C SCI 316 (Kong): Lisp Assignment 1 [Homework Exercises: Not for Submission]

Try to do these exercises before the second Lisp class. You should definitely do them before the third Lisp class. You can test your expressions on **venus**.

Before you work on these exercises, logon to **venus** and *enter the following command at* **venus***'s shell prompt* (if you have not already done so): **/home/faculty/ykong/316setup** After doing that, you should be able to start Clisp on **venus** by entering **cl** at the shell prompt. [If this doesn't work, it may be because your login shell is not **/bin/bash**—in that case you might be able to fix the problem by entering the **chsh -s /bin/bash** command to change your shell, logging out, and then logging back in. ***Otherwise, try entering*** **./cl** or **~/cl** instead of **cl** but also ***let me know*** so I can look into the problem.]

Alternatively, you can test your expressions on **euclid**: After entering the command **/home/faculty/ykong/316setup** on **venus** as the previous paragraph asked you to do, activate your **euclid** account by following the instructions on page 6 of the first-day handout. Then you can start Clisp on **euclid** by entering **cl** at **euclid**'s shell prompt. (**Note**: You must activate your **euclid** account no later than **Monday, September 9** *regardless* of whether or not you plan to use **euclid** for these exercises.)

Students who are able to install GNU Clisp on their PCs will also have the option of testing their expressions on their PCs. [Versions of Clisp are available for other systems—see **http://clisp.sourceforge.net** for more information.] A Windows installer for GNU Clisp can be downloaded by pointing your web browser to

https://sourceforge.net/projects/clisp/files/clisp/2.49/clisp-2.49-win32-mingw-big.exe/download

or, equivalently, **https://bit.ly/clisp2-49win32**

After downloading the installer, open it (e.g., by double-clicking on its icon) to install Clisp. The installer creates a desktop icon named **GNU CLISP 2.49**; double-click on that desktop icon whenever you want to run Clisp.

Write solutions to the following problems *on paper*. Then *test your expressions* by entering them at Clisp's > prompt:

1. Write a Lisp expression that multiplies 30 by the result of 7 minus 2. To do this, you must write a call of the function * with two arguments; the second argument will be a call of the function – with two arguments.

2. **[Exercise 1 on page 15 of Wilensky]** Write Lisp expressions to evaluate each of the following:
   (a) $3^2 + 4^2$ (b) 3*17 + 4*19 (c) $12^3+1^3 - (9^3+10^3)$

3. (a) Write a Lisp expression that divides thirty by the result of seven minus three, in such a way that the result is a rational number (i.e., 15/2).
   (b) Write a Lisp expression that divides thirty by the result of seven minus three, in such a way that the result is a floating-point number (i.e., 7.5). Do ***not*** use the FLOAT function.

4. **[Exercise 3 on page 16 of Wilensky]** Write a Lisp expression that computes the mean of the five numbers eighty-three, eighty-three, eighty-five, ninety-one, and ninety-seven. Do this in such a way that the result is rational, and then in such a way that the result is a floating-point number. Do ***not*** use the FLOAT function.

5. Lisp always prints rational numbers (fractions) in *lowest terms*; for example, if you enter the number `20/12` at Clisp's > prompt (with no spaces before or after the "`/`") then Clisp will print the equivalent number `5/3`. Write a number that you can enter at Clisp's > prompt to determine whether or not the two integers 7,360,001,711 and 58,879,948,151 are relatively prime. If they are not relatively prime, then find their greatest common divisor. (Two integers are said to be *relatively prime* or *coprime* if no integer greater than 1 divides both of them. For example, 20 and 9 are relatively prime; but 10 and 8 are not relatively prime because 2 divides both 10 and 8.)

6. **[Exercise 4 on p. 16 of Wilensky]** The Lisp function SQRT returns the non-negative square root of any non-negative real argument. Use this function to write two expressions that are respectively equivalent to the following:

   (a) $\dfrac{-(-11)+\sqrt{(-11)\times(-11)-4\times1\times(-1302)}}{2\times1}$ (b) $\dfrac{-(-11)-\sqrt{(-11)\times(-11)-4\times1\times(-1302)}}{2\times1}$

7. **[Exercise 7 on p. 17 of Wilensky]** Write a Lisp expression that can be entered at Clisp's > prompt to determine the order in which Common Lisp evaluates function arguments when it calls a function. You may assume that the order is either *left-to-right* or *right-to-left*, and that the same order is used in all Common Lisp function calls. [**Hint**: Consider a function call (– <expr>$_1$ <expr>$_2$). To evaluate this call with *left-to-right* argument evaluation order, do the following: First, evaluate <expr>$_1$; next, evaluate <expr>$_2$; finally, subtract the second result (i.e., the value of <expr>$_2$) from the first result (i.e., the value of <expr>$_1$). To evaluate this call with *right-to-left* argument evaluation order, do the following: First, evaluate <expr>$_2$; next, evaluate <expr>$_1$; finally, subtract the first result (i.e., the value of <expr>$_2$) from the second result (i.e., the value of <expr>$_1$). *Normally, both evaluation orders would of course produce the same final value. But if evaluation of one of the two arguments has a side-effect, then evaluation order may make a difference. Try to think of an expression* (– <expr>$_1$ <expr>$_2$) *for which one of the argument evaluation orders would produce an error but the other would not.*]