8.7.1　构造函数初始化语句

默认情况下，在构造对象时，将调用基类的无参数构造函数。但构造函数可以重载，所以基类可能有一个以上的构造函数。如果希望派生类使用一个指定的基类构造函数而不是无参数构造函数，必须在构造函数初始化语句中指定它。

有两种形式的构造函数初始化语句。

- 第一种形式使用关键字 base 并指明使用哪一个基类构造函数。
- 第二种形式使用关键字 this 并指明应该使用当前类的哪一个构造函数。

基类构造函数初始化语句放在冒号后面，跟在类的构造函数声明的参数列表后面。构造函数初始化语句由关键字 base 和要调用的基类构造函数的参数列表组成。

例如，下面的代码展示了类 MyDerivedClass 的构造函数。

- 构造函数初始化语句指明要使用有两个参数的基类构造函数，并且第一个参数是一个 int，第二个参数是一个 string。
- 基类参数列表中的参数必须在类型和顺序方面与已定的基类构造函数的参数列表相匹配。

```
                                    构造函数初始化语句
                                          ↓
public MyDerivedClass( int x, string s ) : base( s, x )
{                                          ↑
   ...                                   关键字
```

当声明一个不带构造函数初始化语句的构造函数时，它实际上是带有 base()构造函数初始化语句的简写形式，如图 8-12 所示。这两种形式是语义等价的。

```
class MyDerived: MyBase
{
   MyDerived()
   {
      ...
   }
   ...

隐式使用基类构造函数
MyBase()的构造函数
```

```
class MyDerived: MyBase
{
   MyDerived() : base()
   {
      ...
   }
   ...

显式使用基类构造函数
MyBase()的构造函数
```

图 8-12　等价的构造函数形式

另外一种形式的构造函数初始化语句可以让构造过程（实际上是编译器）使用当前类中其他的构造函数。例如，如下代码所示的 Myclass 类包含带有一个参数的构造函数。但这个单参数的构造函数使用了同一个类中具有两个参数的构造函数，为第二个参数提供了一个默认值。

```
                                  构造函数初始化语句
                                          ↓
public MyClass(int x): this(x, "Using Default String")
{                        ↑
   ...                 关键字
}
```