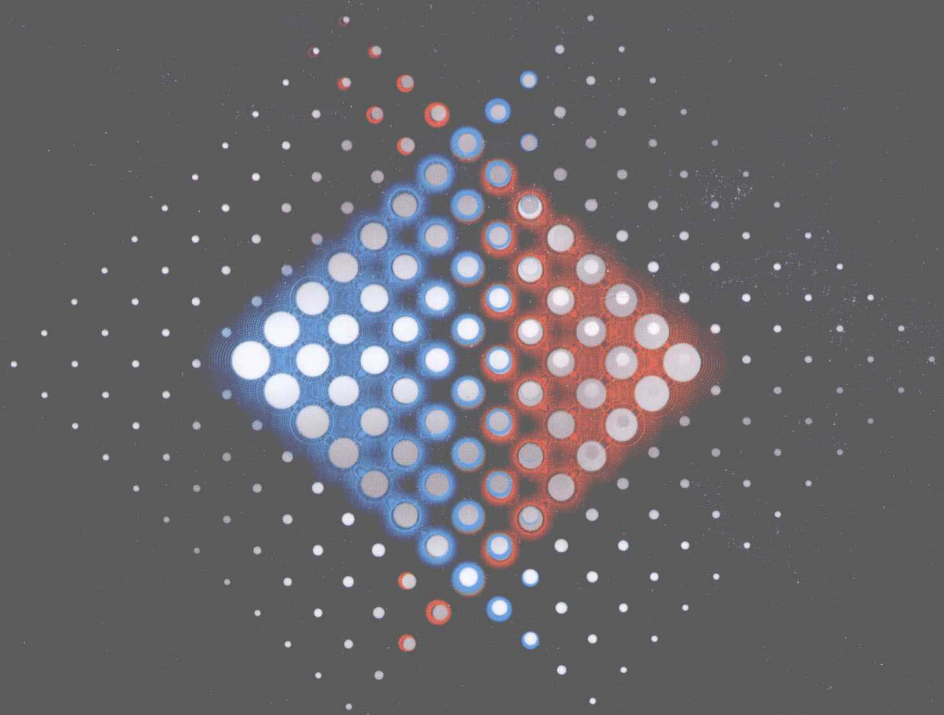
新编计算机类本科规划教材

面向对象程序设计与C++语言

朱战立　宋新爱　编著



電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

新编计算机类本科规划教材

面向对象程序设计与C++语言

朱战立　宋新爱　编著

電子工業出版社
Publishing House of Electronics Industry
北京 • BEIJING

内 容 简 介

本书讨论面向对象程序设计的基本概念，以及使用 Visual C++进行 Windows 用户界面设计的基本方法。本书的内容主要包括：C++语言基础、面向对象程序设计、类和对象、友元和运算符重载、继承、运行时的多态性和抽象类、模板、异常处理、I/O 流类库、MFC 编程、对话框及常用控件。本书的所有例子都用 Visual C++ 6.0 调试通过。

本书既可作为高等院校计算机等专业面向对象程序设计课程的教材，也可作为从事计算机开发和应用的工程技术人员的自学参考书。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有，侵权必究。

图书在版编目（CIP）数据

面向对象程序设计与 C++语言 / 朱战立，宋新爱编著. —北京：电子工业出版社，2010.7
新编计算机类本科规划教材
ISBN 978-7-121-11198-3

Ⅰ. ①面… Ⅱ. ①朱… ②宋… Ⅲ. ①面向对象语言－程序设计－高等学校－教材 ②C 语言－程序设计－高等学校－教材 Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字（2010）第 119560 号

策划编辑：冉 哲
责任编辑：冉 哲
印 刷：
装 订：北京京师印务有限公司
出版发行：电子工业出版社
北京市海淀区万寿路 173 信箱 邮编 100036
开 本：787×1 092 1/16 印张：21.25 字数：540 千字
印 次：2010 年 7 月第 1 次印刷
印 数：4000 册 定价：34.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

前　言

面向对象程序设计方法是目前计算机编程的主流方法。面向对象的软件开发以人类习惯的解决问题方式进行软件开发和设计，从而使软件开发过程与人类的求解问题过程基本一致。面向对象软件开发的出发点和基本原则是：尽可能模拟人类习惯的思维模式，使软件开发过程尽可能接近人类解决问题的过程。面向对象程序设计可以使所设计的程序具有可重用性、灵活性和扩展性。

本书讨论面向对象程序设计的基本概念，以及使用 Visual C++进行 Windows 用户界面设计的基本方法。本书的内容主要包括：C++语言基础、面向对象程序设计、类和对象、友元和运算符重载、继承、运行时的多态性和抽象类、模板、异常处理、I/O 流类库、MFC 编程、对话框及常用控件。为方便学习内容的衔接，本书第 1 章还简要介绍了 C++语言的一些基础概念。本书的所有例子都用 Visual C++ 6.0 调试通过。

和同类教材相比，本教材具有以下 3 个方面的特点。

第一，理论叙述简洁生动，用精心设计的例子说明重要的理论概念。本书前 9 章主要讨论面向对象程序设计的基本概念和基本方法，对于初学程序设计的学生来说，面向对象程序设计的许多概念和设计方法很不容易理解。作者积累多年讲授本课程的经验，以简洁生动的语言、精心设计的例子讲解了这些内容。

第二，以完整的例子引导 Windows 用户界面设计内容的学习过程。本书第 10 章讨论了 Visual C++的 MFC 编程，第 11 章讨论了 Visual C++的对话框及常用控件的设计。这两部分内容学生理解和掌握都比较困难，为此，作者设计了一个又一个相互衔接的例子，通过这些完整的例子，引导学生轻松掌握基于 Visual C++的 Windows 用户界面设计方法。

第三，习题设计丰富。面向对象程序设计的概念较多，要使学生很好地理解和掌握这些概念，需要辅以大量的习题练习，本书前 9 章中，每章都精心选择和设计了大量的习题。第 10 章和第 11 章讨论使用 Visual C++进行 Windows 用户界面设计的基本方法，这部分内容的例题设计包括了程序设计的主要过程，习题主要是上机练习例题，并仿照例题进行一些上机编程。

很多人都感觉，学习面向对象程序设计思想有点像学习哲学，比较抽象，初学时理解有困难。本书内容组织的基本思想是：以面向对象程序设计的理论为指导，以 C++语言面向对象程序设计的具体方法为落脚点，理论结合实际，所有理论概念都通过具体例子说明。希望通过本门课程的学习，既能掌握面向对象的基本概念和基本方法，又能掌握使用 Visual C++进行 Windows 软件开发的基本方法，并能把对面向对象基本概念和基本方法的理解，融会到使用 Visual C++进行 Windows 软件开发的过程中。

根据作者的教学体会，使用本教材授课约需 48～56 学时。

本书第 1～9 章由朱战立编写，第 10 章和第 11 章由宋新爱编写。全书由朱战立修改定稿。

尽管作者在写作过程中非常认真和努力，但由于水平有限，错误和不足之处在所难免，敬请读者批评指正。

作　者

目　录

此为试读，需要完整PDF请访问：www.ertongbook.com

第 1 章　C++语言基础

C++语言是在 C 语言基础上发展起来的一种混合了面向过程语言要素和面向对象语言要素的程序设计语言。C++语言除了支持面向对象技术外，在常规功能方面也做了许多扩充。为方便后面各章的学习，本章在分析 C++语言对 C 语言常规语句功能扩充的基础上，简要介绍 C++语言的一些基础概念。对 C++语言和 C 语言都具有的功能，用 C++/C 语言表示。

本章主要内容包括：C++语言的数据类型、基本语句、变量的使用方法、函数的设计方法、引用变量的功能、new 和 delete 运算符、输入/输出语句，以及程序预处理等。

1.1　数据类型

数据类型规定了一类数据的数据位长度（或称字节个数）、取值范围及对该类数据所能进行的操作。

1.1.1　基本数据类型

基本数据类型是系统已经定义的数据类型。C++/C 语言共定义了 7 种基本数据类型，其中 4 种为整型数，两种为浮点型数，1 种为字符型数。数据类型不同，所定义的变量占用的内存空间、取值范围及对该类数据所能进行的操作也不同。

C++/C 语言定义的 7 种基本数据类型及相应的关键字如下。

- 整型：byte、short、int、long
- 浮点型：float、double
- 字符型：char

C++/C 语言的基本数据类型、字节数和数值范围见表 1.1。

表 1.1　C++/C 语言的基本数据类型、字节数和数值范围

数据类型关键字	字　节　数	数　值　范　围
byte	1	$-128\sim127$，即$-2^7\sim2^7-1$
short	2	$-32\,768\sim32\,767$，即$-2^{15}\sim2^{15}-1$
int	4	$-2\,147\,483\,648\sim2\,147\,483\,647$，即$-2^{31}\sim2^{31}-1$
long	8	$-2^{63}\sim2^{63}-1$
float	4	3.4e–038～3.4e+038
double	8	1.7e–308～1.7e+308

C++/C 语言的字符串用字符数组表示，例如语句

```
char str[10] = "abcdefghi";
```

定义了一个长度为 10 的字符数组变量 str，且该变量的初始值为"abcdefghi"。

另外，C++/C 语言还有空类型，其关键字是 void。空类型主要用来定义函数返回值的类型。

1.1.2 枚举类型

和 C 语言不同的是，C++语言的枚举类型是一个真正的类型名，即在 C++语言中枚举类型可像其他类型一样使用。系统对待枚举类型变量将像对待其他类型的变量一样。

C++语言枚举类型的一般形式是：

```
enum <标识符> {<枚举列表>} ;
```

其中，enum 是枚举类型标识，枚举列表中定义了该枚举类型的所有枚举值。枚举列表的枚举值和编号从 0 开始的整数值相对应。

一旦定义了枚举类型，就可以定义该枚举类型的变量。枚举类型变量允许的操作只有赋值操作。

一个枚举类型变量定义和使用的例子如下：

```
#include <iostream.h>

enum Color{Red, Green, Yellow};
void main(void)
{
    Color myColor;              //Color 是枚举类型名
    myColor = Yellow;           //赋值操作
    cout << myColor;            //myColor 中保存的值是符号数字常量 2
}
```

1.1.3 结构体

高级程序设计语言中只定义了 int、long、float、char 等基本数据类型，但是，在有些程序设计问题中，需要把若干个基本数据类型的数据作为一个整体来考虑。在 C++/C 语言中，这样的问题是通过结构体定义语句来解决的。结构体定义语句的一般格式为：

```
struct <结构体名>
{
    <成员表列>
};
```

其中，结构体名是结构体的标识符，成员表列中的每一项都由已定义数据类型名和成员名两部分组成。由于结构体中所有成员的数据类型都是已定义的，因此可以把结构体看做一个新的、用户自定义的数据类型。换句话说，一旦定义了一个结构体，就可以用该结构体定义变量。

例如，要处理学生信息时，假如要处理的学生信息包括学生的学号、姓名、性别、年龄，就可以把学生的这些信息定义成一个结构体。结构体定义语句如下：

```
struct Student
{
    long  number;                   //学号
    char  name[10];                 //姓名
    char  sex[3];                   //性别
    int  age;                       //年龄
};
```


此为试读，需要完整PDF请访问：www.ertongbook.com

对结构体类型的变量，既可以整体处理，也可以按成员分量处理。整体处理的语句如下：

```
struct Student x = {100001, "张三", "男", 26}, y, *p;
y = x;                                    //结构体赋值
p = &x;                                   //结构体地址赋值
```

按成员分量处理的语句如下：

```
struct Student x = {100001, "张三", "男", 26}, y, z, *p;
y.number = x.number;                      //变量的成员分量赋值
p = &x;                                   //结构体地址赋值
z.number = p->number;                     //指针类型变量的成员分量赋值
```

注意，这里指针类型变量的成员表示方法和非指针类型变量的成员表示方法之间的不同。

C++语言的结构体类型定义方法和 C 语言的定义方法基本类同，主要有两点区别。

① C++语言的结构体类型定义和使用更简单一些，结构体一旦定义，就可以直接使用该结构体名定义变量，而不用在结构体名前加标识符 struct。

② C++语言把结构体看做类的特殊形式，即结构体是没有私有部分的类。换句话说，在 C++语言中，关键字 struct 的含义和后面要讨论的类的关键字 class 基本类同，只是在用 struct 定义的类中，所有成员的权限都是 public。

1.2 基本语句

程序是由各种语句根据问题的要求有机组合而成的。C++/C 语言的基本语句可分为：赋值语句、自加减表达式语句、分支语句、循环语句和流程控制语句。

1.2.1 赋值语句

赋值语句用来给变量赋值。最基本的赋值语句语法格式为：

```
<变量>  <赋值运算符>  <表达式>;
```

其中，最常用的赋值运算符是“=”。另外，还有复合运算符+=，-=，*=，/=等运算符。表达式分为算术表达式、关系表达式和逻辑表达式 3 种。赋值语句中的表达式主要是算术表达式。算术表达式是指由算术运算符组成的表达式。例如：

```
int y = 0, x = 5;              //变量定义并初始化赋值
y = x + 3;                     //把算术表达式 x+3 的值赋给变量 y
x += 5;                        //把算术表达式 x+5 的值赋给变量 x
```

语句 x += 5 也可以写成：

```
x = x + 5;
```

1.2.2 自加减表达式语句

自加减表达式语句是特殊情况下赋值语句的简略形式。例如，对于如下程序段：

```
int i = 0, x = 0, sum = 0;
i++;                           //自加减表达式语句
x += 5;
sum = sum + x;
```

其中，变量 i 的初始值为 0，i++表示变量 i 的值等于变量 i 的值加 1，即 i++也可以写为：

```
i = i + 1;
```

自加减表达式语句的其他例子还有：

```
i++;
++i;
i--;
--i;
```

1.2.3 分支语句

分支语句用来根据不同的条件构造不同的语句执行流程。if 语句的语法形式为：

```
if  (<条件>)
     <语句> ;
```

或

```
if  (<条件>)
     <语句 1>;
 else
     <语句 2>;
```

第一种分支语句的含义是：假如条件成立，则执行语句，否则跳过该语句执行后续语句；第二种分支语句的含义是：假如条件成立，则执行语句 1，否则执行语句 2。

条件是由关系表达式或逻辑表达式组成的一个值为真（即大于 0）或为假（即小于或等于 0）的表达式。

当条件后面的语句多于一条时，要用一对花括号“{}”把这些语句括起来作为一个整体看待。

【例 1.1】 编写一个程序，实现以下功能：对于两个 float 类型的数值 a=0.5 和 b=1.6，首先计算出 a+b 和 a*b，然后判断 a+b 和 a*b 的大小关系，如果 a+b>a*b，则输出 a+b，否则输出 a*b。

程序设计如下：

```
//Exam1-1.c
#include <stdio.h>

void main(void)
{
    float a = 0.5, b = 1.6, s1,s2;
    s1=a+b;                                    //两个数的和
    s2=a*b;                                    //两个数的积
    if (s1 > s2)
        printf("a + b = %f\n", s1);            //输出 a+b
    else
        printf("a * b = %f\n", s2);            //输出 a*b
}
```

程序运行结果为：

```
a + b = 2.100000
```

if 语句可以嵌套使用，即，可以在一个 if 语句中又包含另一个 if 语句，从而构成程序执行的多个分支。但是在大多数情况下，当程序存在多个分支时，将使用 switch 语句。switch 语句的语法形式为：

```
switch (<表达式>)
{
    case  <常量 1>: <语句序列 1>;
    case  <常量 2>: <语句序列 2>;
    ...
    case  <常量 n>: <语句序列 n>;
    default: <语句序列 n+1>;
}
```

switch 语句中，如果表达式的值与某一 case 后面的常量 i 匹配，则执行此 case 后面的所有语句序列；如果表达式的值与任何 case 后面的常量都不匹配，则执行语句序列 $n+1$。

要注意的是，switch 语句序列的最后一条语句通常都是 break 语句。

1.2.4 循环语句

循环语句用来构造程序中需要重复执行的语句部分。循环语句用来构造满足一定条件时，对同一个程序段重复执行若干次的程序结构。

循环语句主要有 while 语句和 for 语句两种。while 语句主要用来构造循环次数不固定的循环，for 语句主要用来构造循环次数固定的循环。实际上，这两种循环语句的功能完全一样，也就是说，既可以把 while 语句构造的循环结构改造成 for 语句构造的循环结构，也可以把 for 语句构造的循环结构改造成 while 语句构造的循环结构。

while 循环的语法形式为：

```
while  (<条件>)
    <循环体>;
```

其含义是，当条件成立时，反复执行循环体中的语句，直到条件不成立时为止。

for 循环的语法形式为：

```
for(<初始表达式>; <终止表达式>; <增量表达式>)
    <循环体>;
```

其循环执行流程为：

① 计算初始表达式（这通常是一个或一组用逗号分隔的赋值语句）。

② 判断终止表达式（这通常是一个条件），若条件成立，则执行循环体中的语句；若条件为假，则跳出循环语句。

③ 计算增量表达式（这通常是一个自加减表达式语句），然后转②。

注意，3 个不同的表达式之间用逗号分隔开。当语句多于一句时，要用一对花括号“{}”把循环体中的语句括起来。

【例 1.2】 编写求 1～1000 累加和的程序，并将计算结果打印出来。

for 循环结构程序如下：

```
//Exam1-2-1.c
#include <stdio.h>

void main(void)
{
    long sum = 0;
    int i;
    for(i = 1; i <= 1000; i++)
        sum = sum + i;
    printf("sum = %ld\n", sum);
}
```

由于本例的循环次数事先已知，因此用 for 语句实现较好。为说明 while 语句和 for 语句的功能完全相同，再用 while 语句设计如下：

```
//Exam1-2-2.c
#include <stdio.h>

void main(void)
{
    long sum = 0;
    int i = 1;
    while(i <= 1000)
    {
        sum = sum + i;
        i++;
    }
    printf("sum = %ld\n", sum);
}
```

两个程序的运行结果均为：

```
sum = 500500
```

循环语句还有 do-while 语句，其实现的功能和 while 语句完全相同，只是语法略有不同。

1.2.5 流程控制语句

流程控制语句主要有 break 语句和 continue 语句。

break 语句的语法形式为：

```
break;
```

其语义是，跳出当前的 switch 语句或循环语句。

continue 语句的语法形式为：

```
continue;
```

其语义是，结束本次循环，即跳过循环语句中尚未执行的语句，接着进行循环条件的判定。continue 语句只用在 for、while、do-while 等循环语句中，一般与 if 语句一起使用，可以加速

循环。

【例 1.3】 设计一个程序实现以下功能：若输入英文字母 Y 或 y，则继续执行；若输入英文字母 N 或 n，则结束；若输入其他字符，则不做任何处理。

```
//Exam1-3.c
#include <stdio.h>

void main(void)
{
    char  ch;
    do
    {
        ch = getche();
        if(ch == 'Y' || ch == 'y')
            continue;
        if(ch == 'N' || ch == 'n')
            break;
    }while(1);
}
```

1.3 变量

C++语言中的变量和 C 语言中的变量相比，功能扩充之处主要体现在：变量的定义方法、const 类型限定符、函数形式的变量类型转换等方面。

1.3.1 变量的定义方法

C 语言只允许变量在程序开始处定义，而 C++语言允许变量在程序的任何位置定义，这就使得 C++语言的变量除全局变量和局部变量外，又增加了块变量。

C++语言把用花括号“{}”括起来的一块区域称为块。块变量就是定义在某个块中的变量。变量的作用域就是变量的作用范围。块变量的作用域就是该变量定义的由花括号“{}”括起来的范围，称做块作用域。块变量在其作用域内是可见的，在其作用域外不可见。

当程序比较大时，在程序的开始处定义程序中要使用的所有变量通常比较麻烦，而 C++语言程序就可以在需要使用一个变量时再定义它。例如，下例程序对 C 语言程序来说是错误的，而对 C++语言程序就是正确的。

```
//Example.cpp
void main(void)
{
    long sum = 0;
    int n = 100;
    for(int i = 0; i < n; i++)
    {
```

```
            for(int j = 0; j < n; j++)
            {
                sum = sum + i + j;
            }
        }
    }
```

注意，C 语言程序的文件名后缀为.c C++语言程序的文件名后缀为.cpp。

1.3.2 const 类型限定符

关键字 const 用来限定变量或函数参数（或函数返回值）的修改。

当用 cons 修饰一个变量时，则限定该变量在定义域范围内为常量，并要求该变量必须在定义时初始化赋值，且以后不允许再重新赋值修改。

用 C 语言编程时通常用宏定义命令来定义常量，例如，要定义 MaxSize 为 100，则使用语句：

```
#define MaxSize 100
```

而用 C++语言编程时通常用 const 来定义常量，如使用语句：

```
const int maxSize = 100;
```

虽然这两种方法均可定义程序中使用的常量，但它们有以下两点主要差别。

① 宏定义命令定义的常量只是文本替换，如用 100 替换 MaxSize，由于没有数据类型，因此 C 程序编译时也不做数据类型检查；而 const 定义的常量带有数据类型，如 maxSize 就是整数类型。定义有数据类型的常量在进行表达式运算时，可以进行类型检查，从而防止程序出错。

② 宏定义命令定义的常量的作用域类似于全局变量；而用 const 定义的常量可像定义变量那样，在程序的任意地方进行定义，其作用域随定义位置的不同而不同。现在通行的程序设计方法主张取消（至少尽量减少）全局变量，因为全局变量不好维护，容易引起错误。

如果用 const 修饰一个函数的形式参数，则限定相应的实际参数在该函数内为常量，即该参数不允许在函数内被修改。

1.3.3 函数形式的变量类型转换

C++语言对变量提供了函数形式的显式类型转换。任何系统定义的基本数据类型或用户自定义的数据类型的名字，都可以作为函数使用，从而把变量（或常量）从一种数据类型转换到另外一种数据类型。函数形式的类型转换例子如下：

```
int i = 20;
float x;
x = float(i);                    //把变量 i 转换为 float 类型
```

而在 C 语言中，上述例子的类型转换方法是：

```
x = (float)i;
```

1.4 函数

C++/C 语言都支持面向过程的程序设计。面向过程的程序设计以函数为程序模块化设计

此为试读，需要完整PDF请访问：www.ertongbook.com