CS 101 Computer Science I (Spring 2001)
Assignment 4

1. Write a deeply recursive Scheme procedure called "**count-occurrences***" that takes two arguments. The first argument **sym** is a symbol. The second argument **exp** is either a symbol or a nested list of symbols. The procedure **count-occurrences*** returns an integer indicating the number of times that **sym** appears anywhere in **exp**.

```
(count-occurrences* 'z  '())              ==>  0
(count-occurrences* 'z  'z)               ==>  1
(count-occurrences* 'z  'x)               ==>  0
(count-occurrences* 'z  '(a z b))         ==>  1
(count-occurrences* 'z  '(a (z) b))       ==>  1
(count-occurrences* 'z  '(a z b z c))     ==>  2
(count-occurrences* 'z  '(a z (b (z)) c)) ==>  2
```

2. Write a deeply recursive Scheme procedure called "**substitute***". This procedure takes three arguments. The first two arguments are a symbol **new** and a symbol **old**. The third argument **exp** is either a symbol or a nested list of symbols. The procedure **substitute*** returns a new expression that results from replacing every occurrence of **old** in **exp** with **new**.

```
(substitute* 'b 'a '())                   ==>   ()
(substitute* 'b 'a 'a)                    ==>   b
(substitute* 'b 'a 'c)                    ==>   c
(substitute* 'c 'd '(a b c d))            ==>   (a b c c)
(substitute* 'c 'd '(a b (c d)))          ==>   (a b (c c))
(substitute* 'b 'a '(c a b a c))          ==>   (c b b b c)
(substitute* 'b 'a '(c (a b (a)) c))      ==>   (c (b b (b)) c)
```

3. Write a deeply recursive Scheme predicate procedure called "**all-greater*?**" that takes two arguments. The first argument **n** is a number. The second argument **exp** is either a number or a nested list of numbers. The procedure **all-greater*?** returns **#t** if every number appearing in **exp** is greater than **n**. Otherwise it returns **#f**.

```
(all-greater*? 3 '())               ==> #t
(all-greater*? 3 4)                 ==> #t
(all-greater*? 3 3)                 ==> #f
(all-greater*? 2 '(4 (3 1) 5))      ==> #f
(all-greater*? 0 '(4 (3 1) 5))      ==> #t
```

Due Dates

- Section 51 (Professor Welty): Monday February 19, 2001
- Section 52 (Professor Ellman): Tuesday February 20, 2001