Lisp handout #1 – 2002 August 21   **Copyright 2002 Michael A. Covington**

Terminology: **Recursion:**   what you have when a data structure or computation contains smaller things of the same type as itself. *(Why smaller?)*

  S-expression: symbolic expression (the things Lisp is made of).

  Lisp: "List Processor," programming language invented by McCarthy in 1958, widely used in AI, still used to some extent.

LISP EVALUATION RULES (version 1, incomplete!)

Type of S-expression	Examples	How evaluated
Number	23 (*integer*) 3.151 (*floating-point*) 2/3 (*rational*)	Itself
Symbol *(Not case sensitive)*	X THIS-OR-THAT	Its value, if a value has been given to it; otherwise impossible to evaluate
Character string	"this is one"	Itself (Seldom used.)

List	(A B C)
	((A B) 234 (C D))
	(+ 2 3)
	(+ (- 5 4) (- 3 2))
		First element must be the name of a function.
		  Evaluate all the subsequent elements,
		  then give those values to the function and
		  compute it.
		If first element is not name of a function, list
		  cannot be evaluated, but can still be used for
		  other purposes.

Quoted expression	'*e* or (QUOTE *e*) where *e* is any expression	The value of '*e* is *e*. That is, the quote blocks evaluation. *Only 1 quote is needed; do you see why?*
Backquoted expression	`*e*	Like quoted expression, but if it is a list, elements within it can be marked for evaluation with commas, thus: `(the answer is ,(+ 2 2)) ➔ (the answer is 4)

SOME USEFUL FUNCTIONS

(+ *a b*)   Addition.   *a* and *b* must be numbers.
(- *a b*)   Subtraction
(* *a b*)   Multiplication
(/ *a b*)   Division   (Result of dividing integers is integer or rational!)

(float *a*)   Converts a number to floating-point, e.g., (float 2/3) ➔ 0.666666667

(eval *e*)   Performs an extra evaluation of *e*