Compound Data
Last Lecture

- Booleans
  - Only two elements
- Symbols
  - Not to be confused with variables
- Strings
- Notions of equality on values
Today’s Goals

- Compound data
  - Where/when it’s needed
  - How to declare it
  - How to construct (create) it
  - How to deconstruct it (take it apart)
  - How to design functions that process compound data
  - How to define such functions
Compound Data

- Again, what can we compute the language constructs we’ve seen so far?
- What information is associated with
  - A university course,
  - A car, and
  - A house?
- What makes something an atom or compound?
Built-in Example

- Cartesian coordinates
  - Mathematically, a pair \((x,y)\)
  - DrScheme
    - \((\text{make-posn } 5 \ 6) \rightarrow (\text{make-posn } 5 \ 6)\)
      - Constructors are like "frozen" functions
    - \((\text{define a } (\text{make-posn } 3 \ 4))\)
    - \((\text{posn-x a}) \rightarrow 3\)
    - \((\text{posn-y a}) \rightarrow 4\)

- Can we define functions that compute:
  - distance-to-origin, mirror-image, etc?
Distance-to-origin

- **Contract:**
  - `distance-to-0: posn -> +ve-number`
- **Purpose:** Takes a position and returns distance from the origin.
- **Examples:**
  - `(distance-to-0 (make-posn 0 10)) → 10`
  - `(distance-to-0 (make-posn 10 0)) → 10`
  - `(distance-to-0 (make-posn 3 4 )) → 5`
Distance-to-origin

- Definition:
  - (define (distance-to-0 p)
    (sqrt (+ (square (posn-x p))
           (square (posn-y p))))))

- This function deconstructs its input and return a simple value. More often programs will ?.
Rolling Your Own Compounds

- What if we want 3D coordinates?
- First you declare this concept to DrScheme:
  - `(define-struct my-posn (x y z))`
- Then you can use it:
  - `(define b (make-my-posn 3 4 5))`
  - `(+ (my-posn-x b) (my-posn-y b)) → 7`