Reminders and announcements:

- The first hour of class on Tuesday, September 19 has been set aside as a review for Exam 1. We will go over Homework 2 as part of the review. Thus Homework 2 is due by the beginning of class. NO LATE SUBMISSIONS will be allowed for this assignment. In addition, come prepared to ask and answer questions about the exam material.

- Programming Assignment 3 is due on Thursday, September 21, at 4:30pm (i.e. regular time).

Exam 1 will be on Thursday, September 21. You may bring one 8.5in x 11in sheet of paper with notes on (only) one side to the exam. You may print out the sheet, but it must be in a 9-point font or larger. E.g., please do not photoreduce or print 4 pages on 1 side. If you handwrite your notes, they can be as small as you like. You may handwrite notes in the margins of a printout.

The exam will consist of questions on the material in Chapters 1-4 (excluding Section 4.6); material covered in lectures and in-class exercises through Thursday, September 14; material in homeworks 1-2; and programming assignments 1-3. Material on function overloading and output formatting will not be on the exam.

The format of the exam will be similar to that of in-class exercise questions, homework questions, and programming assignments. That is, you will be asked to write analyses and designs for programs and/or functions to solve particular problems, and you will be asked to explain and write C++ code examples.

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. Analysis and design of main programs and functions that return values. Differences between received, returned, and local function objects.
2. Declaration and use of constants and variables, use of arithmetic, relational, and logical operators, assignment statements.
3. Basic use of input and output statements (i.e., format manipulators will not be on the exam).
4. Implementation of functions that return values, including how to define and use them, (actual) arguments vs. (formal) parameters.
5. Use of selection constructs: single and multi-branch if constructs, case constructs and their implementations.
6. Use of repetition constructs: pre-test loops, post-test loops, counter-controlled loops and their implementations.

When you are asked to write code, you will not need to write comments, function prototypes, include directives, or output formatting beyond producing newlines in appropriate places.