winner;                       // Who won?

    /*
     * Start the game
     * Display the results after it is completed.
     */
    public void startGame() {
        game.displayBoard();
        playGame();
        winner = game.getWinner();
        printMessage();
    }

    /*
```

```
      * Scanner class is used to get [row,col] from standard input
      * Game is completed if there is a winner or 9 moves have been made.
      */
     public char playGame()
     {
         Scanner in = new Scanner(System.in);
         int count = 0;     // Count number of turns. If it is 9 it is a draw.
         char turn;         // Is it X's turn or O's turn?
         int row, col;      // Hold board position.

       // while no one has won and not yet a draw
         while(game.getWinner() == ' ' && count<9)
         {
             turn = game.whoseTurn();
             System.out.println(turn+"'s turn. Type row and col:");
             do {
                 row = in.nextInt();
                 col = in.nextInt();
     }while(game.getMark(row,col)!=' '); // Is this cell empty?


             game.putMark(row, col);
             game.displayBoard();
             count++;
         }
         in.close();

     }

     /*
      * Print Win or Draw message.
      */
     public void printMessage() {
         if(winner=='X')
             System.out.println("X has won!");
         else if(winner=='O')
             System.out.println("O has won!");
         else
             System.out.println("It's a draw!");
     }

     public static void main(String[] args)
     {
         TTTGame ttt = new TTTGame();
         ttt.startGame();
     }
 }
```

TicTacToe.java

```
public class TicTacToe
{
    char[][] board;           // TicTacToe board has 3 rows and 3 columns.
    char PLAYER_1 = 'X';
    char PLAYER_2 = 'O';
    char turn;                // Whose turn is it?
    /*
     * Initialize the 2D array. X always start's first.
     */
    public TicTacToe() {}

    /*
     * Check 3 rows, 3 cols and 2 diagonals for a win
     * If there is a winner return who won : X or O
     * Otherwise return a blank (space) character.
     */
    public char getWinner() {}

    /*
     * Pretty print the TTT board.
     */
    public void displayBoard() {}

    /*
     * Return the Player who has to put a mark.
     */
    public char whoseTurn() {}

    /*
     * Fill the board at [row,col] with X or O
     * depending on whose turn it is
     * then change turn from X to O or O to X.
     */
    public void putMark(int row, int col) {}

    /*
     * Return the mark at [row,col] in the board.
     */
    public char getMark(int row, int col) {}
}
```