1 Project 5

Due Date: 5/17 before midnight. **No late submissions**

Important Reminder: As per the course *Academic Honesty Statement*, cheating of any kind will minimally result in your letter grade for the entire course being reduced by one level.

1.1 Aims of This Project

The aims of this project are as follows:

- To give you an introduction to programming in Erlang.
- To expose you to concurrent programming.

1.2 Project Specification

Update your github repository with a directory submit/prj5-sol to contain the following files:

`fn_server.erl` This file should implement code for a `fn_server` Erlang module which supports evaluating a fixed function with arbitrary arguments. Specifically, it should export the following functions:

`fn_server:start(Fn)` Start an Erlang server to compute function `Fn` and return its PID. The server should maintain `Fn` as part of its state.

`fn_server:compute(ServerPid, Args)` Use the server identified by PID `ServerPid` to compute and return the value of applying the associated function `Fn` to the arguments specified by the list `Args`.

`fn_server:stop(ServerPid)` Stop the server identified by PID `ServerPid`.

`rand.erl` This file should implement code for a `rand` Erlang module which supports the generation of random integers. Specifically, it should export the following functions:

`rand:start(Seed)` Start an Erlang server to generate random integers and return its PID. The server should maintain the current `Seed` as part of its state.

`rand:rand(ServerPid)` Use the server identified by PID `ServerPid` to generate and return the next random number using the provided `next_rand()` function. The `Seed` stored in the server should be set to the returned value.

`rand:stop(ServerPid)` Stop the server identified by PID `ServerPid`.

The file LOG provides an annotated sample log of the operation of these functions.

1.3 Provided Files

The prj5-sol directory contains the following:

fn_server.erl A starting point for the `fn-server.erl` you are required to complete.

rand.erl A starting point for the `rand.erl` you are required to complete. This file provides the `next_rand()` function which can be used to generate the next random number.

fns.erl Functions which can be used for testing the function server.

README A template README; replace the `XXX` with your name, B-number and email. You may add any other information you believe is relevant to your project submission. In particular, you should document the data-structure used for your word-store.

1.4 Hints

- The amount of code you need to write for this project is well under 50 lines.
- The function server is very similar to the data server covered in the lab.
- To apply a function `Fn` to some list of arguments `Args`, use Erlang's `apply¬(Fn, Args)`.
- The random number generation server needs to maintain the **current** seed as part of its state. This can easily be achieved by having the current seed as an argument to the server function.
- Use `io:format()` when debugging.