C++ Code Structure

Cooperating with the Compiler

C / C++ Compilation

- In Java (and many other modern languages), the compiler is designed to make multiple passes over code files during compilation.
  - In doing this, the compiler first finds all objects, variables, and functions of interest that are available before beginning the actual computation.

C / C++ Compilation

- In C++, you are required to manually "declare" any object, variable, or function *within a code file* ***before*** using it.
  - Note: "declaring" vs "defining."
  - "declare" – "function X exists."
  - "define" – "this is what function X does."
  - If you try to use something before it's declared, a compiler error will result.

C / C++ Compilation

- To simplify the process of declaring relevant code objects, C++ has two core file types.
  - Header files: "*.h"
  - Contains relevant declarations
  - Source files: "*.cpp" ("*.c" in C.)
  - Contains code definitions
  - Source ***and*** header files then #include other header files with needed definitions.

C++ Resources

- Like Java, C++ has a substantial amount of pre-coded resources for use in programs.
  - –This being said, Java's built-in collection is far more extensive than C++'s.
  - –These are also utilized by use of #include, as opposed to Java's import.
  - –However, built-in resources are included through <angle brackets> rather than "quotes."

C++ Resources

- Very common imports:
  - <string>
  - Includes the std::string class, a C++ counterpart to Java's string. This is *not* a fundamental type in C++.
  - <iostream>
  - Includes the std::cout and std::cin output and input streams.
  - As used in class, these are the console output and console input structures, like System.*out* and System.*in* from Java.