CS3L:
Introduction to Symbolic Programming

Lecture 3:
Conditional Expressions

Summer 2008 Colleen Lewis
colleenL@berkeley.edu

Today

- Questions about sentences and words
- Conditionals
  - `if`
  - booleans
  - predicates
  - `cond`
  - `and`, `or`, and `not`

Coming up: conditionals

- Conditionals allow programs to do different things depending on data values
  - *To make decisions*
- "Intelligence" depends on this
  - it is hard to imagine any interesting program that doesn't do different things depending on what it is given

Structure of conditionals

```
(if  <true?>                ;; test
   <do something>           ;; action (if true)
   <do something else>)     ;; action (if false)

(define (mother-says temperature)
   (if (< temperature 90)
     '(you should wear a coat!)
     '(drink plenty of water!))
     )
(mother-says 80)
(mother-says 100)
```

true? or *false?*

- We need Booleans: something that represents truth or 'not truth' to the computer:

  `#t`, `#f`
  `(odd? 3)` ➔ `#t`

- in practice, everything is true except `#f`

  `(if 'joe '(hi joe) '(who are you))`
  ➔ `(hi joe)`

- `false` (the word with 5 letters) is true!
  (really, `false` is `#t`)

Predicates

- Predicates are procedures that return `#t` or `#f`
  - by convention, their names end with a "?"

odd?	`(odd? 3)` ➔ `#t`
even?	`(even? 3)` ➔ `#f`
vowel?	`(vowel? 'a)` ➔ `#t`
	`(vowel? (first 'fred))` ➔ `#f`
sentence?	`(sentence? 'fred)` ➔ `#f`

cond is helps organize lots of different options

```
(cond
   (test-1  return-if-test1-true)
   (test-2  return-if-test2-true)
   ...
   (else    return-if-no-other-test-is-true)
    ))
```

and, or and not

- `and` takes any number of arguments, and returns true only if all are true
- `or` takes any number of arguments, and returns true if any are true
- `not` takes a single argument, and returns true only if the argument is false.

```
(if (not (and #t #t #t #f))
    'yes
    'awwwww)    ➔  yes
```