Interfaces

13 Feb 2009
CMPT166
Dr. Sean Ho
Trinity Western University

TRINITY WESTERN UNIVERSITY

Quiz 3: 15mins

- Evaluate in C++: “My apple” < “MyPear” [2]
- Describe the two meanings of the static keyword in C++. [4]
- What is a pure virtual function in C++?
  How do you specify one in C++?
  Why might such a function be useful? [4]
- Come up with a situation where class inheritance would be useful.
  Design a class hierarchy with a superclass and at least two subclasses.
  Sketch a UML diagram and basic C++ code. [10]

Quiz 3: answers #1-2

- Evaluate in C++: “My apple” < “MyPear” [2]
  - True (<space> is less than 'P')
- Describe the two meanings of the static keyword in C++. [4]
  - On global entities: local binding: scope is limited to current file
  - On local vars inside functions: persistent storage: variable retains its value across calls to the function

Quiz 3: answers #3-4

- What is a pure virtual function in C++? [4]
  - No body (=0), must be overridden by subclasses
  - Superclass container specifies that all instances of any subclass must implement this method
- Come up with a situation where class inheritance would be useful.
  Design a class hierarchy with a superclass and at least two subclasses.
  Sketch a UML diagram and basic C++ code. [10]

Multiple inheritance (arity)

- C++ allows a subclass to inherit from more than one superclass:

```
class Horse { public void eat(); }
class Donkey { public void eat(); }
class Mule : public Horse, Donkey {} // both!
```

- How do disambiguate name collisions?

```
myMule.eat();     // which one?
```

  - Specify superclass name:

```
myMule.Horse::eat();
```

- C++, Python: arity is multiple.
  - Java: arity is single.

Review: abstract classes

- Abstract classes:
  - Too generic to define a real object
    - e.g., TwoDimensionalShape
  - Not intended to be directly instantiated
    - abstract classes have pure virtual methods:
      - No body defined; each subclass must implement
- Concrete classes:
  - Subclass of an abstract class, meant to be instantiated
    - e.g., Square, Circle, Triangle

e.g.: TwoDimensionalShape

- Abstract superclass: TwoDimensionalShape
  - Abstract method: draw()

```
class TwoDimensionalShape {
    virtual void draw() = 0;      // pure virtual
```

- Concrete subclasses: Circle, Square, Triangle
  - Each provide own implementation of draw()

```
class Circle : public TwoDimensionalShape {
    virtual void draw() { drawCircle( x, y, r ); }
}
class Square : TwoDimensionalShape {
    virtual void draw() { drawRect( x, y, w, h ); }
}
```

Interfaces (in Java)

- An interface is a set of methods provided by a class (which may implement several interfaces)
- C++ doesn't have explicit interfaces, but
- In Java: define a set of abstract methods

```
public interface drawableShape {
    public abstract void draw();
    public abstract double area();
}
```

- Classes implement these methods

```
public class Circle implements drawableShape {
    public void draw() { drawOval( x, y, r, r ); }
    public double area() { return 2 * Math.PI * r * r; }
```

Abstract classes vs. interfaces

- Abstract superclasses declare identity:
  - "Circle is a kind of TwoDimensionalShape"
  - Some languages do not allow multiple inheritance
  - Inherit methods, attributes;
    - Get protected access
- Interfaces declare capability:
  - "Circles know how to be drawableShapes"
  - May implement multiple interfaces
  - Interfaces are not ADTs (abstract data types)