MASSACHVSETTS INSTITVTE OF TECHNOLOGY
Department of Electrical Engineering and Computer Science
6.090—Building Programming Experience
IAP 2007

Problem Set 6
Due Friday January 19, 1pm

Special Forms & Procedures

- `(quote` *`expr`*`)` – quote takes one expression, reads it and returns the expression, but does not evaluate it.
- **'** – syntactic sugar for quote. Examples: `'a` returns the symbol `a`, and `'(a b c)` returns a list of the symbols `a`, `b`, and `c`
- **eq?** – test if two symbols are the same: `(eq? 'a 'a)`
- **cond**: `(cond (`*`test1 val1`*`) (`*`test2 val2`*`) ... (else` *`val`*`))`
  First evaluate *test1*. If it is not false, then evaluate *val1* and return it. If *test1* was false, then move on to *test2*. If none of the tests are non-false, then evaluate and return the value of the `else` clause.
- `(map` *`op lst`*`)` - apply *op* to each element of the list, one at a time, and return a list of the resulting values.
- `(filter` *`pred lst`*`)` – apply *pred* to each element of the list, and return a list consisting only of the values for which *pred* returned a non-false value.

Thesaurus

"No, not the dinosaur" you explain to Ben Bitdiddle, "it allows you to look up related words." As usual, Ben Bitdiddle is trying to implement the wrong thing and it's your job to help him out.

Problem 1

A thesaurus is comprised of words and their synonyms. In order to simplify the implementation, we're going to write an abstraction for thesaurus entry, which contains a word and a list of its synonyms. Complete the constructors and selectors for the abstraction:

```
(define (make-entry word synonyms)
  *to-be-completed*)

(define (entry-word entry)
  *to-be-completed*)

(define (entry-synonyms entry)
  *to-be-completed*)

;; example usage:
(define e (make-entry 'victory (list 'conquest 'triumph 'win)))

(entry-word e)
;Value: victory

(entry-synonyms e)
;Value: (conquest triumph win)
```

Problem 2

A thesaurus is a list of entries. Of primary interest are two procedures: adding an entry to the thesaurus and looking a word up in the thesaurus. First, we'll implement adding entries:

```
(define empty-thesaurus null)

(define (add-to-thesaurus entry thesaurus)
  to-be-completed)

(define t
 (add-to-thesaurus
   (make-entry 'ball (list 'globe 'orb 'rondure 'sphere))
   empty-thesaurus))

t
;Value: ((ball (globe ord rondure sphere)))
```

Once you've gotten a working `add-to-thesaurus`, download and evaluate `hw6def.scm`. This will give you two thesauri: `simple-thesaurus` and `full-thesaurus`. The simple thesaurus is for simple testing, and the full version is for serious testing.

Problem 3

In order to look a word up in the thesaurus, we must search through the entries until we find one that matches. You should use `eq?` to compare symbols. Remember to respect the entry abstraction. If the word is in the thesaurus, return the matching entry. If the word is not in the thesaurus, return false.

```
;returns an entry
(define (lookup word thesaurus)
  *to-be-completed*)

(lookup 'victory simple-thesaurus)
```

```
;Value: (victory (conquest triumph win))
(lookup 'defeat simple-thesaurus)
;Value: #f
```

Problem 4

Supposing that you are given the procedure `pick-random`, which randomly picks an element of a list:

```
(define (pick-random lst)
  (if (null? lst)
      null
      (list-ref lst (random (length lst)))))
```

Write the procedure `transmogrify`, which takes in a word and a thesaurus and returns one of the synonyms of the word, picked at random. If the word is not in the thesaurus, return the original word.

```
(define (transmogrify word thesaurus)
  *to-be-completed*)

(transmogrify 'victory simple-thesaurus)
;Value: triumph
(transmogrify 'defeat simple-thesaurus)
;Value: defeat
```

Problem 5

Now we can use this tool to rewrite sentences to make them sound more linguistically impressive. Assume a sentence is given as a list of symbols.

Write the procedure `transmogrify-sentence` that takes in a sentence as a list of words and returns a new list of words with each of the words transmogrified.

```
(define (transmogrify-sentence sentence thesaurus)
  *to-be-completed*)

(transmogrify-sentence '(I went to the store)
                       simple-thesaurus)
;Value: (I went to the outlet)

(transmogrify-sentence
  '(He hit the ball with the bat to end the game with a grand victory)
  full-thesaurus)
;Value: (He strike the rundure with the bludgeon to end the game
         with a impressive triumph)
```