The override Keyword

While always a logic error for a derived class to redefine a non-virtual function, it **is not** a syntax error. C++ 11 added new **contextual keywords** that **allow the compiler to catch such logic errors** that previously were often hidden, and turn them into syntax errors that can be caught at compile time.

To tell the compiler that you **intend to override** a base class function, add the contextual keyword **override** to the end of the member function declaration like this:

```
std::string toString() const override;
```

Now, if you were to forget the **virtual** in the base class, trying to (incorrectly) override a non-virtual inherited member function, or misspell the name of the member function, or provide the wrong arguments, the **compiler catches those errors** and warns you when you compile, like this:

```
error: 'Student::toString()' marked override, but
    does not override std::string Person::toString()
```