# LISP Programming Assignment [50 pts] Due Date: Feb. 16, 2015

1. [**10 pts**] Implement a function that returns the minimum and the maximum as a “cons” (see example below) for a multi-level list (drill down to atom level). Check if atoms are numbers. Otherwise, return null or an error message.

|  |  |
| --- | --- |
|  | Example |
| Input list | (4 (2 (5)) ((8) 3)) |
| Result | (2 . 8) |

1. [**10 pts**] Given a multi-level (hierarchical) list, count all the atoms in the list (drill down in case the list element is a list as well; see example below). A **recursive** approach is required.

|  |  |  |
| --- | --- | --- |
|  | Example #1 | Example #2 |
| Input List | (G D G (A G) (B D) D D (A G)) | (A (B (C) D) (E) F G (((H)))) |
| Result | 11 | 8 |

1. [**15 pts**] Given a list as input, implement a function that returns the frequency of each element in a list as a new list of **cons**es (like buckets). See the two examples below:

|  |  |  |
| --- | --- | --- |
|  | Example #1 (simple list) | Example #2 (hierarchical list) |
| Input List | (1 5 5 3 5 2 3 1 1) | (G D G (A G) (B D) D D (A G)) |
| Result | ((1 . 3) (5 . 3) (3 . 2) (2 . 1)) | ((G . 2) (D . 3) ((A G) . 2) ((B D) . 1)) |

*Note*: the first element in each **cons** is the key/element in the original list, while the second (**cdr**) is the key’s frequency/number of occurrences.

* A recursive approach is ***required***.
* The program should work for hierarchical lists as well (see second example above).
* *Hint*: you may have to implement more than one function (break down the problem).

1. [**15 pts**] Same problem as above (#4), but drill down to atomic level to count frequency of individual atoms:

|  |  |  |
| --- | --- | --- |
|  | Example #1 (simple list) – unchanged since elements = atoms | Example #2 (hierarchical list) |
| Input List | (1 5 5 3 5 2 3 1 1) | (G D G (A G) (B D) D D (A G)) |
| Result | ((1 . 3) (5 . 3) (3 . 2) (2 . 1)) | ((G . 4) (D . 4) (A . 2) (B . 1)) |

*Note*: in this case, the “conses” of your resulting list would each have an atomic “car”.

How to submit your homework:

1. All your LISP code for the entire homework assignment should be saved to a file, say “hw1.lisp”
2. Use the “dribble” command once you have completed the implementation of your assignment (entirely). After you start the output to file, load the LISP source file, and execute in order the functions that prove the correct execution of your implementation for each problem. (see below)
3. Email me (mihaela@smu.edu) the LISP source file and the output file.

*Below you can see the CLISP console where I first started saving the console output to a file “hw1.txt” in a “mylisp” subdirectory using the “dribble” command ([1]), then loaded the source code with the “load” command ([2]), executed two functions ([3] and [4]), then closed the output file by typing again “dribble” without arguments ([5]).*

