西北工业大学
NORTHWESTERN POLYTECHNICAL UNIVERSITY

C++程序设计
Programming in C++

1011018

主讲：魏英，计算机学院

类的定义

1、定义类

2、成员的访问控制

- C语言——面向过程的程序设计思想
  - 自顶向下，逐步求精；
  - 一个main函数+若干子函数。

- C++语言——面向对象的程序设计思想
  - 实现软件设计的产业化；
  - 自然界是由实体（对象）组成的；

- （1）抽象：对具体对象（问题）进行概括，抽出这一类对象的公共性质并加以描述的过程。
  - 数据抽象
  - 行为抽象
- （2）封装：将抽象出的数据成员、行为成员相结合，将他们视为一个整体——类。
  - 使用者不需要了解具体的实现细节，只需要通过接口使用类的成员即可。
- （3）继承与派生：保持原有类特性的基础上，进行更具体的说明。

- 类（class）是用户自定义数据类型。如果程序中要使用类类型（class type），必须根据实际需要定义，或者使用已设计好的类。

- C++定义一个类，其方法与定义一个结构体类型是相似的，一般形式为：

```
class 类名 { //类体
      成员列表
};
```

- 其中成员列表（member list）是类成员的集合，数目可以任意多，由具体应用确定。一对大括号{ }是成员列表边界符，与成员列表一起称为类体（class body）。类体后面必须用分号（；）结束。

- 每个类可以没有成员，也可以有多个成员。
- 类成员可以是数据或函数。
- 所有成员必须在类的内部声明，一旦类定义完成后，就没有任何其他方式可以再增加成员了。

- 类定义时必须给出各个数据成员（data member）的数据类型声明，其一般形式为：

```
class 类名 { //类体
        …
        数据成员类型 数据成员名列表; //数据成员声明
        …
};
```

- 声明时成员名列表允许为多个，用逗号（，）作为间隔，最后必须用分号（；）结束。

- 每个类还可以包含成员函数，能够访问类自身的所有成员。
- 面向对象程序设计一般将数据隐蔽起来，外部不能直接访问，而把成员函数作为对外界的接口，通过成员函数访问数据。即数据成员是属性，成员函数是方法，通过方法存取属性。
- 如果类中有成员函数，则声明成员函数是必需的，而定义成员函数则是可选的，因此类的成员函数有两种形式。

- ①在类中定义（也是声明）成员函数，形式如下：

```
class 类名 { //类体
    …
    返回类型 函数名(形式参数列表)//成员函数定义
    {
        函数体
    }
    …
};
```

- ②成员函数的声明在类中，定义在类外部，形式：

```
class 类名 { //类体
      …
      返回类型 函数名(类型1 参数名1,类型2 参数名2,…);
       //成员函数声明
      返回类型 函数名(类型1, 类型2,…);
      …
};
返回类型 类名::函数名(形式参数列表)
{     //成员函数定义在类外部实现
      函数体
}
```

【例25.1】

```
1  class Data { //Data类定义
2      void set(int d);
3      //成员函数原型声明, 与 void set(int); 等价
4      int get() { //成员函数类内部定义
5          return data;
6      } //get函数定义结束
7      int data; //数据成员
8  }; //Data类定义结束
9  void Data::set(int d) //成员函数类外部定义
10 {
11     data=d; //访问类的数据成员
12 }
13
```

- 类定义一般放在程序文件开头，或者放到头文件中被程序文件包含，此时这个定义是全局的。在全局作用域内，该定义处处可见，因此同作用域内的所有函数都可以使用它。

- 类定义也可以放到函数内部或局部作用域中，此时这个定义是局部的。若在函数内部有同名的类定义，则全局声明在该函数内部是无效的，有效的是局部定义的。

【例25.2】

```
class Data { //全局的Data类定义
    void show(); //成员函数原型声明
    int data; //数据成员
};  //Data类定义结束
void fun()
{    //全局Data在fun函数中无效，有效的是局部定义的Data
    class Data { //局部的Data类定义
        void show() { cout<<data; }//set函数定义
        int data; //数据成员
    };  //Data类定义结束
}
```

- C++规定，在局部作用域中声明的类，成员函数必须是函数定义形式，不能是原型声明。一般地，由于类是为整个程序服务的，因此很少有将类放到局部作用域中去定义。

- 类定义向编译器声明了一种新的数据类型，该数据类型有不同类型的数据成员和成员函数。因此尽管数据成员类似变量的定义，但类型声明时并不会产生该成员的实体，即为它分配存储空间。

- 对类的成员进行访问，来自两个访问源：类成员和类用户。

- 类成员指类本身的成员函数，类用户指类外部的使用者，包括全局函数、另一个类的成员函数等。

- 无论数据成员还是函数成员，类的每个成员都有访问控制属性，由以下三种访问标号说明：public（公有的）、private（私有的）和protected（保护的）。

- 公有成员用public标号声明，类成员和类用户都可以访问公有成员，任何一个来自类外部的类用户都必须通过公有成员来访问。显然，public实现了类的外部接口。

- 私有成员用private标号声明，只有类成员可以访问私有成员，类用户的访问是不允许的。显然，private实现了私有成员的隐蔽。

- 保护成员用protected标号声明，在不考虑继承的情况下，protected的性质和private的性质一致，但保护成员可以被派生类的类成员访问。

- 成员访问控制是C++的类和结构体又一个重要特性。加上访问标号，类定义更一般的形式为：

```
class 类名 { //类体
public: //公有访问权限
    公有的数据成员和成员函数
protected: //保护访问权限
    保护的数据成员和成员函数
private: //私有访问权限
    私有的数据成员和成员函数
};
```

- 如果没有声明访问控制属性，类所有成员默认为private，即私有的。例如：

```
class Data {
    int a, b; //默认为私有的，外部不能直接访问
public://公有的，外部可以直接访问
    void set(int i, int j,int k,int l,int m,int n)
     { a=i,b=j,c=k,d=l,e=m,f=n;}
protected://保护的，外部不能直接访问，派生类可以访问
    int c, d;
private://私有的，外部不能直接访问，派生类也不可以访问
    int e, f;
};
```

- 说明：
- （1）在定义类时，声明为public、private或protected的成员的次序任意。
- （2）在一个类体中不一定都包含public、private或protected部分，可以只有public、private、protected部分或任意组合。
- （3）关键字public、private、protected可以分别出现多次，即一个类体可以包含多个public、private或protected部分。但更通用的做法是将相同访问控制属性的成员集中在一起来写。
- （4）实际编程中，为了使程序清晰，每一种成员访问限定符在类体中只出现一次，且按照public、protected、private顺序组织，形成访问权限层次分明的结构。

C程序设计