CS396: Principles of Languages – Fall 2017

Homework	Points	Announced	Due
#1	8	Sept-21	Oct-05

Introduction

The purpose of this homework is to familiarize you with functional programming and to exercise writing Scheme functions that manipulate lists.

Instructions

Read the following instructions carefully **before** working on this assignment.

1. This is an **individual** assignment. You may discuss ideas, ask questions or explain things to each other. Nevertheless, you should solve the assignments independently.
2. Submissions via email will not be accepted. The homework should be submitted via BBLearn by the due date.
3. You should submit your own work. Material brought from elsewhere (e.g. the Internet[1], a classmate, submission at a previous offering…) is not acceptable.
4. Submit DrRacket files (.rkt)[2]. One file per question. The file name should be HW1#Q{1-5}.rkt
5. You must name your functions exactly as show here.
6. Add comments to your code.
7. A program with syntax errors will earn **zero** points.
8. You may solve any 4 problems for full credit, or solve 5 problems for two extra credit points.
9. Scheme was polluted with imperative programming constructs. You must avoid using these:
    a. Avoid explicit looping and use recursion instead.
    b. Avoid using set functions such as set!, set-car! and set-cdr!
    c. Avoid defining/using global variables in your code.
    d. Avoid using sequencing constructs such as begin.
    e. Use only *lists* to structure your data.
    f. You can use the display function while developing/debugging your code. However, submitted code should not use display.
10. You may use the functions defined in the provided cs396-lib.rkt, but not any other external library.

Questions

Assume that we have the below lists defined:

```
(define lst1 (list 1 2 3 0 4 5 0 6 2 8 0 1))
(define lst2 (list 1 (list 2 3 0) (list (list 4 5) 0) 6 2 8 0 1))
(define lst3 (list 'Apple 'Banana (list 'Mango 'Peach) (list 'Apricot 'Apple) 'Apple 'Orange))
```

[2 points] Question #1

Write a scheme function that takes a list and an atom as parameters and returns a list identical to its parameter list except with all top-level instances of the given atom deleted. The name of this function should be *deleteall*. It should be in a file with the name HW1#Q1.rkt

```
(deleteall 2 lst1) ➔ (1 3 0 4 5 0 6 8 0 1)
(deleteall 2 lst2) ➔ (1 (2 3 0) ((4 5) 0) 6 8 0 1)
(deleteall 'Apple lst3) ➔ (Banana (Mango Peach) (Apricot Apple) Orange)
```

[1] Unless explicitly told to do so.

[2] No zip files. No .doc files. No .docx files. No .pdf files

[2 points] Question #2

Write a Scheme function that takes a list as a parameter and returns the reverse of this list. The name of this function should be *reverse-list*. It should be in a file with the name HW1#Q2.rkt

(reverse-list lst1) ➔ (1 0 8 2 6 0 5 4 0 3 2 1)
(reverse-list lst2) ➔ (1 0 8 2 6 ((4 5) 0) (2 3 0) 1)
(reverse-list lst3) ➔ (Orange Apple (Apricot Apple) (Mango Peach) Banana Apple)

[2 points] Question #3

Write a Scheme function that takes a list of numbers as a parameter and returns the number of zeros in that list. The name of this function should be *count-zeros*. It should be in a file with the name HW1#Q3.rkt

(count-zeros lst1) ➔ 3
(count-zeros lst2) ➔ 1
(count-zeros lst3) ➔ 0

[2 points] Question #4

Write a Scheme predicate function that takes a list and an atom as parameters. The function should search the input list (including nested subsists to all levels of nesting) and returns #t if the list contains the atom. Otherwise, it returns #f. The name of this function should be *deep-member?*. It should be in a file with the name HW1#Q4.rkt

(deep-member? lst1 2) ➔ #t
(deep-member? lst2 4) ➔ #t
(deep-member? lst3 'Peach) ➔ #t
(deep-member? lst3 'Strawberry) ➔ #f

[2 points] Question #5

Write a scheme function that takes a list as a parameter and returns a list with all atoms in the input list moved to the top-level list. The name of this function should be *steamroller*. It should be in a file with the name HW1#Q5.rkt

(steamroller lst1) ➔ (1 2 3 0 4 5 0 6 2 8 0 1)
(steamroller lst2) ➔ (1 2 3 0 4 5 0 6 2 8 0 1)
(steamroller lst3) ➔ (Apple Banana Mango Peach Apricot Apple Apple Orange)

With best wishes
Dr. Mohamed Elwakil