Designing Programs

Design recipe

... but at the same time...

Helper functions and reuse

• Writing writing a function, consider whether existing functions help

• Look for functions that you wish you had written
Another Example

Write the function \texttt{bigger-image?} which checks whether one image has more pixels than a second image

\begin{verbatim}
; bigger-image? : image image -> bool
; Returns true if a has more pixels than b
(define (bigger-image? a b)
  (> (* (image-width a) (image-height a))
      (* (image-width b) (image-height b))))

(check-expect (bigger-image? ■ ■) true)
(check-expect (bigger-image? ■ ■) false)
\end{verbatim}
Another Example

Write the function \texttt{bigger-image?} which checks whether one image has more pixels than a second image

\[
; \texttt{bigger-image? : image image -> bool}
; \text{Returns true if a has more pixels than b}
\]
\[
\text{(define (bigger-image? a b)}
\text{)}
\[
\text{ (> (image-size a) (image-size b)))}
\]

\[
\text{(check-expect} \text{ (bigger-image? □ □) true)}
\]
\[
\text{(check-expect} \text{ (bigger-image? □ □) false)}
\]

\textit{Wish list: image-size}

Fullfill wishes by applying the recipe again
\[(\text{exercise for the reader})\]
Reuse

We should be able to use `bigger-image?` to write the `max-image` function

; max-image : image image -> image
; Returns a if a has more pixels than b, otherwise returns b
(define (max-image a b)
  (cond
   [(bigger-image? a b) a]
   [else b]))

(check-expect (max-image ■ ■) ■)
(check-expect (max-image ■ ■) ■)
Example: Salt Lake City Addresses
Example: Salt Lake City Addresses
Example: Avenues

Convert streets in the Avenues to blocks east of the origin
Example: Avenues

Convert streets in the Avenues to blocks east of the origin
Example: Avenues

Convert streets in the Avenues to blocks east of the origin
Example: Avenues

Convert streets in the Avenues to blocks east of the origin
Example: Avenues

Convert streets in the Avenues to blocks east of the origin
Example: Avenues

Convert streets in the Avenues to blocks east of the origin
Example: Avenues

Convert streets in the Avenues to blocks east of the origin
Example: Avenues

Convert streets in the Avenues to blocks east of the origin

A st = 260 E
Example: Avenues

Convert streets in the Avenues to blocks east of the origin

O st = 1000 E
Example: Avenues

Convert streets in the Avenues to blocks east of the origin

A St. = 260 E
...
O St. = 1000 E
...

• A street at 2.6
• 10 - 2.6 blocks in 14 streets

I St. = 682.85... E

Two problems:
• Converting a letter to a position
• Converting a position to blocks east
Example: Avenues

; street->slc : string -> num
; Converts streets to blocks east of the origin:
(define (street->slc st)
  (+ 2.6 (* (street-index st)
            (/ (- 10 2.6) 14))))

; street-index : string -> num
; Converts "A" to 0, "B" to 1, etc.
(define (street-index st)
  (- (string->int st)
     (string->int "A")))
(check-expect (street-index "A") 0)
(check-expect (street-index "O") 14)

(check-expect (street->slc "A") 2.6)
(check-within (street->slc "I") 6.83 0.01)
(check-expect (street->slc "O") 10)