程序设计实习

郭炜　微博 http://weibo.com/guoweiofpku
http://blog.sina.com.cn/u/3266490431
刘家瑛 微博 http://weibo.com/pkuliujiaying

静态成员变量和静态成员函数

基本概念

- 静态成员：在说明前面加了static关键字的成员。

```
class CRectangle
{
        private:
             int w, h;
            static int nTotalArea;   //静态成员变量
             static int nTotalNumber;
        public:
             CRectangle(int w_,int h_);
              ~CRectangle();
             static void PrintTotal();   //静态成员函数
};
```

基本概念

- 普通成员变量每个对象有各自的一份，而静态成员变量一共就一份，为所有对象共享。

CRectangular C1, C2
C1.w和C2.2不共享
C1.nTotalArea和C2.nTotalArea共享

sizeof 运算符不会计算静态成员变量。

```
class CMyclass  {
      int n;
      static int s;
};
```

则 sizeof(CMyclass) 等于 4

基本概念

- 普通成员变量每个对象有各自的一份，而静态成员变量一共就一份，为所有对象共享。
- 普通成员函数必须具体作用于某个对象，而静态成员函数并不具体作用与某个对象。
- 因此静态成员不需要通过对象就能访问。

如何访问静态成员

1） 类名::成员名

CRectangle::PrintTotal();

2） 对象名.成员名

CRectangle r; r.PrintTotal();

3） 指针->成员名

CRectangle * p = &r; p->PrintTotal();

4） 引用.成员名

CRectangle & ref = r; int n = ref.nTotalNumber;

基本概念

- 静态成员变量本质上是全局变量，哪怕一个对象都不存在，类的静态成员变量也存在。
- 静态成员函数本质上是全局函数。
- 设置静态成员这种机制的目的是将和某些类紧密相关的全局变量和函数写到类里面，看上去像一个整体，易于维护和理解。

静态成员示例

考虑一个需要随时知道矩形总数和总面积的图形处理程序

可以用全局变量来记录总数和总面积

用静态成员将这两个变量封装进类中，就更容易理解和维护

```
class CRectangle
{
    private:
        int w, h;
        static int nTotalArea;
        static int nTotalNumber;
    public:
        CRectangle(int w_,int h_);
        ~CRectangle();
        static void PrintTotal();
};
```

```
CRectangle::CRectangle(int w_,int h_)
{
    w = w_;
    h = h_;
    nTotalNumber ++;
    nTotalArea += w * h;
}
CRectangle::~CRectangle()
{
    nTotalNumber --;
    nTotalArea -= w * h;
}
void CRectangle::PrintTotal()
{
    cout << nTotalNumber << "," <<   nTotalArea << endl;
}
```

```
int CRectangle::nTotalNumber = 0;
int CRectangle::nTotalArea = 0;
// 必须在定义类的文件中对静态成员变量进行一次说明
//或初始化。否则编译能通过，链接不能通过。
int main()
{
        CRectangle r1(3,3), r2(2,2);
        //cout << CRectangle::nTotalNumber; // Wrong , 私有
        CRectangle::PrintTotal();
        r1.PrintTotal();
        return 0;
}
```

输出结果：

```
2,13
2,13
```

注意事项

- 在静态成员函数中，不能访问非静态成员变量，也不能调用非静态成员函数。

```
void CRectangle::PrintTotal()
{
        cout << w << "," << nTotalNumber << "," << nTotalArea << endl;  //wrong
}
CRetangle::PrintTotal(); //解释不通，w 到底是属于那个对象的？
```

```
CRectangle::CRectangle(int w_,int h_)
{
    w = w_;
    h = h_;
    nTotalNumber ++;
    nTotalArea += w * h;
}
CRectangle::~CRectangle()
{
    nTotalNumber --;
    nTotalArea -= w * h;
}
void CRectangle::PrintTotal()
{
    cout << nTotalNumber << "," <<   nTotalArea << endl;
}
```

此CRectangle类写法，有何缺陷？

➢在使用CRectangle类时，有时会调用复制构造函数生成临时的隐藏的CRectangle对象

调用一个以CRectangle类对象作为参数的函数时，

调用一个以CRectangle类对象作为返回值的函数时

➢ 临时对象在消亡时会调用析构函数，减少nTotalNumber 和nTotalArea的值，可是这些临时对象在生成时却没有增加nTotalNumber 和 nTotalArea的值。

- 解决办法：为CRectangle类写一个复制构造函数。

```
CRectangle :: CRectangle(CRectangle & r )
{
        w = r.w;  h = r.h;
        nTotalNumber ++;
        nTotalArea += w * h;
}
```