Lecture 2

Chapter 1.3, 1.4, 1.5
Topics for today

• #include
• Namespace
• cout
• Spaces in output
• Revisit the newline character/escape sequence
• Formatting for double values
• cerr
• cin
• Comments
#include

- What is it used for?
  - C++ has a vast library of functions
  - But these functions are not included by default in your program
  - This is why they have to be included
  - `#include <iostream>`;
    - iostream is the library that contains simple input and output like cin and cout
Namespace

• Even though C++ gives you access to functions like cin and cout, it gives you the ability to define your own cin and cout if you want.

• How does one differentiate between the two?
  – Using namespaces
  – Example on following slide
cout

• C-out
  – As in C++ output
  – Handles output to the console
  – If you have defined your program as cout can be used in the following way
    • using namespace std;
    • cout << “Hello World!”;
  – Otherwise
    • std::cout << “Hello World!”;
cout

• Examples of cout
  – cout << “Today is the “ << (date) << “ of June.\n”;  
  – cout << “$” << price << endl;
    • Note how the newline character was performed  
  – cout << firstNumber << “ “ << secondNumber;
    • Note how a space was added between the two
Newline Character vs. endl

• When to use one vs. the other?
  – Completely up to the programmer
  – Just remember \n always has to be part of a string even is that is the only thing in the string
  – endl cannot be in a string if you want it to work
Formatting for double values

• This is done for readability
• Example confusing output
  – “Tuition cost this term is $5203.91000”
  – Or “Tuition cost this term is $5.20391e03”
• For this example use the following
  cout.setf(ios::fixed);
  cout.setf(ios::showpoint);
  cout.precision(2);
• The last line sets the actual number of decimal points
cerr

• This is the standard output for error messages
• Why?
  – Sometimes the programmer wants to track the errors without having the user know about it
  – Makes it easier to track errors
cin

• Standard input from the console
• Can do multiple inputs in one line
  – cin >> numPuppies >> numKitties;
  – The user can then either type in the number for puppies and press return to type in the number of kitties or write both in one line with spaces to separate the two values.
• Potential issue is if the valid input has a space
  – For example the string “Mr. Fox” for input to a string would result in “Mr.”
Comments

• If the comment is only one line long use
  – // This is a comment
• If it takes up multiple line use /* to start it and */ to end it
• Etiquette dictates that comments shouldn’t run off the screen