【C++】Day78

Class	C++
Date	@April 1, 2022
Material	
Series Number	
Summary	

【Ch15】OOP

15.7.4 Inherited Constructors

A derived class inherits its base-class constructors by providing a `using` declaration that names its (direct) base class. As an example, we can redefine our `Bulk_quote` class to inherit its constructors from `Disc_quote`.

```
class Bulk_quote : public Disc_quote {
public:
  using Disc_quote::Disc_quote; // inherit Disc_quote's constructors
  double net_price(std::size_t) const;
};
```

When applied to a constructor, a `using` declaration causes the compiler to generate code. The compiler generates a derived constructor corresponding to each constructor in the base.

That is, for each constructor in the base class, the compiler generates a constructor in the derived class that has the same parameter list.

These compiler-generated constructors have the form:

```
derived(parms) : base(args) { }
```

If the derived class has any data members of its own, those members are default initialized.

15.8 Containers and Inheritance

When we use a container to store objects from an inheritance hierarchy, we generally must store those objects indirectly.

We cannot put objects of types related by inheritance directly into a container, because there is no way to define a container that holds elements of differing types.

As an example, assume we want to define a vector to hold several books that a customer wants to buy. It should be easy to see that we cannot use a vector that holds `Bulk_quote` objects. We cannot convert `Quote` objects to `Bulk_quote`, so we wouldn't be able to put `Quote` objects into that vector.

We also cannot use a vector that holds objects of type `Quote`. In this case, we can put `Bulk_quote` objects into the container. However, those objects would no longer be `Bulk_quote` objects:

```
vector<Quote> basket;
basket.push_back(Quote("0-201-82470-1", 50));
// ok, but copies only the Quote part of the object into basket
basket.push_back(Bulk_quote("0-201-54848-8", 50, 10, 0.25));
// calls version defined by Quote, prints 750, i.e., 15 * 50
cout << basket.back().net_price(15) << endl;
```