【C++】Day71

Class	C++
Date	@March 9, 2022
Material	
Series Number	
Summary	Derived Class Constructor, Member Using, Static Member, Inheritance Preventing, and Declaration

【Ch15】OOP

15.2 Defining Base and Derived Class

Derived-Class Constructors

Although a derived object contains members that it inherits from its base, it cannot directly initialize those members. A derived class must use a base-class constructor to initialize its base-class part.

Note: Each class controls how its members are initialized.

A derived-class constructor uses its constructor initializer list to pass arguments to a base-class constructor. For example, the `Bulk_Quote` constructor with four parameters.

```
Bulk_quote(const std::string &book, double p, std::size_t qty, double disc) :
  Quote(book, p), min_qty(qty), discount(disc) {}
```

The `Quote` constructor initializes the `Bulk_quote`'s base-class part(i.e., the `bookNo` and `price` members).

Next the direct members `min_qty` and d`iscount` are initialized.

Note: The base class is initialized first, and then the members of the derived class are initialized in the order in which they are declared in the class.

Using Members of the Base Class from the Derived Class

A derived class may access the `public` and `protected` members of its base class:

```
double Bulk_quote::net_price(size_t cnt) const {
  if(cnt >= min_qty)
    return cnt * (1 - discount) * price;
  else
    return cnt * price;
}
```

Inheritance and static Members

If a base class defines a `static` member, there is only one such member defined for the entire hierarchy.

```
class Base {
public:
  static void statmem();
};

class Derived: public Base {
  void f(const Derived&);
};
```

`static` members obey normal access control. If the member is `private`, then derived classes have no access to it.

Assuming the member is accessible, we can use a static member through either the base or derived

```
void Derived::f(const Derived &derived_obj) {
  Base::statmem(); // ok: Base defines statmem
  Derived::statmem(); // ok: Derived inherits statmem

  derived_obj.statmem(); //accessed through a Derived object
```

```
  statmem()p // accessed through this object
}
```

Declarations of Derived Classes

A derived class is declared like any other class. The declaration contains the class name but does not include its derivation list:

```
class Bulk_quote : public Quote; // error: derivation list cannot appear hear
class Bulk_quote; // ok
```

The purpose of a declaration is to make known that a name exists. The derivation list, and all other details of the definition, must appear together in the class body.

Classes Used as a Base Class

A class must be defined, not just declared, before we can use it as a base class:

```
class Quote; //declared but  not defined
// error: Quote must be defined
class Bulk_quote : public Quote {...};
```

The reason for this restriction should be easy to see: Each derived class contains, and may use, the members it inherits from its base class.

To use those members, the derived class must know what they are.

A base class can itself be a derived class:

```
class  Base {...};
class D1 : public Base {...};
class D2 : public D1 {...};
```

In this hierarchy, Base is a direct base to D1 and an indirect base to D2.

A direct base class is named in the derivation list. An indirect base is one that a derived class inherits through its direct base class.

Preventing Inheritance

Sometimes we define a class that we don't want others to inherit from. Under the new standard, we can prevent a class from being used as a case by following the class name with `final`:

```
class NoDerived final {...}; // NoDerived cannot be a base class
class Base{...};
class Last final : base{...}; // Last cannot be a bse class
class Bad : NoDerived {...}; // error: NoDerived is final
class Bad2 : Last {...}; // error: Last is final
```

Exercise

Exercise 15.4: Which of the following declarations, if any, are incorrect? Explain why.

```
class Base { ... };
(a) class Derived : public Derived { ... };
(b) class Derived : private Base { ... };
(c) class Derived : public Base;
```

(a): A class cannot inherit from itself

(b): Correct

(c): Derivation list cannot appear in the declaration of a class.

Exercise 15.5: Define your own version of the `Bulk_quote` class.

Exercise 15.6: Test your `print_total` function from the exercises in § 15.2.1 (p. 595) by passing both `Quote` and `Bulk_quote` objects o that function.

Exercise 15.7: Define a class that implements a limited discount strategy, which applies a discount to books purchased up to a given limit. If the number of copies exceeds that limit, the normal price applies to those purchased beyond the limit.

See 15_5.cpp, 15_6.cpp, 15_7.cpp for code