Review of Object Oriented Programing

CS 247

University of Waterloo

cs247@student.cs.uwaterloo.ca

May 9, 2014

Overview

1. Inheritance
2. Polymorphism
   - Slicing
   - Dynamic Binding
3. Overloading and Overriding
   - Overloading
   - Overriding

Inheritance

Inheritance allows us to create classes which include the data members and functions of other classes. A class which inherits from another class is sometimes called a derived class while the class it inherits from is called a base class.

```
class Base{};

class Derived : public Base {};
```

Polymorphism

When we create a derived class that public inherits from a base class, we are able to use base class pointers to point to objects of the derived type.

Polymorphism

```
class  Meme {
    int i;
};

class Doge : public Meme {
    int wow;
};

int main() {
    Meme* derived = new Doge;

    delete derived;
}
```

Slicing

Object slicing occurs when only base class fields are copied from a subclass object. It can occur as a consequence of assigning a subclass object into a superclass variable, when passing subclass type objects by value to a parameter of the superclass type, or more deceptively when the superclass assignment operator is called between members of the subclass type.
Example : slicing.cpp

Static Binding

By default, the binding of functions occurs at compile time. This means that if we call a member function from a polymorphic base class pointer, the function called will be the base class implementation of that function regardless of the type of the object being pointed to when the program is running.
Example : staticBinding.cpp

Virtual

Virtual functions are an example of dynamic binding, their bindings are resolved at run time. When we create a base class with a virtual function and call that function from a polymorphic pointer the function to be called will be decided at run time based on the class being pointed to.
Example : virtual.cpp

Pure Virtual

When having an instance of a base class object doesn't make sense we can make that class abstract by having a pure virtual function. Pure virtual functions do not need a definition and prevent the class from being instantiated. Any subclass that is to be instantiated must implement the function.
Example : pureVirtual.cpp

Functions with Multiple Definitions

Overloading and overriding are two examples of when multiple functions with distinct definitions can exist with the same name.

Overloading

Overloading occurs when two functions with the same name but different parameters are defined. The parameters can vary in type or number.
Example : overload.cpp

Overriding

Overriding occurs when a function is declared in a subclass with the same number and type of parameters as a superclass function.
Example : override.cpp

The End