
CS3:

Introduction to Symbolic Programming

Lecture 15:
Summary, Exam problems

Spring 2008

Nate Titterton
nate@berkeley.edu

Announcements

- **The FINAL**
 - **Wednesday, May 21st, 12:30-3:30 pm**
 - **Bechtel Auditorium**
 - **Questions will be asked on everything**
 - **With the main emphasis on the post-midterm-2 material (lists, trees, non-functional programming).**
 - **Only 2 hours worth of material – 25% more fattening than a midterm**
 - **Review session, to be announced**
- **Office hours this week: Wed, 11-1:30**
- **Don't forget about the final survey**
 - **This will be worth 1 course point...**
 - **Your answers won't factor into your grade.**

**How are you going to study
for the Final?**

So, what have we done in CS3?

- **Consider the handout of topics**
 - **Common topics**
 - **Pre-recursion**
 - **Recursion**
 - **Higher order procedures**
 - **Lists and beyond**
 - **Case studies**
 - **Working with large programs**

Another list...

- 1. Functional programming**
- 2. Functions as data**
- 3. Recursion**
- 4. Managing large programs**

(1) Functional Programming

- **All that can matter to a procedure is what it returns.**
- **Small functions can be easily tested (isolated)**
- **In other languages, you typically:**
 - **Perform several actions in a sequence**
 - **Set the value of a global or local variable.**
 - **Print, write files, draw pictures, connect to the internet, etc.**
- **Other "paradigms" include: sequential, object-oriented, event-driven, declarative**

(2) Functions as data

- **Higher order procedures take functions as parameters.**
- **It is useful to return functions at times**
- **`lambda` is quite useful, and sometimes necessary.**

(3) Recursion

- **Linear (simple) to quite advanced**
 - They all have base and recursive cases in a conditional
 - Thinking about “inner” recursive calls as possible solutions in their own right can help.
- **In contrast to iteration and looping (where counters or state define looping constraints)**
 - Knowledge of recursion will help these simpler cases.

(4) Managing large programs

- **Style—commenting, naming conventions, etc.**
- **Abstraction—carving up programs into small, coherent units**
 - **for maintenance and collaboration**
- **Iterative and unit testing**
- **Reading the specifications, and communicating often with colleagues**