Notes and Reminders:

- Homework 3 is due at the beginning of class on Wednesday, September 24. **NO LATE SUBMISSIONS** will be accepted. Programming Project 2 is due at regular time, 4:30pm.
- Wednesday, September 24, has been set aside as a review for the exam. We will go over Homework 3 and answer any questions you have about the material.
- The instructor will be out-of-town starting at noon on Thursday, September 25 until Saturday, September 27. Office hours will be extended until 4:45pm on Wednesday, September 24, and extra office hours will be held on Tuesday, September 23, 1:00pm-3:00pm, and Thursday, September 25, 9:00am-11:30am.
- The exam will be proctored by Mr. Mark Randall, the EECS staff engineer. You will be allowed to ask him questions during the exam, but he may or may not be able or willing to answer them.

Exam 1 will be on Friday, September 26. You may bring one 8.5in x 11in size sheet of paper with notes on one side to the exam. You may print out the sheet, but it must be in a **10-point font or larger**. E.g., please do not photoreduce or print 4 pages on a side. If you handwrite your notes, they may be as small as you like. You may handwrite notes in the margins of a printout. No other notes are allowed for the exam.

The exam will be cumulative and comprehensive with respect to basic programming constructs in the sense that you are expected to be able to read and write code or analyses and designs using concepts such as selection, repetition, and functions. Emphasis will on the material in Chapters 1-3, and covered in lectures, and homework and projects assigned through Friday, September 19. The exam will consist of questions similar to the homework problems, programming projects, and exercises in the textbook.

The following is a list of topics that will be emphasized, but it is in no way to be construed as an exclusive list.

1. Makefiles, command line argument processing, and file stream processing
2. Error checking and exception handling
3. Classes - design, implementation, and use, including overloaded operators
4. Algorithm analysis including determining the "Big-O" running time of code fragments and functions, and comparison of various time complexities.
5. Recursion - what is it and how to use it to solve problems.
6. Sequential and binary search algorithms
7. Selection sort algorithm