21世纪普通高等教育精品教材

计算机系列

C语言程序设计教程

主　编　刘　亮　阚永彪

副主编　李小玲　王　凯

WUHAN UNIVERSITY PRESS

武汉大学出版社

图书在版编目(CIP)数据

C语言程序设计教程/刘亮，阚永彪主编．—武汉：武汉大学出版社，2013.9

21世纪普通高等教育精品教材

ISBN 978-7-307-11844-7

Ⅰ.C…　Ⅱ.①刘…　②阚…　Ⅲ.C语言—程序设计—高等学校—教材　Ⅳ.TP312

中国版本图书馆CIP数据核字(2013)第229826号

责任编辑：刘俊杰

出版发行：**武汉大学出版社**　(430072　武昌　珞珈山)

(电子邮件：cbs22@whu.edu.cn 网址：www.wdp.com.cn)

印刷：北京泽宇印刷有限公司

开本：787×1092　1/16　印张：16.5　字数：344千字

版次：2013年9月第1版　2013年9月第1次印刷

ISBN 978-7-307-11844-7　定价：35.00元

内 容 简 介

本书依据《国家中长期教育改革和发展规划纲要(2010—2020 年)》的指导精神,并结合教育部最新颁布的教学指导要求及普通高等院校教学特点编写而成。本书层次清晰、概念准确、内容丰富、图文并茂,注重实验和能力的培养。本书在介绍理论的同时突出应用,其目的是为了培养合格的网络管理员和网络工程师。通过系统地学习帮助读者尽快掌握相关知识点。

全书共分 11 章,内容包括:程序设计和 C 语言,数据类型、运算符与表达式,顺序结构程序设计,选择结构程序,循环结构程序设计,数组,函数,指针,编译预处理,结构体与共用体,以及文件。

本书可作为本科院校(含独立学院)的计算机专业课教材,也可供成人教育和高职高专院校使用,亦可作为广大青年朋友学习的参考用书。

前　言

本书在贯彻落实《国家中长期教育改革和发展规划纲要(2010—2020年)》的前提下,结合教育部《关于深化教育改革培养适应二十一世纪需要的高质量人才的意见》,根据普通高等院校教育人才培养目标及要求,编写了这本《C语言程序设计教程》。

计算机的本质是“程序的机器”,程序和指令的思想是计算机系统中最基本的概念。

程序设计是软件开发人员的基本功。只有懂得程序设计,才能进一步懂得计算机,真正了解计算机是怎样工作的。通过学习程序设计,进一步了解计算机的工作原理,更好地理解和应用计算机;掌握用计算机处理问题的方法;培养分析问题和解决问题的能力;最终具有编制程序的初步能力。即使将来不是计算机专业人员,由于学过程序设计,理解软件生产的特点和生产过程,就能与程序开发人员更好地沟通、合作,开展本领域中的计算机应用,开发与本领域有关的应用程序。

进行程序设计,必须以一种计算机语言作为工具,否则只是纸上谈兵。可供选择的语言很多,各有特点和应用领域。C语言功能丰富、表达能力强、使用灵活方便、应用面广、目标程序效率高、可移植性好,既具有高级语言的优点,又具有低级语言的许多特点,既适于编写系统软件,又能方便地用来编写应用软件。

要进行程序设计,要解决两个问题:

(1)要学习和掌握解决问题的思路和方法,即算法;

(2)学习怎样实现算法,即用计算机语言编写程序,达到用计算机解题的目的。

程序设计课程的核心内容:语法和算法。

怎样学习C程序设计,方法如下。

1.掌握基本要求,注意打好基础。在学校主要是学习程序设计的方法,进行程序设计的基本训练,为将来进一步学习和应用打下基础。不可能通过几十小时的学习,由一个门外汉变成编程高手,编写出大型而实用的程序。如果学时有限,有些较深的内容可以选学或自学,把精力放在最基本、最常用的内容上,打好基本功。

2.重视实践环节。只靠听课和看书是学不会程序设计的,学习本课程既要掌握概念,又必须动手编程,还要亲自上机调试运行来提高对知识的掌握程

度。因为考核方法是编写程序和调试程序。

3.要举一反三。学习程序设计,主要是掌握程序设计的思路和方法。学会使用一种计算机语言编程,在需要时改用另一种语言就不会太困难。不能设想今后一辈子只使用在学校里学过的某一种语言。但是无论用哪一种语言进行程序设计,其基本规律都是一样的。在学习时一定要活学活用,举一反三,掌握规律,在以后需要时能很快地掌握使用其他新的语言进行编程。

本书的编写理念:

本书的创新之处可用“系统阐述编程原理,详细讲解编程思路”两句话来概括。

编程原理阐述不系统、深入和透彻,很多知识点就要死记硬背,这样的学习方式必然枯燥乏味;同时,无法形成自己的知识地图,难以融会贯通。而例题的编程思路如果讲解不详细、不循序渐进、数量不够多,读者必然难以积累和养成自己的编程思路和技巧。这些恰恰是初学者学习编程时的关键因素。因此,本书在内容组织上希望尽量解决这两方面的问题。

需要注意的是,学习编程是一个理论与实践密切结合的过程。认真学习教材、消化书中所有例题是必不可少的环节。但这样做还是不够的,读者还必须认真完成一定数量的编程题(包括上机调试),因为没有一定量的编程实践训练是不可能真正学会编程的。

由于编者水平有限,不足和错漏之处在所难免,恳请广大读者批评指正。

编　者

目 录

第 1 章　程序设计和 C 语言

第 2 章　数据类型、运算符与表达式

第 3 章　顺序结构程序设计

第4章 选择结构程序

第5章 循环结构程序设计

第6章 数 组

第7章　函　数

第8章　指　针

第9章　编译预处理

第10章　结构体与共用体

第11章 文 件

参考文献

第1章　程序设计和C语言

本章导读

C语言是一种计算机程序设计语言。它既有高级语言的特点，又有汇编语言的特点。它可以作为系统设计语言，编写工作系统应用程序，也可以作为应用程序设计语言，编写不依赖计算机硬件的应用程序。因此，它的应用范围广泛。

1.1　程　序

程序(program)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指令序列的集合，指令能被计算机识别和执行，每一条指令会使计算机执行特定的操作。这个指令序列告诉计算机如何完成一个具体的任务。

程序可以看作为实现预期目的而进行操作的一系列语句和指令，一般分为系统程序和应用程序两大类。系统程序就是为了使计算机系统能实现各种功能。计算机软件设计人员根据需要设计好计算机指令的集合作为计算机软件系统的一部分提供给用户使用，由程序计数器(Program Counter)控制。应用程序是用户根据实际需要设计的，一些应用程序是用汇编语言、高级语言等开发语言编制出来的可以运行的文件，在计算机中称为可执行文件(后缀名一般为.exe)。例如学生成绩管理程序、财务管理程序等。

1.2　计算机语言

1.2.1　计算机语言的发展

计算机程序可以用多种语言编写，这些语言有的能够被计算机直接理解、执行，有的需要经过编译后才能被计算机理解、执行。在计算机的发展史上，曾经提出过上千种程序设计语言，今天常用的也有上百种。计算机语言经历了以下几个发展阶段。

1. 机器语言

机器语言能够被计算机直接理解、执行。而一台计算机能够直接理解、执行的仅仅是它自己的机器语言，不同的计算机拥有不同的机器语言。所以说，机器语言是与机器相关的。

机器语言是由计算机体系结构设计者定义，并据此设计与实现计算机硬件的。

计算机工作基于二进制，从根本上说，计算机只能识别和接受由 0 与 1 组成的指令，机器语言是一些由二进制数码 0 和 1 组成的数字串。

2. 汇编语言

显然，机器语言与人们习惯用的语言差别太大，难学、难写、难记、难检查、难修改、难以推广使用，所以早期的程序员尝试着用英语风格的缩写代替数字串来表示计算机的基本操作。这样，这些缩写词就构成了汇编语言的基础。相应地，能够将汇编语言源程序转化成机器语言程序的翻译程序也被开发出来，并被称为汇编程序。

当然这样的程序要想在计算机上运行还需要经过汇编程序的翻译。借助于汇编语言，计算机应用得到了迅速的发展。这时程序员开始发现即便是完成一个很简单的任务，也需要编写很多条汇编指令。

3. 高级语言

为了克服低级语言的缺点，20 世纪 50 年代创造出了第一个计算机高级语言 FORTRAN 语言。它很接近人们习惯使用的自然语言和数学语言。程序中用到的语句和指令是用英文单词表示的，程序中所用的运算符和运算表达式与人们日常所用的数学式子差不多，很容易理解。程序运行的结果用英文和数字输出，十分方便。

高级语言源程序在计算机上运行需要被翻译成机器语言，这个翻译程序被称为编译器，被转换为机器指令的程序称为目标程序。今天，绝大多数程序都是采用高级语言编写的。

具体来说，编译是指将高级语言源程序一次性地、完整地翻译成机器语言，所以编译一个高级语言源程序是很费时的，对计算机的性能要求也很高。

1.2.2 高级语言的发展

数十年来，全世界涌现了 2 500 种以上高级语言，每种高级语言都有其特定的用途，其中应用比较广泛的有 100 多种，影响最大的有 C 语言、C++语言和 JAVA 语言等。

高级语言经历了以下几种不同的发展阶段。

①非结构化的语言。初期的语言属于非结构化的语言，编程风格较随意，只要符合语法规则即可，程序中的流程可以随意跳转。程序难以阅读和维护。

②结构化语言。为了解决以上问题，提出了“结构化程序设计方法”，规定程序必须由顺序结构、分支结构和循环结构等基本结构构成，程序中的流程不允许随意跳转。这种程序结构清晰，易于编写、阅读和维护。C 语言属于结构化语言，支持结构化程序设计方法。

以上两种语言都是基于过程的语言，在编写程序时需要具体指定每一个过程的细节。在实践的发展中，又提出了面向对象的程序设计方法。程序面对的不是过程的细节，而是一个个对象，对象是由数据以及对数据进行的操作组成的。

③面向对象的语言。近十多年来，在处理规模较大的问题时，开始使用面向对象的语言。C++、C# 和 JAVA 等语言就是支持面向对象程序设计方法的语言。

1.3　C语言的发展及其优点

1.3.1　C语言的发展

C语言是在20世纪70年代初问世的。1978年，美国电话电报公司(AT&T)贝尔实验室正式发表了C语言。同时由B. W. Kernighan和D. M. Ritchit合著了著名的“THE C PROGRAMMING LANGUAGE”一书。通常简称为“K&R”，也称为“K&R”标准。但是，在“K&R”中并没有定义一个完整的标准C语言。后来由美国国家标准学会在此基础上制定了一个C语言标准，于1983年发表，通常称为ANSI C。

早期的C语言主要用于UNIX系统。由于C语言的强大功能和各方面的优点逐渐为人们所认识，到了20世纪80年代，开始进入其他操作系统，并很快在各类大、中、小和微型计算机上得到广泛的使用。C语言成为当代最优秀的程序设计语言之一。

C语言是一种结构化语言。它层次清晰，便于按模块化方式组织程序，易于调试和维护。C语言的表现能力和处理能力极强。它不仅具有丰富的运算符和数据类型，便于实现各类复杂的数据结构，还可以直接访问内存的物理地址，进行位(bit)一级的操作。由于C语言实现了对硬件的编程操作，因此它集高级语言和低级语言的功能于一体，既适用于系统软件的开发，又适合于应用软件的开发。此外，C语言还具有效率高、可移植性强等特点。因此，被广泛移植到各种类型的计算机上，从而形成了多种版本的C语言。

目前最流行的C语言有以下几种：

- Microsoft C或称MS C；
- Borland Turbo C或称Turbo C；
- AT&T C。

这些C语言版本不仅实现了ANSI C标准，而且在此基础上各自作了一些扩充，使之更加方便、完美。

自20世纪90年代初，C语言在我国开始推广以来，学习和使用C语言的人越来越多，使其成了学习和使用人数最多的一种计算机语言。

1.3.2　C语言的优点

C语言简洁、紧凑，使用方便、灵活。C语言一共有37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，压缩不必要的成分。C语言程序比其他许多高级语言简练，源程序短，因此输入程序时工作量少。

实际上，C语言是一个很小的内核语言，只包括极少与硬件有关的成分，语言不直接提供输入和输出语句、有关文件操作的语句和动态内存管理的语句等。C语言的编译系统相当简洁。

运算符丰富。C 语言共有 34 种运算符，包含的范围很广泛。C 语言把括号、赋值和强制类型转换等都作为运算符处理，从而使语言的运算类型丰富，表达式类型多样化。灵活使用各种运算符可以实现在其他高级语言中难以实现的运算。

数据类型丰富。C 语言提供的数据类型包括：整型、浮点型、字符型、数组类型、指针类型、结构体类型、共用体类型、复数浮点类型、超长整型和布尔类型等。尤其是指针类型数据，使用十分灵活和多样化，能用来实现各种复杂的数据结构的运算。

具有结构化的控制语句。用函数作为程序的模块单位，便于实现程序的模块化。语言完全模块化和结构化。

C 语言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。因此 C 语言既具有高级语言的功能，又具有低级语言的许多功能，可用来编写系统软件。C 语言的这种双重性，使它既是成功的系统描述语言，又是通用的程序设计语言。

用 C 语言编写的程序可移植性好。由于 C 语言的编译系统相当简洁，因此很容易移植到新的系统。而且 C 语言编译系统在新的系统上运行时，可以直接编译“标准链接库”中的大部分功能，不需要修改源代码，因为标准链接库是用可移植的 C 语言编写的。因此，几乎在所有的计算机系统中都可以使用 C 语言。

对以上特点，待学完 C 语言后再来回顾，就会有比较深的体会。

1.4 C 语言的结构特点

1.4.1 C 语言的结构

一个 C 语言源程序由一个或多个源文件组成。每个源文件由一个或多个函数组成。函数是 C 程序的主要组成部分。程序的几乎全部工作都是由各个函数分别完成的，函数是 C 程序的基本单位，在设计良好的程序中，每个函数都用来实现一个或几个特定的功能。一个源程序不论由多少个文件组成，都有且只有一个 main 函数，即主函数。不论 main 函数处在整个程序中的什么位置，程序总是从 main 函数开始执行的。

源程序中可以有预处理命令(include 命令仅为其中的一种)，预处理命令通常应放在源文件或源程序的最前面。每一个数据声明和语句都必须以分号结尾。标识符、关键字之间必须至少加一个空格以示间隔。

C 语言字符集由字母、数字、空白符、标点和特殊字符组成。在字符常量、字符串常量和注释中，可以使用汉字或其他可表示的图形符号。

为了进一步说明 C 语言源程序结构的特点，先看以下几个程序。这几个程序由简到难，表现了 C 语言源程序在组成结构上的特点。虽然有关内容还未介绍，但可从这些例子中了解到一个 C 源程序的基本组成部分和书写格式。

【例 1-1】 简单 C 语言程序例子。

```
main( )
{
   printf("您好！欢迎学习 C 语言\n");
}
```

main 是主函数的函数名，表示这是一个主函数。每一个 C 源程序都必须有，且只能有一个主函数(main 函数)。printf 函数的功能是把要输出的内容送到显示器去显示。printf 函数是一个由系统定义的标准函数，可在程序中直接调用。

【例 1-2】 C 语言顺序结构。

```
#include<math.h>
/*include 称为文件包含命令。它包含扩展名为.h 的文件，也称为头文件或首部文件*/
#include<stdio.h>
main( )
{
   double x,s;                          /*定义两个实数变量，以被后面程序使用*/
   printf("input number:\n");           /*显示提示信息*/
   scanf("%lf",&x);                     /*从键盘获得一个实数 x*/
   s=sin(x);                            /*求 x 的正弦，并把它赋给变量 s*/
   printf("sine of %lf is %lf\n",x,s);  /*显示程序运算结果*/
} /*main 函数结束*/
```

上述程序的功能是从键盘输入一个数 x，求 x 的正弦值，然后输出结果。

在 main( )之前的两行称为预处理命令。预处理命令还有其他几种，这里的 include 称为文件包含命令，其意义是把尖括号(<>)或引号("")内指定的文件包含到本程序中，成为本程序的一部分。被包含的文件通常是由系统提供的，其扩展名为.h。这个文件也称为头文件或首部文件。C 语言的头文件中包括各个标准库函数的函数原型。因此，凡是在程序中调用一个库函数时，都必须包含该函数原型所在的头文件。本例中使用了三个库函数：输入函数 scanf，正弦函数 sin，输出函数 printf。sin 函数是数学函数，其头文件为 math.h 文件，在程序的主函数前用 include 命令中包含了 math.h。scanf 和 printf 是标准输入/输出函数，其头文件为 stdio.h，因此在主函数前用的 include 命令中也包含了 stdio.h 文件。

需要说明的是，C 语言规定对 scanf 和 printf 这两个函数可以省去对其头文件的包含命令。所以在本例中也可以删去第 2 行的包含命令 #include。同样，【例 1-1】中使用了 printf 函数，也省略了包含命令。

本例中的主函数体又分为两部分，一部分为说明部分，另一部分为执行部分。说明是指变量的类型说明。【例 1-2】中未使用任何变量，因此无说明部分。C 语言规定，源程序中所有用到的变量都必须先说明后使用，否则将会出错。这一点是编译型高级程序设计语言的一个特点，与解释型的 BASIC 语言是不同的。说明部分是 C 源程序结构中很重要的组成部分。【例 1-2】中使用了两个变量 x，s，用来表示输入的自变量和 sin 函数值。由于 sin 函数要

求这两个量必须是双精度浮点型，故用类型说明符 double 来说明这两个变量。说明部分后的 4 行行为执行部分或称为执行语句部分，用以完成程序的功能。执行部分的第 1 行是输出语句，调用 printf 函数在显示器上输出提示字符串，请操作人员输入自变量 x 的值。第 2 行为输入语句，调用 scanf 函数，接受键盘上输入的数并存入变量 x 中。第 3 行是调用 sin 函数并把函数值送到变量 s 中。第 4 行是用 printf 函数输出变量 s 的值，即 x 的正弦值。程序结束。

1.4.2 C 语言的书写规则

从书写清晰以便于阅读、理解、维护的角度出发，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，通常表示程序的某一层次结构。“{”和“}”一般与该结构语句的第一个字母对齐，并单独占一行。

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

(4)程序应当包含注释。

1.5 C 语言的编译运行

用 C 语言编写的程序是源程序。计算机不能直接识别和执行用高级语言编写的指令，必须用编译程序把 C 源程序翻译成二进制形式的目标程序，然后再将该目标程序与系统的函数库以及其他目标程序连接起来，形成可执行的目标程序。

在编好一个 C 源程序后，怎样上机进行编译和运行呢？一般要经过以下几个步骤。

(1)编辑源程序。通过键盘向计算机输入程序，如发现有错误，要及时改正。最后将此源程序以文件形式存放在指定的文件夹内。

(2)对源程序进行编译。先用编译系统提供的“预处理器”(又称“预处理程序”或“预编译器”)对程序中的预处理指令进行编译预处理。由预处理得到的信息与程序其他部分一起，组成一个完整的、可以用来进行正式编译的源程序，然后由编译系统对该源程序进行编译。

编译的作用首先是对源程序进行检查，判定它有无语法方面的错误，如有，则发出“出错信息”，告诉编程人员认真检查改正。修改程序后重新进行编译。如此反复进行，直到没有语法错误为止。这时，编译程序自动把源程序转换为二进制形式的目标程序。如果不特别指定，此目标程序一般也存放在用户当前目录下，此时源文件没有消失。

在用编译系统对源程序进行编译时，自动包括了预编译和正式编译两个阶段。用户不必分别发出二次指令。

(3)连接处理。经过编译所得到的二进制目标文件直接执行。前面已说明：一个程序可能包含若干个源程序文件，而编译是以源程序文件为对象的，一次编译只能得到与一个源程

序文件相对应的目标文件，它只是整个程序的一部分。必须把所有编译后得到的目标模块连接装配起来，再与函数库相连接成一个整体，生成一个可供计算机执行的目标程序，称为可执行程序。

即使一个程序只包含一个源程序文件，编译后得到的目标程序也不能直接运行，要经过连接阶段，因为要与函数库进行连接，才能生成可执行程序。

(4)装载。顾名思义，就是要将程序装载到内存中。因为中央处理器只能访问内存中的指令与数据，而用户的所有程序通常都是存放在硬盘中的，程序要想得到执行，就必须事先装载到内存中去。完成这项“搬运”任务的程序叫做装载程序。由于需要装载的是程序的可执行映像，所以与程序执行相关的库函数也要同时装载到内存中。

(5)执行。计算机在中央处理器的控制下，逐条执行程序中的机器指令。

大多数 C 程序都需要进行数据的输入和输出，C 程序的输入都是通过标准输入流来实现的，通常的标准输入流是键盘。程序的输出都是将数据输出到标准输出流上，通常的标准输出流是显示器。当程序要打印一个结果时，实质上是指将结果显示在计算机屏幕上。当然也可以将标准输出流连接到其他输出设备(如硬盘、打印机等)。

以上过程如图 1-1 所示。

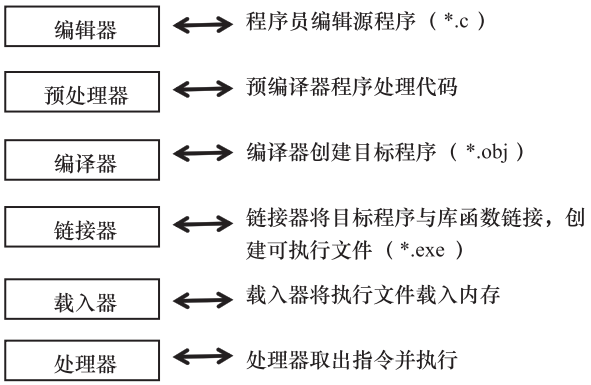

图 1-1　C 程序执行流程图

一个程序从编写到输出结果，并不是一次成功的，往往要经过多次反复修改。编写好的程序并不一定能保证正确无误，除了用人工方式检查外，还须借助编译系统来检查有无语法错误。从图中可以看到：如果在编译过程中发现错误，应当重新检查源程序，找出问题，并修改，然后重新编译，直到无误为止。有时编译过程未发现错误，能生成可执行程序，但是运行结果不正确。一般情况下，这不是语法方面的错误，而可能是程序逻辑方面的错误，例如计算公式不正确、赋值不正确等，应当返回检查源程序，并改正错误。

写出源程序后可以用任何一种编译系统对程序进行编译和连接工作，只要方便、有效即可。

1.6 算法

一个程序主要包括以下两方面的信息。

对数据的描述。在程序中要指定用到哪些数据以及这些数据的类型和数据的组织形式，这就是数据结构。

对操作的描述，即要求计算机进行操作的步骤，也就是算法。数据是操作的对象，操作的目的是对数据进行加工处理，以得到期望的结果。

编程当然是为了解决计算问题，但计算问题强调算法，而程序设计也是在一定抽象层次上的算法设计。这里的抽象层次应理解为数据的描述方式。当人类面临大量数据需要处理时，也就是许多编程问题含有大量纵横交错的数据时，便逐渐意识到数据组织与数据结构的重要性，意识到数据存在的形式必须脱离程序。

作为程序设计人员，必须认真考虑和设计数据结构与操作步骤（即算法）。著名计算机科学家 N. Wirth 提出一个公式：算法＋数据结构＝程序。

1.6.1 算法的概念

由于程序的动作序列包含了对数据的存取访问和算术运算，特别是当今的计算机要处理大量数据，因此对数据的合理描述、组织、存放和读取，关系到程序运行的正确和高效。

算法是求解特定问题的一组有限的操作序列，即一系列解决问题的清晰指令。也就是说，能够对一定规范的输入，在有限时间内获得所要求的输出。如果一个算法有缺陷，或不适合于某个问题，则执行这个算法将不会解决这个问题。一旦用特定的计算机语言来描述算法，就会使该算法成为特定计算机语言下的程序。更多的情况是，算法要靠程序实现来验证，即通过有限的数据，在有限的时间内，得出正确的结果，测试其资源占用与各项性能指标。不同的算法可能用不同的时间、空间或效率来完成同样的任务。一个算法的优劣可以用空间复杂度与时间复杂度来衡量。

1. 目的性

算法是有求解目的动作的序列，因此，算法必须有运算结果。而程序只是强调过程性，也许是不能自行终止的操作序列。例如，操作系统程序随着计算机的开机而运行，随着关闭电源而停止运行，而不是自行终结。

2. 抽象性

算法离不开数据结构，而数据结构在具体的计算机上实现要受到表示范围的限制，所以，算法比具体的程序设计更超脱一些，也就是抽象层次要高一些，算法甚至不在乎用什么编程语言来描述。

3. 研究性

算法许多时候用于理论研究，描述算法的目的是沟通人们的思想，而不是实现。所以，

算法可以用伪编程语言来描述。算法描述也可能是不切实际的。等到算法真正与程序代码对应起来的时候，可能就在一定程度上牺牲了研究的价值。算法事实上与数据相分离，如果数据由数据结构来描述，算法就可以以数据结构为依托来访问数据，从而简化算法和提高逻辑清晰性。

1.6.2　算法的特征

计算机算法可分为两大类别：数值运算算法和非数值运算算法。数值运算的目的是求数值解，例如求方程的根、求一个函数的定积分等。非数值运算包括的面十分广泛，最常见的是用于事务管理领域，例如对一批职工按姓名排序、图书检索、人事管理和行车调度管理等。目前，计算机在非数值运算方面的应用远远超过了在数值运算方面的应用。由于数值运算往往有现成的模型，可以运用数值分析方法，因此对数值运算的算法的研究比较深入，算法也比较成熟。而且对各种数值运算都有比较成熟的算法可供选用。

非数值运算的种类繁多，要求各异，难以做到全部都有现成的答案，因此只有一些典型的非数值运算算法（例如排序算法、查找搜索算法等）有现成的、成熟的算法可供使用。许多问题往往需要使用者参考已有的类似算法的思路，重新设计解决特定问题的专门算法。

一个算法应该具有以下五个重要的特征。

①有穷性：一个算法必须保证执行有限步之后结束；

②确切性：算法的每一步骤必须有确切的定义；

③输入：一个算法有 0 个或多个输入，以刻画运算对象的初始情况；

④输出：一个算法有一个或多个输出，以反映对输入数据加工后的结果。没有输出的算法是毫无意义的；

⑤可行性：算法原则上能够精确地运行。

1.6.3　伪代码

为了表示一个算法，可以用不同的方法。常用的方法有：自然语言、传统流程图、结构化流程图和伪代码等。

用传统的流程图和 N-S 图表示算法直观易懂，但画起来比较费事，在设计一个算法时，可能要反复修改，而修改流程图是比较麻烦的。因此，流程图适于表示一个算法，但在设计算法过程中使用不是很理想。为了更方便的设计算法，常用一种称为伪代码的工具。

伪代码是用介于自然语言和计算机语言之间的文字与符号来描述算法，是一种人工的、非正式的辅助进行算法设计的语言。它如同一篇文章，自上而下地写下来。每一行（或几行）表示一个基本操作。它不用图形符号，因此书写方便，格式紧凑，修改方便，容易看懂，也便于向程序过渡。

这里将要介绍的伪代码特别适合于设计可直接转换成结构化 C 程序的算法，是专门面向这类算法设计的。伪代码与日常使用的英语极为类似，虽然不是一种真正的计算机程序设计语言，但是它易学、易懂，书写方便。

伪代码不是用在计算机上执行的，只是帮助用计算机程序设计语言编写程序前“思考”程序应该如何设计。

【例 1-3】 求 5!，用伪代码表示算法。

```
begin
n=1
i=2
while i<5
    {
    i×n=n
    I+1=i
          }
print   n
end（算法结束）
```

用伪代码写算法并无固定的、严格的语法规则，可以用英文，也可以中英文混用。只要把意思表达清楚，便于书写和阅读即可，书写的格式为清晰易读的形式。

1.7 结构化编程

程序设计采用从上到下，逐步细分的方法展开，即过程化的程序设计方法。在细化的过程中，应充分运用前面的过程体控制结构和模块划分原则。当然，理论归理论，在实际运用的时候，还要考虑到规模适中，不能为了模块化而将过程划分成只有一两条语句的模块。

在一定的数据结构下设计对应算法，然后分别实现数据结构设计和算法设计，这种方法总能符合模块化的要求。

结构化程序设计强调程序设计风格和程序结构的规范化，提倡清晰的结构。而怎样才能得到一个结构化的程序呢？如果面临一个复杂的问题，是难以一下写出一个层次分明、结构清晰、算法正确的程序的。结构化程序设计方法的基本思路是把一个复杂问题的求解过程分阶段进行，每个阶段处理的问题都控制在容易理解和处理的范围内。

具体来说，采取以下方法来保证得到结构化的程序。

(1)自顶向下；

(2)逐步细化；

(3)模块化设计；

(4)结构化编码。

在程序设计中常采用模块设计的方法，尤其当程序比较复杂时，更有必要。在拿到一个程序模块以后，根据程序模块的功能将它划分为若干个子模块，如果这些子模块的规模还大，可以再划分为更小的模块。

程序中的子模块在 C 语言中通常用函数来实现。划分子模块时应注意模块的独立性，

即使用一个模块完成一项功能，耦合性越少越好。模块化设计的思想实际上是一种“分而治之”的思想，即把一个大任务分为若干个子任务，每一个子任务就相对简单了。

结构化程序设计方法用来解决人脑思维能力的局限性和被处理问题的复杂性之间的矛盾。

在设计好一个结构化的算法之后，还要善于进行结构化编码(coding)。所谓编码就是将已设计好的算法用计算机语言来表示出来，即根据已经细化的算法正确地写出计算机程序。

1.8　Visual C++ 6.0

1.8.1　Visual C++ 6.0 的开发环境

下面介绍 Visual C++ 6.0 的常用菜单命令项。

1. File 菜单

New：打开“new”对话框，以便创建新的文件、工程或工作区。

Close Workspace：关闭与工作区相关的所有窗口。

Exit：退出 Visual C++ 6.0 环境，将提示保存窗口内容等。

2. Edit 菜单

Cut：快捷键 Ctrl+X。将选定内容复制到剪贴板，然后再从当前活动窗口中删除所选内容。与“Paste”联合使用可以移动选定的内容。

Copy：快捷键 Ctrl+C。将选定内容复制到剪贴板，但不从当前活动窗口中删除所选内容。与“Paste”联合使用可以复制选定的内容。

Paste：快捷键 Ctrl+V。将剪贴板中的内容插入(粘贴)到当前鼠标指针所在的位置。注意，必须先使用 Cut 或 Copy 使剪贴板中具有准备粘贴的内容。

Find：快捷键 Ctrl+F。在当前文件中查找指定的字符串。顺便指出，可按快捷键 F3 寻找下一个匹配的字符串。

Find in Files：在指定的多个文件中查找指定的字符串。

Replace：快捷键 Ctrl+H。替换指定的字符串(用某一个串替换另一个串)。

Go To：快捷键 Ctrl+G。将光标移到指定行上。

Breakpoints：快捷键 Alt+F9。弹出对话框，用于设置、删除或查看程序中的所有断点。断点将告诉调试器应该在何时何地暂停程序的执行，以便查看当时的变量取值等现场情况。

3. View 菜单

Workspace：如果工作区窗口没显示出来，选择执行该项后将显示出工作区窗口。

Output：如果输出窗口没有显示出来，选择执行该项后将显示输出窗口。输出窗口中将随时显示有关的提示信息或出错警告信息等。

4. Project 菜单

Add To Project：选择该项将弹出子菜单，用于添加文件或数据链接等到工程中去。例如子菜单中的 New 选项可用于添加“C++ Source File”或“C/C++ Header File”；而子菜单中的 Files 选项则用于插入已有的文件到工程中。

Settings：为工程进行各种不同的设置。当选择其中的“Debug”标签（选项卡），并通过在“Program arguments：”文本框中填入以空格分割的各命令行参数后，则可以为带参数的 main 函数提供相应参数。

5. Build 菜单

Compile：快捷键 Ctrl+F7。编译当前处于源代码窗口中的源程序文件，以便检查是否有语法错误或警告，如果有的话，将显示在 Output 输出窗口中。

Build：快捷键 F7。对当前工程中的有关文件进行连接，若出现错误，也将显示在 Output 输出窗口中。

Execute：快捷键 Ctrl+F5。运行（执行）已经编译、连接成功的可执行程序（文件）。

Start Debug：选择该项将弹出子菜单，其中含有用于启动调试器运行的几个选项。例如其中的 Go 选项用于从当前语句开始执行程序，直到遇到断点或遇到程序结束；Step Into 选项开始单步执行程序，并在遇到函数调用时进入函数内部再从头单步执行；Run to Cursor 选项使程序运行到当前鼠标光标所在行时暂停其执行（注意，使用该选项前，要先将鼠标光标设置到某一个希望暂停的程序行处）。执行该菜单的选择项后，就启动了调试器，此时菜单栏中将出现 Debug 菜单（而取代了 Build 菜单）。

6. Debug 菜单

启动调试器后才出现该 Debug 菜单（而不再出现 Build 菜单）。

Go：快捷键 F5。从当前语句启动继续运行程序，直到遇到断点或遇到程序结束而停止（与 Build→Start Debug→Go 选项的功能相同）。

Restart：快捷键 Ctrl+Shift+F5。重新开始对程序进行调试执行（当对程序做过某些修改后往往需要这样做）。选择该项后，系统将重新装载程序到内存，并放弃所有变量的当前值（而重新开始）。

Stop Debugging：快捷键 Shift+F5。中断当前的调试过程并返回正常的编辑状态（注意，系统将自动关闭调试器，并重新使用 Build 菜单来取代 Debug 菜单）。

Step Into：快捷键 F11。单步执行程序，并在遇到函数调用语句时，进入函数内部，并从头单步执行（与 Build→Start Debug→Step Into 选项的功能相同）。

Step Over：快捷键 F10。单步执行程序，但当执行到函数调用语句时，不进入函数内部，而是直接执行完该函数后，接着再执行函数调用语句后面的语句。

Step Out：快捷键 Shift+F11。与“Step Into”配合使用，当执行进入到函数内部，单步执行若干步之后，若发现不再需要进行单步调试的话，通过该选项可以从函数内部返回（到函数调用语句的下一语句处停止）。

Run to Cursor：快捷键 Ctrl+F10。使程序运行到当前鼠标光标所在行时暂停其执行

(注意,使用该选项前,要先将鼠标光标设置到某一个希望暂停的程序行处)。事实上,相当于设置了一个临时断点,与 Build→Start Debug→Run to Cursor 选项的功能相同。

Insert/Remove Breakpoint:快捷键 F9。本菜单项并未出现在 Debug 菜单上(在工具栏和程序文档的上下文关联菜单上),列在此处是为了方便掌握程序调试的手段,其功能是设置或取消固定断点——程序行前有一个圆形的黑点标志,表示已将该行设置了固定断点。另外,与固定断点相关的还有 Alt+F9(管理程序中的所有断点)、Ctrl+F9(禁用/使用当前断点)。

7. Help 菜单

通过该菜单来查看 Visual C++ 6.0 的各种联机帮助信息。

8. 上下文关联菜单

除了主菜单和工具栏外,Visual C++ 6.0 开发环境还提供了大量的上下文关联菜单,用鼠标右键单击窗口中很多地方都会弹出一个关联菜单,里面包含了与被单击项目相关的各种命令,建议在工作时可以试着多单击几次鼠标右键,说不定会发现很多有用的命令,从而极大加快一些常规操作的速度。

1.8.2　Visual C++ 6.0 的主要工作窗口

1. Workspace 窗口

Workspace 窗口显示了当前工作区中各个工程的类、资源和文件信息,当新建或打开一个工作区后,Workspace 窗口通常就会出现三个树视图:ClassView(类视图)、ResourceView(资源视图)和 FileView(文件视图),如果在 Visual C++ 6.0 企业版中打开了数据库工程,还会出现第四个视图 DataView(数据视图)。如同前面所述,在 Workspace 窗口的各个视图内单击鼠标右键可以得到很多有用的关联菜单。

ClassView 显示当前工作区中所有工程定义的 C++类、全局函数和全局变量,展开每一个类后,可以看到该类的所有成员函数和成员变量,如果双击类的名字,Visual C++ 6.0 会自动打开定义这个类的文件,并把文档窗口定位到该类的定义处;如果双击类的成员或者全局函数及变量,文档窗口则会定位到相应函数或变量的定义处。

ResourceView 显示每个工程中定义的各种资源,包括快捷键、位图、对话框、图标、菜单、字符串资源、工具栏和版本信息,如果双击一个资源项目,Visual C++ 6.0 就会进入资源编辑状态,打开相应的资源,并根据资源的类型自动显示出 Graphics、Color、Dialog、Controls 等停靠式窗口。

FileView 显示了隶属于每个工程的所有文件。除了 C/C++源文件、头文件和资源文件外,还可以向工程中添加其他类型的文件,例如 Readme.txt 等,这些文件对工程的编译连接不是必需的,但将来制作安装程序时会被一起打包。同样,在 FileView 中双击源程序等文本文件时,Visual C++ 6.0 会自动为该文件打开一个文档窗口。双击资源文件时,Visual C++ 6.0 也会自动打开其中包含的资源。

在 FileView 中对着一个工程单击鼠标右键后,关联菜单中会出现“Clean”命令,在此

需要解释一下它的功能。Visual C++ 6.0 在建立(Build)一个工程时,会自动生成很多中间文件,例如预编译头文件、程序数据库文件等,这些中间文件加起来的大小往往有数兆,很多人在开发一个软件期间会使用办公室或家里的数台机器,如果不把这些中间文件删除,在多台机器之间使用软盘复制工程就会很麻烦。“Clean”命令的功能就是把 Visual C++ 6.0 生成的中间文件全部删除,避免了手工删除时可能会出现误删或漏删的问题。另外,在某些情况下,Visual C++ 6.0 编译器可能无法正确识别哪些文件已被编译过,以致在每次建立工程时都进行重建,很浪费时间,此时执行“Clean”命令删除中间文件就可以解决这一问题。

应当指出,承载一个工程的还是存储在工作文件夹下的多个文件(物理上),在 Workspace 窗口中的这些视图都是逻辑意义上的,它们从不同的角度去自动统计总结工程的信息,以方便和帮助查看工程以及更有效地开展工作。如果开始不习惯且工程很简单(学习期间很多时候都只有一个.cpp 文件),则完全没有必要去看这些视图,只需要在.cpp 文件内容窗口中进行工作。

2. Output 窗口

与 Workspace 窗口一样,Output 窗口也被分成了数栏,其中前面 4 栏最常用。在建立工程时,Build 栏将显示工程在建立过程中经过的每一个步骤及相应信息,如果出现编译连接错误,那么发生错误的文件及行号、错误类型编号和描述都会显示在 Build 栏中,用鼠标双击一条编译错误,Visual C++ 6.0 就会打开相应的文件,并自动定位到发生错误的那一条语句。

工程通过编译连接后,运行其调试版本,Debug 栏中会显示出各种调试信息,包括 DLL 装载情况、运行时警告及错误信息、MFC 类库或程序输出的调试信息、进程中止代码等。

两个 Find in Files 栏用于显示从多个文件中查找字符串后的结果,当想看看某个函数或变量出现在哪些文件中时,可以从“Edit”菜单中执行“Find in Files…”命令,然后指定要查找的字符串、文件类型及路径,单击“查找”按钮后结果就会输出在“Output”的“Find in Files”栏中。

3. 窗口布局调整

Visual C++ 6.0 的智能化界面允许用户灵活配置窗口布局,例如菜单和工具栏的位置都可以重新定位。在菜单或工具栏左方类似于把手的两个竖条纹处或其他空白处单击鼠标左键并按住,然后试着把它拖动到窗口的不同地方,就可以发现菜单和工具栏能够停靠在窗口的上方、左方和下方,双击竖条纹后,它们还能以独立子窗口的形式出现。独立子窗口能够始终浮动在文档窗口的上方,并且可以被拖到 Visual C++ 6.0 主窗口之外,如果有双显示器,甚至可以把这些子窗口拖到另外一个显示器上,以便进一步加大编辑区域的面积。Workspace 和 Output 等停靠式窗口(Docking View)也能以相同的方式进行拖动,或者切换成独立的子窗口,此外,这些停靠式窗口还可以切换成普通的文档窗口模式,不过文档窗口不能被拖出 Visual C++ 6.0 的主窗口,切换的方法是选中某个停靠式窗口后,在“Windows”菜单中把“Docking View”置于非选中状态。

1.8.3　C语言程序的调试

为了把程序代码输入后交给计算机执行，需要使用 Visual C++ 6.0 的编辑器来完成。首先要创建工程以及工程工作区，而后才能输入具体程序完成所谓的“编辑”工作。

1. 新建 Win32 Console Application 工程

执行菜单“File”下的“New”命令，会出现一个选择界面，选择“Projects”标签后，会看到近 20 种的工程类型，只需选择其中最简单的一种“Win32 Console Application”，然后在右上方的“Location”文本框和“Project name”文本框中填入工程相关信息所存放的磁盘位置（目录或文件夹位置）以及工程的名字，设置好的界面信息如图 1-2 所示。

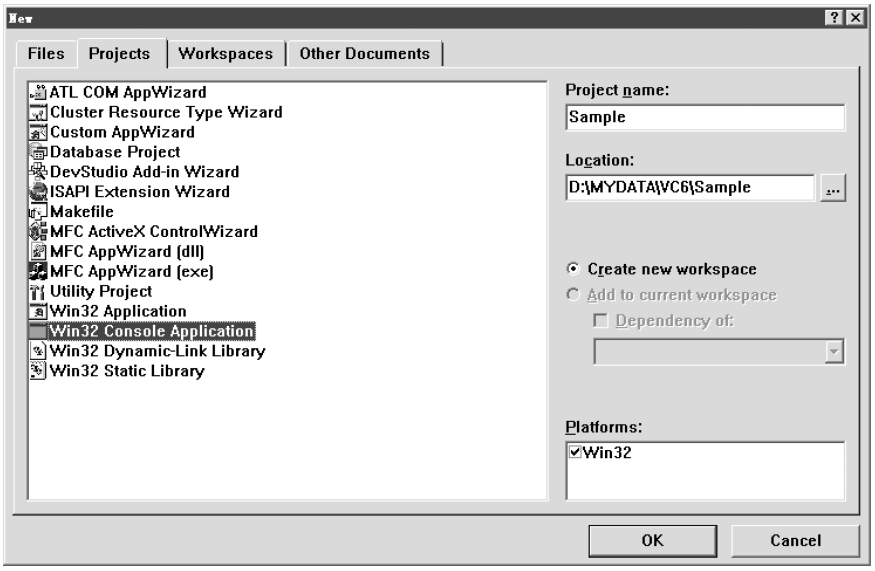

图 1-2　填写工程名及位置

在图 1-2 中的“Location”文本框中填入如“D:\MYData\VC6\Sample”，这是假设准备在 D 磁盘的\MYData\VC6\Sample 文件夹即子目录下存放与工程工作区相关的所有文件及其相关信息，当然也可通过单击其右部的“…”按钮去选择并指定这一文件夹即子目录位置。“Project name”文本框中填入如“Sample”的工程名（注意，名字根据工程性质确定，此时 Visual C++ 6.0 会自动在其下的“Location”文本框中用工程名“Sample”建立一个同名子目录，随后的工程文件以及其他相关文件都将存放在这个目录下）。

单击“OK”按钮进入下一个选择界面。这个界面主要是询问想要构成一个什么类型的工程，其界面如图 1-3 所示。

若选中“An empty project”单选按钮将生成一个空的工程，工程内不包括任何东西。若选中“A simple application”单选按钮将生成包含一个空的 main 函数和一个空的头文件的工程。选中“A″Hello World!″application”单选按钮与选中“A simple application”单选按钮没有什么本质的区别，只是需要包含有显示出“Hello World!”字符串的输出语句。选中“An

application that supports MFC"单选按钮的话，可以利用 Visual C++ 6.0 所提供的类库来进行编程。

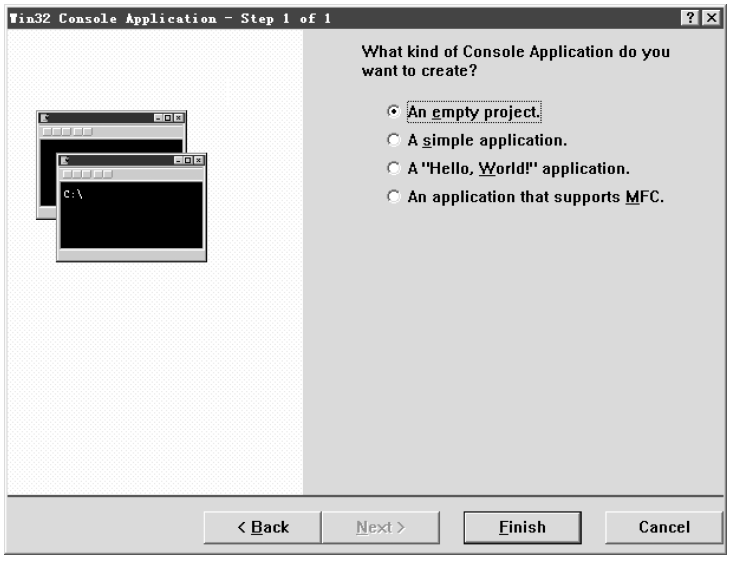

图 1-3　选择创建一个工程

为了更清楚地看到编程的各个环节，选中"An empty project"单选按钮，从一个空的工程开始工作。单击"Finish"按钮，这时 Visual C++ 6.0 会生成一个小型报告，报告的内容是刚才所有选择项的总结，并且询问是否接受这些设置。如果接受则单击"OK"按钮，否则单击"Cancel"按钮。若单击"OK"按钮可进入到真正的编程环境下。界面如图 1-4 所示。

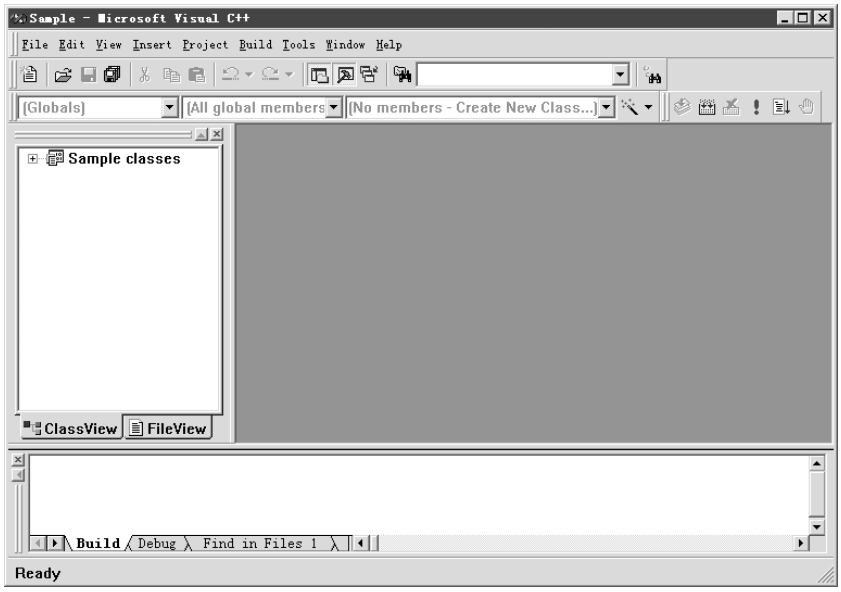

图 1-4　集成开发环境窗口

注意屏幕中的 Workspace 窗口，该窗口中有两个标签，一个是 ClassView，另一个是 FileView。ClassView 中列出的是这个工程中所包含的所有类的有关信息，当然程序将不涉及到类，这个标签中现在也是空的。单击“FileView”标签后，将看到这个工程所包含的所有文件信息。单击“＋”图标打开所有的层次会发现有三个逻辑文件夹：Source Files 文件夹中包含了工程中所有的源文件；Header Files 文件夹中包含了工程中所有的头文件；Resource Files 文件夹中包含了工程中所有的资源文件。所谓资源就是工程中用到的位图、加速键等信息，在编程中不会牵扯到这一部分内容。现在“FileView”中也不包含任何东西。

逻辑文件夹是逻辑上的，只是在工程的配置文件中定义的，在磁盘上并不存在这三个文件夹。也可以删除不使用的逻辑文件夹；或者根据项目需要，创建新的逻辑文件夹来组织工程文件。这三个逻辑文件夹是 VC 预先定义的，就编写简单的单一源文件的 C 程序而言，只需要使用 Source Files 一个文件夹就够了。

下面介绍生成一个“Hello.cpp”的源程序文件，而后通过编辑界面来输入所需的源程序代码。选择菜单“Project”中子菜单“Add To Project”下的“new”选项，在出现的对话框的“Files”标签（选项卡）中，选择“C＋＋ Source File”选项，在右中处的 File 文本框中为将要生成的文件取一个名字，例如这里取名为“Hello”（其他遵照系统隐含设置，此时系统将使用 Hello.cpp 文件保存所键入的源程序），此时的界面如图 1-5 所示。

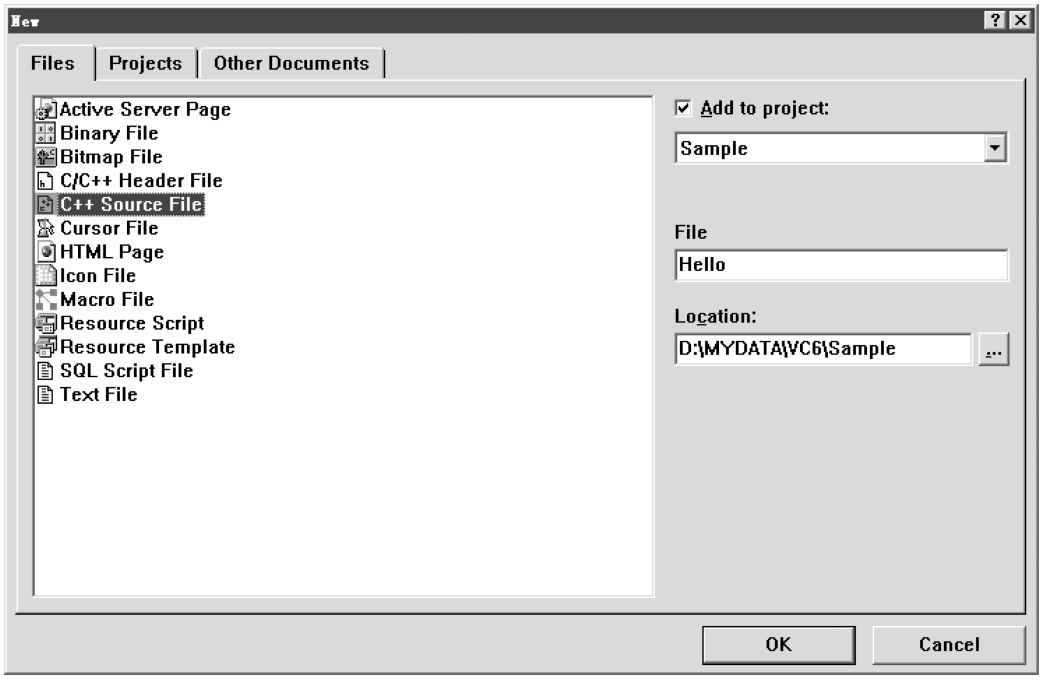

图 1-5　新建“C＋＋ Source File”

然后单击“OK”按钮，进入输入源程序的编辑窗口（注意所出现的呈现“闪烁”状态的输入位置光标），此时只需通过键盘输入所需要的源程序代码：

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

```
main()
{
   printf("Hello World! \n");
}
```

可通过 Workspace 窗口中的 FileView 标签，看到 Source Files 文件夹下的文件 Hello.cpp 已经被加了进去，此时的界面如图 1-6 所示。

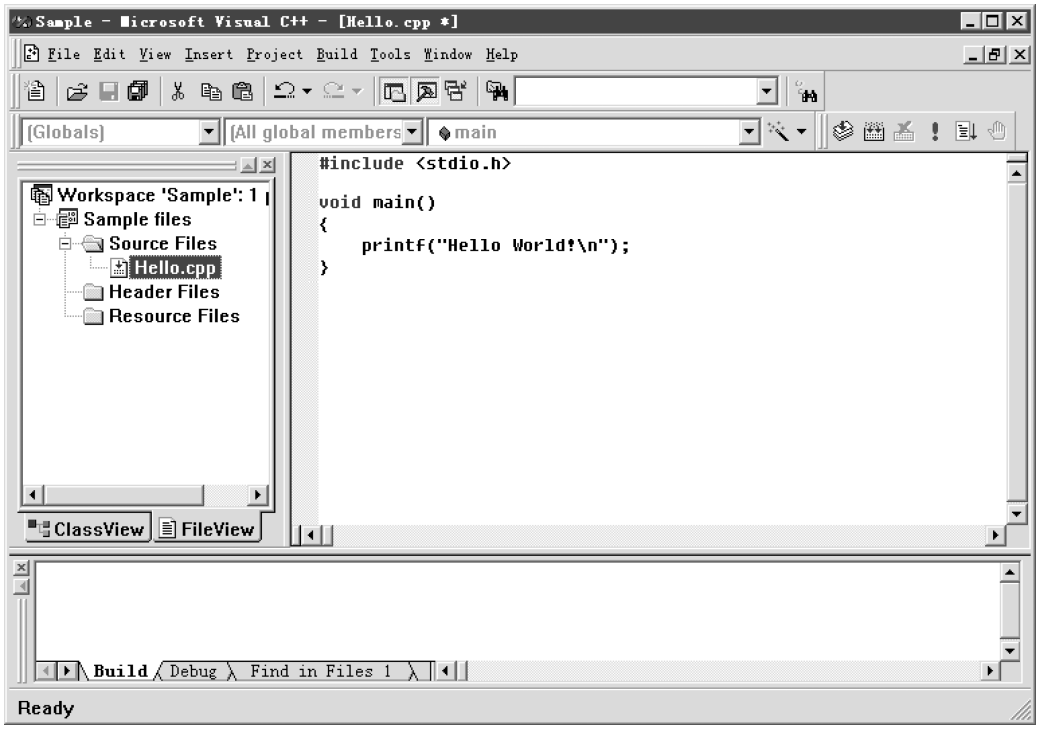

图 1-6　输入程序代码

实际上，这时在“Workspace”窗口的“ClassView”标签中的“Globals”文件夹下，也可以看到刚才键入的 main 函数。

2. 不创建工程，直接输入源程序代码

不需要像前面描述的那样去创新一个工程，对于新编写一个程序，只需要在图 1-2 所示的界面中，选择“Files”标签，再选择“C++ Source File”选项，其界面与图 1-5 相似（仅“Add to project”复选框是灰色，无法选择），后续操作则与前述相同。

最简单的做法是：直接单击工具栏上的“新建文件”选项按钮新建空白文件，紧接着单击工具栏上的“保存”按钮保存此空文件。注意，保存时一定要以“.c”或“.cpp”作为扩展名，否则逻辑程序时自动格式化和特殊显示等很多特性将无法使用，程序无法被运行。

这种方式新建的 C 源程序文件在编译时，会提示用户，要求允许系统为其创新一个默认的工程（含相应的工作区）。

3. 编译、链接后运行程序

程序编制完成(即所谓“四部曲”中第一步的编辑工作完成)之后,就可以进行后三步的编译、链接与运行了。所有后三步的命令项都处在菜单“Build”中。注意,在对程序进行编译、链接和运行前,最好先保存工程(执行“File→Save All”命令)以避免程序运行时系统发生意外而使之前的工作付之东流,应让这种保存做法成为习惯。

首先选择菜单第一项“Compile”,此时将对程序进行编译。若编译中发现错误(error)或警告(warning),将在“Output”窗口中显示出它们所在的行以及具体的出错或警告信息,可以通过这些信息的提示来纠正程序中的错误或警告(注意,错误是必须纠正的,否则无法进行下一步的链接;而警告则不然,它并不影响下一步操作,当然最好还是把所有的警告也“消灭”掉)。当没有错误与警告出现时,“Output”窗口所显示的最后一行应该是“Hello.obj－0 error(s), 0 warning(s)”。

编译通过后,可以选择菜单的第二项 Build 来进行链接生成可执行程序。在链接中出现的错误也将显示到“Output”窗口中。链接成功后,“Output”窗口所显示的最后一行应该是“Sample.exe－0 error(s), 0 warning(s)”。最后就可以运行所编制的程序了,选择“Execute”选项(该选项前有一个深色的感叹号标志“!”,实际上也可通过单击窗口上部工具栏中的深色感叹号标志“!”来启动该选项),Visual C＋＋ 6.0 将运行已经编好的程序,执行后将出现一个结果界面(类似于 DOS 窗口的界面),如图 1-7 所示,其中的“press any key to continue”是由系统产生的,可使用户浏览到输出结果,直到按任意一个键盘按键后,第二次又按为止(那时又将返回到集成界面的编辑窗口处)。

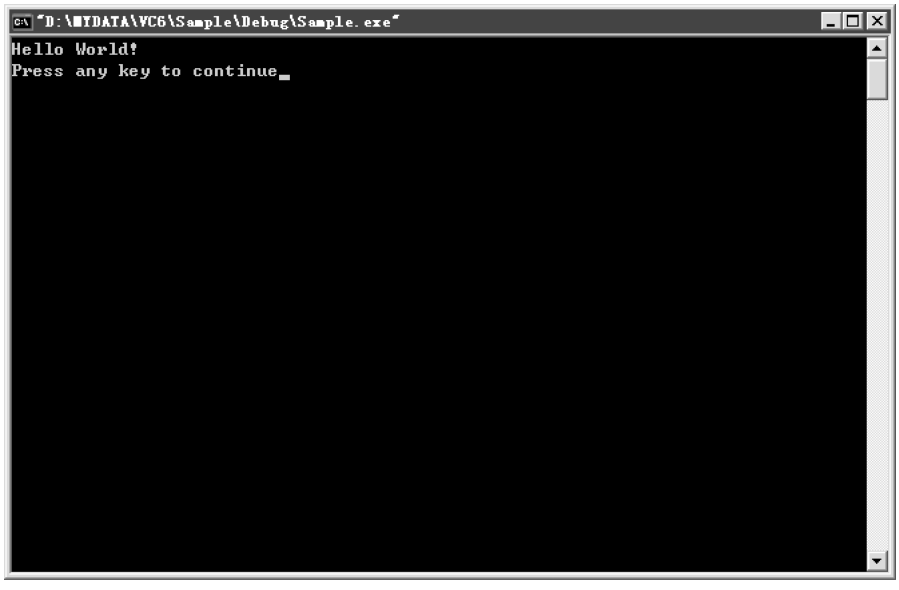

图 1-7　程序 Hello.cpp 的运行结果界面

至此已经生成并运行了一个完整的程序,完成了一个“回合”的编程任务。此时应执行“File→Close Workspace”命令,待系统询问是否关闭所有的相关窗口时,单击“是”按钮,则

结束了一个程序从输入到执行的全过程，回到了刚刚启动 Visual C++ 6.0 的那一个初始画面。

1.9 实 训

实训 1

【内容】上机操作。

【目的】掌握运行 C 语言程序的各个环节。

【题目】运行【例 1-1】、【例 1-2】和【例 1-3】。

实训 2

【内容】简单程序设计。

【目的】编写 C 语言程序。

【题目】参照【例 1-1】，编写一个输出下面字符的程序：

```
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
This is a C program!
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

习 题

一、选择题

1. C 程序的执行是从(　　)。

A. 本程序的 main 函数开始，到 main 函数结束

B. 本程序文件的第一个函数开始，到本程序文件的最后一个函数结束

C. 本程序的 main 函数开始，到本程序文件的最后一个函数结束

D. 本程序文件的第一个函数开始，到本程序 main 函数结束

2. 在 C 语言中，每个语句必须以(　　)结束。

A. 回车符　　B. 冒号　　C. 逗号　　D. 分号

3. C 语言规定：在一个源程序中，main 函数的位置(　　)。

A. 必须在最开始　　B. 必须在系统调用的库函数的后面

C. 可以任意　　D. 必须在最后

4. 以下叙述正确的是(　　)。

A. 系统默认的 C 语言源程序文件的扩展名是. obj

B. 在 C 程序的每一行只能写一条语句

C. C 语言本身没有输入/输出语句

D. 在对一个 C 程序进行编译的过程中，可发现注释中的拼写错误

5. 以下叙述不正确的是(　　)。

A. 一个 C 源程序可由一个或多个函数组成

B. 一个 C 源程序必须包含一个 main 函数

C. C 程序的基本组成单位是函数

D. 在 C 程序中，注释说明只能位于一条语句的后面

6. 一个 C 语言程序是由(　　)。

A. 一个主程序和若干个子程序组成　　B. 若干函数组成

C. 若干过程组成　　D. 若干子程序组成

7. C 源程序的基本单位是(　　)。

A. 函数　　B. 文件　　C. 语句　　D. 控制结构

二、判断题

1. C 语言的源程序在运行之前必须进行编译。(　　)

2. C 语言的程序可以由多个文件组成，每个文件中都可以有一个主函数 main()。(　　)

3. 在一个 C 源程序中，注释部分两侧的分界符分别为/ * 、* /，程序注释部分不影响程序的运行结果。(　　)

4. 在 C 语言中，输入操作是由库函数 printf 完成的，输出操作是由库函数 scanf 完成的。(　　)