CS3:
Introduction to Symbolic Programming

Lecture 2:
Introduction, and Conditionals

Spring 2006

Nate Titterton
nate@berkeley.edu

Announcements

- **Nate's office hours:**
  - Wednesday, 2 - 4
  - 329 Soda
- **Tue/Wed is a Catch-up day.**
  - Use this day to catch up! That is, go back over the last two weeks and fill in places you missed
  - You will all be ready to go on Thur/Fri, right?
- **Our reader is Jonathan Chan**
  - He will be stopping by the lab in the next few weeks

Schedule

2	Jan 23-27	Lecture: Introduction, Conditionals Lab: Conditionals
3	Jan 30-Feb 4	Lecture: Case Studies Reading: Difference between Dates Lab: Work with Difference between Dates
4	Feb 6-10	Lecture: Data abstraction in DbD Lab: Miniproject I
5	Feb 13-17	Lecture: Introduction to Recursion Lab: Recursion
6	Feb 20-24	Lecture: <HOLIDAY> Lab: Recursion II
7	Feb 27-Mar 3	Lecture: *Midterm 1* Lab: Recursion III

A video resource

- http://wla.berkeley.edu

  Weiner lecture archives

- **The "course" is an earlier CS3**
  - Different emphasis; early lectures may work better than later ones
  - Very different lab experience
  - Same book

Some nice comments

- **"In English, when something is in quotes we think about it differently. Same in scheme"**

- **"In order to remember how to parenthesize a cond statement... think of each statement as an *if* without the 'if' "**

Testing

- **There is much more to programming than writing code**
  - *Testing* is crucial, and an emphasis of this course

  - Analysis
  - Debugging
  - Maintenance.
  - "Design"

Write an answer procedure.

Write a procedure named answer that, given a sentence that represents a question, returns a simple answer to that question. (A question's last word ends with a question mark.) If the argument sentence is not a question, answer should merely return the argument unchanged.

- Given `( am i ...? )`, `answer` should return `( you are ...)`.
- Given `( are you ...? )`, `answer` should return `( i am ...)`.
- Given `( some-other-word i ... ? )`, `answer` should return `( you some-other-word ...)`.
- Given `( some-other-word you ... ? )`, `answer` should return `( i some-other-word ...)`.
- Given any other question, `answer` should return the result of replacing the question mark by a period.

Conditionals

```
(define (walk light city cops-present)
  (cond ((equal? city 'berkeley) 'strut)
        ((equal? light 'green) 'go)
        ((equal? light 'not-working)
         'go-if-clear)
        ((and (equal? light 'flashing-red)
              cops-present)
         'wait)
        ((equal? light 'flashing-red)
         'hurry)
        (else 'just-stand-there)))
```

You are writing big programs now. But, what can’t you do yet?

What does “understand a program” mean?

Case Studies

- **Reading!?**
- **A case study:**
  - **starts with a problem statement**
  - **ends with a solution**
  - **in between, a …story… (narrative)**
  - ***How a program comes to be***
- **You will write "day-span", which calculates the number of days between two dates in a year**

You need to read this

- **The lab will cover the case study through a variety of activities.**
  - **This will culminate in the first "mini-project"**
- **We just may base exam questions on it**
- **It will make you a better programmer!**
  4 out of 5 educational researchers say so.

Some important points

- **There is a large "dead-end" in this text**
  - Like occur in many programming projects
  - Good "style" helps minimize the impacts of these

- **There is (often) a difference between good algorithms and between human thinking**