

Hints For LISP Assignments

- Stay with the built-in LISP functions we discussed in class plus those suggested in the Homework Assignment (like “+” and “*”) and the ones here in these Hints. In particular, do not use the built-in LISP functions “append” or “list”. When you are taking apart and building lists, use only “car”, “cdr”, and “cons”. This doesn’t mean you can’t use other arithmetic functions (like “-”) or other predicates (like “zerop”) but don’t use LISP functions that make the problems trivial.
- You should not be using the LISP function “set” for these assignments. In particular, if you are using “set” inside LISP functions, you are probably programming in a PASCAL style, not a LISP style. You will lose points for this. The one exception to this is the last part of Homework 2, but that homework gives you specific directions on using “set” for that particular problem. The “hint” here is that if you are using “set” inside your functions, you do not yet understand how to do this particular style of programming.
- There are loops in LISP. **DON’T USE THEM!** The point of these homeworks is to let you become familiar with a functional style of programming. That means you should use recursion, not iteration. If it helps give you some incentive to learn recursion, you won’t get any credit if you do these problems in a non-recursive fashion.
- Here are two more predicates that may be helpful:
 - “atom” tests to see if a LISP entity is an atom — that is, “(atom A)” returns “T” if A is bound to an atom and “NIL” if it is not (remember the LISP interpreter will evaluate the “A” when you feed it “(atom A)”). One warning: “(atom NIL)” will return “T”.
 - “equal” tests to see if two LISP expressions are equal. So “(equal 6 3)” will return “NIL” and “(equal (quote (A B C)) (quote (A B C)))” will return “T”.

- Here is a shortcut for “quote” (I get tired of typing it too): Instead of typing “(quote A)”, type “A” — that is, “A” preceded by an apostrophe. This works for anything you want to quote, for example “(A B C)” is the same as “(quote (A B C))”.
- LISP makes no distinction between upper and lower case for characters. “foo”, “FOO”, “Foo”, “fOo”, and “fOO” are all the same to the LISP interpreter.
- Comments in Lisp start with a “;” and end with the end of the line. Like this:

```
;
; This is a LISP comment.
;
```

- If you get a LISP structure that looks something like “(A . B)”, this means you did a “cons” where the second parameter was not a list. If you want to see this as an experiment, try “(cons 'A 'B)”.
- If you have additional LISP hints, send them to me and I will add them here.