Programming language HW1 common LISP

Installation

- My environment: Windows 10
- Installed SBCL form here using MSI package for 64-bit Windows
- Excute SBCL with Windows Powershell, using `sbcl --script .\<filename>.lsp`
- The code is uploaded on Github, if any error occurs, you can contact me here.
- Note that `End Of File(EOF)` in Windows can be send by `ctrl+z` instead of `ctrl+d` in Linux.
    - Also, for each EOF signal, you have to press `enter` in Windows due to "blocking IO method"

HW Problems

For each problem, the `.lsp` file(script) is provided.

I just use `(print <function>)` to show the result on terminal. Feel free to modify the parameter to check the result.

Problem 1.1

- filename: prime.lsp
    - method: using recursion to checking every number that smaller than the number input.
    - if the number can be divide only by itself and 1, it\'s a prime number.

Problem 1.2

- filename: palindrome.lsp
    - method: compare the input content with reversed itself, if all equal, it\'s palindrome.

Problem 1.3

- filename: fib.lsp (using `(trace <function>)` to trace how function works)
    - original recursion: Just original recursion call.
    - tail recursion: Add two more arguments, `a` and `b` meaning the zeroth and the first number in the fibonacci series.
      when using the function I write, remember to give `a b` => `0 1` respectively.
    - the tail recursion method is quite similar to iteration method, the number we want is computed through the recursion call

Problem 2

- filename: mergesort.lsp
    - usage: use `sbcl --script .\mergesort.lsp` in command line.
    - the first input number meaning the total items in the list.
    - the following input is the content in the list.
- code explain:
    - first of all, we need a `single` function to judge if a list contains only one atom.
        - the `consp` funcion is used to judge if the input is a `Cons` object
            - if yes, then check if the sequence contains only 1 atom, and return the result.
            - this function will be used for slicing the input sequence

    - then in the mergesort function, use the single and null function to judge if the sequence is sliced.
        - if not, cut in half and recursive call the mergesort function.
        - if sliced, then we can start merge the sequence.
            - the merge function is self provided by common lisp itself.
            - the first argument means their type.
            - the second and the third means the input sequence.
            - the fourth means their comparing method
        - after all merged, return the merged sequence. then the job is done.
- exameple input and output:

  Programming_Language_Homework\HW1_F74056328> sbcl --script .\mergesort.lsp

5

6 5 3 0 4

0 3 4 5 6