CSC 435 Spring 2014

Assignment 6 - Erlang

Due: Tuesday, April 15, 2014 by 11:59 p.m.
Grade: 50 points

Objective:
The objective of this assignment is for students to get a deeper understanding of the differences between implementations of the functional programming paradigm by exploring a new language, Erlang.

Requirements:
You may discuss this assignment in class as it is being done, but you will complete and submit this assignment individually. You are also encouraged to ask questions on Piazza.

For each programming part of this assignment, create a new *.erl* file named as specified. Create a script file to demonstrate that your solution for each programming problem works as per the specifications. You may choose to have separate script files for each problem.

The written parts of the assignment should all be submitted in a single document.

See the **Deliverables** section below for more details on submission.

Grade:

Part 1 - Erlang Basics. (10 points)

Part 2 - Research Questions. (15 points)

Part 3 - Simple Modules. (15 points)

Part 4 - Recursion. (10 points)

Deliverables:
Submit on Canvas in the "Assignment 6" category, a zip file containing:

- All source code files.
- Script file(s) showing the correct loading and execution of your Erlang code.
- The written responses in a doc, docx or pdf file.
- A `readme.txt` (text) file detailing the programs that do and do not function properly (organized by Part).

Details:
Part 1 - Erlang Basics (10 points)

1. Explain the Erlang type system.
2. References cannot be defined with words starting in lowercase letters. Explain why.
3. What is the difference between a `list` and a `tuple` in Erlang? Give an example of when you would use each.
4. Explain the differences between the following built-in functions: `foreach`, `filter`, `map` and `fold`. Write out the syntax for each function in BNF or EBNF.

CSC 435 Spring 2014

Part 2 - Research Questions (15 points)

1. Erlang is known for its mantra "Let it crash." What does this mean? How is Erlang designed to prefer this method of maintenance?
2. List three examples of the types of systems that are built using Erlang. Do you believe that Erlang is a good choice for these systems? Why or why not?
3. Erlang is gaining popularity due to its capacity to create incredibly reliable, concurrent systems. However, Erlang takes a different approach to concurrency, opting to use lightweight processes over the threads used by the languages like Java and C. Discuss the benefits of this philosophy.
4. Erlang is optimized for tail recursion. What is the advantage of this?
5. Choose another functional language that we have studied and identify three differences from, and three similarities with Erlang.

Part 3 - Simple Modules (15 points)

1. Copy the following code into a file called **`calculator.erl`**. Write a new function called **`exp`** that takes two arguments, a number and a magnitude, and computes the value. Use guard statements for the cases of positive and negative magnitudes.

```
-module(calculator).
-export([add/2, subtract/2, multiply/2, divide/2]).
add(A,B) -> A+B.
subtract(A,B) -> A-B.
multiply(A,B) -> A*B.
divide(A,B) -> A/B.
```

   Example output:

```
calculator:exp(3,6). returns 729
calculator:exp(4,0). returns 1
calculator:exp(5,1). returns 5.
```

2. Create a new Erlang file named **`basic_module.erl`**.
   a. Write a function called **`light_mixing`** that takes in 3 variables, which represent the 3 primary colors of light (Red, Blue, and Green). The user will enter 3 integers into the function and the function will return the resulting color.

      For example: (1,0,1) will return Yellow, and (1,1,1) will return White.

      Recreate the color wheel with the 3 primary colors and 4 combination colors Yellow, Magenta, Cyan, and White.

   b. Add a function called **`palindrome`** that takes in a string of characters and uses recursion to determine whether that string is a palindrome. The function should be non-case-sensitive.

      For example: "Racecar" will return true, and "baseball" will return false.

CSC 435 Spring 2014

Part 4 - Recursion (10 points)

Create a new Erlang file called **`recursion.erl`**.

Write a function called **`pascal`** that takes in the number of lines and displays that many lines of the Pascal's Triangle in a file called "**`Pascal.txt`**".

The rules for Pascal's Triangle are:

a. The first row is '1'.

b. The second row is '1 1'.

c. Any row down has one more element than the previous row.

d. Any number in a row under two elements is the sum of those elements.

e. Any number not under two elements is a '1'.

For example, if the user inputs 11, the Triangle displayed is:

```
                    1
                  1   1
                1   2   1
              1   3   3   1
            1   4   6   4   1
          1   5   10  10  5   1
        1   6   15  20  15  6   1
      1   7   21  35  35  21  7   1
    1   8   28  56  70  56  28  8   1
  1   9   36  84  126 126 84  36  9   1
1   10  45  120 210 252 210 120 45  10  1
```