Assignment 7
2D Arrays & For Loops

CEC – Computer Science
Instructor: M. Wolverton

Items Due

- **Source Code for Tic-Tac-Toe**

Instructions

Complete the coding directive in an Intellij IDEA project with the assignment name (e.g. Tic-Tac-Toe) and package name ceccs. Login to the file server at cec-code-lab.aps.edu and create a folder with the assignment name (e.g. Assignment 7). Locate the `fileName.java` (e.g. `Main.java`) files of your source code, then upload them to into the assignment folder. Code will be downloaded and archived for grading on the assignment due date.

Be careful to look at the specific requirements for each program!

Tic-Tac-Toe

Write a command line version of the game Tic-Tac-Toe. Leverage methods, 2d arrays and for-loops to help organize your code. Program the game with the intent that it will be played by two human players. A computer player may be optionally added later.

Additional criteria:

- Have the players use the keyboard numpad coordinates to select a play location.
- Game data should be stored in a 2d char array.
- Use a pair of nested for( ) loops for all operations that need to access the whole game data array. For instance initialization.
- Create a method to print the game board with the following returns and parameters:

```
static void printGameGrid(char[3][3] gameData) {
       // logic to print out the game board in a visually appealing way
}
```

- Create a method to check for win and draw states. It should return a char with the following return codes:

```
static char checkWin(char[3][3] gameData) {
       // logic to look for win states or draw.
       // returns ‘X’ if X wins, ‘O’ if O wins, ‘D’ if draw and ‘N’ for none.
}
```

- Ask players If they want to keep playing at the end of each round
- X plays first, but the losing player should always play X’s the following round. For a draw, players switch their marks.
- Print the score count (P1 wins, P2 wins, Draws) at the end of each round.

Sample Output

```
[[ Tic-Tac-Toe ]]
Player 1 (X): use Numpad (1-9) to select a location.
         - | - | -
        -----------
         - | - | -
        -----------
         - | - | -
3
         - | - | -
        -----------
         - | - | -
        -----------
         - | - | X
...
```

Optional Challenges

- Create a computer opponent that makes random moves.
- Create a computer opponent that makes strategic moves.
- Create a difficulty setting menu (easy, normal, hard).