Computer Science II — CSci 1200 — Sections 1-4,6
Week 2, Tuesday Class — September 4, 2001
Information Hiding, Abstract Data Types, Classes

Rest of Today’s Class
• Separating interface from implementation
• C++ class declarations
• Using C++ classes

Reading — Today’s Material
• C&P, Chapter 1, pages 15-45:
  – introduction to modular design
  – UML as a means of specifying objects
  – key issues in programming
• C&P, Chapter 3:
  – ADTs (pp 112-117)
  – class declarations (pp 131-137)

Reading — Friday’s Material
• C&P, Chapter 3:
  – the ADT list (pp 117-123)
  – designing an ADT (pp 125-129)
  – implementation (pp 131-142)
  – array-based implementation of the ADT list (pp 144-150)
• Deitel&Deitel, Chapters 6 and 7 (section numbers in parentheses):
  – intro (6.1)
  – structs (6.2-6.4, we will only touch on these briefly)
  – Time class (6.5, introductory example)
– class scope and access to class members (6.6, important)
– separating interface from implementation (6.7, important)
– public and private (6.8, important)
– accessor functions and utility functions (6.9)
– initialization through constructors (6.10, important)
– default arguments with constructors (6.11)
– destructors (6.12, we will come back to these later)
– calling constructors and destructors (6.13)
– using data members and member functions (6.14, important)
– references to private members (6.15, just a warning)
– default assignment (6.16, just a note)
– const and constant member functions (7.1, we will come back to this)
– objects as members of other classes (7.2, important)

Abstract data types and information hiding

• You know how to use physical objects without knowing how they work. Examples include cars, televisions, telephones.

• The method of operation can be thought of as a “user interface”.

• The same holds in using software programs such as spread-sheets.

• This idea extends to writing our own programs:
  – We create new types (beyond integers, floats, chars) by specifying (declaring) the interface to them and separately defining their implementation.
  – We use classes defined in the standard library by knowing their interface, but not their implementation.

The specification of a type and a set of operations on it, without a specification of its implementation is called an abstract data type (ADT).
Example: A Date Class

- Many programs require information about dates.
- Information stored about the date includes the month, the day and the year.
- Operations on the date include recording it, printing it, asking if two dates are equal, flipping over to the next day (incrementing), etc.

C++ Classes

A C++ class consists of
- a collection of member variables, usually private, and
- a collection of member functions, usually public, which operate on these variables.

We will look at the example of the Date class declaration.

Using C++ classes

Continuing with the Date example, we will demonstrate
- Defining class objects.
  - Each object we create will have its own distinct member variables!
- Calling class member functions.
- Using the member function that compares one object to another.

After looking at the example, we still will not have examined how the class member functions are implemented! This is an important feature of ADTs and C++ class design.

Review of Today’s Lecture

- Separating interface from implementation — ADTs
- C++ class declarations, including member variables and functions
- Using C++ class member functions