Branwin DuBose & Josh Echols

CS 401 Programming Languages

Assignment 4- Erlang

Statement of Work:

We are asked to write two Erlang program. One program computes the value of pi over N iterations and X actors. This requires a pi:montecarlo(N, X) function to be made. The other Erlang program we need to write allows to players to play tic tac toe and send messages to each other.

Description of Design:

For calculating pi using montecarlo method, we visited the link provided in the assignment report. That link pretty much explains to us what the monte carlo method is and also provides some sample code. From that sample code, we created a module pi. In pi there are 2 methods defined, montecarlo/1 and again/2. montecarlo/1 pretty much goes through the calculations of pi. again/2 repeatedly calls the montecarlo/1 method over and over again. To run this portion of the program, the user should start with pi:again(X, Y). X is the amount of darts to be thrown, and Y is the amount of iterations.

For the tic tac toe program, we implement the board as an Erlang map. It starts out empty and gradually get coordinates that are validated in the place\_token/2 function. Drawing this board is handled by the draw\_board/1 function, with a default drawing of “\_” to indicate that the space is not taken yet. place\_token/2 prints out the current board, asks the player which coordinate they want to place their token at, validates the position (is it already occupied? Is it a valid location?) and then return the new board with the new position filled in. check\_winner/1 obviously checks the winner and returns an atom tie, win, or continue. The continue atom is used to indicate that there is no winner or tie yet, and that the game should continue. msg\_center/2 handles the reception of ALL messages (continue game, customer play messages from tell/1) in the program. It will print out or send messages from tell/1. It also handles the gameplay loop of picking a position, telling the opponent, and showing the opponent’s latest position. wait\_opponent/0 and connect\_opponent/1 are the functions used to set up the initial connection between the players. Tell/1, new\_game/0, and play\_with/1 are implemented per the assignment requirements and pretty basic.

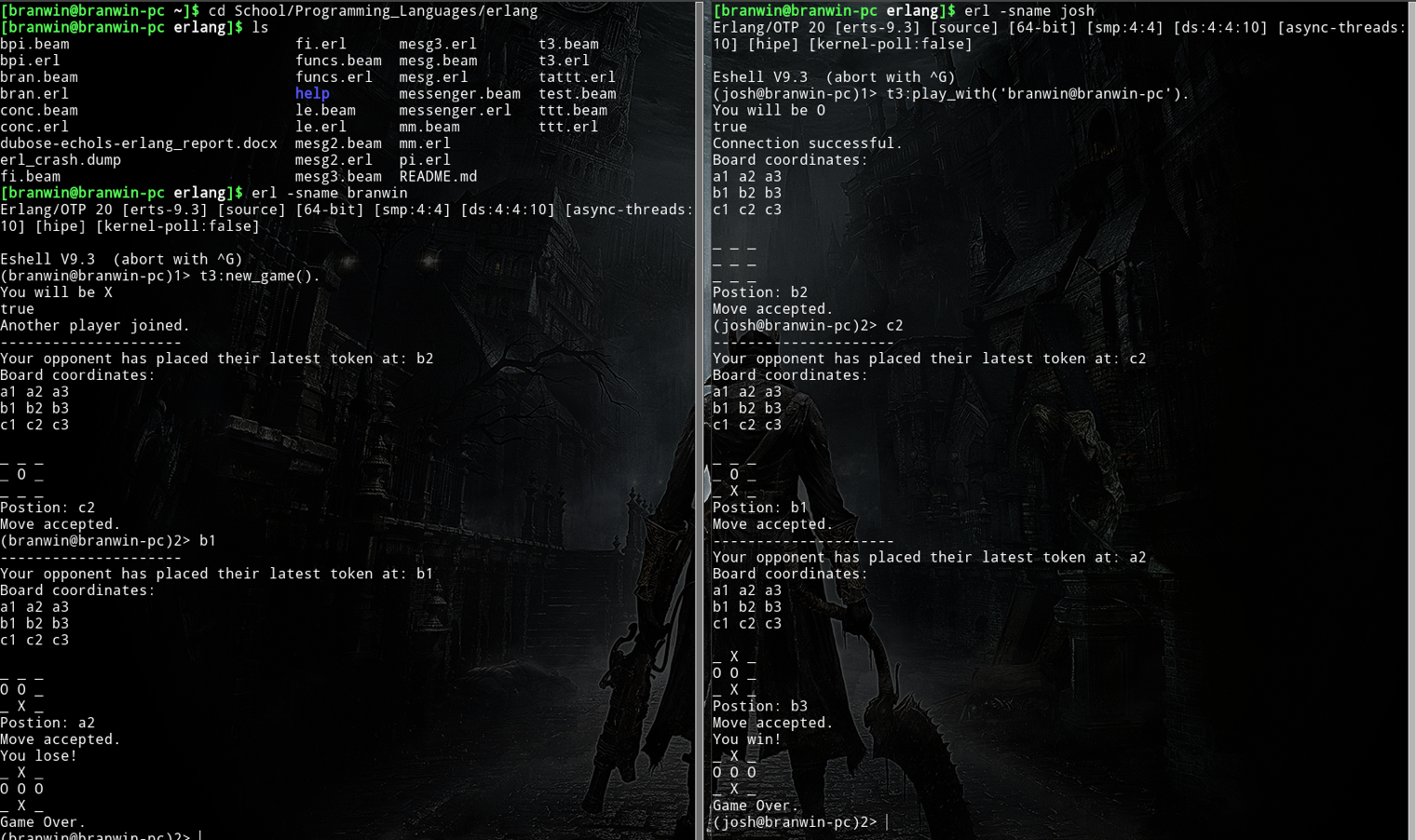
Difficult: The difficult part of Erlang was obviously getting the different spawned processes to run concurrently.

Liked about Erlang: Matching in Erlang proved to be very useful when finding the winner of the tic-tac-toe game.

Disliked about Erlang: We disliked nothing about Erlang.

Test Cases:

Tic – Tac - Toe game play and winning





Tic- Tac- Toe Communication between players



Montecarlo

