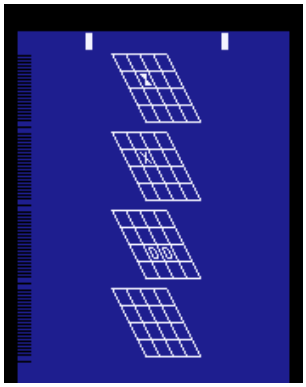
The College of Informatics
Northern Kentucky University
NKU Summer Programming Workshop 2015



Project 6: Tic Tac Toe

On the website are two skeletons of the Tic-Tac-Toe game. One uses a 1-D array ([9]) and the other uses a 2-D array [3][3]. Your task is to finish either one of these (the program with [9] is recommended) by replacing the comments with the proper code. This will require code for each of the following tasks:

1. Print the board

2. Input user's choice, make sure it is a valid choice
   Input the location (0-8 or 0-2, 0-2 – you could also input 1-9/1-3, 1-3 and subtract 1). Make sure that the value entered is legal (within range) and that the given location in the grid is free. We will use a char array so a free space will store the value ' ' while a taken space will store either 'X' or 'O'.

3. Check for winning move
   You will have to see if all 3 squares in the same row, column or diagonal, have the same entry ('O' or 'X') but not blank. For instance, you might have:
   `if(grid[0][0] = = grid[0][1] && grid[0][1] = = grid[0][2] && grid[0][0] != ' ')`
   `or if(grid[0] = = grid[1] && grid[1] = = grid[2] && grid[0] != ' ')`
   Both sets of code check the first row to make sure that the 3 squares are the same character and not ' ' (which means no one has placed an 'X' or 'O' there yet).

4. Check for a draw
   A draw happens if no one has won and 9 moves have been made.

5. If no one has won or the game is not a draw, repeat 1-4 for the other player

6. At the end of the game, output the result
   Output the final board and output who won or whether it was a draw.

If you complete the game, here are several enhancements you can try.

1. Replace the second human player with the computer. You can use one of these strategies:
   a. Random selection (remember that you have to check to make sure the selection is legal)
   b. Check to see if you can win in this move and if so, take it, else check to see if you can block the human from winning and if so, take it, else random selection
   c. Same as 2 but instead of random selection, check to see if middle is taken and if not, take it, else check to see if one of the 4 corners is available and if any are available take it, finally randomly pick one of the other squares
2. Expand the game to be a 4x4 grid or 3 dimensions (this would require a 3-D array where each dimension is 0-2 (or 0-3 if you want to try a 4x4 grid like the image above).
3. Instead of printing the grid out in text, use a PaintComponent method and draw the grid. Ask for help if you want to try this.