Multiple representations for abstract data

- Implementation of complex numbers as an example
- Illustrates how one representation can be better for one operation, but another representation might be better for another operation
- (Scheme already has complex numbers, but we’ll pretend that it doesn’t)

Complex numbers (math view)

- \( z = x + i \, y \) (rectangular form)
- \( r \, e^{i\alpha} \) (polar form)
- \( x \): real part of \( z \)
- \( y \): imaginary part of \( z \)
- \( r \): magnitude of \( z \)
- \( \alpha \): angle of \( z \)

Complex number arithmetic

- **Addition** – addition of coordinates – add real parts and imaginary parts
  - \( z_1 + z_2 = x_1 + iy_1 + x_2 + iy_2 \)
  - \( = (x_1 + x_2) + i(y_1 + y_2) \)

- **Multiplication** – easier to think of in polar form
  - \( z_1 \cdot z_2 = r_1 e^{i\alpha_1} \cdot r_2 e^{i\alpha_2} \)
  - \( = (r_1 \cdot r_2) e^{i(\alpha_1 + \alpha_2)} \)

SO?

- There are two different representations of complex numbers.
- Some operations on complex numbers are easier to think of in terms of one operation and others in terms of the other representation.
- Yet all operations for manipulating complex numbers should be available no matter which representation is chosen.
- Want to have access to each part: real, imaginary, magnitude, angle no matter which representation is chosen.

Two representations

- **Rectangular**
  - `make-from-real-imag` - constructor
  - `real-part` - selector
  - `imag-part` - selector
- **Polar**
  - `make-from-mag-ang` - constructor
  - `magnitude` - selector
  - `angle` - selector

Two different representations possible for the same number.
Addition
; adds together two complex numbers
; uses the representation of addition of coordinates
; in terms of real and imaginary parts
(define (add-complex z1 z2)
  (make-from-real-imag
   (+ (real-part z1) (real-part z2))
   (+ (imag-part z1) (imag-part z2)))))

Subtraction
; subtract one complex number from another
; uses the representation of subtraction of
; coordinates in terms of real and
; imaginary parts
(define (sub-complex z1 z2)
  (make-from-real-imag
   (- (real-part z1) (real-part z2))
   (- (imag-part z1) (imag-part z2)))))

Multiplication
; multiplies two complex numbers
; uses the representation as polar form
; in terms of magnitude and angle
(define (mul-complex z1 z2)
  (make-from-mag-ang
   (* (magnitude z1) (magnitude z2))
   (+ (angle z1) (angle z2)))))

Division
; divides one complex number from another
; uses the representation as polar form
; in terms of magnitude and angle
(define (div-complex z1 z2)
  (make-from-mag-ang
   (/ (magnitude z1) (magnitude z2))
   (- (angle z1) (angle z2)))))

Choose a representation
• We must implement constructors and selectors in
terms of primitive numbers and primitive list structure.

Which representation should we use??
• Rectangular form (real part, imaginary part – good for
  addition and subtraction)
• Polar form (magnitude and angle – good for
  multiplication and division)
• Either representation OK as long as we can select
  out all of the pieces we need – real, imaginary,
magnitude, angle

Rectangular Representation
;; lower level implementation
; RECTANGULAR FORM REPRESENTATION
; takes a real and imaginary part and
; creates a complex number represented
; in rectangular form
(define (make-from-real-imag x y)
  (cons x y))
Rectangular Representation (cont)

; given an imaginary number in rectangular form
; returns the real part
(define (real-part z) (car z))

; given an imaginary number in rectangular form
; returns the imaginary part
(define (imag-part z) (cdr z))

Rectangular Representation (cont)

; given an imaginary number in rectangular form
; return the magnitude (using trigonometric rels)
(define (magnitude z)
  (sqrt (+ (square (real-part z))
           (square (imag-part z)))))

; given an imaginary number in rectangular form
; return the angle (using trigonometric rels)
(define (angle z)
  (atan (imag-part z) (real-part z)))

Rectangular Representation (cont)

; takes a magnitude and an angle and
; creates a complex number represented in rectangular form
(define (make-from-mag-ang r a)
  (make-from-real-mag
   (* r (cos a))
   (* r (sin a))))

Polar representation

;; lower level implementation
; POLAR FORM REPRESENTATION

; takes a magnitude and an angle and
; creates a complex number represented in polar form
(define (make-from-mag-ang r a) (cons r a))

Polar Representation (cont)

; given an imaginary number in polar form
; returns the real part
(define (real-part z)
  (* (magnitude z) (cos (angle z))))

; given an imaginary number in polar form
; returns the imaginary part
(define (imag-part z)
  (* (magnitude z) (sin (angle z))))
Polar Representation (cont)

; takes a real and imaginary part and
; creates a complex number represented
; in polar form (harder)
(define (make-from-real-imag x y)
  (make-from-mag-ang
    (sqrt (+ (square x) (square y)))
    (atan y x)))

Which Representation?

• Note – either representation will work fine.
• Notice that some of the selectors/constructors are easier with one representation over the other
• But, no matter which is used, our basic operations will still work.