COP4020 Programming Assignment 1

1. Write four Scheme functions that take a word (represented by a Scheme atom) as an argument and return either `#t` (=true) or `#f` (=false) depending on the grammatical classification of the word in four categories: determiner, noun, verb, and adjective. The four functions should be named `det?`, `noun?`, `verb?`, and `adj?`. You may assume that the vocabulary is limited to the following words: `a, an, the, apple, car, dog, road, eats, occupies, rides, walks, hairy, hot, red`.

   Save your Scheme functions in the file named `pr1.scm`. Login with ssh to `linprog` and type 'scheme'. Test your Scheme functions from the Scheme command prompt. For example:

   ```
   linprog2> scheme
   MIT/GNU Scheme running under GNU/Linux
   Type `^C' (control-C) followed by `H' to obtain information about inter

   Copyright 2005 Massachusetts Institute of Technology.
   This is free software; see the source for copying conditions.  There is
   warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURP

   Image saved on Sunday March 20, 2005 at 10:14:46 PM
     Release 7.7.90.+ || Microcode 14.15 || Runtime 15.6

   1 ]=> (load ``pr1.scm'')
   ;Loading ``pr1.scm'' -- done
   ;Value: reject
   1 ]=> (det? 'the)
   ;Value: #t
   1 ]=>
   ```

2. Consider the following function:

   ```
   (define \/
    (lambda (a b)
     (if a #t b)
    )
   )
   ```

   What is the output of the following program executed by the Scheme interpreter when entered at the prompt:

   ```
   (reduce \/ (map det? '(the hairy dog eats a red apple)))
   ```

What is the output of the program:

```
(reduce \/ (map det? '(hot red car)))
```

Explain in detail how these programs use the `reduce`, `map`, and `\/` functions to derive the answers.

Note: The implementation of the `reduce` function can be found in the lecture notes on Scheme.

3. Copy the `filter` function from the course notes into your `pr1.scm` file. Write a new function that uses the `filter`, `length`, and `adj?` functions to count the number of adjectives in a sentence. Name your function `adjectives`.

```
1 ]=> (load "pr1")
2 ]=> (adjectives '(a hairy red dog eats a hot dog))
;Value: 3
```

4. Write a function named `reject` that returns `#t` when more than 25% of the words in a sentence are adjectives and `#f` otherwise. For example:

```
1 ]=> (load "pr1")
2 ]=> (reject '(a hairy red dog occupies the hot red car))
;Value: #t
3 ]=> (reject '(a red car rides the road))
;Value: #f
```