Programming Assignment #2: Scheme

Due at the beginning of the class on Friday, November 14.

In order to complete this assignment, you must download and install MIT/GNU Scheme **(version 7.7.1)** on your own personal machine. MIT/GNU Scheme is free software, and can be downloaded from http://www.gnu.org/software/mit-scheme/ .You may also use any of the Lehigh public site computers. In this case you have to use the install software facility available in Lehigh public sites to install Scheme in your home directory. Please note that if you directly download the software from its website into your home directory, it will not work. There is a FAQ page at http://fp1.cc.lehigh.edu/abq2/faq.html to help you get the system up and running.

You will need to submit both electronic versions of your program file and a hardcopy of your source code.

- You may develop your functions individually but when you turn in your assignment you need to merge them all into one file. If you have named the functions as specified below, there will be no problem with putting them in a single file. Include a comment at the beginning of this file that states your name.
- Save your file as **<*yourLastName*>.scm**
- The electronic version of your program file must be submitted by using the course webpage on the Blackboard Learning System (see https://ci.lehigh.edu/). From the CSE 262 page, select Assignments, and then click on "View/Complete". You will then be able to attach your file for submission. Be sure to press the "Submit" button when you are absolutely sure that your code is in its final form. If you do not press this button, your work will not be available for us to grade. **Also, please make sure that you name the Scheme file and your functions exactly as requested.** We will be running scripts to process it and points may be taken off if the script fails to locate your code.

1) Write a Scheme function `area` that takes the base length and the height of a triangle as two of its parameters and returns the area of the triangle. You may assume that your functions will be tested only with numeric literals. An example call to your function is given below:

```
(area 5 4)
; should return 10
```

2) Write a Scheme function `xor` that takes two parameters that have the value of either #T or #F and performs an "exclusive or" operation on them. Note, the result of an "exclusive or" operation of two propositions is true if and only if exactly one of the propositions is true, otherwise it is false. Some example calls to your function are given below:

```
(xor #T #F)
; should return #T

(xor #T #T)
; should return ()
```

3) Write a Scheme function `pow` that takes two parameters that are numeric literals, and returns the value of the first parameter raised to the power of the second parameter. An example call to your function is given below:

```
(pow 2 3)
; should return 8
```

4) Write a Scheme function `zeroes` that takes a simple list of numeric literals as its only parameter and returns the number of zeroes in the list. An example call to your function is given below:

```
(zeroes '(7 6 0 7 0))
; should return 2
```

5) Write a Scheme function `subst` that takes two atoms and a list as parameters and replaces all occurrences of the first given atom in the list with the second given atom, no matter how deeply the first atom is nested. An example call to your function is given below:

```
(subst 4 3 '(5 4 2 3 4))
;should return the list (5 3 2 3 3)
```