C++: Inheritance II

Logistics

- Project
  - Part 2 (water) due Sunday, Oct 16th
- Questions?

Plan for this week

- This is "abstraction week"
  - Monday: Inheritance in C++
  - Tuesday: Exam
  - Today: Inheritance in C++ (cont’d)

Subclassing

- Define a more general class “Performer”.
- Both Actors and Musicians are specializations of Performer

Subclassing and Inheritance

- When you define a class as a subclass:
  - The subclass inherits all of the data members and methods of the superclass.
  - In addition, a subclass can have data/methods that are it’s own.
  - Inheritance is transitive:
    - I.e. If B is a subclass of A and C is a subclass of B, then C inherits the data/methods from both B and A.
Inheritance

• Behind the scenes
  – The memory allocated for an object of a derived class consists of:
    • An area for the base class’s data
    • An area for the derived class’s data

Inheritance

```cpp
class Performer
{
private:
    float basePay;
    char *name;
    char *talent;
    ...
public:
    Performer (char *name, char *talent);
    virtual float calculatePay();
}
```

```cpp
class Musician : public Performer
{
private:
    Instrument *axe;
    bool isRecorded;
    ...
public:
    Musician (char *name);
    virtual float calculatePay();
    virtual void setInstrument (Instrument *I);
}
```

Inheritance

• What the memory for Musician looks like

Inheritance

• Let’s add a Drummer
```cpp
class Drummer : public Musician
{
private:
    Drum kit[];
public:
    Drummer (char *name);
    virtual void setInstrument (Instrument *I);
}
```

Inheritance

• What the memory for drummer looks like

Inheritance and Construction

• When a member of a derived class is constructed, the constructor of it’s base class is called first
  – This fills in the memory area containing members of the base class.
Inheritance and Construction

Musician *M = new Musician("Ringo");

Musician’s constructor is called
Performer’s constructor is called
Musician’s constructor continues

Inheritance and Construction

Drummer *D = new Drummer("Ringo");

Drummer’s constructor is called
Musician’s constructor is called
Performer’s constructor is called
Musician’s constructor continues
Drummer’s constructor continues

Constructing Derived Class Objects

Musician::Musician (char *name) :
   Performer (name, "music"), axe (0),
   isRecorded(false),...
{
}
• There is no super function in C++
• Call to base class constructor required unless base class has a default constructor.

• Questions?

Slicing

• Recall that polymorphism can only be achieved using pointers rather than objects themselves.
• Attempts to copy a base class object with a derived class object will cause slicing.
  – Only the base class section of the derived object will be copied.

Slicing

Musician M ("Ringo");
Performer P (M);
Space allocated for P
Copy constructor called

Correct Polymorphism

Musician *M = new Musician ("Ringo");

Performer *P (M);
M allocated on heap
Pointer copy.
Slicing

• Questions?

Multiple Inheritance

• In C++, a class can be derived from more than 1 base class.
• Recall that C++ has no notion of an interface.

```
class MusicBuyer
{
    private:
        float cashOnHand;
        ...
    public:
        MusicBuyer (float cash);
        ...
}
```

Multiple Inheritance

```
class Musician : public Performer, public MusicBuyer
{
    public:
        Musician (char *name);
        ...
}
```

Multiple Inheritance

• Objects of classes that have multiple base classes will have a data section for each Base class.
  – The constructor for each base class will need to be called in the constructor for the derived class.

```
class Musician : public Performer, public MusicBuyer
{
    public:
        Musician (char *name);
        ...
}
```

Multiple Inheritance

```
class Musician : public Performer, public MusicBuyer
{
    public:
        Musician (char *name);
        ...
}
```

```
class Musician : public Performer, public MusicBuyer
{
    public:
        Musician (char *name);
        ...
}
```

```
class Musician : public Performer, public MusicBuyer
{
    public:
        Musician (char *name);
        ...
}
```
Multiple Inheritance

Musician *M = new Musician("Ringo");

Musician’s constructor is called
Performer’s constructor is called
MusicBuyer’s constructor is called
Musician’s constructor continues

Multiple Inheritance

• Why some think Multiple Inheritance is evil
  – Data member ambiguity
  – Can possibly derive from a base class twice
  – Extra work for compiler.
  – Most multiple inheritance heirarchies can be done using single inheritance
• Java chose to disallow
  – Created interfaces instead
  – I suggest you do the same!

Multiple Inheritance

• Questions

Summary

• Inheritance
  – What really goes on…
  – Slicing
• Multiple Inheritance

• Questions