Lists, Part I: A New Datatype
===
For bundling a bunch of values together into one value.
More precisely: a list contains a sequence of zero or more values.

The list “constructor” function ‘list’
---
To create a list from any number of values.
list : anys -> list

(define picturing-programs)

A list with four “elements”, aka “items”, aka “members”.
(list 104 "hello there" true )
That list itself is a single value.

Contrast that with the following four separate values.
104 "hello there" true

To emphasize the difference, this works:
(define our-list (list 104 "hello there" true ))
But this is an error:
(define another-list 104 "hello there" true )

‘list’ is a function: the argument values can be produced in any way.
(list (* 2 52)
 (string-append "hello " "there")
 (= (+ 1 1 1) 3)
 (ellipse 20 40 "solid" "maroon"))

(check-expect our-list
 (list (* 2 52)
  (string-append "hello " "there")
  (= (+ 1 1 1) 3)
  (ellipse 20 40 "solid" "maroon")))

We can put any type of value inside a list, even lists.
(list 123 our-list 456)
But this week we concentrate on lists containing only non-lists.
The Number of Elements in a List
-----------------------------
The unary function ‘length’ counts the number of elements in a list.

\[
\text{length : list -> number}
\]

(check-expect (length our-list) 4)

This one is interesting to wonder about for later in the course:

\[
\text{(length (list 123 our-list 456))}
\]

Order and Repetition
-----------------
Unlike mathematical sets, lists can have repeated elements, and order matters.

There's a binary predicate ‘equal?’ to compare whether two values are equal.

\[
\text{equal? : any any -> boolean}
\]

Both of these produce false:

\[
\text{(equal? (list 123 456 123) (list 123 123 456))}
\]
\[
\text{(equal? (list 123 123 456) (list 123 456))}
\]

Empty List
----------
\[
\text{(list)}; \text{ A list with no elements, prints in Interaction as: empty .}
\]
\[
\text{empty; \text{ There's a variable referring to it.}}
\]
\[
\text{(check-expect (length (list)) 0)}
\]
\[
\text{(check-expect (length empty) 0)}
\]
\[
\text{(check-expect (list) empty)}
\]

When showing us the empty list you may write it in either way:

\[
\text{(list)}
\]
\[
\text{empty}
\]