Language Study: Erlang

CMSC 233

-Midterm Project

Goals

To continue bask in the glory that is recursion whilst enjoying the challenge of developing a "perfect" Tic-Tac-Toe game. "Perfect" in this context means that the computer should never lose; it should always be able to tie the human player or win.

Instructions

Use a divide and conquer approach here, much like we do when applying recursion to solve a problem. Break the task into small parts and solve those parts individually. Here's some advice on doing that:

Part One

- Represent the game with Erlang functions and constructs
- Use a list of integer values to represent X and O and blank.
- Be able to address the board positions in the list.
- Write functions to get values from the board.

Part Two

- Develop a natural interface to the game. Write a ttt() or play() (or whatever you want to name it) function that launches the game.
- The computer should:
  - display the board
  - prompt the user for the X player's move
  - make the move
  - display the board
  - determine the O (computer's) move
  - make the move
  - display the board
  - repeat until the end of the game

Part Three

- Check all input for range validity.
- Check that the user is not moving into an already occupied space.

Part Four

- Check for a winner. You should probably do this after every move.
- Detect ties as soon as possible. You can certainly tell if it's a tie after move nine because the board will be full and there will be no winner. You can detect a tie after move eight in most cases. There are some cases where you can detect a tie after move seven or perhaps sooner.

Submitting

Print out . . .

- your source code
- a transcript of two successful runs with expected data
- a transcript of two successful runs with unexpected data

. . . and **staple it all together** and hand it in at the start of the class in which it is due. Remember to include your name.