Christopher Aranda

11/2/18

Edward Peir

CSCI 3731

HW09

1. What is the difference between “downcasting” and “upcasting”?
   1. Upcasting refers to an object as if it were its superclass and converts a derived-class reference or pointer to a base-class. Downcasting converts a base-class pointer or reference to a derived-class pointer or reference.
2. Why would you mark a method as virtual? Why wouldn’t you?
   1. The pro to marking methods virtual would make a subclass override a method the way Java does. Without it, the class wouldn’t call the right destructor. Place the virtual keyword in the header file, but not the .cc file! A con to mark methods as virtual is that it’s slower.
3. What is the difference between a dynamic cast and a static cast? Why would you use one over the other?
   1. From the textbook definition, Static casts check what they can at compile time, but do not look at the actual type of the object at runtime. Dynamic casting looks at the actual type of the object being cast and checks if the cast to the requested type is legal. Dynamic casting is strongly recommended to use unless there’s a reason to not use it. Though dynamic casting is said to be slow, the reason to this is because of how it searches through the structure of the code’s inheritance tree. Static casting is much faster, but increases the chance of getting compile errors. It’s not wise to use this unless the cast given is valid and the programmer is desperate for faster speed.
4. When might you use a const cast?
   1. It’s used to get rid of a code’s const-ness. Pretty much breaking a promise to not modify a piece of data. One instance in which a programmer can use the const cast is when a caller calls a library in which the caller doesn’t control the source code. The author might have forgotten to mark a method argument as const. If there’s a const object, the method is unusable, even if the method doesn’t modify the object.
5. Suppose you have a Fruit base class and a Banana subclass. Suppose you have a banana declared like this:  
   Banana b;  
   What is the difference between what happens when you pass your banana to a method declared like this:

void eat(Fruit f);

vs. one declared like this:

void eat(const Fruit& f);.

The first statement returns nothing and its parameter is a Fruit f. The second one uses dynamic casting. returns nothing as well but it’s parameter is a memory address that points to a Fruit called f. (To be revised.)

1. What is the difference between how Java and C++ handle multiple inheritance?
   1. In Java, every class has one parent class. C++, on the other hand, can have no parent to multiple parents. Java has a “universal base class, ” whereas C++ does not.