Polymorphic Inheritance

Public inheritance is a form of specialization. The derived class inherits both the member functions and the data members from the base class, while optionally adding more of both. The derived class **IS-A specialized form** of the more general base class.

A derived class **may override** a `virtual` member function to add specialized behavior, as we did with `Student::toString()`. This is called **polymorphic inheritance**, it provides **specialized behavior** in response to the same messages.

Let's see if that's true. Let's use our simple `Person<-Student` hierarchy from the last few lessons and see what happens with some experiments. Click the Running Man on the left to open a copy of the lab for this lesson. Make sure you **Fork** it so that you have your own copy.

Change `toString()` in each class so it identifies the class at the beginning of the method. Here are the modified `toString()` member functions. Notice that this version of the `Student::toString()` no longer calls its base class version; it **entirely replaces it**.

```
string Person::toString() const
{
    return "Person::Name: " + name;
}

string Student::toString() const
{
    return "Student::Name: " + getName()
        + ", ID: " + to_string(studentID);
}
```