Last Lecture

• We illustrated
  ◦ Abstraction
    • Hides all but the relevant (to client code) data about an object in order to reduce complexity and increase efficiency.
  ◦ Effect of Abstraction
    • Separating purpose (what) of a class from its implementation (how) such that client code is able to use a class without knowing its implementation
  ◦ Result of Abstraction -&gt; Abstract Data Type (ADT)
    • Information Hiding -&gt; hidden section
    • Specifications -&gt; visible section

using 2 versions of the Temperature class
  ◦ Non-ADT version
  ◦ ADT version
Example of ADT: Temperature class

class Temperature {

private:
    double myDegrees;
    char myScale;
    bool isValidTemperature( const double degrees,
                            const char scale );

public:
    static const double ABSOLUTE_ZERO_FAHRENHEIT = -459.67;
    static const double ABSOLUTE_ZERO_CELSIUS = -273.15;
    Temperature();
    Temperature(double degrees, char scale);
    double getDegrees() const;
    char getScale() const;
    void setDegrees( const double degrees );
    void setScale( const char scale );
    Temperature inFahrenheit() const;
    Temperature inCelsius() const;
    void raise( const double amount );

}; // end Temperature
// End of header file

Note that Temperature is an ADT, but it is not a data collection ADT
Hidden part of Temperature ADT—its implementation

private:
    double myDegrees;       // >= ABSOLUTE_ZERO for myScale
    char    myScale;         // 'F' or 'C'

    // Description: Temperature validation utility.
    // Returns true if Temperature( degrees, scale ) represents a valid temperature, false otherwise.
    bool isValidTemperature( const double degrees, const char scale );
Visible part of Temperature ADT – its specifications

public:
    static const double ABSOLUTE_ZERO_FAHRENHEIT = -459.67;
    static const double ABSOLUTE_ZERO_CELSIUS = -273.15;
    Temperature();
    Temperature(double degrees, char scale);
    double getDegrees() const;
    char getScale() const;
    void setDegrees(const double degrees);
    Temperature inCelsius() const;
    Temperature inFahrenheit() const;
    void raise(const double amount);

* Documentation removed due to restricted real estate
Learning Outcomes

At the end of this lecture, a student will be able to … :

- Describe what we do in each of the 4 steps of software development process
- Apply the first few steps of this process while solving a problem using a data collection designed as an abstract data type (ADT)
- Describe our first data collection -> List
Today’s Menu

- Data Collection versus Data Structure
- The 4 Steps of the Software Development Process
  - Step 1 – Problem Statement
  - Step 2 – OO Design
- Introduce List data collection
Data Collection

• Definition
  ◦ Conceptual

• Examples
  ◦ List

• Data Collection designed as ADT
Data Structure

- **Definition**
  - Data structures are constructs available as part of a programming language
  - 
  - 
  -

- **Examples:**
  - array
  - linked list (nodes and pointers)
Review:
The 4 Steps of the Software Development Process

- Step 1: Problem statement
- Step 2: Design
- Step 3: Implementation
- Step 4: Testing
Step 1 - Problem Statement

- FriendsBook Application
  - Design and implement an application that maintains the data for a simple social network.
  - Each person in the network must have a profile that contains the person’s name, optional image, current status and a list of friends.
  - Your application must allow a user to join the network by creating a profile, leave the network, modify the profile, search for other profiles, and add friends.

Source: Textbook - Programming Problem 11 - Chapter 9 - Page 288
Step 1 - Question

- Is the problem statement clear and unambiguous?
Why is Step 1 important!

- This quote demonstrates why problem statement must be clear and unambiguous before we move on to the other steps of software development process.

Feet or miles?

During a laser experiment, a laser beam was directed at a mirror on the Space Shuttle Discovery. The test called for the laser beam to be reflected back toward a mountain top. The user entered the elevation of the mountain as “10,023,” assuming the units of the input were in feet. The computer interpreted the number in miles and the laser beam was reflected away from Earth, toward a hypothetical mountain 10,023 miles high.
Step 2 - OO Design – Overview

1. Figure out the Classes
   ◦ Model “entities” from problem statement into classes

2. Figure out the Data Collection
   ◦ Choose appropriate data collection and design it as an ADT

3. Record our design
   ◦ Use UML class diagram

4. Specify behaviour of our program
   ◦ Write algorithm (in pseudocode)
Summary

- Defined Data Collection and Data Structure
- Described what happens in each of the 4 steps of software development process
- Started on the first few steps of this process while solving a problem using a data collection designed as an abstract data type (ADT)
Next Lecture

- Continue creating our solution to the simple social network problem while following the steps of the software development process and designing data collection List as a ADT