

编程实战与技巧系列

2



Visual C++ .NET

精彩案例 237

北京希望电子出版社 总策划
臧桂鹏 编写



北京希望电子出版社
Beijing Hope Electronic Press
www.bhpe.com.cn

编程实战与技巧系列

2

Visual C++ .NET

精彩案例 237

北京希望电子出版社 总策划
臧桂鹏 编写



北京希望电子出版社
Beijing Hope Electronic Press
www.bhp.com.cn

内 容 简 介

本书以 237 个 Visual C++ 精彩实例全面解析了 C++ 语言、编程的思路、方法和技巧。

该书内容涉及 Visual C++ 编程语言，文档与视图结构，界面、对话框与控件，菜单、工具栏和状态栏，文件与系统，COM 组件，数据库，多媒体，ATL 与 ActiveX 控件，以及网络和 Internet 编程等各个方面。选例经典而具有代表性，分析透彻，编程逻辑思路简练，具有很强的实用性、可操作性和参考价值。

本书非常适合 Visual C++ 初学者和中级程序员，亦可作为业余爱好者、高校计算机专业师生自学、教学用书，以及专业程序员参考书。

本版 CD 内容为本书 237 个实例的全部源代码。

盘 书 系 列 名：编程实战与技巧系列（2）

盘 书 名：Visual C++ .NET 精彩案例 237

总 策 划：北京希望电子出版社

文 本 著 作 者：藏桂鹏

责 任 编 辑：但明天

出版、发行者：北京希望电子出版社

地 址：北京市海淀区知春路甲 63 号卫星大厦三层 100080

网 址：www.bhp.com.cn E-mail: lwm@bhp.com.cn

电 话：010-62520290, 62521724, 62528991, 62630301, 62524940, 62521921, 82610344（发行）
010-82675588-202（门市） 010-82675588-501, 82675588-201（编辑部）

经 销：各地新华书店、软件连锁店

排 版：希望图书输出中心 马 君

CD 制 作 者：希望多媒体开发中心

CD 测 试 者：希望多媒体测试部

CD 产 生 者：北京中华新联光盘有限责任公司

文 本 印 刷 者：北京媛明印刷厂

开 本 / 规 格：787 毫米×1092 毫米 1/16 26.5 印张 611 千字

版 次 / 印 次：2003 年 3 月第 1 版 2003 年 3 月第 1 次印刷

印 数：1~3000 册

本 版 号：ISBN 7-89498-045-5

定 价：43.00 元（本版 CD）

说明：凡我社产品如有残缺，可执相关凭证与本社调换。

前 言

Visual C++是 Microsoft 公司开发的 Microsoft Visual Studio 套件的一部分, 它提供了一个集成的开发界面和许多有效的辅助开发工具, Visual C++.NET 是目前最新版本。Visual C++.NET 开发平台是一个具有高度综合性的软件开发工具, 在高性能的执行效率与底层控制和快速可视化开发方面均表现出色。其应用程序向导为很大一部分类型的程序提供框架代码, 用户不用编写很多繁琐的程序代码。

现在的图书市场上关于 Visual C++的书籍可谓浩如烟海, 然而真正能够满足读者需要的却是少之又少, 如果您正在寻找一本简明实用、内容丰富的参考书, 本书将是一个最好的选择。

作者从实际的应用开发中精心选取了大约 250 个典型的、非常实用的编程技术, 涉及了 Visual C++程序开发的几乎各个方面, 每种编程技术作为一个相对完整的知识点, 由浅入深, 简明透彻, 从基础方法到高级编程技巧, 内容丰富翔实, 能够满足不同层次读者的各种需要。

每个知识点代表了一种典型的编程技术, 熟练掌握它们的同时, 读者还可以得到很多的编程的启发, 通过大胆创新、组合扩充, 读者完全可以创建自己的很有特色的解决方案, 这也正是作者选取部分知识点, 借以抛砖引玉的目的。

本书内容包括 10 章, 主要包括以下内容:

第 1 章是 C++语言编程基础, 这是本书的语言基础, 简要介绍了 C++编程的特点、方法和注意事项, 有利于深入的学习 Visual C++编程。

第 2、3、4 章分别介绍文档与视图结构、对话框与控件编程以及菜单、工具栏和状态栏的编程, 全面分析了 Visual C++的界面部分的编程方法。

第 5 章是文件与系统编程, 主要介绍常用的文件操作以及系统编程的方法。

第 6、7 章分别介绍 COM 组件、ATL 与 ActiveX 控件编程, COM 技术应用非常广泛, 是目前的主流编程技术之一。

第 8、9、10 章分别介绍多媒体技术、Internet 网络开发和数据库编程等内容。

本书由臧桂鹏执笔编写, 此外, 张忠东、李晓裳、范智育、王宏生、李毅、王瑾、吴浩等同志在整理材料方面给予了作者很大的帮助, 在此一并致以感谢!

由于时间仓促, 加之编者的水平有限, 缺点和错误在所难免, 恳请专家和广大读者不吝赐教, 批评指正。

编 者
2002.8

目 录

第1章 C++语言编程基础	1	35 创建动态切分窗口	63
1 C++的封装性	2	36 创建静态切分窗口	63
2 C++的继承性	3	37 使用 CScrollView 类	66
3 C++的多态性	4	38 使程序处于最前面	68
4 类的声明	8	39 移动窗口	68
5 初始化对象	9	40 创建不规则窗口	70
6 虚函数	11	41 为文档创建多个视图	73
7 运算符重载	13	42 获知与 FormView 关联的改变	73
8 使用静态变量	16	43 实现橡皮区矩形	74
9 函数重载	18	第3章 对话框与控件编程	76
10 使用函数指针实现回调	20	44 理解对话框	77
11 定义和使用函数对象	21	45 生成动态对话框	79
12 处理内联虚函数	24	46 对话框显示不出来	81
13 理解位运算符	26	47 在对话框中显示图片	82
14 给表达式赋值	29	48 改变对话框的背景颜色	83
15 理解字符串	31	49 创建模式对话框	84
16 C++数组和指针	32	50 创建非模式对话框	85
第2章 文档与视图结构	35	51 制作提示对话框	86
17 在窗口中心输出字符串	36	52 控制对话框的大小	88
18 设置窗口的初始化大小	37	53 为对话框加入位图按钮	88
19 用 SDI 实现两个文档模板	38	54 改变控件的颜色	90
20 设置窗口最小化显示	40	55 实现彩色按钮	91
21 改变视图背景	40	56 生成自绘制的控件类	94
22 控制主窗口的最大和最小尺寸	41	57 运行时指定对话框的按钮	96
23 使用多个定时器	43	58 在对话框中使用菜单和工具栏	96
24 创建文档模板	45	59 使用组合框控件	99
25 定制文档和视图类	47	60 定制编辑框控件中的字符	100
26 改变窗口图标	50	61 检验列表框是否滚动	101
27 实现动画图标	52	62 使用列表框控件	102
28 启动时自动打开上次的文档	54	63 使用选项卡控件	103
29 启动时不创建空文档	55	64 判断树形控件的展开和收缩	105
30 理解串行化	56	65 使用树形控件	107
31 以不同格式保存及显示文件	57	66 使用列表控件	109
32 设置视图类的空背景画刷	58	67 单选按钮和复选框	111
33 在属性页中添加字体对话框	59	68 使用 UpdateData	114
34 理解切分窗口	61	69 创建对话框	114

第4章 菜单、工具栏和状态栏 117

70 创建自定义菜单	117
71 处理弹出菜单消息	118
72 创建动态菜单	120
73 给系统菜单添加菜单项	122
74 响应鼠标消息	123
75 添加自定义消息	124
76 消息传递和消息循环	125
77 确定菜单占据的行数	126
78 使用浮动菜单	127
79 动态追加菜单项	128
80 获取菜单弹出的位置	129
81 分开类型的 MRU 菜单	131
82 控制菜单的大小	132
83 创建浮动工具栏	133
84 更新工具栏的状态	135
85 创建自定义工具栏	137
86 在工具栏中嵌入组合框	138
87 控制工具栏的拖动停靠	139
88 不加载菜单、工具栏和状态栏	142
89 在工具栏上添加文本标签	144
90 工具栏停靠	146
91 让工具栏显示 256 图像	148
92 在状态栏中显示进度条	151
93 在状态栏显示系统时间	153
94 在状态栏中显示鼠标位置	154

第5章 文件与系统编程 156

95 文件读写	157
96 使用文件对话框	158
97 遍历整个目录树	159
98 打开文件提示	162
99 调用 html 文件	163
100 修改目录的日期和时间	164
101 获取文件或文件夹属性	167
102 将路径转换为长路径名	169
103 分割文件成多个小文件	171
104 合并多个文件成一个可执行文件	175
105 实现文件拖放	182
106 访问和修改注册表	183
107 使用注册表保存信息	186

108 识别操作系统环境	187
109 检测硬件设备	188
110 使改变的鼠标光标不闪烁	190
111 使用应用程序模拟键盘和鼠标按键	191
112 设置系统时间	192
113 将应用程序的图标加入到系统 托盘中	193
114 设定计算机名称	195
115 创建临时文件	196
116 启动和等待线程结束	197
117 检测系统中的当前进程	199

第6章 COM 组件编程 202

118 理解 COM 本质	203
119 重用 C++ 对象	204
120 定义 COM 接口	206
121 使用接口描述语言 IDL	209
122 定义 IUnknown 接口	210
123 实现 IUnknown 接口	211
124 获取接口指针	214
125 定义 IClassFactory 接口	216
126 创建永久对象	217
127 利用类厂创建 COM 对象	219
128 对 COM 库进行初始化	220
129 实现包容	221
130 实现聚合	222
131 COM 客户如何使用 COM 对象	224
132 定义接口映射表	225
133 类厂在 MFC 中实现	226
134 使用 MFC 建立 COM 组件	227
135 测试 COM 组件	229
136 编写 COM 组件	232

第7章 数据库编程 236

137 连接 ODBC 数据源	237
138 动态加载 ODBC 数据源	238
139 处理记录集与对应表	239
140 查询 ODBC 数据源中的数据	240
141 在 MFC ODBC 中进行事务处理	243
142 配置 ODBC 数据源	244
143 利用 SQL 语句删除记录	246
144 绕过 ODBC 口令提问	247



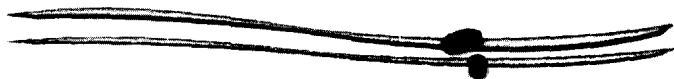
145	分配连接句柄.....	247	182	理解 ATL 技术.....	325
146	执行 SQL 语句.....	248	183	使用 ATL 窗口类.....	327
147	释放语句句柄.....	250	184	创建 ATL 项目.....	327
148	断开并释放数据源连接.....	250	185	使用 ATL 实现一个窗口.....	330
149	使用 DAO 进行数据库编程.....	251	186	实现一个对话框.....	331
150	操作数据库对象.....	253	187	实现容器窗口.....	332
151	使用 DAO 进行事务处理.....	255	188	增加一个连接点.....	333
152	直接调用 DAO 函数.....	257	189	创建 ATL Server 项目.....	334
153	使用 ADO 访问数据库.....	257	190	使用 ATL 实现接口.....	336
154	利用 UDL 文件建立 ADO 连接.....	262	191	创建 ATL Server Web 项目.....	340
155	使用 ADO 操作记录集.....	263	192	添加和改变 ActiveX 控件储备事件.....	341
156	使用 ADO 进行事务处理.....	266	193	添加 ActiveX 控件的自定义事件.....	342
157	使用 ADO 执行 SQL 命令.....	267	194	创建 ActiveX 控件项目.....	342
158	获取数据链接的属性.....	268	195	定义 ActiveX 控件的属性.....	344
159	降低记录集文件数量.....	272	196	显示 ActiveX 控件的属性页.....	345
160	建立自定义数据库类.....	275	197	创建 ActiveX 容器程序.....	346
第 8 章 多媒体编程.....	278		198	在容器程序中管理嵌入对象.....	347
161	多媒体系统的关键技术.....	279	199	使用鼠标来修改对象.....	349
162	多媒体的文件格式.....	280	200	创建 ActiveX 服务器项目.....	353
163	创建字体对象.....	281	第 10 章 Internet 应用程序编程....	358	
164	显示旋转文本.....	282	201	初始化 WinSock.....	359
165	输出空心字.....	284	202	实现网络聊天室服务器.....	360
166	显示渐变字.....	286	203	创建套接字.....	361
167	输出艺术字.....	288	204	传递套接字描述符和指针.....	362
168	制作应用程序真彩封页.....	291	205	从数据报套接字接收数据.....	364
169	设置窗口字体属性.....	294	206	向数据报套接字发送数据.....	364
170	实现马赛克效果.....	297	207	流式套接字连接发送和接收数据.....	365
171	实现浏览位图.....	300	208	异步接收数据.....	366
172	创建 OpenGL 项目.....	302	209	使用 WinInet 函数.....	368
173	播放视频文件.....	305	210	连接 Internet 服务器.....	369
174	播放无声 AVI 动画文件.....	308	211	回调函数与句柄建立链接.....	370
175	创建多媒体播放器.....	309	212	实现客户端通信功能.....	371
176	制作 MIDI 文件播放程序.....	311	213	从 FTP、HTTP 或 Gopher 服务器中获取 数据.....	374
177	利用 MCI 播放大型 WAV 文件.....	313	214	从文件句柄读取数据并移动文件指针..	375
178	制作 CD 播放器.....	314	215	获取和设置 FTP 服务器的当前目录.....	376
179	访问 MCI.....	317	216	FTP 下载文件.....	377
180	创建纹理场景.....	319	217	FTP 上传文件.....	380
181	绘制 Bezier 线框曲面.....	320	218	FTP 打开文件.....	382
第 9 章 ATL 与 ActiveX 控件编程....	324				



219	发送 HTTP 请求	383	228	控制浏览器	399
220	处理 HttpSendRequest()函数的调用 错误	384	229	使用 POP 协议接收电子邮件	400
221	获取服务器的响应信息	385	230	处理发送和接收数据的超时	402
222	实现 HTTP 协议	386	231	创建 Internet 服务器扩展程序	403
223	使用 CInternetSession 对象检索文件	389	232	从 Internet 上下载文件	404
224	HTTP 服务器处理	390	233	实现串行通信	406
225	Gopher 服务器处理	393	234	在单线程中实现串口通信	407
226	发送电子邮件	395	235	在多线程下实现串行通信	409
227	实现网页浏览	398	236	读取网卡的 MAC 地址	410
			237	使用 CSockets 进行文件传送	411



第1章 C++语言编程基础



在学习 Visual C++.NET 编程之前,应该了解一下 C++语言的基础知识,尤其是面向对象的一些基本概念。本章简要介绍了 C++编程的特点、方法和注意事项。

读者应该熟练掌握本章的内容,以便为后续章节的 Visual C++.NET 编程打下良好的基础。

本章主要内容

- C++的特点
- 类说明
- 构造函数
- 析构函数
- 虚函数
- 运算符重载
- 静态变量





1 C++的封装性

在 C 语言中，编程思想是结构化的。C++比 C 有许多优点，主要有封装性、继承性和多态性，这也是 C++具有面向对象程序设计语言的特点。

封装把数据与操作结合成一体，使程序结构更加紧凑，同时避免了数据紊乱带来的调试与维护的困难。

支持数据封装，就是支持数据抽象。在 C++中，类是支持数据封装的工具，对象则是数据封装的实现。在结构化程序设计中，数据只被看作是一种静态的结构，它只有等待到调用函数来对它进行处理。在面向对象程序设计中，将数据和对该数据进行合法操作的函数封装在一起作为一个类的定义。另外，封装还提供一种对数据访问严格控制的机制。因此，设计将被隐藏在封装体中，该封装体通过操作接口与外界交换信息。

对象被说明为具有一个给定类的变量。每个给定类的对象包含有这个类所规定的若干个私有成员、公有成员以及保护成员。

在 C 语言中，可以定义结构，但这种结构只包含数据，而不包含函数。C++中的类是数据和函数的封装体，结构可作为一种特殊的类，可以包含函数，也包含私有或保护的成员。

封装简化了程序员对对象的使用，只需要知道输入什么和输出什么，而对类内部进行什么操作不必追究。封装还使得一个对象可以像一个部件一样用在各个程序中，而不用担心对象的功能受到影响。例如类 Demo：

```
#include "math.h"
#include "iostream.h"
class Demo
{
public:
    int GetValue()
    {
        SetValue();
        Value=5;
        return Value;
    };
protected:
    void SetValue()
    {
        Value=4;
    };
private:
    //数据是私有的
    int Value;
};
```



```
void main()
{
    //定义 Demo 类的对象
    Demo demo;
    int value_1;
    double value_2;
    value_1=demo.GetValue();
    //开平方根
    value_2=sqrt(value_1);
    cout<<value_2<<"\n";
}
```

输出结果是3, 对于 main()里的操作, 无法更改类 Demo 的 Value 值, 这样保护了数据 Value, 实现了数据的隐藏。函数 SetValue()和数据 Value 结合在一起操作, 实现了封装。

2 C++的继承性

继承增强了软件的可扩展性, 并且为代码重用提供了强有力的手段。

在面向对象的语言中, 可以从一个类派生另一个类。派生类(也称子类)继承了其基类和祖先类的数据成员和成员函数。派生类代表基类的一种改良, 因为增加了新的属性或新的操作。派生类一般通过声明新的数据成员和成员函数来增加新的功能。另外, 派生类在继承的成员函数不合适时, 也可以弃之不用。

举例说明: 水果类是一个基类, 它有一些水果基本的共有特性如有营养、好吃等; 桔子类、苹果类和香蕉类是水果类的派生类, 它们在继承水果类的一些基本特性的基础上, 增加了各自的特性, 如特殊的形状、味道等。香蕉苹果和富士苹果是苹果类的派生类, 它们又增加了各自的新特性。

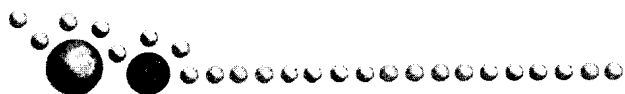
在设计苹果类时, 可以利用继承水果类的一些基本功能, 再添加一些苹果的特性; 同样, 在苹果类的基础上, 添加一些特殊的功能, 就可以完成香蕉苹果类和富士苹果类的设计, 从而减少了很大的工作量。

下面, 首先设计水果类 Fruit:

Class Fruit

```
{
public:
    void Eat()
    {
        cout<<"Can be eaten.\n";
    }
    void Taste()
    {
        cout<<"Taste is good.\n";
    }
}
```





```
    }  
};
```

苹果类 Apple 需要水果类 Fruit 同样的成员函数 void Eat()和 void Taste(), 另外增加了新函数 void Shape()和 void Color()。在设计时, 不需要重写 void Eat()和 void Taste()函数, 只需要继承水果类 (public Fruit), 苹果类的实现代码如下:

```
class Apple: public Fruit  
{  
public:  
    void Shape()  
    {  
        cout<<"The shape is a circle.\n";  
    }  
    void Color()  
    {  
        cout<<"Diferent kinds of color.\n";  
    }  
};
```

其中, “calss Apple:public Fruit” 表示类 Apple 公有继承类 Fruit, 关键字 public 可以替换为 private, 其不同的含义如表 1-1 所示。

表 1-1 访问权限的继承

派生类型	基类中的访问权限	派生类中的访问权限
public	public	Public
	protected	protected
	private	private
private	public	全是 private
	protected	
	private	

这样苹果类 Apple 具有 4 个成员函数: void Eat(), void Taste(), void Shape()和 void Color() 函数, 大大减少了代码的编写工作量。

3 C++的多态性

多态性使程序员在设计程序时可以对问题进行更好的抽象, 以设计出重用性和维护性最佳的程序。多态性是一种面向对象程序设计的功能, C++可以建立具有相同成员函数名的对象, 这些函数在概念上相似, 但在功能上却完全不同。

例如, 在各种几何形状类中, 比如如点、线、正方形、矩形、圆和椭圆等, 每一种类都有自己的 Draw()函数, 它负责正确地画出所需要图形。调用点类的 Draw()函数, 它画出的是点; 调用圆类的 Draw()函数画出的是圆。简单地说, 多态性就是“函数名相同, 实现却不同”。



多态性的实现一般是通过在派生类中重定义基类的虚函数来实现的。

例如基类 CDrawTool 定义了 3 个虚函数 DrawBegin(), DrawTo() 和 DrawEnd(), 它们用于实现绘画的功能。

```
class CDrawTool
{
public:
    CDrawTool() {}
protected:
    virtual void DrawBegin(const CPoint& point) {}
    virtual void DrawTo(const CPoint& point) {}
    virtual void DrawEnd() {}
};
```

类 CPenTool 继承了 CDrawTool 类, 用同样的函数实现了“画点”的功能。

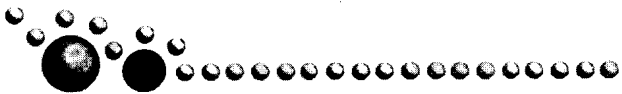
```
class CPenTool: public CDrawTool
{
public:
    CPenTool(): CDrawTool () {}
protected:
    virtual void DrawBegin(const CPoint& point);
    virtual void DrawTo(const CPoint& point);
    virtual void DrawEnd();
};
```

类 CFillTool 继承了 CDrawTool 类, 用同样的函数实现了“填充”的功能。

```
class CFillTool: public CDrawTool
{
public:
    CFillTool(): CDrawTool() {}
protected:
    virtual void DrawBegin(const CPoint& point);
};
```

类 CBoxTool 继承了 CDrawTool 类, 并扩充了虚函数 DrawObject(), 用来实现“画封闭曲线”的功能。

```
class CBoxTool: public CDrawTool
{
public:
    CBoxTool(bool filled): CDrawTool(), Filled(filled) {}
protected:
    virtual void DrawBein(const CPoint& point);
```



```
virtual void DrawTo(const CPoint& point);  
virtual void DrawEnd();  
virtual void DrawObject(CDC& dc) {}
```

protected:

```
bool Filled;  
int X1;  
int X2;  
int Y1;  
int Y2;
```

};

类 CRectTool 继承了 CBoxTool 类，具体实现了“画矩形”的功能。

```
class CRectTool: public CBoxTool  
{  
public:  
    CRectTool(bool filled): CBoxTool(filled) {}
```

protected:

```
virtual void DrawObject(CDC& dc);
```

};

类 CEllipSCTool 继承了 CBoxTool 类，具体实现了“画椭圆”的功能。

```
class CEllipseTool: public CBoxTool  
{  
public:  
    CEllipseTool(bool filled): CBoxTool(filled) {}
```

protected:

```
virtual void DrawObject(CDC& dc);
```

};

类的函数具体实现如下：

```
void CPenTool::DrawBegin(const CPoint& point)
```

```
{  
    State->DC->MoveTo(point);
```

```
}  
void CPenTool::DrawTo(const CPoint& point)
```

```
{  
    State->DC->LineTo(point);
```

```
}  
void CPenTool::DrawEnd()
```

```
{  
    State->DC->RestorePen();
```



```
}  
void CFillTool::DrawBegin(const CPoint& point)  
{  
    .....  
    //实现填充的代码  
}  
void CBoxTool::DrawBegin(const CPoint& point)  
{  
    X1=X2= point.X;  
    Y1=Y2= point.Y;  
    .....  
    DrawObject( *DC );  
}  
void CBoxTool::DrawTo(const CPoint& point)  
{  
    DrawObject( *DC );  
    X2 = point.X;  
    Y2 = point.Y;  
    DrawObject( *DC );  
}  
void CBoxTool::DrawEnd()  
{  
    DrawObject( *DC );  
    .....  
    DrawObject( *DC );  
}  
void CRectTool::DrawObject(CDC& dc)  
{  
    //画矩形  
    dc.Rectangle(x1,Y1,X2,Y2);  
}  
void CEllipseTool::DrawObject(CDC& dc)  
{  
    //画椭圆  
    dc.Ellipse(X1,Y1,X2,Y2);  
}
```

从上例可以看出，在不同的类中，同样的函数名可以实现不同的功能。



4 类的声明

在 C++ 中，类声明以关键词 `class` 开始，在类中所声明的内容以花括号括起来，花括号后的分号作为类说明语句的结束标志。类中定义的数据和函数称为这个类的成员，包括成员变量和成员函数。例如类名为 `Hello` 的声明形式如下：

```
class Hello
{
public:
    //公有的数据和函数
protected:
    //保护的数据和函数
private:
    //私有的数据和函数
};
```

C++ 类成员提供了 3 种可访问的权限，其中关键字分别是 `public`、`protected` 和 `private`。对于定义这些成员的访问权限，说明如下：

(1) `public`（公共部分）：只有类的成员函数、对象、派生类的成员函数及其对象可访问；

(2) `protected`（保护的部分）：只有类及其派生类的成员函数能访问保护的成员，对象无权访问保护的成员。

(3) `private`（私有部分）：只有类的成员函数能访问私有成员，对象、基类无权访问私有成员。

各个类声明的组成部分有以下原则：

(1) 在一个类说明中，关键字 `public`、`protected` 和 `private` 出现的顺序和次数可以是任意的。例如：

```
class Hello
{
public:
    void init(int a,int b);
private:
    int X;
pubic:
    int GetA();
private:
    int Y;
protected:
    void SetupWindow();
};
```

以上的类说明与下面的类说明等同：



```
class Hello
{
public:
    void init(int a,int b);
    int GetA();
private:
    int X;
    int Y;
protected:
    void SetupWindow();
};
```

(2) 如果没有指定类的属性，C++编译器则把成员当作保护的。

(3) 成员变量一般应该放在保护的部分，避免放在公有部分，允许它们被派生类的成员函数访问。

(4) 应该使用成员函数来设置和查询数据成员的值。设置操作数据成员的成员函数有助于执行合法性检查和更改其他数据成员。

(5) 类可以有多个构造函数，一般放在公共部分，当只有一个时，必须放在公共部分声明。

(6) 成员函数的定义一般放在一个单独的源文件中。

在类中，可以像说明普通变量一样说明类中的数据成员，数据成员可以具有任何类型，但是不能在类说明中，对数据成员使用表达式进行初始化。在类中声明的任何成员不能使用 `extern`、`auto` 和 `register` 关键字。一个类说明的数据成员描述了对对象的内部数据结构，类中说明的成员函数用于操作对象的这些数据。除了在类中对这些成员函数进行函数声明外，还必须在程序中定义这些成员函数的功能实现。定义成员函数的一般形式如下：

返回类型 类名::成员函数名（参数说明）

```
{
    函数体
}
```

其中，“::”是作用域运算符。“类名::”用于表明其后的成员函数是属于该类的。在“函数体”中可以直接访问类中说明的成员函数和数据成员，以描述该成员函数对它们所进行的操作。

类一旦被说明后，就可以像变量 `int`、`float` 一样使用，声明它的对象。例如：

`Hello A,B;`

该语句声明了 `A` 和 `B` 是类 `Hello` 的对象。

5 初始化对象

构造函数和析构函数是在类体中说明的两种特殊的成员函数。

构造函数用于创建对象，也就是类的入口，在创建对象时，使用给定的值来将对象初始化。要调用类的成员函数，首先要通过构造函数对对象进行初始化。C++中有下列关于

