Cogs 502 Fall 2018
Symbols and Programming Assignment 03 due Oct 24, 8pm

Question 1

Given the list `((A B) (C D) (E F))`,[1]

(a) Write what you would get from it by applying the following in order,[2]

1. `CAR`
2. `CDR CDR`
3. `CAR CDR`
4. `CDR CAR`
5. `CDR CDR CAR`
6. `CDR CAR CDR CDR`

(b) Which sequtences of `CAR`s and `CDR`s would get you `A`, `B` and `F`?

Question 2

Write down what the following expressions evaluate to; work them out before trying on the computer.

(a) `(if (listp '(list 1 2)) 'ok 'not-really)`

(b) `(if (null (nil)) 'vice 'versa)`

(c) `(and (listp (if (> 2 4) (- 2 4) (+ 2 4))) (if (> 2 4) (- 2 4) (+ 2 4)))`

(d) `(or (listp (if (> 2 4) (- 2 4) (+ 2 4))) (if (> 2 4) (- 2 4) (+ 2 4)))`

(e) `(or (and (or 'or) 'and) 'or)`

Question 3

The function `wrap-2` takes a two element list, and wraps each element inside a list:

```
* (wrap-2 '(a (b)))

((A) ((B)))
* (wrap-2 '(a b))

((A) (B))
* (wrap-2 (wrap-2 '(a b)))

(((A)) ((B)))
* (wrap-2 (wrap-2 (wrap-2 '(a b))))

((((A))) (((B))))
*
```

Write two versions of `wrap-2`:

(a) one using `LIST`, `CAR` and `CADR`

(b) the other using `CONS`, `CAR` and `CADR`

Question 4

Define a function `dewrap-2` that takes a two element list and removes the outer parentheses of the lists, *if there are any*. You are free to use `LIST` or `CONS` in constructing your resulting lists.

[1]Touretzky, ex. 2.15.
[2]Do not directly check on the computer, first attempt it on paper.

Question 5

Define a function that takes a list and swaps its first two elements. Return the list itself if it has less than two elements.

Question 6

Define a function that takes two arguments and returns the greater of the two.[3] Use the predicate >= for comparison and return a symbol that indicates error, in case any of the arguments is not a number.

Question 7

Define a function that takes three arguments and returns the greatest of the three. Use the predicate >= for comparison and return a symbol that indicates error, in case any of the arguments is not a number.

Question 8

`APPEND` may be used to concatenate zero or more lists together; try and see how it works. Write a function `AFTER-FIRST` that takes two lists and inserts all the elements in the second list after the first element of the first list. Given `'(A D E)` and `'(B C)`, it should return `(A B C D E)`.

Question 9

The built-in `MEMBER` takes and object and a list and checks whether the object appears in the list or not; discover how it works. Using `MEMBER`, define a function `MY-MEMBER` that behaves as follows:

```
* (my-member 'b '(a b c))

(B IS A MEMBER OF (A B C))
* (my-member 'z '(a b c))

(Z IS NOT A MEMBER OF (A B C))
*
```

Question 10

Using `MEMBER` and `LENGTH`, write a function `ORDER` which gives the order of an item in a list. You can do this by combining `LENGTH` and `MEMBER` in a certain way. It should behave as follows:

```
* (order 'a '(a b c))

1
* (order 'c '(a b c))

3
* (order 'z '(a b c))

NIL
```

[3] Graham (1996), p. 29, ex. 4.