Specification Inheritance

In English, the word "inheritance" has several meanings. There is biological inheritance, where you inherit you eye color from your parents, but there is also cultural inheritance and legal inheritance.

Similarly, in C++ you've seen **specialization** inheritance, **polymorphic** inheritance and **implementation** inheritance. In this lesson, we'll look at another form of inheritance, called **specification inheritance**.

With specification inheritance, a base class may **specify a set of responsibilities** that a derived **must fulfill**, but not provide any actual implementation. **The interface** (method signatures) are inherited. This is similar to legal inheritance, where your grandparents may leave you some money to be used only for college.

The specification relationship is used **in combination** with regular specialization:

- the derived class **inherits the interface** of the base class, as in specification
- it also inherits a **default implementation** of, at least, some of its methods

A derived class **may** override a `virtual` member function to add specialized behavior, as we did with `Student::toString()`, **or**, it may be **required** to implement a particular member function, which could not be provided in the base class.