Assignment 0913
This assignment is somewhat longer than usual, because you have more time than usual to complete it 😊. Two goals: get your head into thinking seriously about programming languages, and to settle the infrastructure you will use for your research paper so that from here on, you can “just write it.”

Not for Submission
1. Read Chapters 1 and 2 in Scott.
   a. You may skip 2.3 if you run out of time, but you won’t regret reading it.
   b. Do read all of 2.2.
   c. Scan through the review questions at the end of Chapters 1 and 2, and do them in your head just to make sure you picked up the material.
   d. Scan through the exercises at the end of Chapters 1 and 2, also doing them in your head.
   e. Selected exercises are for submission; see below.
2. Get LaTeX up and running on whatever system(s) you will use to work on your paper.
4. Get your hands dirty with the programming languages discussed in today’s class: setup working C, C++, Java, and Perl environments, type in the sample programs, and run them.

For Submission
Please submit all assignments on hardcopy. E-mail (even of PDFs) does not count. I know it seems anti-computer science to require this, but there are other forces at work here. Be prepared to talk through your work in class.

1. Type, customize, and print paper.tex and prospectus.tex, and submit the printouts to me (no problem, because you did #2 above, right?)
2. Choose a favorite algorithm similar in complexity to greatest common denominator and implement it in C, C++, Java, and Perl. For example, least common multiple, manual modulo, etc. Don’t get too fancy — consider this a “learning by example” exercise, with some practice looking things up on the Web if you get stuck on a language point.
3. Work on the following exercises in Scott, and submit your work to me:
   a. Exercise 1.1: for each type of error except 1.1.e, provide both source code showing the error and the error message that is generated by the specific language implementation you are using, whether at build time or run time. 1.1.e will only need source code.
   b. Exercise 1.5: in addition, how specific to Unix make are these issues, or do they apply to other build environments as well? (i.e. ant, IDEs such as Eclipse)
   c. Exercise 1.8
   d. Exercise 2.1
   e. Exercise 2.3 (more of a preview to future material)