Course Summary

This course introduced the science of computing. We learned tools and techniques affecting all parts of the software lifecycle.

Specifications:

- Writing precise specifications (more of that in csc238)

Design:

- Looking at a problem abstractly.

- Standard abstractions that have proven useful in computer science. Abstract data types.
  - stack
  - list
  - queue
  - graph
  - tree

- Some new data structures that offer alternative ways to implement an abstract idea.
Summary (Cont’d)

- linked lists
- circular queues
- trees
- binary search trees

(more of these in csc378)

• Analyzing the efficiency of algorithms
  - Big-Oh notation
  - Overestimating
  - Reasoning about non-recursive and recursive programs

• Reasoning about correctness of algorithms and data structures.
  - Why is it necessary?
  - Proof mechanisms
    * proofs by cases, by contradiction, universal proofs, conditional proofs
    * proofs by induction
Summary (Cont’d)

Implementation

● Properties of a good program.
  - abstraction
  - reuse
  - information-hiding
  - encapsulation
  - design by contract, documentation (see later)

● Designing a program to have these properties.
  - interfaces
  - packages
  - abstract classes
  - exception handling
  - specific Java features:
    * everything is an object
    * static/instance variables and methods
      * Comparable, Iterator, Vector

● New algorithms
  - greedy algorithms
  - merge sort
Summary (Cont’d)

- A new programming technique: recursion
  - thinking recursively
  - termination, base case
  - recursion vs induction
  - recursion vs iteration
  - tail recursion

Code Reviews

Testing

- Choosing a systematic and thorough set of test cases.
- Documenting testing so that it will be convincing.

Documentation

- Design by contract

- Internal comments
  - data invariants
  - representation invariants
  - how much to comment and where

- external comments
  - pre- and post-conditions
Final Exam

3 hours, no aids allowed. Material from lectures, tutorial exercises, assignments, online homework, assigned readings in the handbook

Particular questions (not necessarily a complete list):
- write code (perhaps with pointers, or recursion, or interesting object-oriented features)
- trace code
- write specifications for iterative or recursive code
- do big-oh analysis of code
- talk about pros and cons of various implementations for an ADT
- do a memory model trace
- do some class design
- implement a class
- use a given class
- talk about any of the above things
- correct mistakes in examples of any of the above things
How to study for the exam

• Go back over anything you never felt good about; ask us about it
• Practice! Do the sample exams on the web. Both old midterms and finals will be helpful.
• Go to office hours, even if you do not have specific questions. You are welcome to attend anyone's hours, regardless of whose section you are in.
• We will be holding additional office hours - watch the announcements on the course web page.
How to write the exam

• Read over the whole exam before you start.
• Budget your time. You could run out of time if you lose track of time (even though ours is not intended to be a speed test).
• Read questions with care, and be sure to do what we asked for!
• If you are at all unsure about what we mean, feel free to ask a question
End-of-Term Administrative Stuff

● Getting things back:
  - Your professor (see the web page for what is available)
● CSSU - Engineering Annex
  - Office number: 978-5354
  - Office hours: see our web page
● Marks:
  - Watch the web page for announcements when various marks have been posted.
  - Check your marks!
  - You are responsible for ensuring that we have correct marks for you
  - Report errors immediately, by bringing the relevant assignment or test to me
    - during office hours or
    - under the door in my office (LP384)
  - Please do this BEFORE the final!
● Final marks
  - We will post *unofficial* course grades on the web (watch the announcements on the web page)
  - We expect to be done with marks before Christmas
What is ahead?

- System support
  - Operating systems
  - Compilers
  - Programming languages and paradigms
  - Databases

- Communications and security
  - Network protocols
  - Security
  - Concurrent systems (how to write them? how to test them? how to reason about them?)

- Theory
  - Which things can be computed and which cannot?
  - How to analyze code?
    - Solving recurrence relations
    - Proving algorithms correct
    - Analyzing running times
    - Time/space trade-off
  - New data structures
  - Logic (the centre of it all)
What is ahead (cont’d)?

• Creating usable and scalable systems
  - Human-computer interaction
  - Requirements analysis
  - Design patterns, architecture, modeling languages
  - Metrics to access/predict cost, quality, complexity
  - Simulations

• And much much more...
  - Artificial intelligence (robotics, planning, vision)
  - Graphics
  - Numerical analysis
And in conclusion...

Hope you enjoyed this class!

Hope you will choose computer science as your future career!

Good luck on the final exam!

See you during office hours!

And … happy holidays!