CS 101 Computer Science I (Spring 2001)
Assignment 3

1. Write a recursive procedure called "**match-all**" that takes two lists **lst1** and **lst2** of equal length as arguments. The procedure match-all returns a new list that results from combining corresponding members of **lst1** and **lst2** as shown below.

```
(match-all '(1 2 3) '(-1 -2 -3))      ==>   ((1 -1) (2 -2) (3 -3))
(match-all '(a b) '(c d))             ==>   ((a c) (b d))
(match-all '() '())                   ==>   ()
```

2. Write a recursive procedure called "**substitute**" that takes three arguments: a symbol **new**, a symbol **old**, and a list of symbols **lst**. The procedure substitute returns a new list that results from replacing every occurrence of **old** with new on **lst**.

```
(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 '())                ==>   ()
```

3. Write a recursive procedure called "**count-occurrences**" that takes two arguments: a symbol **sym** and a list of symbols **lst**. The procedure **count-occurrences** returns an integer indicating the number of times that **sym** occurs on **lst**.

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

4. Write a recursive predicate called "**suffix?**" that takes two arguments, both of which are lists of symbols or numbers. The predicate returns **#t** if the first list is a suffix of the second, and returns **#f** otherwise.

```
(suffix? '() '(foo bar))              ==> #t
(suffix? '(7) '(10 7))                ==> #t
(suffix? '(7) '(7 10))                ==> #f
(suffix? '(c d e)  '(a b c d e))      ==> #t
(suffix? '(x y z)  '(a b c y z))      ==> #f
```

Due Dates

- Section 51 (Professor Welty): Monday February 12, 2001
- Section 52 (Professor Ellman): Tuesday February 13, 2001