第1章 C语言与程序设计概述

内容提要

(1)知识点：本章主要讨论什么是程序、程序设计语言、程序设计语言和C语言的发展历史、C语言程序的基本框架、使用Visual C++开发C语言程序的步骤。

(2)难点：熟练掌握C语言程序的开发环境和开发步骤。

1.1 程序与程序设计语言

1.1.1 程序与程序设计

1. 程序

程序是为实现特定目标或解决特定问题而用计算机语言编写的命令序列的集合。由于计算机还不能理解人类的自然语言，所以还不能用自然语言编写计算机程序。一个程序应该包括以下两方面的内容。

(1)对数据的描述。在程序中要指定数据的类型和数据的组织形式，即数据结构。

(2)对操作的描述。即操作步骤，也就是算法。

著名计算机科学家沃思提出一个公式：数据结构+算法=程序。实际上，一个程序除了以上两个主要的要素外，还应当采用程序设计方法进行设计，并且用一种计算机语言来表示。

2. 程序设计

程序设计是给出解决特定问题程序的过程，是软件构造活动中的重要组成部分。程序设计往往以某种程序设计语言为工具，给出这种语言下的程序。程序设计过程应当包括分析、设计、编码、测试、排错等不同阶段。

为解决某一问题编写的程序不是唯一的，不同的用户编写程序的思路也不会完全一样，因此不同的程序有不同的执行效率，这涉及到程序的优化、程序所采用的数据结构以及算法等多方面的因素。

1.1.2 程序设计语言

程序设计语言用于书写计算机程序的语言。计算机程序设计语言的发展经历了机器语言、汇编语言、高级语言三个阶段。

1. 机器语言

机器语言指令是由二进制的0和1构成，不同的CPU具有不同的指令系统。机器语言程序难编写、难修改、难维护，需要用户直接对存储空间进行分配，编程效率极低。但由于机器语言使用的是针对特定型号计算机的语言，它具有灵活、直接执行和运算效率高等特点。

2. 汇编语言

汇编语言指令是机器指令的符号化，与机器指令存在着直接的对应关系，所以汇编语言同样存在难学难用、容易出错、维护困难等缺点。但由于汇编语言针对计算机特定硬件而编制程序，能准确发挥计算机硬件的功能和特长，程序精炼效率高。

使用汇编语言编写的程序，机器不能直接识别，要由一种程序将汇编语言翻译成机器语言，这种起翻译作用的程序叫汇编程序。汇编程序是语言处理系统软件，汇编程序把汇编语言翻译成机器语言的过程称为汇编。

3. 高级语言

人们把机器语言和汇编语言称为低级语言，它们都是面向机器的语言，和具体机器的指令系统密切相关。高级语言与计算机的硬件结构及指令系统无关，在形式上接近于算术语言和自然语言，具有更强的表达能力，可方便地表示数据的运算和程序的控制结构，能更好的描述各种算法，而且容易学习掌握。但高级语言编译生成的程序代码一般比用汇编程序语言设计的程序代码要长，执行的速度也慢。

高级语言经历了从早期语言到结构化程序设计语言，从面向过程的程序设计语言到面向对象的程序设计语言的过程。FORTRAN语言是目前国际上广为流行、也是使用得最早的一种高级语言，在工程与科学计算中占有重要地位，备受科技人员的欢迎。BASIC语言是在20世纪60年代初为适应分时系统而研制的一种交互式语言，可用于一般的数值计算与事务处理。BASIC语言结构简单，易学易用，并且具有交互能力，成为许多初学者学习程序设计的入门语言。

(1)结构化程序设计语言

20世纪70年代以来，结构化程序设计和软件工程的思想日益为人们所接受和欣赏。在它们的影响下，先后出现了一些很有影响的结构化语言，这些结构化语言直接支持结构化的控制结构，具有很强的过程结构和数据结构能力。PASCAL、C、FORTRAN语言就是它们的突出代表。

PASCAL语言是第一个系统地体现结构化程序设计概念的现代高级语言，软件开发的最初目标是把它作为结构化程序设计的教学工具。由于它模块清晰、控制结构完备、有丰富的数据类型和数据结构、语言表达能力强、移植容易，不仅被国内外许多高等院校定为教学语言，而且在科学计算、数据处理及系统软件开发中都有较广泛的应用。

C 语言功能丰富，表达能力强，有丰富的运算符和数据类型，使用灵活方便，应用面广，移植能力强，编译质量高，目标程序效率高，具有高级语言的优点。同时，C 语言还具有低级语言的许多特点，如允许直接访问物理地址，能进行位操作，能实现汇编语言的大部分功能，可以直接对硬件进行操作等。用 C 语言编译程序产生的目标程序，其质量可以与汇编语言产生的目标程序相媲美，具有“可移植的汇编语言”的美称，成为编写应用软件、操作系统和编译程序的重要语言之一。

(2)面向对象程序设计语言

以“对象 + 消息”程序设计范式构成的程序设计语言，称为面向对象语言。比较流行的面向对象语言有 Delphi、Visual Basic、Java、C + + 等。

Delphi 语言具有可视化开发环境，提供面向对象的编程方法，可以设计各种具有 Windows 内格的应用程序(如数据库应用系统、通信软件和三维虚拟现实等)，也可以开发多媒体应用系统。

Visual Basic 语言简称 VB，是为开发应用程序而提供的开发环境与工具。它具有很好的图形用户界面，采用面向对象和事件驱动的新机制，把过程化和结构化编程集合在一起。它在应用程序开发中的图形化构思，无需编写任何程序，就可以方便地创建应用程序界面，且与 Windows 界面非常相似，甚至是一致的。

Java 语言是一种面向对象的、不依赖于特定平台的程序设计语言，简单、可靠、可编译、可扩展、多线程、结构中立、类型显示说明、动态存储管理、易于理解，是一种理想的、用于开发 Internet 应用软件的程序设计语言。

C + + 语言进一步扩充和完善了 C 语言，成为一种面向对象的程序设计语言。C + + 提出了一些更为深入的概念，它所支持的这些面向对象的概念容易将问题空间直接地映射到程序空间，为程序员提供了一种与传统结构程序设计不同的思维方式和编程方法。C + + 目前流行的最新版本是 Borland C + + 、Symantec C + + 和 Microsoft Visual C + + 。

1.2　C 语言的发展过程

C 语言的发展颇为有趣，它的原型是 1960 年出现的 ALGOL 60 语言(也成为 A 语言)。和汇编语言相比，A 语言的可读性、可移植性较好，但离硬件远，不适合编系统程序。

1963 年，剑桥大学将 ALGOL 60 语言发展成为 CPL(Combined Programming Language)语言。CPL 更接近硬件一些，但规模大。

1967 年，剑桥大学的 Matin Richards 对 CPL 语言进行了简化，于是产生了 BCPL 语言。

1970 年，美国贝尔实验室的 KenThompson 将 BCPL 进行了修改，并为它起了一个有趣的名字“B 语言”。

1970 年，美国贝尔实验室的 Ken Thompson 将 BCPL 进行了修改，并为它起了一个有趣的名字“B 语言”，并且用 B 语言写了第一个 UNIX 操作系统。

1973 年，美国贝尔实验室的 D. M. RITCHIE 在 B 语言的基础上最终设计出了一种新的语言，他取了 BCPL 的第二个字母作为这种语言的名字，这就是 C 语言。

为了使 UNIX 操作系统推广，1977 年 Dennis M. Ritchie 发表了不依赖于具体机器系统的 C 语言编译文本《可移植的 C 语言编译程序》。

1978 年 Brian W. Kernighian 和 Dennis M. Ritchie 出版了名著《The C Programming Language》，从而使 C 语言成为目前世界上流行最广泛的高级程序设计语言。

1988 年，随着微型计算机的日益普及，出现了许多 C 语言版本。由于没有统一的标准，使得这些 C 语言之间出现了一些不一致的地方。为了改变这种情况，美国国家标准研究所(ANSI)为 C 语言制定了一套 ANSI 标准，成为现行的 C 语言标准。

1.3 C 语言程序基本结构

1.3.1 C 语言字符集与标识符

1. C 语言字符集

字符是组成语言的最基本的元素。C 语言字符集由字母、数字、空格、标点和特殊字符组成。在字符常量，字符串常量和注释中还可以使用汉字或其它可表示的图形符号。

(1)字母

小写字母 a ~ z 共 26 个，大写字母 A ~ Z 共 26 个。

(2)数字

0 ~ 9 共 10 个数字。

(3)空白符

空格符、制表符(Tab 键)、换行符等统称为空白符。空白符只在字符常量和字符串常量中起作用。在其它地方出现时，只起间隔作用，编译程序对它们忽略不计。在程序中使用空白符与否，对程序的编译不产生影响，但在程序中适当地使用空白符将增加程序的清晰性和可读性。

(4)标点和特殊字符

比如：加号(+)、减号(-)、分号(;)、逗号(,)等。

2. 标识符

标识符是给程序中的某个对象所起的名字，这些对象可以是数据类型、变量、常量、函数、数组结构体和文件等。

C 语言规定：一个标识符由字母、数字和下划线组成，第一个字符必须是字母或下划线。通常以下划线开头的标识符是编译系统专用的，所以在编写 C 语言程序时，最好不要使用以下划线开头的标识符。

C 语言规定标识符的长度最大可达 255 个字符，但是在实际编译时，只有前面 32 个字符能够被正确识别，所以标识符的长度不要超过 32 个字符。对于一般的应用程序来说，32 个字符的标示符长度足够用了。

C 语言对大小写字符敏感，所以在编写程序时要注意大小写字符的区分。例如：对于 student 和 Student 这两个标识符来说，C 语言会认为这是两个完全不同的标识符。

C 语言程序中的标识符命名应做到简洁明了、含义清晰。这样便于程序的阅读和维护。比如，在比较最大值时，使用 max 来定义该标识符。

C 语言中的标识符分为以下三类：

(1)关键字

关键字是由 C 语言规定的具有特定意义的字符串，通常也称为保留字。用户定义的标识符不应与关键字相同。标准 C 语言有 32 个关键字，根据关键字的作用，可分其为数据类型关键字、控制语句关键字、存储类型关键字和其它关键字四类。

auto double int struct break else long switch case enum register typedef char extern return union const float short unsigned continue for signed void default goto sizeof volatile do if while static

(2)标准标识符

C 语言中规定了一类标准标识符，通常被用作库函数名和预编译命令名等。虽然 C 语言允许用户另作它用，但是这将使这类标识符失去系统规定的原意而引起不必要的误解，因此，建议用户不要将这些标准标识符另作它用。比如，sin、printf、scanf 等。

(3)用户自定义标识符

由用户自己根据需要定义的标识符。用户给标识符命名时除了要遵循标识符的命名规则外，还应该考虑到标识符“见名知义”的原则。C 语言区分大小写字母。通常用小写字母表示变量、函数、数组和文件等的名称，而用大写字母表示符号常量名，以便增强程序的可读性。

例：合法标识符和非法标识符。

以下能用作用户标识符的是：

WORD	as_ b3	_ 224N	Else	int_ 2_	Li_ ming
_ main	If	22A	lea_ 1	avg3	Xyw8

以下不能用作用户标识符的是：

sizeof	- wer	born365. com	x - i - a - o	2_ int_
printf	$ _ 238	M. J. YORK	a * b	8Tea

1.3.2　C 程序的基本结构

一个 C 语言源程序可以由一个或多个源文件组成，每个源文件可由一个或多个函数组成，一个源程序不论由多少个文件组成，都有一个且只能有一个 main 函数，即主函数。

为了说明 C 语言源程序结构的特点，先看以下 2 个程序。

【例 1 - 1】一个简单的 C 程序示例(exp1 - 1. c)。

```
/* 第 1 个程序:显示 This is my first C program! */
#include <stdio.h>          //预处理,包含文件
void main()                 //主函数
{
    printf("This is my first C program! \n");      //调用输出函数
```

```
}
```

【例1-2】输入2个整数，调用函数add计算它们的和(exp1-2.c)。

```
/* 第2个程序:计算和* /
#include  < stdio. h >
/* 下面的函数是计算x与y的和* /
int add(int x, int y)
{   int z;
      z = x + y;
      return (z);
}
/*     下面是主函数    * /
void main()
{int a, b, sum;
      scanf("%d%d", &a, &b);
      sum = add(a, b);
      printf("sum =  %d\n", sum);
}
```

本程序由主函数main和被调用函数add组成，在主函数中输入2个整数到a和b中，然后通过语句sum = add(a，b)调用函数add，计算结果由return语句返回给主函数。这两个函数在位置上是独立的，可以把主函数main放在前面，也可以把主函数main放在函数add的后面。

scanf和printf是C语言提供的标准输入输出函数，&a中的“&”的含义是取地址，程序中scanf函数的作用是将从键盘上键入的2个数输入到变量a和b所标志的内存单元中，或者称对a和b赋值。

从以上两个例子中可以看到，一个C语言程序的基本结构分为3个部分：

1. 预处理部分

#include < stdio. h > 是一条预处理命令，用“#”号开头，后面不能加“;”号，stdio. h是系统提供的头文件，其中包含有关输入输出函数的信息，如程序中用到的printf输出函数。输出语句中的“%d”为输出格式符，它指定输出结果时的数据类型和格式，程序在执行时，该位置由具体数据替代。

2. 函数

(1)函数是C语言程序的基本单位，用于描述程序所完成的功能。

程序中main是主函数名，C语言规定必须用main作为主函数名，函数名后的一对圆括号不能省略，圆括号中内容(参数)可以是空的。

一个C程序可以包含任意多个函数，但必须有且只有一个main主函数。一个C程序总是从主函数开始执行，最后在主函数结束。主函数的位置在程序中是任意的，其他函数总是通过函数调用语句来执行。

主函数可以调用任何其他函数，任何非主函数之间也可以相互调用，但是均不能调用主函数。

(3)函数包含两部分：函数首部和函数体。函数首部包括函数类型、函数名和形式参数；函数体需要用花括号括起来，左括号表示函数体的开始，右括号表示函数体的结束。函数定义形式可表示如下：

```
函数类型 函数名(形式参数列表)
{
    函数体
}
```

上面2个程序中的主函数的类型为void，表示是空类型。C语言规定：空类型的函数不需要返回值。函数体中包含各种语句，语句是程序的基本执行单位，语句可分为定义(声明)部分和执行语句部分。每一条语句都必须用分号“;”结束，语句的数量不限，程序中由这些语句向计算机系统发出指令，如程序中printf输出语句。C语言本身没有输入输出语句。输入和输出操作是由调用系统提供的输入输出函数来完成的。

3. 注释

为了对程序代码进行说明，可以添加注释，作用是帮助用户阅读程序，它对程序的运行不起作用，在对源程序进行编译时，注释会被忽略。注释内容可以是西文，也可以是中文，注释通常用于说明变量的含义、程序段的功能。注释部分可以放在程序中任意合适的位置，一个好的程序应该有必要的注释，这样可以增加可读性。

C程序中用“/*……*/”表示注释一个程序块，注释内容可以是一行或多行。

“/*”和“*/”必须成对出现，且“/”和“*”之间不能有空格。

1.3.3　C程序的书写格式

为了便于阅读、理解和维护程序，在书写程序时应遵循以下规则。

(1)一个说明或一个语句占一行。

(2)用{}括起来的部分，通常表示了程序的某一层次结构。{}一般与该结构语句的第一个字母对齐，并单独占一行。

(3)正确使用缩进。低一层次的语句或说明可比高一层次的语句或说明缩进若干格后书写，以便看起来更加清晰，增加程序的可读性。

在编程时应力求遵循这些规则，以养成良好的编程风格。

1.4　C语言程序开发过程

在写好一个C程序后，要经过编辑、编译、链接等步骤才能运行。

1. 编辑

选择适当的编辑程序(如Visual C++6.0)，将C语言源程序通过键盘输入到计算机中，并以文件的形式存入到磁盘中(.cpp或者.c)。

2. 编译

将源程序翻译成二进制形式的目标程序(.obj)。

3. 连接

编译后生成的目标文件与系统的函数库和其他目标程序，经过连接后生成最终的可执行程序(.exe)。

1.4.1 Visual C++6.0 集成开发环境简介

Visual C++ 6.0 是美国微软公司开发的 C++集成开发环境，它集源程序的编写、编译、连接、调试、运行，以及应用程序的文件管理于一体，是当前 PC 机上最流行的 C 程序开发环境。Visual C++ 6.0 也可以编写控制台程序，系统中也包含 C 语言的编译器，可以用来编译 C 程序，源程序文件的扩展名可以是.cpp 和.c。

1. Visual C++ 6.0 界面

Visual C++ 6.0 集成开发环境被划分成 4 个主要区域：菜单栏和工具栏、工作区窗口、代码编辑窗口和输出窗口，如图 1-1 所示。

图 1-1 Visual C++ 集成开发环境

(1)菜单栏。Visual C++菜单栏包含了开发环境中几乎所有的命令，它为用户提供了代码操作、程序的编译、调试、窗口操作等一系列的功能。与一般 Windows 应用程序一样有文件、编辑、视图、插入、工程、编译、工具、窗口、帮助等菜单。

(2)工具栏。通过工具栏，可以迅速地使用常用的菜单命令。最常用的工具栏是标准工具栏，当鼠标指向这些工具时，通常有信息提示工具的含义，因此也比较容易掌握。若要显示或隐藏某个工具栏，则在任一工具栏的快捷菜单中选择相应的命令即可。

(3)项目工作区。项目是开发一个程序时需要的所有文件的集合，而工作区是进行项目组织的工作空间。利用项目工作区窗口可以观察和存取项目的各个组成部分。在 Visual C++中，一个工作区可以包含多个项目。

项目工作区有 Class View 和 File View 选项卡，Class View 显示当前项目的类，全局的变量和函数也在这里显示。File View 显示当前项目的源文件、头文件、资源文件等。

在 Visual C++中，项目中所有的源文件都是采用文件夹的方式进行管理的，它将项目名作为文件夹，在此文件夹下包含源程序代码文件(.cpp、.h)、项目文件(.dsp)以及项目工作区文件(.dsw)等。若要打开一个项目，只需打开对应的项目工作区文件即可。

(4)代码窗口。一般位于开发环境中的右边，各种程序代码的源文件、资源文件、文档文件等都可以通过该窗口显示。

(5)输出窗口。输出窗口有多个选项卡，最常用的是“编译”。在编译、连接时，这里会显示有关的信息，供调试程序用。

(6)状态栏。状态栏一般位于开发环境的最底部，它用来显示当前操作状态、注释、文本光标所在的行、列号等信息。

1.4.2　在 Visual C++ 6.0 中创建一个 C 程序的步骤

使用 Visual C++ 6.0 平台开发 C 语言程序的主要步骤如下：

1. 运行 Visual C++ 6.0

安装好 Visual C++6.0 后，可以通过 Windows 开始菜单中的程序子菜单找到 Visual C++的启动菜单项，启动 Visual C++6.0。

2. 新建一个工程

在 Visual C++ 6.0 中以工程为单位开发 C 语言程序。每个工程可以包含一个或多个 C 语言源程序文件，其中只有一个 C 语言源程序文件中包含 main 函数。

选择 Visual C++ 6.0 的 File 主菜单中的 New 菜单项将会弹出新建工程对话框，可以选择开发各种类型的应用程序，本书只通过 Win32 Console Application 类型的工程演示程序的开发。在对话框的右边的 Location 中选择工程创建的位置，在 Project Name 文本框中输入工程名称，如图 1-2 所示。

最终 Visual C++ 6.0 会在 Location 指定的位置创建以工程名称命名的文件夹，工程所有的文件都位于这个文件夹中。

当单击 OK 按钮后，将有向导引导创建这个工程的基本文件。

在向导的第一步，选择 An empty project，然后单击 OK 按钮，这将创建一个没有任何源程序文件的空工程。

3. 添加 C 语言源程序文件

当空工程创建完成后，可以在 Visual C++ 6.0 集成环境的左边以文件视图方式显示的工程作空间中查看目前工程包含的文件情况，这时没有任何包含文件，如图 1-3 所示。

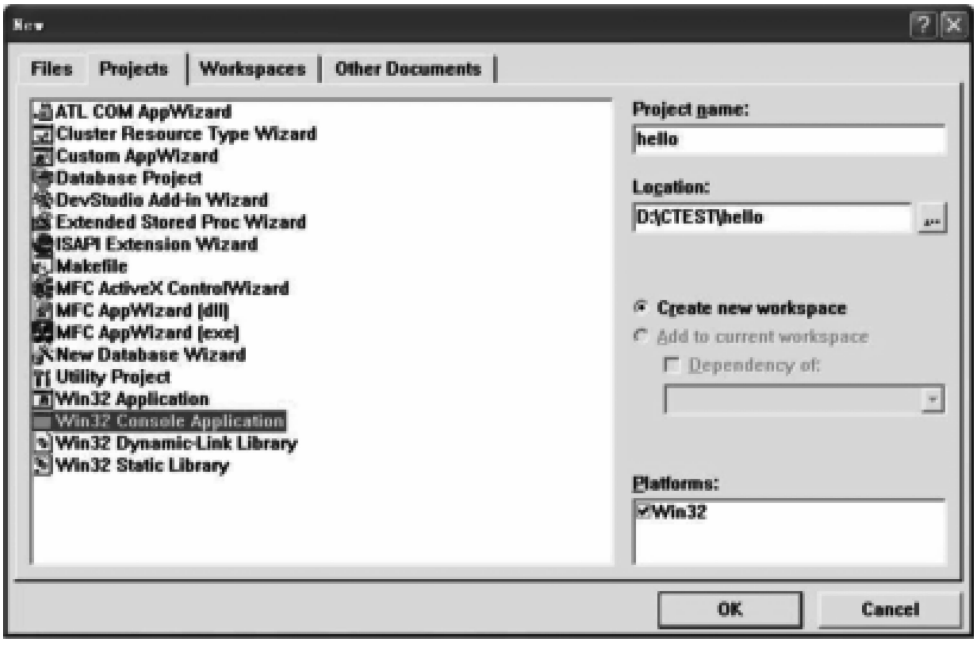


图 1 – 2　在 VC + +6.0 中新建工程

图 1 – 3　VC + + 6.0 中的新工程

要添加一个 C 语言源程序文件，可以通过“New Text File”快捷按钮新建一个文本文件，然后将其保存为 C 语言源程序文件，即后缀为“.C”的文件。然后将新建的 C 语言源程序文件添加到工程工作空间的 Source Files 中，如图 1 – 4 所示。

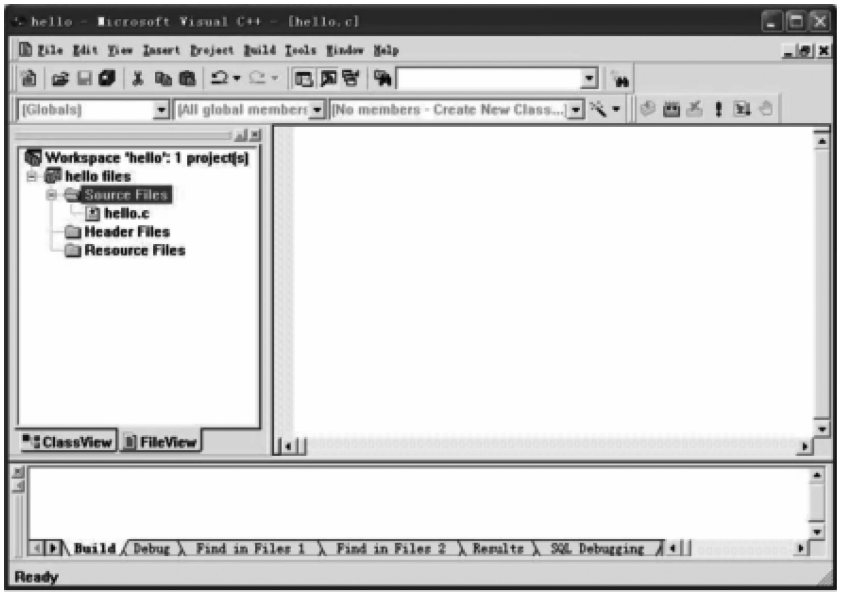


图 1－4　VC＋＋6.0 中添加源程序文件

4. 编译、连接工程文件

在新添加的 hello. c 源程序文件中输入和编辑源程序完毕后，在主菜单 Build 中选择相应菜单选项，或者通过按下 Build 快捷按钮（或者快捷键 F7）完成对源程序的编译和连接工作，如图 1－5 所示。

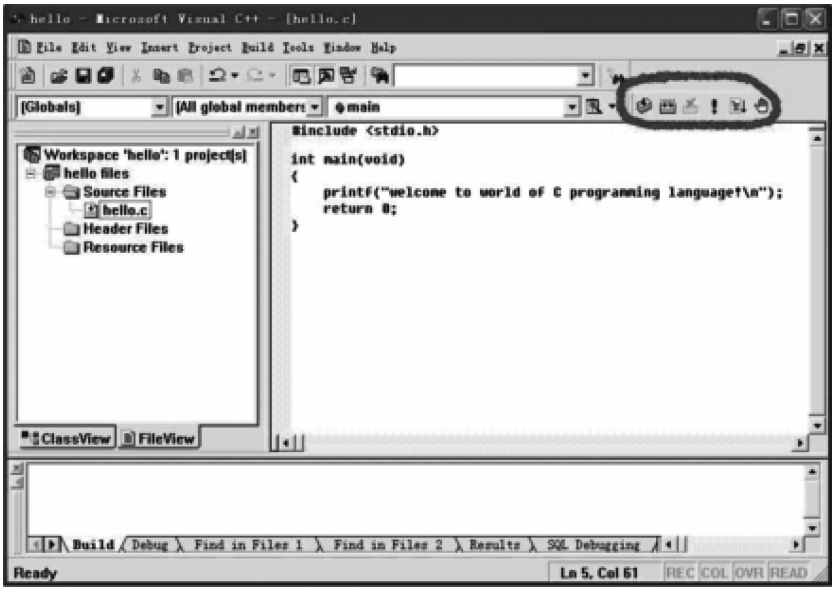


图 1－5　VC＋＋6.0 中编译和连接

5. 运行程序

当编译、连接都正确完成后，可以按下“Execute Program”快捷按钮（或者快捷键 Ctrl＋

F5)运行刚生成的应用程序。Visual C++6.0 将会生成一个对应的进程和一个对应窗口，在窗口中可看到程序运行输出到屏幕上的信息，如图 1-6 所示。

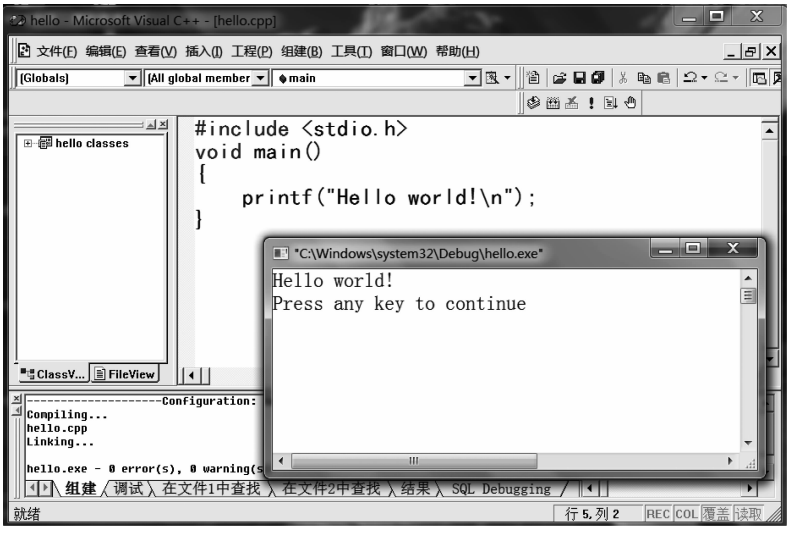


图 1-6　VC++6.0 中运行程序

1.5　在线评测系统介绍

在课程实验和 ACM 竞赛中，将编写好的程序提交到在线评测系统(Online Judge)中，在线评测系统将自动判断程序的正确性。

1.5.1　样题

在浏览器中输入地址 http://192.168.51.51，进入韶关学院在线评测系统，单击“题目”、“题目列表”，选择相应的练习题，如图 1-7 所示。

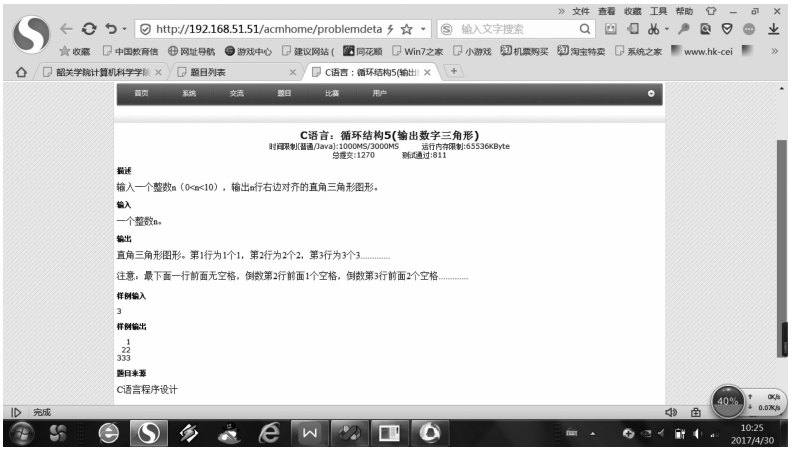


图 1-7　在线评测系统中的样题

1.5.2　提交代码

单击题目网页下的“提交”链接，选择源代码的语言为 GCC，从 Visual C++6.0 中将代码复制到提交框中，点击下方的“Submit”，如图 1-8 所示。

图 1-8　提交代码

本章小结

本章介绍了 C 语言程序设计的基本概念、C 语言的开发过程、使用 Visual C++ 开发 C 语言程序的步骤。

(1)计算机是由程序控制的，程序是用程序设计语言来编写的。

(2)程序设计语言的发展经历了机器语言、汇编语言、高级语言三个阶段。

(3)机器语言和汇编语言依赖于具体的计算机，没有通用性，高级语言不依赖于具体的计算机，通用性强。

(4)C 语言是一种高级语言，既具有高级语言的特点，又具有低级语言的特点，既可以作为系统设计语言，又可以作为应用程序设计语言。

(5)C 语言程序是由函数组成，一个 C 语言程序可以包含一个或者多个函数，但必须有一个 main 函数。

(6)开发一个 C 语言程序需要经过编辑、编译、链接和运行四个步骤。

习 题

一、选择题

1. 下列四组选项中，均不是 C 语言关健字的选项是(　　)。

A）define　　B）gect　　C）include　　D）while
　　IF　　char　　scanf　　go
　　type　　printf　　case　　pow

2. 以下选项中不合法的用户标识符是(　　)。

A）abc. c　　B）file　　C）Main　　D）PRINT

3. 下列标识符组中，合法的用户标识符为(　　)。

A）_ 0123 与 ssiped　　B）del-word 与 signed
C）list 与 * jer　　D）keep% 与 wind

4. C 语言程序是由(　　)组成。

A）程序　　B）语句　　C）字符　　D）函数

第2章 数据类型与表达式

内容提要

(1)知识点：本章主要讨论C语言的基本数据类型及表达式。要求学生清楚C语言为什么要划分数据类型，掌握变量、常量的概念，掌握整型、浮点型、字符型3种数据类型的特点以及使用。掌握C语言主要运算符的功能和特点，掌握主要运算符的优先级和结合性。了解数据类型的相互转换规则。

(2)难点：对数据类型概念的理解，变量的定义与使用，运算符与表达式的掌握与理解，运算符的优先级以及结合性的理解。

2.1 数据类型概述

所谓数据类型是按各种数据的性质，表示形式，占据存储空间的多少，构造特点来划分的。在C语言中，数据类型可分为：基本数据类型，构造数据类型，指针类型，空类型四大类，如图2-1所示。

- 数据类型
  - 基本类型
    - 字符型(char)
    - 数值类型
      - 整型(short，int，long，unsigned)
      - 实型
        - 单精度型(float)
        - 双精度型(double)
    - 枚举类型(enum)
  - 构造类型
    - 数组类型([ ])
    - 结构体类型(struct)
    - 共用体类型(union)
  - 指针类型(*)
  - 空类型(void)

图2-1 C语言数据类型

基本数据类型：基本数据类型最主要的特点是，其值不可以再分解为其它类型。也就是说，基本数据类型是自我说明的。

构造数据类型：构造数据类型是根据已定义的一个或多个数据类型用构造的方法来定义的。也就是说，一个构造类型的值可以分解成若干个“成员”或“元素”。每个“成员”都是一个基本数据类型或又是一个构造类型。

指针类型：指针是一种特殊的，同时又是具有重要作用的数据类型。其值用来表示某个变量在内存储器中的地址。虽然指针变量的取值类似于整型量，但这是两个类型完全不同的量，因此不能混为一谈。

空类型：调用后不需要向调用者返回函数值的函数，可以定义为“空类型”。其类型说明符为 void。

2.2 基本数据类型

2.2.1 整型数据

整型(INTEGER)数据是不包含小数部分的数值型数据，只用来表示整数，以二进制形式存储。根据整数在计算机内存中所占用的空间大小，可分为短整型(short int 或 short)、基本整型(int)和长整型(long int 或 long)。不同的编译器，整型数据所占的字节数是不一样。ANSI 标准定义 int 是占 2 个字节，TC 是按 ANSI 标准的，int 是占 2 个字节的。但是在 VC 里，int 占 4 个字节。根据整数的最高位是否作为符号位，可将整数分为有符号(signed)整型和无符号(unsigned)整型。各种数据类型占用的内存空间的大小和数据的取值范围如表 2－1 所示：

表 2－1　VC 中各类整型数据所占内存字节数及数的表示范围

类型说明符	字节数	数的范围	
int	4	－2147483648～2147483647	$-2^{31}\sim(2^{31}-1)$
unsigned int	4	0～4294967295	$0\sim(2^{32}-1)$
short int	2	－32768～32767	$-2^{15}\sim(2^{15}-1)$
unsigned short int	2	0～65535	$0\sim(2^{16}-1)$
long int	4	－2147483648～2147483647	$-2^{31}\sim(2^{31}-1)$
unsigned long	4	0～4294967295	$0\sim(2^{32}-1)$

2.2.2 实数类型

实数类型的数据即实型数据。在 C 语言中，实型数据分为单精度和双精度两种，单精度用 float 表示，双精度用 double 表示，实型数据又称为浮点数，它们所占的字节数及取值范围如表 2－2 所示。

表 2－2　C 语言实型数据分类

类型说明符	字节数	有效数字(位)	数的范围
float	4	6～7	3.4×10^{-38}～3.4×10^{38}
double	8	15～16	1.7×10^{-308}～1.7×10^{308}

在一般情况下，float 型的数据在内存中占 4 个字节的存储空间，double 型的数据在内存中占 8 个字节的存储空间。对于程序中的实型数据来说，float 型的数据提供 7 位有效数字，double 型的数据提供 15～16 位的有效数字。

一个实型常量赋值给一个实型变量时，只根据实型变量表示的有效数字的位数，在实型常量中按从左向右的方向截取数据。float 型的变量只能保存 7 位有效数字，double 型的变量能保存 15～16 位的有效数字。

2.2.3　字符型数据

字符型数据类型的标识符为 char，在内存中占 1 个字节。它包括中文字符、英文字符、数字字符和其他 ASCⅡ字符，其长度(即字符个数)范围是 0～254 个字符。

2.3　常量与变量

对于基本数据类型量，按其取值是否可改变又分为常量和变量两种。在程序执行过程中，其值不发生改变的量称为常量，其值可变的量称为变量。它们可与数据类型结合起来分类。例如，可分为整型常量、整型变量、实型常量、实型变量、字符常量、字符变量、字符串常量等。在程序中，常量是可以不经说明而直接引用的，而变量则必须先定义后使用。

2.3.1　常量

1. 整型常量

整型常量就是整常数。在 C 语言中，使用的整常数有八进制、十六进制和十进制三种。

(1)十进制整数是由不以 0 开头的 0～9 的数字组成的数据。

(2)八进制整数是由以 0 开头的 0～7 的数字组成的数据。

(3)十六进制整数是由以 0x 或 0X 开头的 0～9 的数字及 A～F 的字母组成的数据。

例如：0，65，83 是十进制数；00，071，0123 是八进制数；0x0，0X0，0X55，0x55，0X3f，0x3f 是十六进制数。

整数常量的取值范围是有限的，它的大小取决于此类整型数的类型，与所使用的进制形式无关。尽管整数可以用十进制、八进制、十六进制来表示，但它在内存中都是按二进制来存储的。

C 语言规定，凡是在整型数后加小写字母 l 或大写字母 L 作为长整数处理；凡是在整

型数后加后缀“U”或者“u”，表示该数为无符号整型。

2. 实型常量

实型也称为浮点型。实型常量也称为实数或者浮点数。在 C 语言中，实数只采用十进制。它有二种形式：十进制小数形式、指数形式。

(1)十进制数形式：由数码 0 ~ 9 和小数点组成。例如：0.0、25.0、5.789、0.13、5.0、300.、-267.8230、.35、43. 等均为合法的实数。注意，必须有小数点。

(2)指数形式：由十进制数，加阶码标志“e”或“E”以及阶码(只能为整数，可以带符号)组成。其一般形式为：

a E n(a 为十进制数，n 为十进制整数)，其值为 $a * 10^n$。

如：2.3E5（等于 $2.3 * 10^5$）、3.4E-2（等于 $3.4 * 10^{-2}$）、0.6E7（等于 $0.6 * 10^7$）、-2.7E-2、(等于 $-2.7 * 10^{-2}$)。而 347（无小数点）、E6（阶码标志 E 之前无数字）、-9（无阶码标志）、53. -E4（负号位置不对）、2.3E（无阶码)都是不合法的实数。

默认状态下，实型常量被识别为双精度 double 类型浮点数，可以使用后缀为“f”或“F”表示单精度 float 类型浮点数，后缀为“l”或“L”表示双精度 double 类型浮点数。比如 34.12 是 double 类型，34.12f 是 float 类型。

3. 字符型常量

字符型常量包括一般字符常量和转义字符常量。

(1)字符常量

字符常量是用单引号括起来的一个字符。例如：'a'、'b'、'='、'+'、'?'都是合法字符常量。

在 C 语言中，字符常量有以下特点：

①字符常量只能用单引号括起来，不能用双引号或其它括号。

②字符常量只能是单个字符，不能是字符串。

③字符可以是字符集中任意字符。但数字被定义为字符型之后就不能参与数值运算。如'5'和 5 是不同的。'5'是字符常量，不能参与运算。

(2)转义字符

转义字符是一种特殊的字符常量，由一对单引号括起来，以反斜杠“\”开头，后面跟若干字符。反斜杠之后的字符被转换为另外的含义，不同于字符原有的意义，故称为“转义字符”。转义字符通常用来表示 C 语言中的一些特殊的字符(例如控制字符)。C 语言常见转义字符及含义如表 2-3 所示。

表 2-3　常见转义字符表

转义字符	含义	ASCII 码(10 进制)
\o	空字符(NULL)	0
\n	换行符(LF)	10
\r	回车符(CR)	13

续表

转义字符	含义	ASCII 码(10 进制)
\t	水平制表符(HT)	9
\v	垂直制表(VT)	11
\a	响铃(BEL)	7
\b	退格符(BS)	8
\f	换页符(FF)	12
\'	单引号	39
\"	双引号	34
\\	反斜杠	92
\?	问号字符	63
\ddd	任意字符	三位八进制
\xhh	任意字符	二位十六进制

字符常量中使用单引号和反斜杠以及字符常量中使用双引号和反斜杠时，都必须使用转义字符表示，即在这些字符前加上反斜杠。

广义地讲，C 语言字符集中的任何一个字符均可用转义字符来表示。表中的 \ddd 和 \xhh 正是为此而提出的。ddd 和 hh 分别为八进制和十六进制的 ASCII 代码。如'\101' 表示字母'A'，'\102' 表示字母'B'，'\134' 表示反斜线，'\x0A'表示换行等。

【例 2－1】转义字符的使用。

```
#include <stdio.h>
void main()
{
   printf("ab c\tde\rf\n");
   printf("hijk\tL\bM\n");
}
```

程序运行结果为：

```
f ab  c  de
hijk      M
```

4. 字符串常量

字符串常量是由一对双引号括起的字符序列。例如："CHINA"，"C program"，"14.5"等都是合法的字符串常量。

字符串常量和字符常量是不同的量。它们之间主要有以下区别：

(1)字符常量由单引号括起来，字符串常量由双引号括起来。

(2)字符常量只能是单个字符，字符串常量则可以含一个或多个字符。

(3)可以把一个字符常量赋予一个字符变量，但不能把一个字符串常量赋予一个字符

变量。在C语言中没有相应的字符串变量，可以用一个字符数组来存放一个字符串常量。

(4)字符常量占一个字节的内存空间。字符串常量占的内存字节数等于字符串中字节数加1。增加的一个字节中存放字符'\0'(ASCII码为0)，这是字符串结束的标志。

如图2-2所示，字符串"C program"在内存中所占的字节为10。

C		p	r	o	g	r	a	m	\0

图2-2　字符串常量"C program"在内存的存储形式

字符常量'a'和字符串常量"a"虽然都只有一个字符，但在内存中的情况是不同的，如图2-3所示。

'a'在内存中占一个字节，可表示为：

a

"a"在内存中占二个字节，可表示为：

a	\0

图2-3　字符常量'a'和字符串常量"a"在内存的存储形式

5. 符号常量

在C语言中，可以用一个标识符来表示一个常量，称之为符号常量。符号常量在使用之前必须先定义，其一般形式为：

#define 标识符 常量

其中#define命令把该标识符定义为其后的常量值。一经定义，其值不能被改变，以后在程序中所有出现该标识符的地方均代之以该常量值。

习惯上符号常量的标识符用大写字母，变量标识符用小写字母，以示区别。

【例2-2】符号常量的使用示例。

```
#define   PI   3.1415926
#define   R   3.0
#include <stdio.h>
void main()
{
   floatc, s;
   c = 2* PI* R;
   s = PI* R* R;
   printf("circumference = %f\n", c);
   printf("area = %f\n", s);
}
```

分析：命令行#define PI 3.1415926的作用是在预编译时将程序中凡出现PI的地方全

部用3.1415926代替。符号常量的优点是使程序容易理解，可读性好且容易维护。

2.3.2　变量

1. 变量的概念

变量是指C语言程序中合法的标识符，是用来存取某种类型值的存储单元，其中存储的值可以在程序执行的过程中被改变。

在C语言中用到的变量必须先定义后使用。对变量的定义就是给变量分配相应类型的存储空间。

定义变量的一般形式为：

＜变量类型说明符＞＜变量列表＞[= ＜初值＞];

其中：

(1)变量类型说明符确定了变量的取值范围以及对变量所能进行的操作规范。

(2)变量列表由一个或多个变量名组成。当要定义多个变量时，各变量之间用逗号分隔。

(3)初值是可选项，变量可以在定义的同时赋初值，也可以先定义，在后续程序中赋初值。

2. 变量的定义

C语言中，必须对所有的变量“先定义，后使用”。如“int a;”表示定义了一个变量，其变量名为a，变量名代表内存中的存储单元，在对程序进行编译连接时，由系统给每个变量分配存储单元。

变量还可以在定义的同时赋初值。如“int a=3;”表示定义了一个变量a，同时给a所对应的地址单元赋初值3，如图2-4所示。请注意区分变量名和变量值，这是两个不同的概念，变量名对应变量的存储地址，而变量值是代表这个地址单元的内容。如果在定义的同时不赋初值，对于自动变量而言，其初值是不确定的。例如：

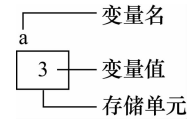


图2-4　变量名及存储单元

```
int, b, c;                  //表示定义了a, b, c 3个整型变量,其初值是不确定的
int a=1, b=2, c=3;  //表示定义了a, b, c 3个整型变量并分别赋初值1, 2, 3
float a, b=3.5;         //表示定义了a, b2个单精度型变量,其中只有b赋初值3.5
char x, y='A';          //表示定义了x, y2个字符型变量,其中只有变量y赋初值'A'
```

C语言对变量强制定义的目的是：

(1)因为只有在定义了变量的类型后，系统才知道如何给变量分配存储空间。如指定变量a为整型，在编译时就能为其分配相应的4个字节的存储空间，并按整数方式存储数据。

(2)指定一个变量属于一个特定的类型，在编译时，能根据该类型进行运算是否合法的检查。例如：

```
float a=4.5, b=8.9;
```

```
int c;
c = a% b;                    //错误
```

求余运算要求两个操作数都是整数，运算结果也要求是整数，而现在 a，b 均为实数，在编译时，系统会给出有关的出错信息。

3. 变量的分类

如果把变量与数据类型结合起来分类，变量可分为整型变量、浮点变量、字符变量。

(1)整型变量

系统根据声明变量时所指定的数据类型为变量分配存储空间，使用整型变量时要注意不能超过变量的取值范围。

如果定义了一个短整型变量 i：

```
shorti；i = 10；
```

则 i 在内存中的二进制表示如下：

0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0

C 语言中的数值在计算机中是以补码形式表示的：

①正数的补码和原码相同；

②负数的补码：将该数的绝对值的二进制形式按位取反再加 1。

【例 2-3】整型数据的溢出。

```
#include < stdio. h >
void main()
{
   shorta = 32767, b;
   b = a + 1;
   printf("a = % d, b = % d \n", a, b);
}
```

程序运行后 a 的值为 32767，内存中的表示如下：

0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

b 的值则为 -32768，内存中的表示如下：

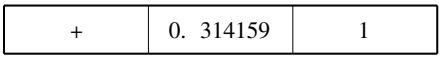
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

由于短整型变量所能表示的数的范围是 -32768 ~ 32767，无法表示大于 32767 的数，此例中得到 -32768 的错误结果，这种情况叫溢出，但程序在运行过程中不会给出出错信息。需要编程者选择恰当的数据类型来保证结果正确。

(2)实型变量

实型变量分为个单精度型和双精度型变量，两者之间的区别在于后者的有效位数比前

者的多、精度更高。实型数据一般占 4 个字节(32 位)内存空间，按指数形式存储，分为小数部分(尾数)和指数部分(阶码)。实数 3.14159 在内存中的存放形式如图 2 -5 所示。



数符小数部分指数部分

图 2 -5　实型数据在内存中的存放形式

因此，使用实型变量时，可能会有误差。实型变量的有效位数越多，与实际值就会越接近，精度就越高。

【例 2 -4】对比 float 型数据和 double 型数据。

```
#include < stdio. h >
void main()
{
   float   x = 5555. 55555,
   doubley = 5555. 555555555555;
   printf("x = %f  \n",x);
   printf("y = %lf  \n",y);
}
```

程序运行结果为：

```
x = 5555. 555664
y = 5555. 555556
```

分析：由于 x 是单精度型，有效位数只有 7 位，整数位已占 4 位，因此小数部分只有 3 位是有效的，3 位以后的其它数字为无效数字。z 是双精度型，有效位数为 16 位，但是 VC + +6. 0 规定小数后最多保留 6 位数字，其余作四舍五入处理。

【例 2 -5】实型数据的舍入误差。

```
#include < stdio. h >
void main()
{
   float   a = 123456789. 0,b;
   b = a + 1;
   printf("a = %f\n", a);
   printf("b = %f\n", b);
}
```

程序运行结果为：

```
a = 123456792. 000000
b = 123456792. 000000
```

分析：因为 float 类型只能保证 7 位有效数字，给 a 赋值时只能保证前 7 位是准确的，后面几位是无效数字，把 1 加在无效数字上，是不会发生任何变化。

(3)字符型变量

字符变量用来存储字符常量，即单个字符。

字符型变量的类型说明符为 char，如 char a = 'b'，在机器中占 8 位，其范围为 0 ~ 255。

将一个字符常量放到一个字符变量中，实际上不是把该字符本身放到内存单元中去，而是将该字符的相应的 ASCII 代码放到存储单元中。因此，字符型数据和整型数据之间可以通用。一个字符数据既可以以字符形式输出，也可以以整数形式输出。例如：

char a = 'A'; 与 char a = 65; 是等价的。因为字符'A'在计算机中的形式是整型 65。

【例 2 - 6】大写字母转换为小写字母。

```
#include < stdio. h >
void main()
{
   char a = 'X', b = 'Y';
   a = a + 32;
   b = b + 32;
   printf("% c, % c \n", a, b);
}
```

程序运行结果为：

x, y

分析：大写字母的 ASCII 码值和小写字母的 ASCII 码值相差 32，大写字母加 32 就是小写字母，将其以字符形式输出即可得到小写字母。

2.4 运算符与表达式

数据是需要加工的，如对数据的加减乘除运算，大小比较等等，这些都是编写程序必需的，否则程序就没有意义了。为解决这个问题，C 语言提供了丰富的运算符，使得 C 的运算十分灵活方便。C 语言提供了以下运算符：

(1)算术运算符：用于各类数值运算。包括加(+)、减(-)、乘(*)、除(/)、求余(或称模运算,%)、自增(+ +)、自减(- -)共七种。

(2)关系运算符：用于比较运算。包括大于(>)、小于(<)、等于(= =)、大于等于(> =)、小于等于(< =)、不等于(! =)六种。

(3)逻辑运算符：用于逻辑运算。包括与(&&)、或(||)、非(!)三种。

(4)位操作运算符：参与运算的量，按二进制位进行运算。包括位与(&)、位或(|)、位非(~)、位异或(^)、左移(< <)、右移(> >)六种。

(5)赋值运算符：用于赋值运算，分为简单赋值(=)、复合算术赋值(+ =， - =，* =，/ =,% =)和复合位运算赋值(& =， | =，^ =， > > =， < < =)三类共十一种。

(6)条件运算符：这是一个三目运算符，用于条件求值(?:)。

(7)逗号运算符：用于把若干表达式组合成一个表达式(,)。

(8)指针运算符：用于取内容(*)和取地址(&)二种运算。

(9)求字节数运算符：用于计算数据类型所占的字节数(sizeof)。

(10)特殊运算符：有括号()，下标[]，成员(→，.)等几种。

在本章中主要介绍算术运算符与算术表达式，关系运算符和关系表达式，逻辑运算符与逻辑表达式，赋值运算符与赋值表达式，条件运算符与条件表达式，逗号运算符与逗号表达式，其他运算符将在以后各章中陆续介绍。

2.4.1　算术运算符与算术表达式

1. 基本的算术运算符

(1)加法运算符“+”：加法运算符为双目运算符，即应有两个量参与加法运算。如 a+b，5+8 等。具有右结合性。

(2)减法运算符“-”：减法运算符为双目运算符。但“-”也可作负值运算符，此时为单目运算，如 -x，-7 等具有左结合性。

(3)乘法运算符“*”：双目运算，具有左结合性。

(4)除法运算符“/”：双目运算具有左结合性。参与运算量均为整型时，结果也为整型，舍去小数。如果运算量中有一个是实型，则结果为双精度实型。

(5)求余运算符(模运算符)“%”：双目运算，具有左结合性。要求参与运算的量均为整型。求余运算的结果等于两数相除后的余数。

【例2-7】除法和求余运算符示例程序。

```
#include < stdio. h >
voidmain()
{
   printf("\n\n%d, %d\n", 20/7, - 20/7);
   printf("%f, %f\n", 20. 0/7, - 20. 0/7);
   printf("%d\n", 100%3);
}
```

分析：20/7，-20/7 的结果均为整型，小数全部舍去。而 20.0/7 和 -20.0/7 由于有实数参与运算，因此结果也为实型。100 除以 3 所得的余数 1。

2. 算术表达式和运算符的优先级和结合性

表达式是由常量、变量、函数和运算符组合起来的式子。一个表达式有一个值及其类型，它们等于计算表达式所得结果的值和类型。表达式求值按运算符的优先级和结合性规定的顺序进行。单个的常量、变量、函数可以看作是表达式的特例。

算术表达式是由算术运算符和括号连接起来的式子。

(1)算术表达式：用算术运算符和括号将运算对象(也称操作数)连接起来的，符合 C 语法规则的式子。例如，a*b+c/d，(x+r)*8-(a+b)/7，c*sin(x)+d*cos(x)都是算术表达式。

(2)运算符的优先级：C 语言中，运算符的运算优先级共分为 15 级。1 级最高，15 级

最低。在表达式中，优先级较高的先于优先级较低的进行运算。而在一个运算量两侧的运算符优先级相同时，则按运算符的结合性所规定的结合方向处理。

算术运算符的优先级是：++，--优先级最高，然后是*，/,%，最后是+，-。

(3)运算符的结合性：C语言中各运算符的结合性分为两种，即左结合性(自左至右)和右结合性(自右至左)。例如算术运算符的结合性是自左至右，即先左后右。如有表达式x-y+z则y应先与"-"号结合，执行x-y运算，然后再执行+z的运算。这种自左至右的结合方向就称为"左结合性"。而自右至左的结合方向称为"右结合性"。C语言运算符中，除了单目运算符、赋值运算符和条件运算符是右结合性外，其它运算符都是左结合性。如x=y=z，由于"="的右结合性，应先执行y=z再执行x=(y=z)运算。

3. 自增、自减运算符

单目算术运算符"++""--"的前缀与后缀方式，对操作数本身的值的影响是相同的，但其对表达式的值的影响是不同的。前缀方式是先将操作数加(或减)1，再将操作数的值作为算术表达式的值；后缀方式是先将操作数的值作为算术表达式的值，再将其加(或减)1。例如：

a的值为4；

b=++a为前缀方式，首先将a的值加1，再得到b的值为5，结果为b=5，a=5；

b=a++为后缀方式，首先得到b的值为4，再将a的值加1，结果为b=4，a=5。

【例2-8】自增自减综合应用。

```
#include <stdio.h>
void main()
{
   int i=5;
   printf("%d\n", ++i);
   printf("%d\n", --i);
   printf("%d\n", i++);
   printf("%d\n", i--);
   printf("%d\n", -i++);
   printf("%d\n", -i--);
}
```

分析：i的初值为5，第2行i加1后输出故为6；第3行减1后输出故为5；第4行输出i为5之后再加1(为6)；第5行输出i为6之后再减1(为5)；第6行输出-5之后再加1(为6)，第7行输出-6之后再减1(为5)。

【例2-9】自增运算的应用。

```
#include <stdio.h>
void main()
{
   inti=5, j=5, a, b;
```

```
    a = (i + +) + (i + +) + (i + +);
    b = ( + +j) + ( + +j) + ( + +j);
    printf("%d, %d, %d, %d \n", a, b, i, j);
}
```

分析：对 a = (i + +) + (i + +) + (i + +)应理解为三个 i 相加，故 a 值为 15。然后 i 再自增 1 三次，相当于加 3，故 i 的最后值为 8。而对于 b 的值则不然，b = (+ +j) + (+ +j) + (+ +j)应理解为：由于加法左结合性质，所以先计算(+ +j) + (+ +j)，j 两次自增 1 得到 7，再参与运算，计算出结果 14；再与第三个(+ +j)求和，则 j 先自增 1 为 8，再与 14 相加的和为 22，j 的最后值仍为 8。

注意：自增运算符和自减运算符的运算对象只能是变量，不能是常量或表达式。形式 3 + +或 + +(i+j)都是非法的表达式，因为 3 是常量，常量的值是不能变的； + +(i+j)也是不能实现的，因为假设 i+j 的值为 10，那么自增后得到的 11 放在什么地方呢？无变量单元可供存放。

2.4.2　关系运算符与关系表达式

1. 关系运算符

关系运算符用来对两个操作数进行比较。运算过程为：如果关系表达式成立，结果为真(true)，否则为假(false)。由于 C 语言没有逻辑型数据，就用 1 代表“真”，0 代表“假”。表 2-4 列出了 C 语言的关系运算符。

表 2-4　关系运算符

运算符	名称	示例	功能
<	小于	a<b	a 小于 b 时返回真；否则返回假
<=	小于等于	a<=b	a 小于等于 b 时返回真；否则返回假
>	大于	a>b	a 大于 b 时返回真；否则返回假
>=	大于等于	a>=b	a 大于等于 b 时返回真；否则返回假
==	等于	a==b	a 等于 b 时返回真；否则返回假
!=	不等于	a!=b	a 不等于 b 时返回真；否则返回假

2. 优先级

(1)“<”、“<=”、“>”和“>=”为同一级，“==”和“!=”为同一级。前者优先级高于后者。

(2)关系运算符优先级低于算术运算符，高于赋值运算符和逗号运算符。

3. 关系表达式

关系表达式就是用关系运算符将两个表达式连接起来的式子。

关系表达式的一般形式为：

表达式　关系运算符　表达式

关系表达式的值是一个逻辑值，即“真”或“假”，分别用“1”和“0”表示。当关系表达式两边的类型不致时，例如一边是整型，一边是双精度型，则系统自动将整型转换为双精度型，然后进行比较。

例如：a+b>c-d,, x>3/2，'a'+1<c，-i-5*j==k+1；都是合法的关系表达式。

由于表达式也可以又是关系表达式，因此也允许出现嵌套。

例如：a>(b>c)，a!=(c==d)等。

【例2-10】写出下列程序的执行结果。

```
#include <stdio.h>
void main()
{
   int a=3, b=4, c=8;
   char d='k';
   printf("%d, %d\n", 'a'+2<d, -a-2*b>=c+1);
   printf("%d, %d\n", 1<b, c-a!=b);
   printf("%d, %d\n", a+b+c==2*a, c=a+b);
}
```

分析：本题要注意字符变量在表达式中的运算，在运算前先将字符转换成该字符对应的ASCII码值，如'a'+2<d，因为'a'的ASCII码值为97，'k'的ASC码值为107，上述表达式等价于：97+2<107，即99<107，结果为真，输出十进制整型数据为1。表达式-a-2*b>=c+1等价于：-3-2*4>=8+1，即-11>9，结果为假，输出十进制整型数据为0。表达式1<b等价于：1<4，结果为真，输出十进制整型数据为1。表达式c-a!=b等价于：8-3!=4，即5!=4，结果为真，输出十进制整型数据为1。表达式a+b+c==2*a等价于：3+4+8==2*3，即15==6，结果为假，输出十进制整型数据为0。表达式c=a+b是一个赋值表达式，等价于：c=3+4，即c=7，将7赋给变量c，再输出变量c的值，因此输出结果为：7。

2.4.3 逻辑运算符与逻辑表达式

1. 逻辑运算符

逻辑运算符应用于逻辑运算，其运算符有：与(&&)、或(||)、非(!)，其中&&和||是双目运算符,!是单目运算符。逻辑运算的结果只有两个：1(逻辑真)，0(逻辑假)。

逻辑运算符的运算规则如表2-5所示。

表2-5 逻辑运算真值表

A	B	!A	A&&B	A||B
0(假)	0(假)	1(真)	0(假)	0(假)
0(假)	1(真)	1(真)	0(假)	1(真)
1(真)	0(假)	0(假)	0(假)	1(真)
1(真)	1(真)	0(假)	1(真)	1(真)

运算规则如下：

(1)逻辑非运算的规则是：真变假，假变真；当运算量为非0时，运算结果为0，当运算量为0时，运算结果为1；

(2)逻辑与运算的规则是：只有当两个运算量都为真是，结果才为真，否则为假；

(3)逻辑或运算的规则是：当两个运算量均为假时，结果才为假，否则为真。

说明：C语言系统对任何非0值都理解为逻辑真，0理解为逻辑假。如：3-1&&2，其等价逻辑表达式为：2&&2，即“真”&&“真”，逻辑结果为1(真)。

2. 优先级

(1)！的优先最高；

(2)&&和‖的优先级低于算数运算符和关系运算符，但是高于赋值运算符。

例如：按照运算符的优先顺序可以得出：

a>b && c>d	等价于	(a>b) && (c>d)
! b==c‖d<a	等价于	((! b)==c)‖(d<a)
a+b>c && x+y<b	等价于	((a+b)>c) && ((x+y)<b)

3. 逻辑表达式

将常量、变量、函数等通过逻辑运算符连接起来组成的表达式称为逻辑表达式，逻辑表达式的一般形式为：

表达式　逻辑运算符　表达式

逻辑表达式的值是式中各种逻辑运算的最后值，以“1”和“0”分别代表“真”和“假”。其中的表达式可以又是逻辑表达式，从而组成了逻辑表达式的嵌套。

【例2-11】用表达式描述下列条件。

(1)x是5的倍数　　//表达式为：x%5==0

(2)x是奇数　　//表达式为：x%2==1或x%2！=0

(3)'A'<=x<='Z'　　// 表达式为：x>='A' && x<='Z'

(4)x能同时被5和7整除　　//表达式为：x%5==0 && x%7==0

分析：(1)中x是5的倍数等价于可以被5整除，即除以5以后，余数为0，因此可用关系表达式x%5==0来描述；(2)中x是奇数等价于除以2的余数为1，也可以用关系表达式x%2==1或x%2！=0来描述；(3)中'A'<x<'Z'是一个数学表达式，相当于描述x是一个大写字母，即x大于或等于'A'，并且x小于或等于'Z'，用逻辑运算符连接起来构成的逻辑表达式为：x>='A'&& x<='Z'；(4)中x能同时被5和7整除，等价于x能被5整除，并且能被7整除，其逻辑表达式为：x%5==0 && x%7==0。

【例2-12】逻辑表达式的运用。

```
#include <stdio.h>
void main()
{
   int a=2, b=0, c=0;
   printf("%d", a&&b);
```

```
    printf("%d", a || b && c);
    printf("%d", ! a&& b);
    printf("%d", (a = 1) || (b = 3) + 10&&(c = 2));
}
```

运行结果为：

0　1　0　1

求解 C 语言逻辑表达式时，按从左到右的顺序计算运算符两侧的操作数，一旦得到表达式的结果，就停止计算。

(1)求解：表达式 1&& 表达式 2 时，先计算表达式 1，若其值为 0，则表达式 1&& 表达式 2 值一定为 0。此时，没有必要计算表达式 2 的值。例 2－12 中，计算表达式！a&&b 时，先算！a，由于 a 的值是 2，！a 就是 0，该逻辑表达式的值一定是 0，不必再计算 b。

(2)求解：表达式 1 || 表达式 2 时，先计算表达式 1，若其值为非 0，则表达式 1 || 表达式 2 值一定为 1。此时，没有必要计算表达式 2 的值。例 2－12 中，计算表达式(a = 1) || (b = 3) +

10&&(c = 2)时，先计算表达式(a = 1)，该表达式的值是 1，也就是说表达式 1 为 1，对于 || 运算而言，不必再计算表达式 2，即(b = 3) +10&&(c = 2)。

也就是说，对于 && 运算符来说，只有表达式 1 的值不等于等 0，才继续进行右面的运算。对 || 运算符来说，只有表达式 1 的值为 0，才继续进行其右面的运算。因此，如下有下面的逻辑表达式：

(m = a > b)&&(n = c > d)

当 a = 1，b = 2，c = 3，d = 4，m 和 n 的原值为 1，由于 a > b 的值为 0，因此 m = 0，而“n = c > d”不被执行，因此 n 的值不是 0 而仍保持原值 1。

熟练掌握 C 语言的关系运算和逻辑运算符后，可以巧妙地用一个逻辑表达式来表示一个复杂的条件。

例如，要判别用 year 表示某一个是否这闰年。闰年的条件是符合下面二个条件之一：①能被 4 整除，但不能被 100 整除，如 2008；②能被 4 整除，又能被 400 整除，如 2000。可以用一个逻辑表达式来表示：

(year%4 = = 0&&year%100! = 0) || year%400 = = 0

如果上述表达式值为真(1)，则 year 为闰年，否则 year 为非闰年。

2.4.4　赋值运算符与赋值表达式

1. 赋值运算符

简单赋值运算符和表达式：简单赋值运算符记为“ = ”。由“ = ”连接的式子称为赋值表达式。其一般形式为：

变量 = 表达式

例如：x = a + b、w = sin(a) + sin(b)、y = i + + + － －j，都是合法的赋值表达式。

赋值表达式的功能是计算表达式的值再赋予左边的变量。赋值运算符具有右结合性。

因此，a=b=c=5，可理解为 a=(b=(c=5))。

在其它高级语言中，赋值构成了一个语句，称为赋值语句。而在 C 中，把“=”定义为运算符，从而组成赋值表达式。凡是表达式可以出现的地方均可出现赋值表达式。

例如：x=(a=5)+(b=8)是合法的。它的意义是把 5 赋予 a，8 赋予 b，再把 a，b 相加，和赋予 x，故 x 应等于 13。

2. 复合的赋值运算符

在赋值符“=”之前加上其它二目运算符可构成复合赋值符。如+=，-=，*=，/=，%=，<<=，>>=，&=，^=，|=。

构成复合赋值表达式的一般形式为：

变量双目运算符=表达式

它等效于：

变量=变量运算符表达式

例如：

a+=3 等价于 a=a+3

x*=y+8 等价于 x=x*(y+8)，而不是 x=x*y+8

x%=5 等价于 x=x%5

【例 2-13】 a=12；则表达式 a+=a-=a*a 的值是多少？

分析计算步骤：

①“a-=a*a”等价于“a=a-a*a”，计算可得 a 的值为 12-144=-132。

②“a+=-132”等价于“a=a+(-132)”，计算可得 a 的值为-132-132=-264。

因此，表达式 a+=a-=a*a 的值为 264。

【例 2-14】 int a=2；a%=4-1；则表达式 a+=a*=a-=a*=3 的值是多少？

分析计算步骤：

①“a%=4-1;”等价于“a=a%3”，计算可得 a 的值为 2。

②“a*=3”等价于“a=a*3”，计算可得 a 的值为 6。

③“a-=6”等价于“a=a-6”，计算可得 a 的值为 0。

④“a*=0”的运算中，a 的值为 0。

⑤再进行“a+=0”的运算中，a 的值为 0。

因此，表达式 a+=a*=a-=a*=3 的值为 0。

复合赋值符这种写法，对初学者可能不习惯，但十分有利于编译处理，能提高编译效率并产生质量较高的目标代码。

2.4.5　条件运算符与条件表达式

条件运算符由“?”和“:”两个字符组成，用来连接 3 个运算对象，是 C 语言中唯一的三目运算符。

用条件运算符将运算对象连接成的式子称为条件表达式。其中运算对象可以是算术表达式、关系表达式、逻辑表达式、赋值表达式和条件表达式。条件表达式的一般形

式为：

＜表达式1＞？＜表达式2＞：＜表达式3＞

其中表达式1是一个关系表达式或逻辑表达式。

条件运算符的执行过程：先求解表达式1的值，若表达式1的值为真，则求解表达式2的值，且作为整个条件表达式的结果；若表达式1的值为假，则求解表达式3的值，且作为整个条件表达式的结果。下面的赋值表达式：

max = (a > b)? a: b

执行结果就是将条件表达式的值赋给max，也就是将a和b二者中大者赋给max。

条件运算符的优先级较低，只比赋值运算符高。它的结合方向是自右向左。例如：

(1)(a > b)? a: b + 1 等价于：a > b? a:(b + 1)

(2)a > b? a: c > d? c: d 等价于：a > b? a:(c > d? c: d)

【例2-15】 a = 5，b = 9，c = 6，则 max = a > b? a: b > c? b: c 为多少？

分析：表达式 a > b? a: b > c? b: c 等价于 a > b? a(b > c? b: c)，由于 a > b 的值为假，所以求整个表达式的值只需求 b > c? b: c 的值；而 b > c 的值为真，所以 b > c? b: c 的值为b，因此max的值为9。

【例2-16】 输入两个数，求其中较大的数和较小的数。

```
#include < stdio. h >
void main()
{
   int x, y, max, min;
   scanf("%d%d", &x, &y);
   min = x < y? x:y;
   max = x > y? x:y;
   printf("max = %d, min = %d \n", max, min);
}
```

如果程序运行时输入的两个数分别为15和37，则程序执行结果为：

max = 37，min = 15

善于利用条件表达式，可以使程序写得精练、专业。

2.4.6 逗号运算符与逗号表达式

在C语言中逗号“,”也是一种运算符，称为逗号运算符。用它将两个式子连接起来，称为逗号表达式。逗号表达式的一般形式为：

表达式1，表达式2

逗号表达式的求解过程是：先求解表达式1，再求解表达式2。整个逗号表达式的值是表达式2的值。

(1)逗号表达式的嵌套。一个逗号表达式又可以与另一个逗号表达式组成一个新的逗号表达式，例如：(a = 3 * 5，a * 4)，a + 5

(2)逗号表达式的可以扩展为：

表达式1，表达式2，表达式3，…，表达式n

它的值为表达式n的值。

逗号运算符的优先级最低，结合方向为从左至右。

【例2-17】求下列表达式的值。

① x=(y=6，y*3)

② x=y=6，y*3

③(a=3*5，a*4)，a+5

分析：①首先将6赋给y，然后执行y*3得18，将整个结果18赋给x，所以整个表达式的值为18(此时，x=18，y=6)。②首先将6赋给y和x，然后执行y*3得18，所以整个表达式的值为18(此时，x=6，y=6)。因此，这两个表达式的作用是不同的，第①个是一个赋值表达式，将一个逗号表达式的值赋给x，第②个是逗号表达式，包括一个赋值表达式和算术表达式。③先计算出a的值为3*5，等于15，再计算a*4得60(但a值仍为15)，再进行a+5得20，即整个表达式的值为20。

2.5　数据的类型转换

C语言中，各种类型数据可以进行混合运算，例如200+'A'+7.65-9876.55*'b'，这时需要先转换成统一类型，再进行运算。C语言提供了3种类型转换方式：自动转换、赋值转换和强制转换。其中自动转换和赋值转换又称为隐式转换，而强制转换又称为显示转换。

2.5.1　自动类型转换

整型、浮点型、字符型数据可以进行混合运算。运算中，不同类型的数据先转化为同一类型，然后进行运算。为了保证精度，转换从低级到高级。各类型从低级到高级的顺序为：char→int→long→float→double。

自动转换由编译系统自动完成。自动转换遵循以下规则：

(1)若参与运算量的类型不同，则先转换成同一类型，然后进行运算。

(2)转换按数据长度增加的方向进行，防止计算过程中数据被截断，以保证精度不降低。如int型和long型运算时，先把int量转成long型后再进行运算。

(3)所有的浮点运算都是以双精度进行的，即使仅含float单精度量运算的表达式，也要先转换成double型，再作运算。

(4)char型和short型参与运算时，必须先转换成int型。

图2-6表示了类型自动转换的规则。其中水平方向上的转换是必须进行的，即所有的float类型数据先转换成double类型，所有的char或short类型数据先转换成int类型，然后进行运算。水平方向转换后，如果仍有不同类型，则按竖直方向的箭头方向进行转换。

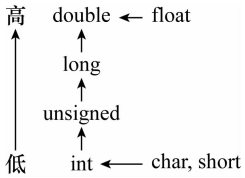


图 2-6　不同类型数据转换顺序

如 3 + 'a' - 10.0/2 * 5，首先将 char 类型的'a'转换成 int 类型数据 97，然后与 int 类型的 3 相加得到 100；接下来将 int 类型数据 2 转换成 double 类型，与 10.0 做除法运算得到 5.0，再将 int 类型数据 5 转换成 double 类型，然后同 5.0 相乘得到 25.0；最后将前面加法运算得到的 100 从 int 类型转换成 double 类型，再相减，最终的结果 75.0 是 double 类型的数值。

2.5.2　赋值转换

如果赋值运算符两边的数据类型不相同，系统将自动进行类型转换，即把赋值号右边的类型转换成左边的类型。具体规定如下：

(1)实型赋值给整型，舍去小数部分，从而丢失一部分数据，降低精度。如：

```
int a;
float b = 3.47;
a = b;
```

a 为整型，赋予实型量 b 值 3.47 后只取整数的值 3。

(2)整型赋值给实型，数值不变，但将以浮点形式存放，即增加小数部分(小数部分的值为 0)。

```
int a = 6;
float b;
b = a;
```

b 为实型，赋予整型量 a 值 6 后增加了小数部分，b 值为 6.000000。

(3)字符型赋予整型，由于字符型为一个字节，而整型为二个字节，故将字符的 ASCII 码值放到整型量的低八位中，高八位为 0。整型赋予字符型，只把低八位赋予字符量。

```
inta, b = 322;
char c1 = 'a', c2;
a = c1;
c2 = b;
```

字符型量 c1 赋予 a 变为整型，a 的值为 97。整型量 b 取其低八位赋予 c2 后成为字符型(b 的低八位为 01000010，即十进制 66，按 ASCII 码对应于字符'B')，所以 c2 的值为'B'。

2.5.3　强制类型转换

强制类型转换是通过类型转换运算来实现的，其功能是把表达式的运算结果强制转换

成类型说明符所表示的类型。其一般形式为：

(类型说明符)(表达式)

例如：(float) a 是把 a 转换为实型，(int)(x + y) 是把 x + y 的结果转换为整型。

说明：

①类型说明符和表达式都必须加括号(单个变量可以不加括号)，(int)(x + y)与(int) x + y 两个式子强制转换的对象是不同的。如把(int)(x + y)写成(int)x + y，则成了把 x 转换成 int 型之后再与 y 相加了。

②无论是强制转换或是自动转换，都只是为了本次运算的需要而对变量的数据长度进行的临时性转换，而不改变数据说明时对该变量定义的类型。即强制转换得到的是所需类型的中间变量，原变量类型不变

③较高类型向较低类型转换时可能发生精度损失问题。

【例 2 -18】 强制转换示例

```
#include <stdio.h>
void main()
{
   int a = 7, b = 2;
   float y1, y2;
   y1 = a/b;
   y2 = (float)a/b;
   printf("y1 = %f, y2 = %f", y1, y2);
}
```

程序的运行结果为：

```
y1 = 3.000000, y2 = 3.500000
```

2.6　运算符与表达式综合应用举例

前面学习了 6 种类型运算符和表达式，学习运算符和表达式的目的重在应用，能够根据实际问题的需求写出对应的表达式，或者能够正确的将数学表达式转换为 C 表达式，这是学习 C 语言编程的第一步。

【例 2 -19】 判断所输入字符是否为英文字母(定义变量 char ch 表示所输入的字符)。

分析：

(1)ch 为大写字母的条件是 ch >= 'A'&&ch <= 'Z';

(2)ch 为小写字母的条件是 ch >= 'a'&&ch <= 'z';

(3)显然，ch 满足条件(1)或条件(2)都符合本题的要求，因此，完整的表达式为：

```
(ch >= 'A'&&ch <= 'Z') || (ch >= 'a'&&ch <= 'z')
```

【例 2 -20】 求 3 个数中最大的数。

```
#include <stdio.h>
```

```
int main()
{
   int a = 2, b = 200, c = - 8;
   int max ;                    //max 用来存放 3 个数中最大的数
   max = (a > b)? a:b;          //条件表达式,求 a, b 中较大的数,并存入变量 max 中
   max = (max > c)? max:c;      //条件表达式,求 max, c 中较大的数,并存入变量 max 中
   printf ("max = %d \n", max);
}
```

运行结果为：

```
max = 200
```

说明：本程序中两次用了条件表达式，这样使程序显得更专业、简练。

【例 2 -21】将下列数学式转换成 C 语言的表达式。

(1)华氏温度转换为摄氏温度的公式：c = 5 (F - 32)/9；

(2)判断一个方程是否有实根的条件是：b2 - 4ac≥0；

(3)3≤a≤9 且 a≠6。

转换后的 C 语言表达式分别是：

```
(1)c = (5.0/9.0) * (f - 32)      //注意 5 和 9 要用实数表示，否则 5/9 值为 0
(2)(b * b - 4 * a * c) > = 0
(3)(a > = 3&&a < = 9)&&a! = 6
```

【例 2 -21】设 a = 1，b = 2，c = 3，d = 4，写出下列逻辑表达式的值。

```
(1)a + b = = c&&a + b < c || ! a! = b
(2)! (a > b) || ! a < b&&(a + b > c - d) || ! a + c < d - a
(3)(! (a + b) + c - 1)&&(b + c/2)
(4)( - - a < 0)&&! (b - - < = 1) || (c - -> 1)
```

分析：

(1)在语句 a + b = = c&&a + b < c 中，a + b 的值为 3，得 a + b = = c 的值为 1、a + b < c 的值为 0，所以 a + b = = c&&a + b < c 的值为 0。在语句! a! = b 中,! a 的值为 0，得! a! = b 的值为 1，所以整个表达式的值为 1。

(2)在语句! (a > b) || ! a < b 中，a > b 的值为 1,! (a > b)的值为 0,! a 的值为 0,! a < b 的值为 1，得! (a > b) || ! a < b 的值为 1；而 a + b > c - d 的值是 1，所以! (a > b) || ! a < b&&(a + b > c - d)的值为 1，其后的 || ! a + c < d - a 就不再执行了，因此，整个表达式的值为 1。

(3)在语句! (a + b) + c - 1 中，a + b 非零,! (a + b)为 0，故! (a + b) + c - 1 的值为 1；算术表达式 b + c/2 的值非零，即为 1；因此，整个表达式的值为 1。

(4)在语句(- - a < 0)&&! (b - - < = 1)中， - - a 的值为 0(先使 a 减 1，再使用 a)，得 - - a < 0 的值为 0，由于其后是 && 运算，得(- - a < 0)&&! (b - - < = 1)的值为 0；而 c - - > 1 的值为 1，因此，整个表达式的值为 1。

本章小结

本章介绍了C语言数据类型、常量与变量的概念、运算符与表达式和数据类型转换。

(1)C语言的数据类型十分丰富，可分为基本数据类型、构造类型、指针类型和空类型，不同类型数据的操作方式和取值范围不同，所占存储空间的大小也不同。

(2)在程序中，数据的表现形式有常量和变量。数据值会发生变化的对象称之为变量，变量要先声明后使用；数值始终保持不变的数据对象称为常量。常量有数值常量和符号常量两种形式，数值常量分为整数、长整数、无符号数、浮点数(实数)、字符、字符串和转义字符；符号常量只是一个字符串，用来代替一个已知的常量。

(3)注意区分变量类型与变量的概念：变量的类型是C语言的一种基本数据类型，它是抽象的，不占具体的存储空间；而变量是C语言中的实体，它占据一定的存储空间，它一定属于一种变量类型。

(4)C语言中具有丰富的运算符和表达式，实现对各种数据的处理。运算符包括算术运算符、赋值运算符、关系运算符、逻辑运算符、条件运算符、逗号运算符等。使用运算符要**注意**：运算符的功能、与操作数(运算对象)的关系、操作数的数据类型、运算符优先级和结合方向等。

一般而言，单目运算符优先级较高，赋值运算符优先级低。算术运算符优先级较高，关系和逻辑运算符优先级较低。多数运算符具有左结合性，单目运算符、三目运算符、赋值运算符具有右结合性。

(5)混合运算时，不同类型数据要按规则转换成同一数据类型，再运算。类型转换方式有三种：自动转换、赋值转换和强制转换。自动转换由系统自动实现转换，由少字节类型向多字节类型转换。强制转换由强制转换运算符完成转换。

(6)ANSI C标准没有具体规定各类数据在内存中所占的字节数，由各编译系统自行决定。常见的有两种：Turbo C 2.0，Visual C++ 6.0。

①Turbo C 2.0中，int：2字节，short：2字节，long：4字节，字符型：1字节，float型：4字节，double型：8字节。

②Visual C++ 6.0中，int：4字节，short：2字节，long：4字节，字符型：1字节，float型：4字节，double型：8字节。

本书以Visual C++ 6.0编译系统为标准，所有例题均在Visual C++ 6.0编译系统中运行通过。

习　题

一、选择题

1. 在C语言中，要求参加运算的数必须是整数的运算符是(　　)。

A) %　　B) /　　C) !　　D) *

2. 在以下运算符中，优先级最高的运算符是(　　)。

A) < =　　B) /　　C) ! =　　D)&&

3. 设 x，y 为均为 float 型变量，则以下不合法赋值语句(　　)。

A) + + x　　B) y = (x%2)/10　　C) x * = y + 8　　D) x = y = 0

4. 假设所有变量均为整型，则表达式(a = 2，b = 5，a + +，b + +，a + b)的值为(　　)。

A) 7　　B) 8　　C) 9　　D) 10

5. 若变量 c 为 char 型，能正确判断出 c 为小写字母的表达式是(　　)。

A) 'a' < = c < = 'z'　　B) (c > = 'a') || (c < = 'z')

C) ('a' < = c) and ('z' > = c)　　D) (c > = 'a')&&(c < = 'z')

6. 若有定义：int x = 3，y = 2；float a = 2.5，b = 3.5；则 (x + y)%2 + (int)a/(int)b 的值为(　　)。

A) 1.0　　B) 1　　C) 2.0　　D) 2

7. 已知 inti；float f;，正确的语句是(　　)。

A) (int f)%i;　　B) int(f)%i;　　C) int(f%i);　　D) (int)f%i;

8. 已知 intj，i = 1；执行语句 j = -i + +；后，j 的值是(　　)。

A) 1　　B) 2　　C) -1　　D) -2

9. 若 a，b，c，d 都是 int 型变量且初值为 10，不正确的赋值语句是(　　)。

A) a = b = c = d　　B) a = b + +

C) a + b + +　　D) d = (a = b = 125) - C + +

10. 若 a，b，c，d 都是 float 型变量且初值为 10.5，不正确的赋值语句是(　　)。

A) a = b = 15　　B) d = int(a + c)

C) a = (int)(b + 1)　　D) + + a

11. 已知 char a；intb；float c；double d；则表达式 a * b + c - d 结果为(　　)型。

A) double　　B) int　　C) float　　D) char

12. 以下程序的输出结果是(　　)。

```
#include < stdio. h >
void main()
{
   int y = 3, x = 3, z = 1;
printf("%d%d\n", ( + + x, y + + ), z + 2);
}
```

A) 3 4　　B) 4 2　　C) 4 3　　D) 3 3

13. 以下程序的输出结果是(　　)。

```
#include  < stdio. h >
void main()
{
   int a = 5, b = 4, c = 6, d;
```

```
printf("%d\n", d = a > b? (a > c? a:c):(b));
}
```

A）5　　B）4　　C）6　　D）不确定

14. 执行下列程序中的输出语句后,a 的值是(　　)。

```
#include < stdio. h >
void main()
{
  int a;
  printf("%d\n", (a = 3 * 5, a * 4, a + 5));
}
```

A）65　　B）20　　C）15　　D）10

15. 以下程序的输出结果是(　　)。

```
#include < stdio. h >
void main()
{
  int a = - 1, b = 4,K;
  k = ( + + a < 0)&&! (b - - < 0);
  printf("%d, %d, %d\n",K, a, b);
}
```

A）0，0，3　　B）0，1，4　　C）0，－1，4　　D）0，0，4

二、填空题

1. 下面不合法的整型常数是_____。

160，－01，0668，011，0x，01a，－0xfffa，3.4E2

2. 下面不合法的浮点数是_____。

160.，e3，123，2e4.5，.e5，1e3

3. 下面合法的字符常量是_____。

"c"，'\\'，'W'，'\011'，'\xab'

4. 若 a 是 int 型变量，则执行表达式 a=25/3%3 后 a 的值为_____。

5. 若 x 和 n 均是 int 型变量，且 x 和 n 的初值均为 5，则执行表达式 x+=n++后 x 的值为_____，n 的值为_____。

6. 若 a 是 int 型变量，则表达式(a=4*5，a*2)，a+6 的值为_____。

7. 若有定义 int m=5，y=2;，则执行表达式 y+=y-=m*=y 后 y 的值是_____。

8. 若 x=6，y=4，z=2，则表达式!(x-y)+z-1&&y+z/2 的值是_____。

9. 表示条件 10<x<100 或 x<0 的 C 语言表达式为_____。

10、设，a，b，c 均为整型变量，“a 和 b 都小于 c”的 C 语言表达式为_____。

三、程序运算题

1. 设 a=1，b=2，c=3，d=4，写出下列表达式的值。

(1)！a&&b‖！c

(2)a‖b+c&&b-c

(3)a！=11&&b<4

(4)++a‖b++&&C++‖d++

(5)(a+b>c)&&a+b<c‖！a！=b

(6)！(a！=b)‖！a<b&&(a+b>c-d)‖！a+c<d-a

(7)(a+b+！c-1)&&(b+c/2)

(8)(++a<0)&&！(b--<=1)‖(C++>1)

2. 设 x=y=z=-1，则执行逻辑表达式++x‖++y&&++z 后，x，y，z 的值分别为多少?

3. 设 x=y=z=1，则执行逻辑表达式++x‖y--‖++z 后，x，y，z 的值分别为多少?

4. 若 a，b，c 均为 int 型变量，则执行以下语句后的 a，b，c 值分别为多少?

```
a=b=c=1;
++a‖++b&&++c;
```