Name:Osbaldo Gonzalez Jr.
Name:Sasha Saaed-Marmorstein
Looked at: Talal AlShumais And Jerry Li

Class Hierarchies – Worksheet

1. Explain the relationship between levels in an inheritance tree.
   All the child/sub classes are able to add to their public interface all the public interface methods of the base/parent class, assuming inheritance is public. In essence all subclasses of the base class are able to take on the responsibilities and actions of the base class. Of Course there can be sub sub classes that inherit from subclasses, which inherit from a base class; in essence making a tree of inheritance in which every class continues to get more and more detailed and specific to the programmer's needs.

2. What does it mean to say that the C++ language allows the programmer to create an inheritance forest, and not a tree?
   it means that multiple sets of functions/classes can be created that don't all inherit from each other in a linear way. inheritance from multiple classes is allowed.
3. What is a pure virtual member function? What is an abstract class?
   A pure virtual member function is not defined when it is first declared. Classes that inherit from the class where the pure virtual function is declared will have to define it, unless they themselves want to be abstract classes. An abstract class is a class with at least one a pure virtual member function.
4. What is a dynamic cast? What is the purpose of this operation?
   It is an operator that during run time converts an object of one data type to another and checks to make sure that it is safe to downcast. This can be useful for having pointers of one class become pointers to a derived class.

5. What are the problems that arise with statements that branch on different types? Why is it better to use polymorphism in this situation?
   with polymorphism, you would not have to rewrite your statement for each possible type, and instead you can write one statement that can interact with different variable types

6. When an overridden virtual member function is executed, the vtable helps determine which function to execute. Explain how this mechanism works.

   Since this is usually used to enact polymorphism, in essence what this does is ensure that even when an instance of a child class object is being “converted” and used through the interface of the base class, if the sub class has another overridden definition of the base class method, then the object’s vtable will point to that implementation, and use that implementation (it is very similar to how we would use function pointers, but in this case, it is being handled for us). If there is no new definition then the object’s vtable will simply point to the base classes’ implementation for the method. In essence the vtable is used to resolve what method to use for that object (even when “seen” as a base class object) during runtime, based on what is available.
7. What does it mean to say that a class uses multiple inheritance?

The derived class now takes upon the public interface (assuming public inheritance) of multiple "base" classes all in one. This allows for basically the programmer to shove the implementation of multiple classes into one without having to make multiple classes that only serve to inherit from one class, one by one growing it's interface, and then inheriting that class to the new class. Although, it will be noted that it is often not recommended, and can often lead to unforeseen complexity.

8. Why are the function name and data ambiguity problems inherent in multiple inheritance not an issue with single inheritance?

in multiple inheritance, since the 2 base classes could both have a function with the same name, such as display, we have to be clear about what we want to do when we want to display in the inherited class. if the inherited class was only using single inheritance there could be no conflicting definitions from base classes.

9. What is virtual inheritance? What problem does this feature address?
Virtual inheritance solves the problem of when a class inherits from two or more classes where those two classes also inherit from another base class. This tries to solve the diamond shaped inheritance problem (which is what I described in the first sentence). This allows the derived class (derived from the two or more classes which themselves are derived) to resolve which method to use: the method from the base class. In essence this allows for the implementation of a method (as an example) to trickle down all the way to the most derived class, even if the most derived class itself inherits from two other classes which inherit from the base class. This gets confusing quickly!

10. Investigate the use of multiple inheritance in the iostream hierarchy. Why does iostream inherit from both istream and ostream? Why doesn't fstream inherit from ifstream and ofstream?

istream and ostream both take output or send output to be displayed in the console. iostream allows both, so it can benefit from istream and ostream's member functions. ifstream is derived from istream and ofstream is derived from ostream. So rather than derive from those, fstream follows their pattern and derives from iostream. The reason that fstream doesn't inherit from ifstream and ofstream an instead chooses to inherent from iostream, is because the ifstream anb ofstream classes have class specific method implementations for reading and writing that might result in ambiguity for the derived fstream classes (kind of like the diamond shape problem from a previous question)