CS 256 – Programming Languages and Translators
Assignment 6

- This assignment is due by 1 p.m. on Wednesday April 16, 2014
- This assignment will be worth 5% of your grade
- You are to work on this assignment by yourself

Basic Instructions

The Collatz Conjecture states that if you have a positive integer n and apply the following rules:

- If the number is even, divide it by 2; or
- If the number is odd, multiply it by three and add 1

repeatedly, the result will eventually (but in a finite number of steps) be 1. For example, if you start with $n = 24$, you get

$$24 \rightarrow 12 \rightarrow 6 \rightarrow 3 \rightarrow 10 \rightarrow 5 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1$$

It takes 10 steps to go from 24 to 1 using the specified ruleset.

You must write a program in LISP that contains a function called `longest_collatz` that takes a single number as an argument. For the sake of brevity, we will call this number n. That is a fine variable name. Why not? Your function should find the number `from 1 to n` that takes the most steps to reduce to 1 by applying the Collatz Conjecture. (An easy (but hardly efficient) way to determine the number of steps is to build the actual Collatz sequence, and then determine the `length` of the resulting sequence.) Your function should then return a list structured as follows: `(longest_length starting_number)`

This problem is a generalization of the problem presented at `http://projecteuler.net/problem=14`. You can check to see if your generator works for the specific case presented there by creating an account and checking your answer. A warning, my (admittedly inefficient) code took over a minute to execute for the large number specified.

Submission

With this writeup, I have supplied a runner lisp script. Your code should be contained to a **single** file with a `.lisp` extension, though your file name may be whatever you wish it to be. If your file is named `pe14.lisp`, I should be able to run `gcl -load pe14.lisp -load runner.lisp`. A sample run is included:

```
~/Documents/teaching/assignments/hw6$ gcl -load pe14.lisp -load runner.lisp


(8 3)
(20 9)
(112 27)
```

I will use a modified version of the runner script to generate output for different numbers.

You will use `cssubmit` to submit this assignment. From the directory containing your lisp file with your function definition(s), run

```
cssubmit 256 a 6
```

To submit your assignment to me.

Extra Credit

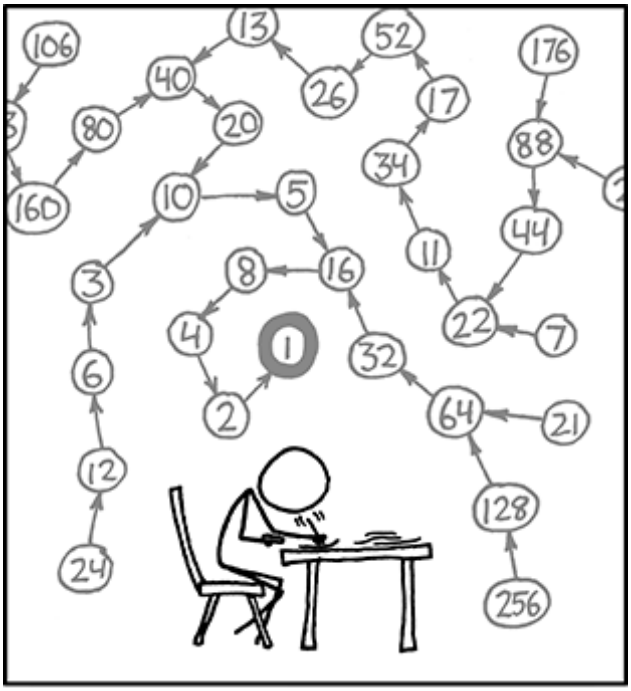
You may choose to implement an additional Project Euler problem in LISP for extra credit. The amount of extra credit depends on the difficulty of the problem chosen.

Any functions used to solve the problem should reside in your lisp file. Your solution to whichever problem you choose to solve should NOT automatically run when `gcl -load <your_file>.lisp` is executed. Instead, you should show me that you have extra credit by including a `README.txt` file that describes:

- Which problem you chose to solve
- How to execute your solution (THIS IS IMPORTANT)
- The answer you obtained when running your solution
- A warning if your code takes a long time to run.

I will add any extra credit you get to the lowest assignment score you have received with the highest weight.

And, because XKCD



THE COLLATZ CONJECTURE STATES THAT IF YOU PICK A NUMBER, AND IF IT'S EVEN DIVIDE IT BY TWO AND IF IT'S ODD MULTIPLY IT BY THREE AND ADD ONE, AND YOU REPEAT THIS PROCEDURE LONG ENOUGH, EVENTUALLY YOUR FRIENDS WILL STOP CALLING TO SEE IF YOU WANT TO HANG OUT.