The practical exam is a take-home exam. It will be given out on Tuesday, October 9, at the end of the class period and will be due (hardcopy printouts and submitted on the Linux server) by 5pm on Thursday, October 11. **Class on October 11 is canceled,** but the instructor will be available in Library Reception from 3pm to 4pm, and in the Manor Computer Lab at 5pm. Otherwise, submit hardcopies by having Reception place them in the instructor's mailbox.

For the Programming Practical Exam, you will be expected to write programs using `g++` on the Linux server to the specifications given in the exam. **The exam is open textbooks (Ford & Topp and Stroustrup), open class notes (both your own and those on the course website, but not anyone else's notes). You also are allowed hardcopy printouts of your assignments, but you are not allowed to access electronic copies of your assignments or the notes on the course website. (I.e., you can look at your assignments and the course notes, but you may not cut and paste from these assignments or notes.)** These are the only aids you may use in completing the exam. You will be asked to sign a statement certifying that you did not receive or give any unauthorized aid for the exam.

The exam will consist of three problems. For some problems, you may be given files with some code already written. For other problems, you may be asked to write an entire program. It should not take more than about 2 hours to complete.

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 using concepts such as selection, repetition, and functions. Material covered in lecture and assignments through Exam 1 (including Homework 4 and Project 3, but not pointers) are emphasized. The exam will be similar to the programming projects and programming homework exercises.

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
2. Command line argument processing
3. File stream processing
4. Exception handling
5. Classes - implementation and use, including overloaded operators
6. Template functions (but not template classes)
7. Vectors - declaration and use (e.g. with template sorting functions)