Multidimensional Arrays Practice
Let’s make a game!
Let’s make a game!

- Connect Four!
Connect Four

• How do we start?
Connect Four

- Pseudocode framework
Connect Four

// Set up board and variables
// Fill the board with empty characters

// 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…
Connect Four

Variables:

```java
// Set up our game board
char[ ][ ] board = new char[6][7];
Scanner input = new Scanner(System.in);
char player = 'X';
```
Connect Four

• Methods:

```java
public static char[][] fillBoard(char[][] board, char myChar) {}

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

public static boolean tryDropPiece(char[][] board, int col, char player) {}

public static boolean checkForWin(char[][] board) {}

public static char switchPlayer(char currentPlayer) {} 
```
Connect Four

• Fill the game board with an empty character (□):

```java
public static char[][] fillBoard(char[][] board, char myChar)
{
    for (int row = 0; row < board.length; row++){
        java.util.Arrays.fill(board[row], 0, board[row].length, myChar);
    }
    return board;
}
```

• Call method in main method:

```java
// Fill with empty character
board = fillBoard(board, '□');
```
Connect Four

- Show the game board:

```java
public static void showGameBoard(char[][] board){
    System.out.println();
    for (int row = 0; row < board.length; row++){
        System.out.print("|");
        for (int col = 0; col < board[row].length; col++){
            System.out.print(" "+ board[row][col] + "|");
        }
        System.out.println();
    }
}
```
Connect Four

- Try to drop a game piece:

```java
public static boolean tryDropPiece(char[][] board, int col, char player) {
    boolean result = false;

    // Check if the column is full
    if (board[0][col] != '□') {
        System.out.println("That column is already full.");
        return false;
    }

    // Drop the piece as far as it will go. Find the first empty space,
    // starting from the bottom
    for (int row = board.length - 1; row >= 0; row--) {
        if (board[row][col] == '□') {
            board[row][col] = player;
            return true;
        }
    }

    return result;
}
```
Connect Four

• Check for win:
  • How many ways can a player win?
Connect Four

- Check for win (horizontal):

```java
// Check for win horizontally
for (int row = 0; row < board.length; row++){
    for (int col = 0; col < board[row].length - 3; col++){
        if (board[row][col] != '□'
            && board[row][col] == board[row][col+1]
            && board[row][col] == board[row][col+2]
            && board[row][col] == board[row][col+3]){ return true; }
    }
}
```
Connect Four

- Check for win (vertically):

```java
// Check for win vertically
for (int col = 0; col < board[0].length; col++) {
    for (int row = 0; row < board.length - 3; row++) {
        if (board[row][col] != '□'
           && board[row][col] == board[row+1][col]
           && board[row][col] == board[row+2][col]
           && board[row][col] == board[row+3][col]) {
            return true;
        }
    }
}
```
Connect Four

- Check for win (diagonally - first direction):

```java
// Check for win diagonally, from top left
for (int row = 0; row < board.length - 3; row++){
    for (int col = 0; col < board[row].length - 3; col++){
        if (board[row][col] != '□'
            && board[row][col] == board[row+1][col+1]
            && board[row][col] == board[row+2][col+2]
            && board[row][col] == board[row+3][col+3]){
            return true;
        }
    }
}
```
Connect Four

- Check for win (diagonally - second direction):

```java
// Check for win diagonally, from top right
for (int row = 0; row < board.length - 3; row++){
    for (int col = 3; col < board[row].length; col++){
        if (board[row][col] != '□'
            && board[row][col] == board[row+1][col-1]
            && board[row][col] == board[row+2][col-2]
            && board[row][col] == board[row+3][col-3]){
            return true;
        }
    }
}
return result;
```
Connect Four

- Switch player:

```java
public static char switchPlayer(char currentPlayer) {
    if (currentPlayer == 'X') {
        return 'O';
    } else {
        return 'X';
    }
}
```
Connect Four

- Put it all together in the game loop:

```java
while (true){
    // Show the state of the board
    showGameBoard(board);

    // Ask player for move
    System.out.print("Player " + player + ", please enter the column where you'd like to drop your piece: ");
    int col = input.nextInt();

    if (tryDropPiece(board, col, player)){ // Try move
        if (checkForWin(board)){ // Check for winner
            System.out.println("Player " + player + " wins!");
            showGameBoard(board);
            return; // End game
        }
    }
    player = switchPlayer(player); // Switch players
}
```