# Visual C++数据库开发基础与应用

周启涛　葛卉娟　编　著

人 民 邮 电 出 版 社

需要全本请访问 www.ertongbook.com

**图书在版编目（CIP）数据**

Visual C++数据库开发基础与应用 / 周启涛，高英编著．—北京：人民邮电出版社，2005.8

ISBN 7-115-13736-6

Ⅰ．V… Ⅱ．①周…②高… Ⅲ．C 语言－程序设计 Ⅳ．TP312

中国版本图书馆 CIP 数据核字（2005）第 089626 号

## 内 容 简 介

本书详细介绍了如何使用 Visual C++进行数据库应用程序设计。本书由 4 部分组成：第一部分为 Visual C++程序设计的基础（第 1 章～第 3 章）；第二部分为数据库基础（第 4 章～第 6 章），详细介绍了 Oracle 和 SQL Server 数据库的相关知识，以及常用数据库结构化查询语言 SQL 语句的使用方法；第三部分为 Visual C++数据库开发基础（第 7 章），详细介绍了两大数据库开发技术 MFC ODBC 和 ADO；第四部分为案例（第 8 章和第 9 章），详细介绍了考勤管理系统和家庭财务管理系统的开发过程，从系统设计、数据库设计，再到系统的实现，给读者呈现了一个完整的数据库管理系统的开发过程。

本书内容丰富、实用性强、讲解透彻、易于掌握，适合使用 Visual C++进行数据库开发的初学者学习。

**Visual C++数据库开发基础与应用**

- 编　　著　周启涛　高　英
  责任编辑　刘　浩
- 人民邮电出版社出版发行　　北京市崇文区夕照寺街 14 号
  邮编　100061　　电子函件　315@ptpress.com.cn
  网址　http://www.ptpress.com.cn
  北京鸿佳印刷厂印刷
  新华书店总店北京发行所经销
- 开本：787×1092　1/16
  印张：20.5
  字数：498 千字　　2005 年 8 月第 1 版
  印数：1 – 5 000 册　　2005 年 8 月北京第 1 次印刷

ISBN 7-115-13736-6 /TP・4856

定价：30.00 元

**读者服务热线：(010)67132692　印装质量热线：(010)67129223**



# 前　言

Visual C++是计算机界公认的优秀的应用程序开发工具之一，不仅可以开发系统级应用，还可以开发性能优良、界面友好的数据库应用程序。

本书浓缩了作者多年的项目开发经验，讲解循序渐进，重点突出，每章都辅以丰富的实例，力图使读者更容易掌握 Visual C++数据库应用程序开发的技能。

本书共 9 章，可以分成 4 部分。

第 1 章至第 3 章介绍了 Visual C++程序设计的基础，包括语法基础（如数据类型、基本语句、数组、指针、类、对象、继承、派生、重载、多态性、虚函数）和 MFC 基础。

第 4 章和第 5 章介绍了 Oracle 和 SQL Server 的安装、常用工具以及数据库的管理。第 6 章介绍了结构化查询语言 SQL 的常用语句以及 PL/SQL 和 Transact-SQL 的常用函数。

第 7 章介绍了 Visual C++数据库开发技术，并重点介绍了 MFC ODBC 和 ADO 数据库开发技术、ADO 数据绑定控件，以及如何封装自己的 MFC ODBC 和 ADO 数据库访问类。利用 MFC ODBC 或 ADO 编写数据库应用程序时，会多次使用 MFC ODBC 或 ADO 的一些类库操作数据库，而且还会用到一些公共的常量、变量以及函数，对这些常用的 MFC ODBC 或 ADO 类库、常量、变量以及函数进行封装，将会使数据库编程更为轻松。

第 8 章介绍如何利用封装的 MFC ODBC 数据库访问类开发考勤管理系统。本章使用 Oracle 作为后台数据库。

第 9 章介绍如何利用封装的 ADO 数据库访问类开发家庭财务管理系统。本章使用 SQL Server 作为后台数据库。

本书中的程序代码可以在 Http://www.ucbook.com 处下载。

由于时间仓促，加之水平有限，不足之处在所难免，敬请读者批评指正（可发电子函件至：book_better@sina.com）。

编者

2005.8



# 目　录


第 1 章　C 语言基础 ..... 1

1.1　C 语言概述 ..... 1
1.1.1　C 语言的特点 ..... 1
1.1.2　C 程序的构成 ..... 2
1.2　数据类型、运算符和表达式 ..... 2
1.2.1　标识符 ..... 2
1.2.2　数据类型 ..... 3
1.2.3　常量和变量 ..... 3
1.2.4　运算符和表达式 ..... 4
1.2.5　注释 ..... 8
1.2.6　在 Visual C++中运行示例程序 ..... 8
1.3　基本语句 ..... 11
1.3.1　条件语句 ..... 11
1.3.2　开关语句 ..... 12
1.3.3　循环语句 ..... 14
1.3.4　转向语句 ..... 15
1.4　数组 ..... 17
1.4.1　数组的定义 ..... 17
1.4.2　使用数组元素 ..... 17
1.4.3　数组赋值 ..... 18
1.5　函数 ..... 21
1.5.1　函数的定义和声明 ..... 21
1.5.2　函数的参数和返回值 ..... 22
1.5.3　函数的调用 ..... 23
1.5.4　变量的作用域 ..... 26
1.6　指针 ..... 27
1.6.1　指针的概念和定义 ..... 27
1.6.2　指针的运算 ..... 28

1.6.3 指针与数组 ..... 30
1.6.4 指针与函数 ..... 33
1.7 结构 ..... 35
1.7.1 结构的概念和定义 ..... 35
1.7.2 结构变量的初始化和使用 ..... 36
1.7.3 结构数组 ..... 37
1.7.4 结构指针 ..... 38
1.8 Visual C++程序调试 ..... 40
1.8.1 调试环境 ..... 40
1.8.2 调试程序 ..... 42

第 2 章 C++面向对象程序设计 ..... 46

2.1 类和对象 ..... 46
2.1.1 类的概念和定义 ..... 46
2.1.2 数据成员和成员函数 ..... 48
2.1.3 对象 ..... 53
2.1.4 this 指针 ..... 53
2.1.5 new 和 delete 运算符 ..... 54
2.2 继承和派生 ..... 55
2.2.1 基本概念 ..... 55
2.2.2 派生类的定义 ..... 56
2.2.3 公有继承和私有继承 ..... 56
2.2.4 派生类的指针 ..... 61
2.2.5 多继承 ..... 63
2.3 重载 ..... 66
2.3.1 函数重载 ..... 66
2.3.2 运算符重载 ..... 68
2.4 多态性与虚函数 ..... 70
2.4.1 多态性的概念与应用 ..... 70
2.4.2 虚函数的使用 ..... 72
2.4.3 纯虚函数和抽象类 ..... 73

第 3 章 Visual C++程序设计基础 ..... 76

3.1 MFC 基础 ..... 76
3.1.1 什么是 MFC ..... 76
3.1.2 MFC 的类层次 ..... 77
3.1.3 CString 类 ..... 82
3.1.4 CStringArray 类 ..... 84
3.1.5 CStringList 类 ..... 85



需要全本请访问 www.ertongbook.com

3.1.6 CTime 类 ..... 86
3.2 基于对话框的应用程序 ..... 88
3.2.1 创建项目 ..... 88
3.2.2 工作区窗口 ..... 90
3.2.3 执行项目 ..... 91
3.2.4 常用控件 ..... 92
3.2.5 控件示例 ..... 94
3.3 单文档应用程序 ..... 101
3.3.1 创建项目 ..... 101
3.3.2 主要程序类 ..... 103
3.3.3 添加鼠标事件 ..... 105
3.3.4 设置菜单 ..... 108
3.3.5 添加对话框 ..... 113
3.4 高级控件 ..... 114
3.4.1 CTreeCtrl 控件 ..... 114
3.4.2 CImageList 控件 ..... 116
3.4.3 控件示例 ..... 116

**第 4 章 Oracle 数据库管理** ..... 121

4.1 安装 Oracle 数据库服务器 ..... 121
4.2 常用工具 ..... 126
4.2.1 数据库配置助手 ..... 126
4.2.2 网络配置助手 ..... 128
4.2.3 网络管理器 ..... 132
4.2.4 企业管理器 ..... 134
4.2.5 SQL* Plus ..... 136
4.2.6 SQLPlus Worksheet ..... 137
4.3 数据库管理 ..... 138
4.3.1 表空间管理 ..... 138
4.3.2 用户管理 ..... 140
4.3.3 数据表管理 ..... 143
4.3.4 索引管理 ..... 145
4.3.5 视图管理 ..... 147

**第 5 章 SQL Server 数据库管理** ..... 150

5.1 安装 SQL Server ..... 150
5.2 常用工具 ..... 153
5.2.1 服务管理器 ..... 153
5.2.2 企业管理器 ..... 153

5.2.3 向导 ..... 155
5.2.4 查询分析器 ..... 158
5.3 数据库管理 ..... 159
5.3.1 系统数据库和示例数据库 ..... 159
5.3.2 创建数据库 ..... 160
5.3.3 修改数据库 ..... 162
5.3.4 删除数据库 ..... 163
5.4 表和视图管理 ..... 163
5.4.1 数据类型 ..... 163
5.4.2 数据表管理 ..... 164
5.4.3 索引管理 ..... 167
5.4.4 视图管理 ..... 169

第 6 章 常用 SQL 语句 ..... 172

6.1 INSERT 语句 ..... 172
6.2 UPDATE 语句 ..... 173
6.3 DELETE 语句 ..... 174
6.4 SELECT 语句 ..... 177
6.4.1 简单的 SQL 语句 ..... 177
6.4.2 使用 WHERE 子句指定查询条件 ..... 178
6.4.3 使用 ORDER BY 子句排序查询结果 ..... 179
6.4.4 使用统计函数 ..... 180
6.4.5 使用 GROUP BY 子句分组查询 ..... 182
6.4.6 使用 HAVING 子句指定组查询条件 ..... 183
6.4.7 连接查询 ..... 184
6.4.8 子查询 ..... 185
6.4.9 使用 UNION 运算符建立联合 ..... 187
6.5 PL/SQL 的常用函数 ..... 188
6.5.1 数值型函数 ..... 188
6.5.2 字符型函数 ..... 189
6.5.3 日期型函数 ..... 190
6.6 Transact-SQL 的常用函数 ..... 190
6.6.1 数值型函数 ..... 190
6.6.2 字符型函数 ..... 191
6.6.3 日期型函数 ..... 192

第 7 章 Visual C++数据库开发基础 ..... 193

7.1 数据库开发技术简介 ..... 193
7.2 MFC ODBC 数据库开发技术 ..... 193

7.2.1 MFC ODBC 主要类介绍 ..... 194
7.2.2 ODBC 数据源的配置 ..... 194
7.2.3 数据库的连接 ..... 196
7.2.4 查询记录 ..... 197
7.2.5 添加、修改和删除记录 ..... 198
7.3 封装自己的 MFC ODBC 数据库访问类 ..... 199
7.3.1 通用定义类文件 ..... 200
7.3.2 封装 ODBC 类 ..... 201
7.4 ADO 数据库开发技术 ..... 205
7.4.1 ADO 对象模型 ..... 205
7.4.2 设置 ADO 数据库访问环境 ..... 207
7.4.3 数据库的连接 ..... 207
7.4.4 查询记录 ..... 208
7.4.5 添加、修改和删除记录 ..... 210
7.5 使用 ADO 数据绑定控件 ..... 213
7.5.1 ADO Data 控件 ..... 213
7.5.2 DataCombo 控件 ..... 215
7.5.3 DataList 控件 ..... 216
7.5.4 DataGrid 控件 ..... 217
7.5.5 Chart 控件 ..... 219
7.6 封装自己的 ADO 数据库访问类 ..... 220
7.6.1 通用定义类文件 ..... 220
7.6.2 封装 ADO 类 ..... 222

**第 8 章 考勤管理系统** ..... 229

8.1 系统分析与设计 ..... 229
8.1.1 功能描述 ..... 229
8.1.2 功能模块设计 ..... 230
8.2 数据库设计与实现 ..... 230
8.2.1 数据库需求设计 ..... 230
8.2.2 数据库表的设计 ..... 231
8.2.3 数据库表的实现 ..... 232
8.3 系统的实现 ..... 234
8.3.1 创建应用程序 ..... 234
8.3.2 创建主对话框的界面 ..... 235
8.3.3 显示数据到界面上 ..... 238
8.3.4 系统信息管理 ..... 243
8.3.5 出勤信息管理 ..... 248
8.3.6 加班信息管理 ..... 251

8.3.7 请假信息管理 ..... 253
8.3.8 出差信息管理 ..... 256
8.3.9 考勤统计信息管理 ..... 257

第 9 章 家庭财务管理系统 ..... 267

9.1 系统分析与设计 ..... 267
9.1.1 功能描述 ..... 267
9.1.2 功能模块设计 ..... 267
9.2 数据库设计与实现 ..... 268
9.2.1 数据库需求设计 ..... 268
9.2.2 数据库表的设计 ..... 268
9.2.3 数据库表的实现 ..... 270
9.3 系统的实现 ..... 271
9.3.1 创建应用程序 ..... 271
9.3.2 创建多文档视图框架 ..... 273
9.3.3 系统设置管理 ..... 284
9.3.4 账户信息管理 ..... 290
9.3.5 收支信息管理 ..... 300
9.3.6 明细查询管理 ..... 308
9.3.7 统计分析管理 ..... 313

# 第1章 C语言基础

C++语言是目前使用广泛的一种高级程序设计语言，它既可以进行结构化程序设计，也可以进行面向对象程序设计。C++是从C语言发展而演变而来的，是C语言的超集，即C++除了提供新的特性外，几乎支持所有的C语言特性。因此，掌握C语言基础是非常重要的。

## 1.1 C语言概述

C语言是由Dennis Ritchie创造并首先在配备UNIX操作系统的DEC PDP-11计算机上实现的。C语言是国际上广泛流行的，很有发展前途的计算机高级语言，它适合于作为系统描述语言，既可用来写系统软件，也可以用来写应用软件。

### 1.1.1 C语言的特点

C语言是一种强大的专业化编程语言，其主要特点如下：

**1．语言简洁紧凑、使用方便灵活**

C语言只有32个关键字，9种控制语句，程序书写形式自由，主要用小写字母表示。学过PASCAL或FORTRAN的读者可以看出，C语言简练，源程序短。

**2．数据结构丰富**

C语言的数据类型有整型、浮点型、双精度浮点型、字符型、数组类型、指针类型、结构体类型和共用体类型等，足以用来实现各种复杂的数据结构（如队列、链表、树、栈等）的运算。

**3．理想的结构化语言**

C语言具有丰富的结构化控制语句（如条件语句、开关语句和循环语句等）。C语言使用函数作为程序的模块单位，便于实现程序的模块化。

**4．运行速度快**

C程序往往很紧凑而且运行速度快，可以与汇编语言相媲美，许多硬件的驱动程序都是使用C语言编写的。

**5．可移植性强**

C是可移植的语言，在一个系统上编写的C程序，经过很小的改动或者不改动就可以在其他的系统上运行。

### 1.1.2 C 程序的构成

C 程序是由函数构成的。C 程序结构是模块化的，即一个 C 源程序是由一个主函数 main 和零个、一个或多个其他函数组成的。函数可以是系统的库函数（如 printf 函数），也可以是用户自定义的函数。

一个简单的 C 程序如下：

```
#include "stdio.h"
int main(void)   /*主函数*/
{
  printf("这是一个简单的 C 程序\n"); /*输出"这是一个简单的 C 程序",并另起一行*/
}
```

其中，利用 include 关键字引入标准 I/O 定义文件 stdio.h，该文件包含了一些标准 I/O 函数的定义，如 printf 函数。在编写一个 C 程序时，通常要包含该文件。main 是主函数，函数体由花括号括起来，只有一个输出语句，利用函数 printf 来实现。/*……*/表示注释部分，对编译和运行不起作用。该程序的作用是输出一行文字“这是一个简单的 C 程序”。

又如：

```
#include "stdio.h"
int main(void)  /*主函数*/
{
  int a,b,sum;  /*定义变量*/
  a = 20;       /*给变量 a 赋值 20*/
  b = 30;       /*给变量 b 赋值 30*/
  sum = a + b;  /*求 a 与 b 的和*/
  printf("两个数的和为: %d\n",sum); /*输出"两个数的和为: 50"*/
}
```

本程序作用是求两个数 a 和 b 的和，程序的第 4 行定义了变量 a、b、sum，第 5、6 行对变量 a 和 b 进行赋值，第 7 行对 a 和 b 求和，并赋值给 sum，第 8 行中的“%d”是输入输出格式字符，表示输出数据的格式是一个十进制整数，printf 函数最右端的 sum 是要输出的变量，对应的输出格式为“%d”，因此输出的信息为“两个数的和为：50”。

## 1.2 数据类型、运算符和表达式

本节介绍 C 语言的数据类型、运算符和表达式。

### 1.2.1 标识符

标识符是程序中使用的变量、函数、自定义类型和宏等的名字，是由字母、下划线和数字组成的字符系列。C 语言程序是由大量标识符构成的。

在使用标识符的时候要注意以下事项：

- 标识符只能由字母、下划线和数字组成，不能出现非法的字符，如#、*、&和%等字符。如标识符 w#4、u&3 和!kk 等都是非法的标识符。



需要全本请访问 www.ertongbook.com

- 标识符的名称要有意义，以便于程序的阅读和维护。例如表示“名称”的标识符可以用 name，表示“长度”的标识符可以用 length。
- C 语言对标识符大小写是敏感的。标识符 name 和标识符 Name 是不同的。初学者一定要注意。
- 标识符不能是 C 关键字。关键字是预先有特定含义的标识符，又称保留字，不能用作他途，如 int、char、float、double、long、short、unsigned、struct、goto 和 break 等。这些关键字在 C 编译器中通常会显示特殊的颜色。

### 1.2.2　数据类型

C 语言的数据类型可以分为两大类：基本数据类型和构造数据类型。

基本数据类型包括整型（int）、浮点型（float）、双精度浮点型（double）和字符型（char）。整型又分为长整型和短整型，有符号整型和无符号整型；字符型又分为有符号字符型和无符号字符型两种，如表 1-1 所示。

表 1-1　　基本数据类型

| 类型名称 | 全称类型标识符 | 简化类型标识符 | 取值范围 |
|---|---|---|---|
| 短整型 | short int | short | –32768~32767 |
| 整型 | int | | –32768~32767 |
| 长整型 | long int | long | $-2^{31}$~$(2^{31}-1)$ |
| 无符号短整型 | unsigned short int | unsigned short | 0~65535 |
| 无符号整型 | unsigned int | unsigned | 0~65535 |
| 无符号长整型 | unsigned long int | unsigned long | 0~$(2^{32}-1)$ |
| 有符号字符型 | char | | –128~127 |
| 无符号字符型 | unsigned char | | 0~255 |
| 单精度浮点型 | float | | $10^{-37}$~$10^{38}$ |
| 双精度浮点型 | double | | $10^{-307}$~$10^{308}$ |

用户可以构造数据类型，或者叫做自定义数据类型，即将若干个基本数据类型的变量按照一定的规则组合起来的，如后面要讲到的数组类型。

### 1.2.3　常量和变量

在程序运行过程中，其值不能被改变的数据，称为常量。而在程序运行过程中值可以改变的数据，称为变量。

**1．常量**

常量（constant）指程序运行中不改变的固定值，可以是任何基本数据类型。每一种常量的表示方法依其基本数据类型而定。整型常量比较简单，如 111。浮点型常量要求小数点和小数分量，如 3.1415926。浮点型常量还可以用科学计数法表达，如 2.15e-10，6.23e。字符型常量用一对单引号（'）表示，如'a'、'n'。

- 8 进制和 16 进制常量

以 8 为基的数值系统称为 8 进制，使用数字 0~7。8 进制常量必须以 0 开始，和紧随其后的 8 进制数字串构成，如 0132，就是 10 进制的 90。

以 16 为基的数值系统称为 16 进制，使用数字 0~9 和字母 A~F，代表 10 进制中的 0 到 15。16 进制常量由 0x（或 0X）和紧随其后的 16 进制数字串构成，如 0x5A，就是 10 进制数据的 90。

- 字符串常量

字符串是由一对双引号（"）表示，如"This chapter is about C Language"是一个字符串常量。字符串常量和字符常量是有区别的，前者使用双引号，后者使用单引号，初学者一定要注意这一点。

- 反斜线字符常量

C 语言有一些特殊的反斜线字符常量，也称转义字符，例如'\n'表示换行符，'\t'表示水平制表符，'\r'表示回车，'\\'表示反斜线\，'\0'表示空值，'\"表示单引号，'\'"表示双引号。

**2．变量**

变量是指在程序中其值可以变化的量。C 语言中规定，变量在使用之前必须定义或说，其格式如下：

```
<变量类型> <变量名表>;
```

＜变量名表＞中可以有多个变量名，其间用逗号分隔，例如：

```
int length,width;       /*定义了两个整型变量 length，width。*/
float radius,circle;    /*定义了两个浮点型变量 radius，circle。*/
double money;           /*定义了一个双精度浮点型变量 money。*/
char w1 , w2;           /*定义了两个字符变量 w1，w2。*/
```

**3．变量赋初值**

没有确定值的变量是不能够使用的，所以需要对没有确定值的变量赋初值，例如：

```
length = 10;    /*给整型变量 length 赋初值 10*/
width = 20;     /*给整型变量 width 赋初值 20*/
radius = 2.5;   /*给浮点型变量 radius 赋初值 2.5*/
circle = 18.6;  /*给浮点型变量 circle 赋初值 18.6*/
money = 18.20;  /*给双精度浮点型变量 money 赋初值 18.20*/
w1 = 'a';       /*给字符变量 w1 赋初值'a*/
w2 = 'b';       /*给字符变量 w2 赋初值'b*/
```

### 1.2.4 运算符和表达式

C 语言中的运算符种类多、功能强，这为 C 语言编程带来了方便。表达式是常量、变量、函数和运算符组合起来的有意义的式子。

**1．运算符**

运算符根据用途分类可以分为算术运算符、增 1 和减 1 运算符、关系运算符、逻辑运算符、位操作符、赋值运算符和条件运算符。其中，增 1 和减 1 运算符也是算数运算符，不过它们的用法比较独特，所以单独给以详细的介绍。

- 算术运算符

算术运算符是用来实现算术操作的，又可分为单目运算符和双目运算符。

单目运算符包括求负（–）、增 1（++）和减 1（––）运算符。增 1 和减 1 运算符稍后介绍。求负运算符是用来改变一个操作数的正、负号的。在正数前，该数变成负数；在负数前，

该数变成正数。

双目运算符有+、-、×、/、% 5 种。

+：加法运算符，求两个数的和，如 5+6，其和为 11。

–：减法运算符，求两个数的差，如 9-2，其差为 7。

×：乘法运算符，求两个数的积，如 22×5，其积为 110。

/：除法运算符，求两个数的商，如 6/2，其商为 3。

%：求余运算符，求两个数相除的余数，如 7%5，余数为 2。

- 增 1 和减 1 运算符

增（减）1 运算符，只作用一个变量，使该变量的值增加（减少）1。这两种运算符有两种作用方式，一种是前缀式，运算符放在变量的前面；一种是后缀式，运算符放在变量的后面。

使用前缀运算符的时候，先对变量进行增（减）1 操作，然后使用变量的值；使用后缀运算符的时候，先使用变量的值，然后再进行增（减）1 操作。

初学者往往对前缀式和后缀式混淆，可以从下面的代码中加深了解。

```
#include "stdio.h"
int main(void)
{
  int value = 10;
  printf("使用前缀自增，当前 Value 值为： %d\n",++value);
  printf("增加后的 Value 值： %d\n",value);
  value = 10;
  printf("使用后缀自增，当前 Value 值为：%d\n",value++);
  printf("增加后的 Value 值： %d\n",value);
  value = 10;
  printf("使用前缀自减，当前 Value 值为： %d\n",--value);
  printf("减少后的 Value 值： %d\n",value);
  value = 10;
  printf("使用后缀自减，当前 Value 值为：%d\n",value--);
  printf("减少后的 Value 值： %d\n",value);
}
```

编译并运行这段代码，输出的结果为：

```
使用前缀自增，当前 Value 值为：11
增加后的 Value 值： 11
使用后缀自增，当前 Value 值为：10
增加后的 Value 值： 11
使用前缀自减，当前 Value 值为：9
减少后的 Value 值： 9
使用后缀自减，当前 Value 值为：10
减少后的 Value 值： 9
```

- 关系运算符

关系运算符用来对两个操作数的大小进行比较，共有以下 6 种。

<：小于运算符。左边操作数的值小于右边操作数的值时为真，否则为假。如 4<3，其值为假。

<=：小于等于运算符。左边操作数的值小于或等于右边操作数的值时为真，否则为假。如 23<=25，其值为真。

>：大于运算符。左边操作数的值大于右边操作数的值时为真，否则为假。如 20>18，其值为真。

>=：大于等于运算符，左边操作数的值大于或等于右边操作数的值时为真，否则为假。如 22>=26，其值为假。

==：等于运算符。左边操作数的值等于右边操作数的值时为真，否则为假。如 15==15，其值为真。

!=：不等于运算符。左边操作数的值不等于右边操作数的值时为真，否则为假。如 15!=12，其值为真。

- 逻辑运算符

逻辑运算符是对操作数进行逻辑运算的，有以下 3 种。

!：逻辑非运算符，如果操作数为真，经过该运算符作用之后为假；如果操作数为假，经过该运算符作用之后为真。如!（4<3）的值为真。

&&：逻辑与运算符，只有两个操作数都为真时，结果才为真，否则结果为假。如（'a'<'b'）&&（4<5）的值为真。

||：逻辑或运算符，只要有一个操作数为真，结果就为真；只有两个操作数都为假，结果才为假。如（'a'<'b'）||（4>5）的值为真。

- 位操作符

位操作符是用来实现二进制位的某些操作的，可分为逻辑操作运算符和移位运算符。逻辑运算符包括以下 3 种运算符。

~：求反运算符，该运算符将其操作数的所有二进制位取反，即 1 取反后为 0，0 取反后为 1。如 10 的二进制表示为 00001010，取反后的二进制表示为 11110101，即~10 为 245

&：按位与运算符，该运算符将其两个操作数的所有二进制位从低位到高位对齐，再将每位的两个二进制数相与，只有两个二进制位都为 1 时结果才为 1，其他情况结果为 0。如 9&10，因为 9 的二进制表示为 00001001，所以结果为 00001000，即 8。

|：按位或运算符，该运算符将其两个操作数的所有二进制位从低位到高位对齐，再将每位的两个二进制数相或，当两个二进制位均为 0 时结果为 0，其他情况结果为 1。

移位运算符包括左移和右移两种。>>表示右移运算符，<<表示左移运算符。移位运算符是将运算符左边的操作数化为二进制数，然后按该运算符右边的操作数指定的移动位数进行移位。如 4>>2，相当于 0110（4）右移 2 位，结果为 0001，即为 1。

- 赋值运算符

赋值运算符的功能是给某个变量传送值。该运算符的左边往往是变量名，右端是表达式，即把右边表达式的值传给左边的变量值。

最常见的运算符就是=，用来给左边的变量赋值。还有一些常用的复合赋值运算符如+=、—=、*=、/=、%=、&=、|=、^=、>>=、<<=。复合赋值运算符实际上是一种赋值表达式的简写方法。如 a+=b 等价于 a=a+b，a&=b 等价于 a=a&b，其他类似，不再一一举例。

- 条件运算符

条件运算符是一个三目运算符，由？和：组成的，格式如下：

```
Expr1?Expr2:Expr3
```

Expr1、Expr2 和 Expr3 三个表达式是操作数，先计算 Expr1 表达式的值，如果表达式非零，则整个三目运算符组成的表达式的值为 Expr2，否则为 Expr3。

如下面的代码所示：

```
#include "stdio.h"
int main(void)
{
  int a = 10;
  int b = 20;
  int c = a>b?a:b;    /*三目运算符取 a 和 b 的最大值*/
  printf("a 和 b 的最大值为： %d\n",c);
}
```

编译并运行后，运行结果为：

```
a 和 b 的最大值为：20。
```

**2．表达式**

C 语言的表达式主要有算术表达式、关系表达式、逻辑表达式、赋值表达式和条件表达式。

- 算术表达式

算术表达式的类型由表达式中各操作数的类型决定。如果各操作数的类型不同，则表达式的类型是各操作数中的优先级最高的类型。

类型的优先级顺序，由低到高分别为 int 类型、unsigned 类型、long 类型、double 类型。如果遇到 char 类型、short 类型、float 类型需要对它们进行转换。char 类型和 short 类型转换为 int 类型，float 类型转换为 double 类型。如，一个 int 类型和一个 char 类型数据运算，先将 char 类型转换为 int 类型，例如，计算 10+'b'，先将字符类型'b'转换为 int 类型值 98，相应的算术表达式的值为 108。

同理，int 类型和 double 类型数据进行运算，先将 int 类型转换成 double 类型，如下面的代码所示：

```
#include "stdio.h"
int main(void)
{
  int a = 10;
  double f1= 3;
  printf("Value 值： %.2f\n",a/f1);  /*输出的结果以浮点型格式输出*/
}
```

编译并运行后，运行结果为：

```
Value 值： 3.33
```

其中，a 是整型的，f1 是双精度浮点型的（尽管 f1 的值是整数 3），它们的商将会是双精度浮点类型。“%.2f”表示变量 f1 将以浮点型数据格式输出，且四舍五入保持两位小数。

- 关系表达式

由关系运算符组成的表达式称为关系表达式。关系表达式的值是 0 或 1，可用来作为语句的执行条件，在程序控制语句中经常用到。

- 逻辑表达式

逻辑表达式是由逻辑运算符组成的表达式。逻辑表达式的值是 0 或 1。如（'a'<'b'）||（4>5）的值为 1。

- 赋值表达式

赋值表达式是由赋值运算符组成的表达式。当赋值运算符左右类型不一致时，系统将赋值运算符右端表达式强制转换成左端变量的类型，并将转换类型之后的表达式值赋给左边变量。

- 条件表达式

由条件运算符组成的表达式称为条件表达式。条件表达式在程序中相当于一个简单的条件语句。如

```
c = a>b?a:b;
```

同下面的代码是等价的：

```
if(a>b)
  c = a;
else
  c = b;
```

### 1.2.5 注释

使用注释的目的是偏于阅读和理解程序，其内容会被编译器忽略。注释最好是放在它要解释的语句附近。注释可以单独成行或放在解释的语句之后，称之为行注释。如果注释较长，可以分行，称之为块注释。

初学者要养成良好的书写注释的习惯。这一方面可以增加程序的可读性，另一方面在程序调试的时候，也可以利用注释跳过某些语句。

在代码中添加行注释比较简单，注释以“/*”开始，以“*/”结束，注释的内容在“/*”和“*/”之间，方法如下：

```
/*这是一行注释*/
```

也可以使用“//”进行行注释，注释的内容在“//”之后，方法如下：

```
//这是一行注释
```

如果是块注释，只需在块的开始处添加“/*”，在块的结尾处添加“*/”，方法如下：

```
/*这是第一行注释
这是第二行注释*/
```

或者：

```
/*
这是第一行注释
这是第二行注释
*/
```

### 1.2.6 在 Visual C++中运行示例程序

Visual C++是 Microsoft Visual Studio 开发组件中最强大的编程工具，方便地编辑、编译和运行 C 语言的程序代码。



需要全本请访问 www.ertongbook.com