University of Virginia 24 October 2011
cs1120: Introduction of Computing
Explorations in Language, Logic, and Machines

Class 26: Objectifying Objects

Upcoming Schedule

- **All regular office hours this week**
- **Wednesday, 26 October: Quiz 3** (covers the course book through Chapter 10, *The Information* through the end of Chapter 8)
- **Problem Set 6** is now due on **Monday, 7 November**

Spring Courses

There will be an option for Problem Sets 8 and 9 to use Java, and we will cover some Java in class. This will be enough for you to satisfy the prerequisite for cs2110: *Software Development Methods*.

cs2110 is offered this Spring with lectures Mondays, Wednesdays, and Fridays at 11am in Rice Auditorium, taught by Prof. Tom Horton.

Students who are pursuing the *BA Computer Science* major or interested in taking further computing courses are expected to do the Plan J option for the final assignments to be prepared to take cs2110. If you do this, you will be well prepared to take cs2110 (and much better prepared on many of the topics than the students who enter from cs1110).

If you have questions about what courses you should take next, feel free to stop by office hours to discuss this or contact me by email.

Encapsulation

State is *encapsulated* if it can only be read and modified by certain procedures.

Why is it better for our counter to encapsulate its state?

```
(define (make-counter)
  ((lambda (count)
     (lambda ()
       (set! count (+ 1 count))
       count))
   0))
```

Objects package state and procedures.

```
(define (make-counter)
  (let ((count 0))
    (lambda (message)
      (cond ((eq? message 'reset!) (set! count 0))
            ((eq? message 'next!)
               (set! count (+ 1 count)))
            ((eq? message 'current) count)
            (else
             (error "Unrecognized message"))))))
```

PS1-PS4: ***Functional Programming***
Focused on **procedures**
Break a problem into procedures that can be composed

PS5: ***Imperative Programming***
Focused on **data**
Design data for representing a problem and procedures for updating that data

PS6: ***"Object-Oriented Programming"***
Focused on **objects** that package state and procedures
Solve problem by designing objects that model the problem
Lots of problems in real (and imaginary) worlds can be thought of this way

What are some advantages of functional programming over imperative programming?

What are some advantages of imperative programming over object-oriented programming?

What are some advantages of object-oriented over functional programming?

What are some advantages of object-oriented over imperative programming?

(Note: I plan to ask at least one of the last four questions on Wednesday's quiz.)