COMP-335
Bryan Jensen
September 6, 2014

Homework 0

1. (* 1 (* 2 (* 3 (+ 4 (+ 5 6)))))
2. Because, as in the typical linked-list format, the each list is the pointer to a single node. Since each node has two data members, a cargo and a pointer to the next node in the list (and therefore the rest of the list), it makes sense that the majority of the interaction performed on the list, first and rest, would retrieve the cargo (a number) and pointer to the rest of the list (a list), respectively.
3. "second":
    a. (second list) == (first (rest list)), because the "rest" retrieves the list of the second element onwards, and then calling "first" on that list retrieves just the second's "cargo".
    b. (4 5), because it is the second element of a list of various types, much as is possible in Python, and the second element is a simple list of numbers.
4. Since everything in racket is surrounded by parentheses, the order of operation is *very* explicit and therefore there is no confusion as to what operations are performed first. For example:
    a. 3 * 5 + 4 in Python would be fairly straight forward, as is it done in order ((3 * 5) + 4).
    b. But 3 + 5 * 4 would not be, unless one was aware of the order of operations, since it is implicitly (3 + (5 * 4)) and **not** ((3 + 5) * 4)
    c. However, with Racket, every parenthesis is explicitly there, so either can be represented, e.g. (+ 3 (* 5 4)) or (* (+ 3 5) 4), and there is no confusion with implicit rules.