C++ Inheritance Introduction

Some Functions are Not Inherited by derived classes

- Constructors
- Copy constructor
  - If not defined → default
- Assignment operator
  - If not defined → default
- Destructors

Assignment Operator Example

- Given "Derived" is derived from "Base":
- Derived& Derived::operator =(const Derived & rightSide)
- {
-         Base::operator =(rightSide);
-         …
- }
- Notice code line
  - Calls assignment operator from base class
    - This takes care of all inherited member variables
  - Would then set new variables from derived
  - class…

Copy Constructor Example

- Consider:
- Derived::Derived(const Derived& Object)
- : Base(Object), …
- {…}
- After : is invocation of base copy constructor
  - Sets inherited member variables of derived
  - class object being created
  - Note Object is of type Derived; but it's also of
  - type Base

Destructors in Derived Classes

- If base class destructor functions correctly
  - Easy to write derived class destructor
- When derived class destructor is invoked:
  - Automatically calls base class destructor!
  - So no need for explicit call
- So derived class destructors need only be
- concerned with derived class variables
  - And any data they "point" to
  - Base class destructor handles inherited data
  - automatically

Destructor Calling Order

- Consider:
- class B derives from class A
- class C derives from class B
- A ← B ← C
- When object of class C goes out of scope:
  - Class C destructor called 1st
  - Then class B destructor called
  - Finally class A destructor is called
- Opposite of how constructors are called

Multiple Inheritance

- Derived class can have more than one
- base class!
  - Syntax just includes all base classes
  - separated by commas:
  - class derivedMulti : public base1, base2
  - {…}
- Possibilities for ambiguity are endless!
- Dangerous undertaking!
  - I have never seen it used
  - DO NOT USE!

Summary- Not Inherited Functions

- Overloaded assignment operator
  - But can be invoked from derived class
- Constructors are not inherited
  - Are invoked from derived class’s constructor
- Destructors
  - Automatically called from most to derived to base

- Multiple Inheritance
  - Difficult to get right
  - Do not use!

Go to “10_Inheritence_Composition_model_professor” doc