Introduction to Scheme

Abhinav Ashar
CS 61A: Structure and Interpretation of Computer Programs

January 18, 2019

1 Scheme

Scheme is a functional programming language and is a dialect of the Lisp programming language. It is defined by parenthesis (and a lot of them!) and is structured in such a way that it is easy to interpret for the computer.

a) Defining a variable: (define pi 3.14) → pi = 3.14 in Python

b) Defining a function: (define (pi 3.14)) - the difference between variables and functions in Scheme is the extra pair of parentheses that come with the function

c) When first defining a function name or variable name, you return the name. If you are defining a variable which is assigned to a number that is equal to something that would cause an error, this would cause an error. The same is not true for functions because you do not evaluate the inside of a function until later, just like in Python.

d) The apostrophe means to not evaluate whatever follows it, but rather treat it like a string (kind of)

e) Scheme uses prefix notation, so 2 + 2 in Python translates to (+ 2 2) in Scheme

f) When trying to interpret Scheme, I find it useful to work inside out for each line

g) If Statement: (if <condition> <true result> <false result>)

h) Try to format your Scheme code, especially the if statements, like Python code by doing the if and <condition> in one line, the true result in the next line, and the false result in the last line. It makes reading the code much easier.

i) To make sure that your parentheses match up accordingly, when you draw and opening parenthesis, press enter a few times and put the closing parenthesis directly underneath it. Then continue coding within the space of these two parentheses.

j) The **ONLY** false value in Scheme is #f. Any other values, including 0, are considered true.

k) (quotient ...) is used to do floor division

l) (remainder ...) is the same as mod

m) Just like in Python, Scheme has short circuiting. Remember: With OR, you are looking for the First True or Last False (FTLF). With AND, you are looking for First False or Last True (FFLT).