Constructor Chaining

Before a derived class constructor can do any of its work, it must first initialize **all of the base class data members**. This must happen **before** the derived constructor ever runs. (If this sounds familiar, it should. It is the same reason that your class data members are already initialized before the first line of your constructor runs!)

You needn't do anything special to make this happen. When your constructor runs, if **implicitly calls the default or no-argument constructor** in the base class as its first line of code, and that constructor calls **its** base class constructor, and so on, all the way up to the first class in your hierarchy. This is called **constructor chaining**.

So, consider for a second, the `Student` class constructor in the previous section. Even though not explicit in the source code, the following commands are executed when the `Student` constructor is invoked (after memory for both the `Person` part of the `Student` and the `Student` part of the `Student` has been allocated).

1. call the `Person` default constructor
2. call the `setName()` inherited member function
3. assign the `sid` parameter to the data member
4. print the diagnostic message

That's why you saw **three diagnostic messages** when **you only created two objects**. When you created the `Student` object named `steve`, the `Student` constructor **first** called the **default** constructor for the `Person` class.