CS3: Introduction to Symbolic Programming

Lecture 15: Summary, Exam problems

Fall 2006

Nate Titterton
nate@berkeley.edu
Announcements

• The FINAL
  - Thursday, May 17, 5-8pm
  - F295 Haas
  - Questions will be asked on everything
    - With emphasis on later material (lists).
    - Only 2 hours worth of material – 50% more than a midterm
  - A review session time will be posted on the portal

• A survey will be up on the course portal tomorrow
  - This will be worth 3 course points! (from the random step). That is more than many homeworks!
  - It 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
  - Case studies
  - Working with large programs
2. Functional programming
3. Functions as data
4. Recursion
5. Abstraction
6. 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": sequential, object-oriented, declarative
(2) Functions as data

• Higher order procedures take functions as parameters.

• It is useful to return functions at times

• \texttt{lambda} is quite useful, and sometimes necessary.
(3) Recursion

- Linear (simple) to quite advanced

- In contrast to iteration and looping (where counters or state define looping constraints)
  - Knowledge of recursion will help these simpler cases.
(4) Abstraction

• The big idea that is related to everything!

• A design practice that makes it possible to carve up a problem, and therefore focus on only part of it.
  - Makes working collaboratively more efficient
(5) Managing large programs

- **Style**: commenting, naming conventions, etc.
- **Abstraction**: for maintenance and collaboration
- **Iterative testing**
- **Reading the specifications, and communicating often with colleagues**
Lecture 15:
Summary, Exam problems
**Announcements**

• The FINAL
  - Thursday, May 17, 5-8pm
  - F295 Haas
  - Questions will be asked on everything
    - With emphasis on later material (lists).
    - Only 2 hours worth of material – 50% more than a midterm
  - A review session time will be posted on the portal

• A survey will be up on the course portal tomorrow

  - This will be worth 3 course points! (from the random step). That is more than many homeworks!
  - It won't factor into your grade.
How are you going to study for the Final?

Click to add text
So, what have we done in CS3?

• **Consider the handout of topics**
  - Common topics
  - Pre-recursion
  - Recursion
  - Higher order procedures
  - Lists
  - Case studies
  - Working with large programs
Another list…

2. Functional programming
3. Functions as data
4. Recursion
5. Abstraction
6. 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": sequential, object-oriented, 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

- In contrast to iteration and looping (where counters or state define looping constraints)
  - Knowledge of recursion will help these simpler cases.
(4) Abstraction

• The big idea that is related to everything!

• A design practice that makes it possible to carve up a problem, and therefore focus on only part of it.
  - Makes working collaboratively more efficient
(5) Managing large programs

- **Style**: commenting, naming conventions, etc.
- **Abstraction**: for maintenance and collaboration
- **Iterative testing**
- **Reading the specifications, and communicating often with colleagues**