The Functional Paradigm Revisited

- Computation is performed by applying mathematicological functions to values
- A program is a composition of functions
- Functions are first-class values that can be created/passed/returned with complete generality
- Computation is expression-oriented rather than statement-oriented

Scheme Values

- Every value is either an *atom* or a *list*
- There are many categories of atoms:
  - *Symbols*
  - *Numbers*
  - *Booleans*
  - *Characters* and *Strings*
  - *Vectors*
  - *Procedures*
  - Others
- Anything that's not an atom is a list

Atoms: Symbols

- Perhaps the most common category of atom
- Like identifiers or words, but with a more liberal syntax:

```
ab
#ab
this-is-a-symbol
****
-a+b*c
```

(Most blocks of non-whitespace, non-parenthesis characters act as symbols)

Atoms: Numbers

- Includes signed integers, reals, rationals, and complex's
- Again, a pretty liberal syntax:

```
123
-123
123.456
17/3
3+4i
```

(Integers can effectively be arbitrarily large)

Atoms: Booleans

- The usual two values: truth and falsehood
- Simple syntax:

```
#f
#t
```

Atoms: Strings

- The usual: sequences of zero or more characters
- Pretty typical syntax:

```
""

"this is a string"

"a \"quote-containing\" string"
```

Lists

- Defined recursively
- `()` is a list, the *empty list* (aka the *null list*)
- $(x_1\ \ x_2\ \ \ldots\ \ x_n)$ is a list, where each x_j is an atom or a list
- Examples:

```
()
(a b c)
(a (b c))
((a b) c)
(((a) b) c)
```

Lists (cont.)

- We'll need some notation for our examples
- *s*, *t*, *l*, *x*, x_2, *y*

  s, t, l, x, x_2, y

  Italicized, variable-width font letters (possibly subscripted) stand for arbitrary lists or atoms
- `s, t, l, x, x2, y`

  Unitalicized, fixed-width font letters (never subscripted) stand for specific symbols

S-Expressions

- Collectively, the notation used to write down explicit atoms and lists is referred to as *S-expressions* (abbrev *sexpr's*)
- As we'll see, everything about a Scheme program is written as an S-expression, including
  - Program input
  - Program output
  - Program source code