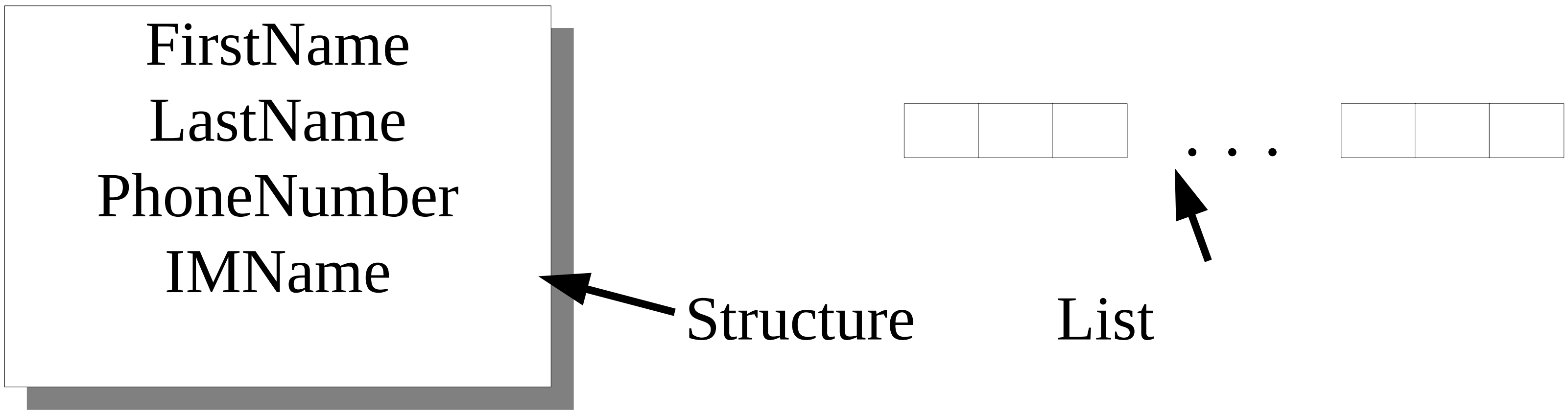
Structures (Ch. 6)

- Scheme supports a number of different ways to organize data into containers:
  - structure: collection of a fixed number of *named* values.
  - list: ordered collection of values (of arbitrary length)



FirstName
LastName
PhoneNumber
IMName

Structure

List

Structures

- Collection of named values (fields).
- Scheme allows you to define your own *type* of structure:
  - need to specify the name of each *field*.
  - new functions are created (automatically) for:
    - creating a new structure (a *constructor* function)
    - extracting the value of a field in the structure (one *selector* function for each field)

A simple structure

- Holds x and y coordinates of a point.
- Two named values: *x* and *y*.
- Name of structure is `posn`
- Constructor: `make-posn`
- Selectors: `posn-x`   `posn-y`

Using the `posn` structure type

```
(make-posn 32 48) ; x is 32, y is 48
;; create new variable named center
(define center (make-posn 10 10))

(posn-x center)
(posn-y (make-posn 17 33))
```

Possibly useful function

```
;; distance-to-0 consumes posn
;;    and produces a number
;; (distance-to-0 p) produces the
;; distance from the point
;; represented by p from the origin.
(define (distance-to-0 p) ... )
; tests
(distance-to-0 (make-posn 0 10))  ; 10
(distance-to-0 (make-posn 3 4))  ; 5
```

Writing `distance-to-0`

- We know the distance is: $\sqrt{x^2+y^2}$

```
(define (distance-to-0 p)
   (sqrt (+ (* (posn-x p) (posn-x p)))))
```

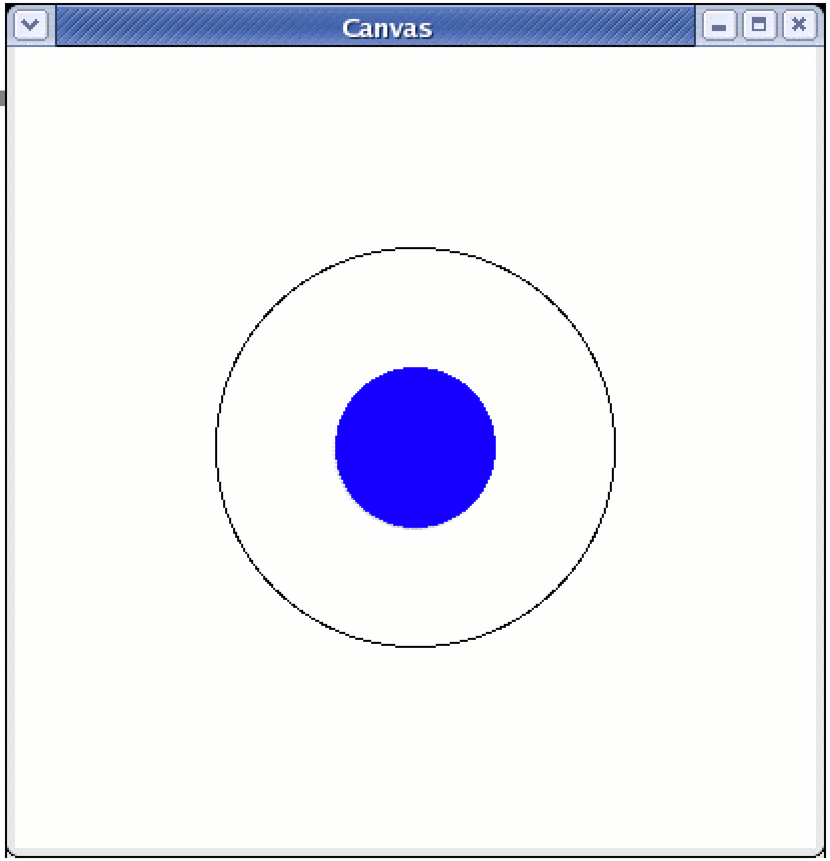
The `posn` structure and `draw.ss`

- A small library of simple graphics functions.
  - Can draw lines, rectangles and circles.
  - Coordinates are provided to these functions as `posn` structures:

```
(draw-line (make-posn 0 0) (make-posn 10 10))
(draw-circle (make-posn 0 0) 10)
(draw-solid-disk (make-posn 0 0) 100 'red)
```

Exercises

- 1. Create this:



- Ex 6.2.1, 6.2.2