CS14 – Homework 4

Description:

In this assignment, you will implement another simple calculator. This calculator will be based on trees instead of stacks.

Specifications:

Your program will prompt the user to enter a formula with a single ">" character. Formulas use the "LISP notation." That means that the formula is a tree structure represented with parenthesis. The first "token" inside the parenthesis is the operator we will apply. You must implement +, -, /, and *. The remainder of the things in parenthesis are what the arguments. There must be at least 1 argument, but there is no limit on the number of arguments. You will first parse this equation into a tree structure called an "abstract syntax tree," then you will evaluate the tree and output the answer to the screen.

You are free to design the internals of this program however you would like. It is probably a good idea to create a class that represents the syntax tree. It should have a function to create the tree from a string, and a function to evaluate the tree. Both of these functions should be recursive.

Example Outputs:

```
> (+ 1 2 3)
6
> (+ (+ 1 2) 3)
6
> (* 2 2 3)
12
> (+ (* 1 2) 3)
5
> (- 1 2)
0.5
> (- 1 2 3)
-4
> (/ 1 2)
0.5
> (/ 1 2 2)
0.25
> (+ (* 2 3) (* 2 3 4))
30
> 20
20
> (+ 1 2
Error: Invalid tree
> (+ 1)
1
> (1)
1
```

```
> (1 1)
Error: 1 is not a function
> (+)
Error: + function requires at least one argument
> (happy 2)
Error: happy is not a function
```

Extra credit:

It is possible to use this tree notation to implement an entire programming language. You will get up to 5 pts of extra credit if you include support for if statements in your calculator. To do this, first you must support comparison operations. To get the extra credit, there are three comparisons that you should support: greater, less, and equals. These functions will compare two numbers, and return 1 for true and 0 for false.

For example:

```
> (equals 5 5)
1
> (equals 5 4)
0
> (greater 5 4)
1
> (greater 4 5)
0
> (less 5 4)
0
> (less 4 5)
1
```

Then, you must implement the actual if statement. The first token in the parenthesis should be the word "if." The second token is the condition. If the condition evaluates to 1 (i.e. true), then the if statement is equivalent to the third statement. If the condition evaluates to 0 (i.e. false), then the if statement is equivalent to the fourth statement.

```
> (if 1 4 5)
4
> (if 0 4 5)
5
> (if (equals 4 5) 1 2)
2
> (if (less 4 5) 1 2)
1
> (if 0 1)
Error: if statement requires exactly 3 parameters
> (if 0 1 2 3)
Error: if statement requires exactly 3 parameters
```

Submission instructions: You should submit a single tarball containing a Makefile and your source code. Your submission should contain nothing else.

In the submission box on ilearn, you should specify whether you did the extra credit or not.

File headers: Every file you submit must begin with the following information.

```
// Course: CS 14 Spring 2013
//
// First name: <<INSERT>>
// Last name: <<INSERT>>
// Course username: <<INSERT>>
// Email address: <<INSERT>>
//
// Lecture section: <<INSERT>>
// Lab section: <<INSERT>>
// TA: <<INSERT>>
//
// Assignment: <<INSERT>>
//
// I hereby certify that the code in this file
// is ENTIRELY my own original work.
//=============================================================
```