Syracuse University
L.C. Smith College of Engineering and Computer Science

CIS 252 – Introduction to Computer Science
Spring 2004

Catalog Description
Programming emphasizing recursion, data structures, and data abstraction. Elementary analysis of and reasoning about programs. Public policy issues. Extensive programming. Three hours of lectures and one hour of computer laboratory.

Instructor Information
Prof. Susan Older  (sueo@ecs.syr.edu)
Office: CST 3-179, x4679
Office hours: TBA

Prof. Jim Royer  (royer@ecs.syr.edu)
Office: CST 3-132, x1028
Office hours: TBA

Course Web Address
http://www.cis.syr.edu/courses/cis252

Textbook

Additional References
Other notes will be distributed during the semester as necessary.

Course Objectives
As its name suggests, this course serves as an introduction to many of the core concepts of computer science. Broadly speaking, the practice of computer science is about data structures and supported operations on them, small (or large) languages and environments, and component-based problem solving. A computer scientist must continually choose among alternatives, and to do so successfully requires exposure to multiple options, knowledge of how to make wise choices, and experience.

We will be using the programming language Haskell for this course, because it is especially well suited for the two main themes we want to emphasize: *patterns* and *problem solving*. Haskell supports data patterns through its type system and through an extensive pattern-matching facility. It also supports many programming patterns through higher-order functions. Finally, Haskell supports a view of component-based design that is especially well-suited for problem solving and for logical analysis. Consequently, you will begin to think about programming in a different, more systematic and logical way, even as you move on to imperative and object-oriented programming.

History suggests that you’ll be tempted to call this course “the Haskell course”. Our goal
is that, by the end of the semester, you will be able to see below the surface of Haskell and recognize core concepts that you will take with you to other courses and into your professional lives.

Prerequisites

Course Outcomes
After completion of the course, students should be able to do the following:

- Use the Unix command-line interpreter to perform basic tasks, including: managing files and directories, changing file permissions, checking a print queue, printing files (both postscript and non-postscript), checking available options for a Unix program, and using basic pipes and filters.
- Use Emacs, including being able to configure it via the file ~/.emacs and using apropos to locate available functions.
- Recall the 4 steps of Polya’s problem-solving method.
- Recall the $n$ steps of the program-development process.
- When given an informal English specification of a problem, extract a relevant collection of abstract types from that specification.
- When given a formal specification, construct concrete examples that satisfy that specification.
- When given an informal but fairly precise English description, write a Haskell program that accurately captures the desired behavior.
- When given a Haskell expression, determine whether or not it is well-typed.
- When given a Haskell expression that is well-typed, give its type.
- When given a Haskell expression that is ill-typed, explain why it fails to be well-typed.
- When given a moderate-sized Haskell program and relevant input, calculate the result of that program.

Outcome Measurement
Your final grade will be based on a variety of activities:

- Homework assignments (??% of final grade)
  There will be a homework assignment approximately every week: we will drop the lowest homework grade at the end of the semester. Occasionally, students may be asked to explain their homework to us or to the TA: in such cases, the homework grade will be based on the results of this explanation.

- Labs (??% of final grade)
  There will be a lab assignment approximately every week: we will drop the lowest lab grade at the end of the semester.

- Quizzes (??% of final grade)
  There will be approximately six in-class quizzes during the semester. There will also be a two-hour optional final exam: the quiz/exam portion of your final grade will be the greater of (1) your cumulative average of in-class quizzes, and (2) your score on the final exam.
You should also be aware of the following submission practice and policy:

On some homework and lab assignments, you may work singly or in pairs; on other assignments, you must work alone. If you work with someone else, you should submit a single solution: each person’s name must appear clearly on the first page. All members of a group will receive the same grade (even if only one student is asked to explain a homework); groups of more than two will receive no credit.

Labs and homework assignments should be placed in the marked bin in CST 3-116 (the printer room by the Foundry); you will also need to submit code electronically (details are available on the class web site). Assignments are due by the date and time specified on them: No late assignments will be accepted.

Course Topics

Other Information
Ethics
We expect all students to behave ethically: do not cheat, plagiarize, or commit fraud. Fraud includes faking program transcripts to make it appear that code works correctly when it does not; plagiarism includes using someone else’s work without proper credit.

If we discover any instances of cheating, fraud, or plagiarism, we will give the guilty parties failing grades for the course and report the culprits to the director and the dean. If you are unsure whether a certain action constitutes cheating, fraud, or plagiarism, assume that it does: you may ask us for clarification at any time.

Every student must read and sign a copy of the course Honor Policy, which details your obligations to behave ethically. Students will receive zeroes on all assignments/exams until this sheet is turned in to us.

Accommodations
Students who may need accommodations because of a disability should make an appointment to meet with the instructor and register with the Office of Student Assistance (306 Steele Hall) and/or the Center for Academic Achievement (804 University Avenue, Room 303).

Adventure
To get the most out of this course, be prepared to keep an open mind and a sense of adventure.

In short, this semester’s version of CIS 252 is an experiment. We are making substantial changes—including the course textbook and programming language—that we believe will help you in subsequent courses and beyond.
We anticipate that, like most experiments, some things will go wonderfully, and some things will go horribly wrong. We hope to minimize the latter, but we do expect to make mistakes. *Speak up, and let us know what’s on your mind. We may not agree with all of your comments and suggestions, but we want to hear them.*