**c++中冒号(:)和双冒号(::)的使用说明**

1.c++中冒号的用法

（1）表示机构内位域的定义（即该变量占几个bit空间）

代码如下:

typedef struct \_XXX{

          unsigned char a:4;

          unsigned char c;

} ; XXX

（2）构造函数后面的冒号起分割作用，是类给成员变量赋值的方法，初始化列表，更适用于成员变量的常量const型。

代码如下:

struct \_XXX{

          \_XXX() : y(0xc0) {}

};

（3） public:和private:后面的冒号，表示后面定义的所有成员都是公有或私有的，直到下一个"public:”或"private:”出现为止。"private:"为默认处理。

（4）类名冒号后面的是用来定义类的继承。

代码如下:

class 派生类名 : 继承方式 基类名

{

    派生类的成员

};

继承方式：public、private和protected，默认处理是 public。

2．双冒号（::）用法

（1）表示“域操作符”

例：声明了一个类A，类A里声明了一个成员函数void f()，但没有在类的声明里给出f的定义，那么在类外定义f时，

就要写成void A::f()，表示这个f()函数是类A的成员函数。

（2）直接用在全局函数前，表示是全局函数

例：在VC里，你可以在调用API 函数里，在API函数名前加：：

（3）表示引用成员函数及变量，作用域成员运算符

例：System::Math::Sqrt() 相当于System.Math.Sqrt()

VC中如下

::是C++里的“作用域分解运算符”。比如声明了一个类A，类A里声明了一个成员函数voidf()，但没有在类的声明里给出f的定义，那么在类外定义f时，就要写成voidA::f()，表示这个f()函数是类A的成员函数。

:: 一般还有一种用法，就是直接用在全局函数前，表示是全局函数。当类的成员函数跟类外的一个全局函数同名时，考试,大提示在类内定义的时候，打此函数名默认 调用的是本身的成员函数；如果要调用同名的全局函数时，就必须打上::以示区别。比如在VC里，你可以在调用API函数时，在API函数名前加::。