关于面向对象编程
类和对象

第2章 类和对象

2.1 面向对象程序设计方法概述
2.2 类的声明和对象的定义
2.3 类的成员函数
2.4 对象成员的引用

2.1 面向对象程序设计方法概述

抽象是对具体对象（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。

- 先注意问题的本质及描述，其次是实现过程或细节。
- 数据抽象：描述某类对象的属性或状态（对象相互区别的物理量）。
- 代码抽象：描述某类对象的共有的行为特征或具有的功能。
- 抽象的实现：通过类的声明。

2.1 面向对象程序设计方法概述

- 抽象实例——钟表
- 数据抽象： int Hour,int Minute,int Second
- 代码抽象： SetTime(),ShowTime()

钟表类

```
class  Clock
{
  public:
   void SetTime(int NewH,int NewM,int NewS);
   void ShowTime();
  private:
   int Hour,Minute,Second;
};
```

2.1 面向对象程序设计方法概述

- 封装：将抽象出的数据成员、代码成员相结合，将它们视为一个整体。
  - 目的是曾强安全性和简化编程，使用者不必了解具体的实现细节，而只需要通过外部接口，以特定的访问权限，来使用类的成员。
  - 实现封装：类声明中的{}

2.1 面向对象程序设计方法概述

- 继承与派生
- 是C++中支持层次分类的一种机制，允许程序员在保持原有类特性的基础上，进行更具体的说明。

2.1 面向对象程序设计方法概述

- 多态：同一名称，不同的功能实现方式
- 目的：达到行为标识统一，减少程序中标识符的个数。
- 实现：重载函数和虚函数

2.2 类的声明和对象的定义

```
class A
{   int x,y;
    public:
       A(int a){  x=a; cout<<“1\n”;}
       A(int a, int b) {  x=a,y=b;cout<<“2\n”;}
};
A   a1(3);
void  f(void)  {   A  b(2,3);}
void main(void)
{  A  a2(4,5);
   f();  f();
}
```

```
1
2
2
2
```

2.2 类的声明和对象的定义

类的定义

- 类是一种复杂的数据类型，它是将不同类型的数据和与这些数据相关的运算封装在一起的 集合体。
- 类将一些数据及与数据相关的函数封装在一起，使类中的数据得到很好的“保护”。在大型程序中不会被随意修改。

2.2 类的声明和对象的定义

类的定义格式：

```
class  类名
{    private :
        成员数据;
        成员函数;
     public :
        成员数据;
        成员函数;
    protected:
        成员数据;
        成员函数;
};
```

关键字

类名

私有

公有

保护

分号不能少

```
class  Student
{ private :
     char Name[20];
     float Math;
     float Chiese;
  public :
     float  average;
     void SetName(char *name);
     void  SetMath(float math);
     void  SetChinese(float ch);
     float GetAverage(void);
};
```

2.2 类的声明和对象的定义

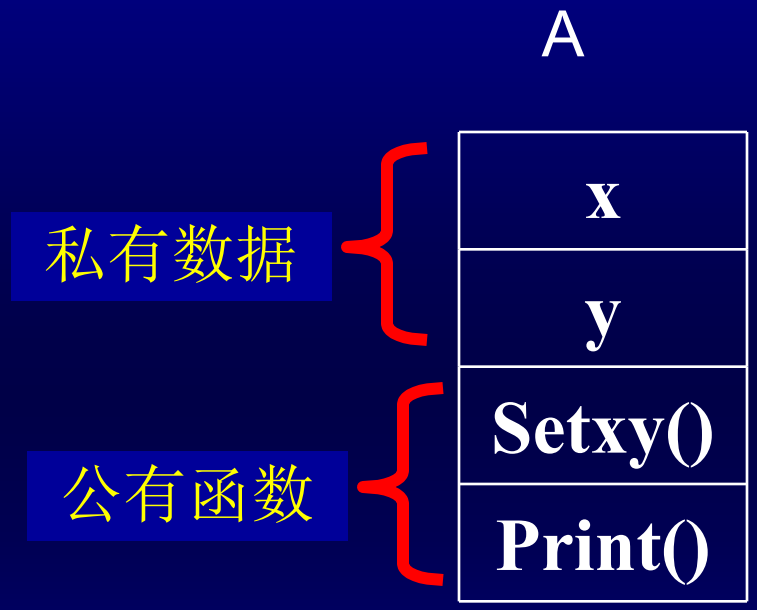
- priviate限定的成员称为私有成员，私有成员只能在该类的内部使用，即私有成员数据只允许该类中成员函数使用，私有成员函数只能被该类内的成员函数调用；类就相当于私有成员的作用域。
- public限定的成员称为公有成员，公有成员的数据或函数不受类的限制，可以在类内或类外自由使用；对类而言是透明的。
- protected限定的成员称为保护成员，只允许在类内及该类的派生类中使用保护的数据或函数。即保护成员的作用域是该类及该类的派生类。

2.2 类的声明和对象的定义

- 每一个限制词(private等)在类体中可使用多次。一旦使用了限制词，该限制词一直有效，直到下一个限制词开始为止。
- 如果未加说明，类中成员默认的访问权限是private，即私有的。

2.2 类的声明和对象的定义

```
class  A
{     float  x, y;
  public:
      void Setxy(float a,float  b)
     {  x=a;   y=b;   }
      void  Print(void)
    {   cout<<x<<'\t'<<y<<endl; }
};
```



在类外不能直接使用 x 或 y ，必须通过Setxy()给 x 或 y 赋值，通过Print()输出 x 或 y 。

2.2 类的声明和对象的定义

在定义一个类时，要注意如下几点：

1、类具有封装性，并且类只是定义了一种结构（样板），所以类中的任何成员数据均不能使用关键字extern，register限定其存储类型。

2、在定义类时，只是定义了一种导出的数据类型，并不为类分配存储空间，所以，在定义类中的数据成员时，不能对其初始化。

```
如：class  Test
    {   int  x=5,y=6;   //是不允许的
        extern  float  x;  //是不允许的
    }
```

2.2 类的声明和对象的定义

对象

- 只有在定义了属于类的变量后，系统才会为类的变量分配空间。
- 对象是类的实例，定义对象之前，一定要先说明该对象的类。

类的变量我们称之为对象。

2.2 类的声明和对象的定义

- 不同对象占据内存中的不同区域，它们所保存的数据各不相同，但对成员数据进行操作的成员函数的程序代码均是一样的。
- 对象的定义格式：

  类名　对象名1，对象名2，…；

  class　类名　对象名1，对象名2，…；

  例 Student　st1，st2；

  类名　　对象名

- 在建立对象时，只为对象分配用于保存数据成员的内存空间，而成员函数的代码为该类的每一个对象所共享。

以上内容仅为本文档的试下载部分，为可阅读页数的一半内容。如要下载或阅读全文，请访问：https://d.book118.com/798116131075006072