# CS-141 Introduction to Programming
# TicTacToe

Goals:

1. Practice using 2D arrays.
2. Practice working with code composed of multiple methods.

Requirements:

- You are to write a program that plays TicTacToe where the board is stored as a 2D array of chars. The code should be well structured using methods. *Each method should be fairly short* (e.g. fit easily on one screen).
- The TicTacToeGame.java is given below as a reference, but please start with the Netbeans project available on the class webpage. Before beginning, carefully read the comments so you know what each method does!
- *Do not use any global variables* other than the constants and Scanner which are provided. You should create the board in play() and pass the board as a parameter to the methods which need to access the board. This will give you practice using arrays as parameters.
- The output should look very similar to the sample output below.

Suggestions:

- Feel free to make small changes in the code structure, if you want, as long as the final code is well structured (e.g. short methods, no globals).
- The game loop in play() should continue looping as long as the number of moves is less than 9 and no one has won. Each time through the loop, one of the players (user or computer) takes a turn. An if-statement will be needed to select the turn method (userTurn vs computerTurn). See the comments in the code.
- Start implementing the methods that don't rely on other methods, testing thoroughly as you proceed. For example, it makes sense to start with the openingGreeting(), initBoard() and printBoard() methods. (Note some of these methods are already implemented for you) Test that the methods work by calling them from play(). Then implement initPlayer() and readValue().
- Use tabs (\t) as a way to format the columns when printing the board.
- Pick whatever strategy you want to implement the computerTurn() method. It doesn't have to play intelligently. It just has to be sure to play in an empty space. (If you take our Machine Learning class, you will learn how to create game trees).
- Try generating the Javadoc based on the comments in the code. Go to the menu:
  Run->Generate Javadoc
- If you want, add some of the following extras (not required):
  - Have a way for the user to quit midway through the game.
  - Allow the player to play multiple games.

**Sample Structure - TicTacToeGame.java**
**(please download the entire Netbeans project from our website.)**

```
package tictactoe;
import java.util.Scanner;

/** TicTacToe
 *  The user ('X') plays against the computer ('O')
 *  The first player is chosen randomly.
 *  @author Your Name
 */
public class TicTacToeGame {

    /** the game character symbol for the player */
    public final char PLAYER = 'X';
    /** the game character symbol for the computer  */
    public final char COMPUTER = 'O';
    /** the game character symbol for an empty place on the board */
    public final char EMPTY = '-';
    /** Use for reading input from the user */
    Scanner in = new Scanner(System.in);

    /**
     * Plays the game of TicTacToe. Stops when either 1) a player wins or
     * 2) no spaces are left on the board.
     */
    public void play() {
        // Declare and create the board
        openingGreeting();  // Greeting
        // Pick starting player
        // Initialize the board
        // Print the board

        int moves = 0;           // used to control game loop
        boolean hasWon = false;  // used to control game loop

        // game loop over moves and hasWon. In the loop:
        //    Have user or computer take their turn
        //    Print the board
        //    Check to see if anyone has won (update hasWon)
        //    swap players
        //    increment the moves
        //

        if (!hasWon) {
            System.out.println("No one has won.");
        }
    }
```

```
/** Welcome the player */
public void openingGreeting() {
    System.out.println("\nWELCOME TO TICTACTOE\n");
    System.out.println("You will be X, the computer will be O.");
    System.out.println("The starting player will be picked randomly.");
    System.out.println("Enjoy the game and good luck!\n");

}

/** Initialize the values in the board to all blanks (EMPTY).
 * Note, each board position will eventually be either:
 *   'X' - this is the user (PLAYER)
 *   'O' - this is the computer (COMPUTER)
 *   ' ' - a space indicates the position is empty (EMPTY)
 *   @param board
 */
public void initBoard(char[][] board) {
    // Loop over rows and columns of the board, setting each to EMPTY
    // YOUR CODE GOES HERE
}

/**
 * Swap the current player. That is, if the current player is 'X', it
 * is swapped to be 'O', and vice versa.
 * @param curPlayer the currentPlayer
 * @return the new current player
 */
public char swapPlayer(char curPlayer) {
    if (curPlayer == PLAYER) {
        curPlayer = COMPUTER;
    } else {
        curPlayer = PLAYER;
    }
    return curPlayer;
}

/**
 * Randomly pick which player (user or computer) goes first. Set the
 * currentPlayer to the first player. It is assumed that the user
 * is always "X" and the computer is always "O".
 * @return the first player
 */
public char initPlayer() {
    System.out.println("You are X and the computer is O.");
    char currentPlayer;
    if (Math.random() < .5) {
        currentPlayer = PLAYER;
        System.out.println("You are the first player.");
    } else {
```

```
        currentPlayer = COMPUTER;
        System.out.println("The computer is the first player.");
    }
    return currentPlayer;
}

/**
 * Chooses which position the computer will play. Sets that position
 * of the board to "O"
 * @param board the board
 */
public void computerTurn(char[][] board) {
    // YOUR CODE GOES HERE -
    //         Pick some strategy (doesn't have to be smart!)
}

/**
 * Have the user choose which position to play. Sets that position of
 * the board to "X"
 * @param board the board
 */
public void userTurn(char[][] board) {
    int row=-1, col=-1;
    // COMPLETE THE CODE
    // Prompt user for row and column, making sure they enter
    //   proper values (by calling method readValue twice).
    System.out.println("\nIt is your turn.");

    // If that location is already taken, you need to ask again.
    // Use a loop for this!

    // Once they have entered a row and column, replace
    // that item in the board with the PLAYER character.
    board[row][col] = PLAYER;
}

/**
 * Utility method to prompt user for a value in the range of 0 to 2
 * It makes sure that the user enters a number between 0 and 2.
 * If not, it loops and asks for another input.
 *
 * @return the value of the row or column
 */
public int readValue() {
    int val = -1;
    while (true) {
        val = in.nextInt();
        if (val < 0 || val > 2) {
            System.out.println("You have not entered a value in the "
```

```
                    + "range 0 to 2. Please try again.");
            } else {
                break;
            }
        }
        return val;
    }

    /**
     * Checks to see if the currentPlayer has won.
     *
     * @param board         the board
     * @param currentPlayer the currentPlayer
     * @return true if the currentPlayer has won.
     */
    public boolean checkWin(char[][] board, char currentPlayer) {
        // the winning sequence:
        String seqWin = "" + currentPlayer + currentPlayer + currentPlayer;
        boolean hasWon = false;
        String seq = "";

        // here is how you might check the rows
        for (int i = 0; i < board.length; i++) {
            seq = "";
            for (int j = 0; j < board.length; j++) {
                seq += board[i][j];
            }
            if (seq.equals(seqWin)) {
                hasWon = true;
            }
        }

        // check cols - ADD YOUR CODE GOES HERE

        // check diags - ADD YOUR CODE GOES HERE


        // Say who has won:
        if (hasWon) {
            if (currentPlayer == PLAYER) {
                System.out.println("Congratulations - you have won!");
            } else {
                System.out.println("Sorry, the computer won and you lost. "
                        + "\nBetter luck next time");
            }
        }
        return hasWon;
    }
```

```
    /**
     * Print the board. You should have a heading to show the column
     * indices. Each row should have a row index.
     * @param board
     */
    public void printBoard(char[][] board) {
        // YOUR CODE GOES HERE
    }
}
```

**Sample Output**

```
WELCOME TO TICTACTOE

You will be X, the computer will be O.
The starting player will be picked randomly.
Enjoy the game and good luck!

You are X and the computer is O.
You are the first player.
The starting board is:
     0    1    2
--------------------------
0 |  -    -    -
1 |  -    -    -
2 |  -    -    -

It is your turn.
Enter the row (0-2) and column (0-2) of your move: 1 1
     0    1    2
--------------------------
0 |  -    -    -
1 |  -    X    -
2 |  -    -    -

It is the computer's turn:
     0    1    2
--------------------------
0 |  O    -    -
1 |  -    X    -
2 |  -    -    -

It is your turn.
Enter the row (0-2) and column (0-2) of your move: 0 0
This space is not empty. Try again.
Enter the row (0-2) and column (0-2) of your move: 2 2
```

```
     0    1    2
--------------------------
0 |  O    -    -
1 |  -    X    -
2 |  -    -    X

It is the computer's turn:
     0    1    2
--------------------------
0 |  O    O    -
1 |  -    X    -
2 |  -    -    X

It is your turn.
Enter the row (0-2) and column (0-2) of your move: 0 2
     0    1    2
--------------------------
0 |  O    O    X
1 |  -    X    -
2 |  -    -    X

It is the computer's turn:
     0    1    2
--------------------------
0 |  O    O    X
1 |  O    X    -
2 |  -    -    X

It is your turn.
Enter the row (0-2) and column (0-2) of your move: 1 2
     0    1    2
--------------------------
0 |  O    O    X
1 |  O    X    X
2 |  -    -    X
Congratulations - you have won!
```