Multidimensional Arrays Practice
Let’s make a game

- Tic Tac Toe!
Tic Tac Toe

• How do we start?
Tic Tac Toe

• Pseudocode framework
Tic Tac Toe

// Set up board and variables

// Game loop

// Show board
// Ask player for move
// Try the move

// If it works
// Check for a winner
// If there is a winner
// End game
// Switch players
// If it doesn’t work, ask player for move, etc...
Tic Tac Toe

• Variables:

```java
// Set up board
char[][] board = {
    {' ', ' ', ' '},
    {' ', ' ', ' '},
    {' ', ' ', ' '}
};

Scanner input = new Scanner(System.in);
char player = 'X';
```
Tic Tac Toe

• game loop

```java
// Game loop
while(true){

  // Show the board
  showBoard(board);

  // Ask player for move
  System.out.println("Please enter your move by row and column (ex: 1 1): ");
  int row = input.nextInt();
  int column = input.nextInt();

  if (tryMove(row, column, board, player)){
    // Check for winner
    char winner = checkForWinner(board);

    if (winner != ' '){
      System.out.println(winner + "'s win!");
      showBoard(board);
      break;
    } else {
      // Switch players
      if (player == 'X'){
        player = 'O';
      } else {
        player = 'X';
      }
    }
  }
}
```
Tic Tac Toe

• Methods:

    public static void showBoard(char[][] board) {}

    public static boolean tryMove(int row, int column, char[][] board, char player) {}

    public static char checkForWinner(char[][] board) { }
Tic Tac Toe

- showBoard

```java
public static void showBoard(char[][] board){
    System.out.println("\n" +"------------");
    for (int row = 0; row < board.length; row++){
        System.out.print("| ");
        for (int column = 0; column < board[row].length; column++){
            System.out.print(board[row][column] + " | ");
        }
        System.out.println("\n" +"------------");
    }
}
```
public static boolean tryMove(int row, int column, char[][] board, char player) {
    boolean wasSuccessful = false;

    if (board[row][column] == ' '){
        board[row][column] = player;
        wasSuccessful = true;
    } else {
        System.out.println("That spot is already taken");
        wasSuccessful = false;
    }

    return wasSuccessful;
}
public static char checkForWinner(char[][] board){

    // Check horizontal winning combos
    for (int row = 0; row < board.length; row++){
        if (board[row][0] == board[row][1] && board[row][0] == board[row][2]){
            return board[row][0];
        }
    }

    // Check vertical winning combos
    for (int column = 0; column < board[0].length; column++){
        if (board[0][column] == board[1][column] && board[0][column] == board[2][column]){
            return board[0][column];
        }
    }

    // Check diagonal winning combos
    if (board[0][0] == board[1][1] && board[0][0] == board[2][2]){
        return board[0][0];
    }

    if (board[0][2] == board[1][1] && board[0][2] == board[2][0]){
        return board[0][0];
    }

    return ' ';}