

C 语言程序设计

邓晨曦 郑火胜 林 烨 主编



湖南大学出版社·长沙
HUNAN UNIVERSITY PRESS

PREFACE

PREFACE

前言

党的“二十大”报告中，再次明确了职业教育的重要地位。报告中提到要“办好人民满意的教育，统筹职业教育、高等教育、继续教育协同创新，推进职普融通、产教融合、科教融汇，优化职业教育类型定位”，进一步为职业教育发展指明了前进方向，绘就了美好蓝图。

要结合职业教育教学，扎实推进习近平新时代中国特色社会主义思想 and 党的二十大精神进教材、进课堂、进头脑，将党的二十大精神有机融入思政课教学和专业课教育教学，强化课程思政，培养职业精神、工匠精神，让党的二十大精神内化于心、外化于行，确保党的二十大精神落地生根，推动我省职业教育高质量发展。

20 世纪 90 年代以来，C 语言迅速在全世界普及推广。C 语言是一种用途广泛、功能强大、数据类型丰富、运算符多且使用灵活的结构化程序设计语言，它既具有高级语言程序设计的特点，又具有汇编语言的功能，所以使用 C 语言既可以开发应用软件，又可以开发系统软件。目前绝大多数学校都将 C 语言作为学生学习程序设计语言的一门专业基础课，对培养学生的专业素养、专业能力意义重大。所以，C 语言程序设计在计算机教育和计算机应用中发挥着重要的作用。

本书内容分 10 章。第 1 章为程序设计基础知识，主要对程序设计语言的发展、功能做了介绍，同时对 C 程序在 DEV C++ 环境下的编辑、编译、链接、运行的方法进行了介绍。第 2 章为数据类型、运算符、表达式和算法，对基本数据类型、赋值、算术运算符及类型转换和算法做了介绍。第 3 章为顺序结构程序设计，主要对数据的输入和输出做了介绍。第 4 章为选择结构程序设计，结合关系运算符、逻辑运算符及条件运算符，介绍了使用 if 语句及 switch 语句进行选择程序设计的方法。第 5 章为循环结构程序设计，介绍了 for 循环、while 循环、do-while 循环以及循环的嵌套，并对典型的算法（递推法、迭代法）进行了讨论。第 6 章为数组，介绍了一维数组、二维数组、字符数组及字符串的使用方法，同时对数组作为函数的参数进行了讨论。第 7 章为函数，介绍了函数的定义与调用，递归函数以及数组作函数的参数的方法，并对变量与函数的关系进行了讨论。第 8 章为指针，介绍了指针的基本概念、指针与函数、指针与数组、指针与字符串。第 9 章为结构体和共用体，对用指针实现动态内存分配进行了讨论，介绍了结构的概念、结构变量的使用、结构数组、结构指针，同时介绍了共用体的使用方法。第 10 章为文件，介绍了文件的分类、文件的基本操作等。

本书内容翔实、结构完整、概念清晰、实例丰富，便于老师循序渐进地进行教学，也便于学生自学，老师和学生都可按需对内容进行筛选。

本书在编写过程中参考了大量国内外计算机程序设计文献资料，在此对这些文献资料的作者一并表示感谢。

由于编者水平有限，书中难免存在疏漏和不足之处，敬请广大读者批评指正，以便日后不断完善、改进。

编 者

CONTENTS

CONTENTS

目 录

第 1 章 程序设计基础知识	001	4.5 if 语句的嵌套	072
1.1 什么是计算机程序	002	4.6 用 switch 语句实现多分支选择结构	076
1.2 什么是计算机语言	002	4.7 选择结构程序综合举例	079
1.3 C 语言的发展及其特点	004	小 结	084
1.4 C 语言程序的基本结构	005	习 题	085
1.5 运行 C 程序的步骤与方法	009	第 5 章 循环结构程序设计	086
1.6 程序设计的任务	012	5.1 用 while 语句实现循环结构	087
小 结	013	5.2 用 do...while 语句实现循环结构	091
习 题	014	5.3 用 for 语句实现循环结构	096
第 2 章 数据类型、运算符、表达式和算法 ...	015	5.4 循环的嵌套	102
2.1 数据的表现形式	015	5.5 改变循环执行的状态	108
2.2 数据类型	020	小 结	115
2.3 常用运算符与表达式	025	习 题	115
2.4 程序的灵魂——算法	033	第 6 章 数 组	117
小 结	040	6.1 一维数组	117
习 题	040	6.2 二维数组	123
第 3 章 顺序结构程序设计	042	6.3 字符数组	128
3.1 数据的输入 / 输出	042	小 结	138
3.2 C 语言程序的结构	054	习 题	139
3.3 C 语句	056	第 7 章 函 数	140
小 结	061	7.1 函数的定义	141
习 题	062	7.2 函数的调用	145
第 4 章 选择结构程序设计	063	7.3 函数的嵌套调用	150
4.1 关系运算符和关系表达式	063	7.4 函数的递归调用	152
4.2 逻辑运算符和逻辑表达式	065	7.5 局部变量和全局变量	154
4.3 条件运算符和条件表达式	067	小 结	163
4.4 用 if 语句实现选择结构	068	习 题	164

第 8 章 指 针	166	10.1 文件概述	216
8.1 指针的定义	166	10.2 文件的常用操作	222
8.2 指针变量	167	小 结	240
8.3 指针与数组	173	习 题	240
8.4 指针与函数	178	附录 1 C 语言中的 32 个关键字及其含义 ...	241
小 结	186	附录 2 C 语言中的所有的转义字符	243
习 题	187	附录 3 ASCII 码对照表	244
第 9 章 结构体和共用体	188	附录 4 C 语言中运算符优先级对照表	245
9.1 结构体	188	附录 5 C 语言中常见错误中英文对照表	247
9.2 共用体	205	附录 6 C 语言中常用库函数	253
9.3 枚举类型	209	参考文献	257
小 结	214		
习 题	215		
第 10 章 文 件	216		

第 1 章

程序设计基础知识

计算思维 (computational thinking) 是一种新的思维方法, 它利用计算机科学的基础概念解决问题、进行系统设计并理解人类行为。计算思维一词由周以真 (Jeannette M. Wing) 教授于 2006 年提出, 目前受到了广泛的重视。美国的卡内基·梅隆大学早在 2007 年就建立了计算思维中心, 目的是开发计算机学科与其他学科交叉研究的新方法。ACM (association for computing machinery, 国际计算机学会) 在 2008 年公布的《CC2001 计算机科学教学指导草案》中指出, 应该将计算思维作为计算机学科教学的重要组成部分。

计算思维不仅仅属于计算机学科, 它将和阅读、写作及算术一样, 成为 21 世纪每个人必须具备的基本技能。学生的程序设计能力就是计算思维的具体体现。C 语言是国际上广泛流行的、很有发展前途的一种程序设计语言。下面我们将学习 C 语言程序设计, 培养学生的计算思维能力。

C 语言是在 B 语言的基础上发展起来的, 它的根源可以追溯到 ALGOL60。1960 年出现的 ALGOL60 是一种面向问题的高级语言。1963 年, 英国的剑桥大学推出了 CPL (combined programming language), 1967 年对 CPL 作了简化, 推出了 BCPL (basic combined programming language)。1970 年, 美国贝尔实验室又对 BCPL 作了进一步简化, 设计出了简单而且接近硬件的 B 语言, 但 B 语言过于简单, 功能有限。1973 年, 贝尔实验室在 B 语言的基础上设计出了 C 语言。C 语言既保持了 BCPL 和 B 语言的优点, 又克服了它们的缺点。最初的 C 语言只是为描述和实现 UNIX 操作系统而设计的。直到 1975 年 UNIX 第 6 版公布后, C 语言的突出优点才引起了人们的普遍关注。1978 年以后, C 语言已先后移植到大、中、小、微型机上, 现在已成为世界上最优秀的程序设计语言之一。

以 1978 年发表的 UNIX 第 7 版中的 C 编译程序为基础, 由 B.W.Kernighan 和 D.M.Ritchie 合著了著名的 *The C Programming Language* 一书, 通常简称为 “K&R”, 也有人称之为 “K&R” 标准。这本书中介绍的 C 语言成为后来广泛使用的 C 语言版本的基础, 被称为标准 C。1988 年, 随着微型计算机的日益普及, C 语言出现了许多版本。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, 美国国家标准学会 (American national standards institute, ANSI) 为 C 语言制定了一套 ANSI 标准, 成为现行的 C 语言标准。1990 年, ISO (international organization for standardization, 国际标准化组织) 再次采用了这种标准, 所以也称之为 C90。1999 年, ISO 对 C 语言进行了修订, 简称 C99。后来 ANSI 又采用了这种标准。在 ANSI 的标准确立后, C 语言的规范在一段时间内没有大的变动。《标准修正案一》在 1995 年为 C 语言创建了一个新标准, 但是只修正了一些 C89 标准中的细节和增加了更多更广的国际字符集支持。不过, 这个标准引出了 1999 年 ISO 9899: 1999 的发表, 它通常被称为 C99。C99 于 2000 年 3 月被 ANSI 采用。

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, 而且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 C 语言集低级语言和高级语言的功能于一体, 既可用于系统软件的开发, 也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, 因此被广泛地移植到各类型计算机, 从而形成了多种版本的 C 语言。



学习目标

- 掌握 C 语言的基本概念。
- 了解 C 语言程序的基本结构。
- 编写几个简单的 C 程序。



案例描述

编写两个程序，分别实现以下功能。

程序 1：输出“这是我的第一个 C 程序”。

程序 2：从键盘输入一个学生的考试成绩，判断并输出这个学生成绩是否及格。



案例相关知识

- C 程序的基本结构。
- C 程序的运行步骤。

1.1 什么是计算机程序

有人以为计算机是“万能”的，会自动进行所有的工作，甚至觉得计算机神秘莫测，这是很多初学者的误解。其实，计算机的每一个操作都是根据人们事先指定的指令进行的。例如用一条指令要求计算机进行一次加法运算，用另一条指令要求计算机将某运算结果输出到显示屏。为了使计算机执行一系列的操作，必须事先编好一条条指令，输入计算机。

所谓程序，就是一组计算机能识别和执行的指令。每条指令使计算机执行特定的操作。只要让计算机执行这个程序，计算机就会“自动地”执行各条指令，有条不紊地进行工作。一个特定的指令序列用来完成一定的功能。为了使计算机系统能实现各种功能，需要成千上万个程序。这些程序大多数是由计算机软件设计人员根据需要设计好的，作为计算机的软件系统的一部分提供给用户使用。此外，用户还可以根据自己的实际需要设计一些应用程序，例如学生成绩统计程序、财务管理程序、工程中的计算程序等。

总之，计算机的一切操作都是由程序控制的，离开程序，计算机将一事无成。所以，计算机的本质是程序的机器，程序和指令是计算机系统中最基本的概念。只有懂得程序设计，才能真正了解计算机是怎样工作的，才能更深入地使用计算机。

1.2 什么是计算机语言

一种计算机和人都能识别的语言，就是计算机语言。

人和人之间的交流需要通过语言。中国人之间用汉语，英国人用英语，俄罗斯人用俄语，等等。人和计算机交流信息也要解决语言问题，需要创造一种计算机和人都能识别的语言，这就是计算机语言。计算机语言经历了以下几个发展阶段。

1.2.1 机器语言

机器语言时代，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 和 1 组成的指

令。在计算机发展的初期，一般计算机的指令长度为 16，即以 16 个二进制数（0 或 1）组成一条指令，16 个 0 和 1 可以组成各种排列组合。

要使计算机知道和执行自己的意图，就要编写许多条由 0 和 1 组成的指令。然后要用纸带穿孔机以人工的方法在特制的黑色纸带上穿孔，在指定的位置上有孔代表 1，无孔代表 0。一个程序往往需要一卷长长的纸带。在需要运行此程序时就将此纸带装在光电输入机上，当光电输入机从纸带读入信息时，有孔处产生一个电脉冲，指令变成电信号，让计算机执行各种操作。

这种计算机能直接识别和接受的二进制代码称为机器指令。机器指令的集合就是该计算机的机器语言（machine language）。

机器语言的特点：难学，难记，难检查，难修改，难以推广使用，依赖具体机器难以移植。

例如：

```
B8 7F 01
BB 21 02
03 D8
B8 1F 04
2B C3
```

1.2.2 汇编语言

为了克服机器语言的上述缺点，人们创造出符号语言（symbolic language），它用一些英文字母和数字表示一个指令，例如用 ADD 代表“加”，用 SUB 代表“减”，MOV 代表“传送”等。

显然，计算机并不能直接识别和执行符号语言的指令，需要用一种称为汇编程序的软件把符号语言的指令转换为机器指令。一般一条符号语言的指令对应转换为一条机器指令。转换的过程称为“代真”或“汇编”，因此，符号语言又称为符号汇编语言（symbolic assembler language）或汇编语言（assembler language）。

汇编语言的特点：相比机器语言简单好记，但仍然难以普及；汇编指令需通过汇编程序转换为机器指令才能被计算机执行；依赖具体机器难以移植。

例如：

```
MOV AX 383
MOV BX 545
ADD BX AX
MOV AX 1055
SUB AX BX
```

1.2.3 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。它很接近于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和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。

当然，计算机也是不能直接识别高级语言程序的，也要进行“翻译”。一种称为编译程序的软件能把用高级语言写的程序〔称为源程序（source program）〕转换为机器指令的程序〔称为目标程序（object program）〕，然后让计算机执行机器指令程序，最后得到结果。高级语言的一个语句往往对应多条机器

指令。

高级语言的特点：功能强大，便于学习、记忆和修改，不依赖于具体机器。

例如：

```
S=1055-(383+545)
```

高级语言经历了不同的发展阶段：

(1) 非结构化的语言。初期的语言属于非结构化的语言，编程风格比较随意，只要符合语法规则即可，没有严格的规范要求，程序中的流程可以随意跳转。人们往往追求程序执行的效率而采用了许多“小技巧”，使程序变得难以阅读和维护。早期的 BASIC，FORTRAN 和 ALGOL 等都属于非结构化的语言。

(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”。规定程序必须由具有良好特性的基本结构（顺序结构，选择结构，循环结构）构成，程序中的流程不允许随意跳转，程序总是由上而下顺序执行各个基本结构。这种程序结构清晰，易于编写、阅读和维护。QBASIC，FORTRAN77 和 C 语言等属于结构化的语言，这些语言的特点是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。在编写规模较小的程序时，还能得心应手，但在处理规模较大的程序时，就显得捉襟见肘、力不从心了。在实践的发展中，人们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#，Visual Basic 和 Java 等语言是支持面向对象程序设计方法的语言。有关面向对象的程序设计方法和面向对象的语言在本书中不作详细介绍，有兴趣的读者可参考有关专门书籍（如谭浩强编著的《C++ 面向对象程序设计（第 3 版）》）。

进行程序设计，必须用到计算机语言，人们根据任务的需要选择合适的语言，编写出程序，然后运行程序得到结果。

1.3 C 语言的发展及其特点

1972—1973 年间，美国贝尔实验室的 D.M.Ritchie 在 B 语言的基础上设计出了 C 语言。

最初的 C 语言只是为描述和实现 UNIX 操作系统而设计的。随着 UNIX 的日益广泛使用，C 语言也迅速得到推广。1978 年以后，C 语言先后移植到大、中、小和微型计算机上。C 语言便很快风靡全世界，成为世界上应用最广泛的程序设计高级语言。

以 UNIX 第 7 版中的 C 语言编译程序为基础，1978 年，Brian W.Kernighan 和 Dennis M.Ritchie 合著了影响深远的名著 *The C Programming Language*，这本书中介绍的 C 语言成为后来广泛使用的 C 语言版本的基础，它是实际上第一个 C 语言标准。

1983 年，美国国家标准协会（ANSI），根据 C 语言问世以来各种版本对 C 语言的发展和扩充，制定了第一个 C 语言标准草案（'83 ANSI C）。

1989 年，ANSI 公布了一个完整的 C 语言标准——ANSI X3.159—1989（常称为 ANSI C 或 C 89）。

1990 年，国际标准化组织 ISO（international standard organization）接受 C 89 作为国际标准 ISO/IEC 9899: 1990，它和 ANSI 的 C 89 基本上是相同的。

1999 年，ISO 又对 C 语言标准进行了修订，形成了标准 C99，在基本保留原来的 C 语言特征的基础上，针对应用的需要，增加了一些功能，尤其是 C++ 中的一些功能，并在 2001 年和 2004 年先后进行了

两次技术修正。

C 语言的主要特点有：

(1) 语言简洁、紧凑，使用方便、灵活。C 语言一共只有 30 多个关键字（见附录 1）、9 种控制语句，程序书写形式自由，压缩了一切不必要的成分。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。

(2) 运算符丰富。C 语言的运算符包含的范围很广泛，共有 34 种运算符（见附录 4）。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 C 语言的运算类型极其丰富，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。

(3) 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、结构体类型和共用体类型等，C99 又扩充了复数浮点类型、超长整型（longlong）和布尔类型（bool）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（如链表、树、栈等）的运算。

(4) 具有结构化的控制语句（如 if...else 语句、while 语句、do...while 语句、switch 语句和 for 语句）。用函数作为程序的模块单位，便于实现程序的模块化。C 语言是完全模块化和结构化的语言。

(5) 语法限制不太严格，程序设计自由度大。例如，对数组下标越界不进行检查，由程序编写者自己保证程序的正确。对变量的类型使用比较灵活，例如，整型量与字符型数据以及逻辑型数据可以通用。一般的高级语言语法检查比较严，能检查出几乎所有的语法错误。而 C 语言为了使编写者有较大的自由度放宽了语法检查。程序员应当仔细检查程序，保证其正确，不要过分依赖 C 语言编译程序查错。“限制”与“灵活”是一对矛盾。限制严格，就失去灵活性；而强调灵活，就必然放松限制。对于不熟练的人员，编写一个正确的 C 语言程序可能会比编写一个其他高级语言程序难一些。也就是说，对用 C 语言的人要求更高一些。

(6) C 语言允许直接访问物理地址。C 语言能进行位（bit）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。

(7) 用 C 语言实现的程序可移植性好。由于 C 的编译系统相当简洁，因此很容易移植到新的系统。而且 C 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 语言写的。因此，几乎在所有的计算机系统中都可以使用 C 语言。

(8) 生成目标代码质量高，程序执行效率高。

C 原来是专门为编写系统软件而设计的，许多大的应用软件也都用 C 语言编写，这是因为 C 语言的可移植性好，硬件控制能力高，表达和运算能力强。许多以前只能用汇编语言处理的问题，后来可以改用 C 语言来处理了。目前 C 的主要用途之一是编写嵌入式系统程序。由于具有上述优点，使 C 语言应用面十分广泛，许多应用软件也用 C 语言编写。

对 C 语言以上的特点，待学完 C 语言以后再回顾一下，就会有比较深的体会。

1.4 C 语言程序的基本结构

1.4.1 简单的 C 语言程序

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。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。

书写程序时应遵循的规则：

- (1) 一个说明或一条语句占一行。
- (2) 用 {} 括起来的部分，通常表示了程序的某一层结构。{} 一般与该结构语句的第一个字母对齐，并单独占一行。
- (3) 缩进风格。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，使程序看起来更加清晰，增加程序的可读性。
- (4) 注释。注释有两种形式：①以 // 开始的单行注释。这种注释可以单独占一行，也可以出现在一行中其他内容的右侧。此种注释的范围从 // 开始，以换行符结束。如果注释内容一行写不下，可以用多个单行注释。②以 /* 开始，以 */ 结束的块式注释。这种注释可以包含多行内容。它可以单独占一行（在行开头以 /* 开始，行末以 */ 结束），也可以包含多行。编译系统在发现一个 /* 后，会开始找注释结束符 */，把二者间的内容作为注释。

在编程时应力求遵循这些规则，以养成良好的编程风格。

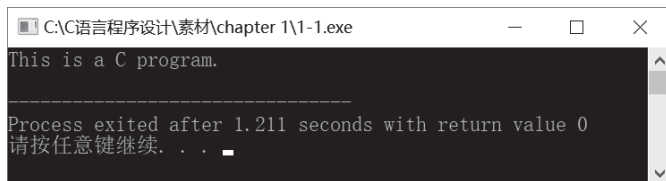
【例 1.1】在屏幕上输出：This is a C program.

解题思路：在主函数中用 printf 函数原样输出以上文字。

程序如下：

```
#include <stdio.h> /* 这是编译预处理指令 */
int main( )           // 定义主函数
{                     // 函数开始的标志
    printf("This is a C program.\n"); // 输出所指定的一行信息
    return 0;         // 函数执行完毕时返回函数值 0
}                     /*main()函数结束*/
```

程序运行结果如下：



程序分析：

- (1) main 是函数的名字，表示“主函数”；每一个 C 语言程序都必须有一个 main () 函数。
- (2) main 前面的 int 表示此函数的类型是 int 类型（整型），即在执行主函数后会得到一个值（即函数值），其值为整型。
- (3) “return 0;”的作用是当 main () 函数执行结束前将整数 0 作为函数值，返回到调用函数处。
- (4) 函数体由花括号 {} 括起来。
- (5) printf () 是 C 编译系统提供的函数库中的输出函数（详见第 3 章）。printf () 函数中双引号内的字符串 "This is a C program." 按原样输出。\\n 是换行符，即在输出 "This is a C program." 后，显示屏上的光标位置移到下一行的开头。
- (6) 每个语句最后都有一个分号，表示语句结束。

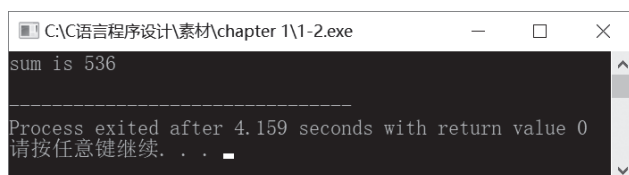
【例 1.2】求 3 个整数之和。

解题思路：设置 3 个变量，a, b, c 用来存放 3 个整数，sum 用来存放和数；用赋值运算符“=”把相加的结果传送给 sum。

程序如下：

```
#include <stdio.h>           // 这是编译预处理指令
int main( )                  // 求 3 个整数之和
{
    int a,b,c,sum;           // 定义变量 a, b, c, sum 为整型变量
    a=123;                   // 以下 3 行是对 3 个变量赋值
    b=456;
    c=-43;
    sum=a+b+c;               // 求 3 个变量的值之和, 放在变量 sum 中
    printf("sum is %d\n", sum); // 输出 sum 的值
    return 0;
}
```

程序运行结果如下：



程序分析：

printf () 函数圆括号内有两个参数。第一个参数是双引号中的内容“sum is %d\n”，它是输出格式字符串，作用是输出用户希望输出的字符和输出的格式。其中 sum is 是用户希望输出的字符，%d 是指定的输出格式，d 表示用“十进制整数”形式输出。圆括号内第二个参数 sum 表示要输出变量 sum 的值。在执行 printf () 函数时，将 sum 变量的值（以十进制整数表示）取代双引号中的 %d。

【例 1.3】输入两个学生的年龄，要求输出其中较大的年龄。

解题思路：从键盘输入两个年龄，用一个函数来实现求两个整数中的较大者；在主函数中调用此函数并输出结果。

程序如下：

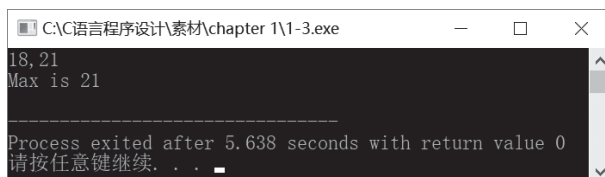
```
#include <stdio.h>
int main( )                  // 主函数
{
    int max(int age_1,int age_2); // 对被调用函数 max 的声明
    int age_1, age_2, age_max;    // 定义整型变量 age_1, age_2, age_max
    scanf("%d,%d", &age_1, &age_2); // 从键盘输入变量 age_1 和 age_2 的值
    age_max=max(age_1, age_2);    // 调用 max 函数, 将得到的值赋给 age_max
    printf("Max is %d\n", age_max); // 输出 age_max 的值
    return 0;
}
// 下面是求两个整数中的较大者的函数
int max(int x, int y)        // 定义 max 函数, 函数值为整型, 形式参数 x 和 y 为整型
{
    int z;                   // 定义本函数中用到的变量 z 为整型
    if(x>y) z=x;              // 如果 x>y, 则将 x 的值赋给变量 z
```

```

else z=y;           // 否则, 将 y 的值赋给变量 z
return(z);          // 将 z 的值返回到主函数中调用 max 函数的位置
}

```

程序运行结果如下:



```

C:\C语言程序设计\素材\chapter 1\1-3.exe
18, 21
Max is 21

-----
Process exited after 5.638 seconds with return value 0
请按任意键继续. . .

```

程序分析:

- (1) 本程序包括两个函数: ①主函数 `main()`; ②被调用的函数 `max()`。
- (2) `max()` 函数的作用是将 `x` 和 `y` 中的大者的值赋给变量 `z`, 最后通过 `return` 语句将 `z` 的值作为 `max()` 的函数值返回给 `main()` 函数中调用 `max` 函数的位置。
- (3) `scanf()` 是输入函数的名字 (`scanf` 和 `printf` 都是 C 的标准输入输出函数)。该 `scanf()` 函数的作用是输入变量 `age_1` 和 `age_2` 的值。其中, `&age_1` 中 `&` 表示取变量的地址。
- (4) `max(age_1, age_2)` 调用 `max()` 函数。在调用时将 `age_1` 和 `age_2` 作为 `max()` 函数的实际参数的值分别传送给 `max()` 函数中的形式参数 `x` 和 `y`。

1.4.2 C 语言程序的结构

通过以上几个程序例子, 可以看到一个 C 语言程序的结构有以下特点:

- (1) 一个程序由一个或多个源程序文件组成。一个规模较小的程序, 往往只包括一个源程序文件, 如例 1.1 和例 1.2 都是一个只有一个函数 (`main()` 函数) 的源程序文件, 例 1.3 中有两个函数, 属于同一个源程序文件。

在一个源程序文件中可以包括 3 个部分:

①预处理指令。如 `#include <stdio.h>` (还有一些其他预处理指令, 如 `#define` 等)。C 编译系统在对源程序进行“翻译”以前, 先由一个预处理器 (也称预处理程序、预编译器) 对预处理指令进行预处理, 对于 `#include <stdio.h>` 指令来说, 就是将 `stdio.h` 头文件的内容读进来, 取代 `#include <stdio.h>`。由预处理得到的结果与程序其他部分一起组成一个完整的、可以用来编译的最后的源程序, 然后由编译程序对该源程序正式进行编译, 才得到目标程序。

②全局声明。即在函数之外进行的数据声明。例如可以把例 1.2 程序中的 “`int a, b, sum;`” 放到 `main()` 函数的前面, 这就是全局声明。在函数外面声明的变量称为全局变量, 如果是在程序开头 (定义函数之前) 声明的变量, 则在整个源程序文件范围内有效。在函数中声明的变量是局部变量, 只在函数范围内有效。

③函数定义。如例 1.1、例 1.2 和例 1.3 中的 `main()` 函数和例 1.3 中的 `max()` 函数。要指定每个函数的功能, 在调用这些函数时, 会完成函数定义中指定的功能。

- (2) C 语言程序主要是由函数构成的, 函数是 C 语言程序的基本单位。

①一个 C 语言源程序必须有且只有一个 `main()` 函数, 可以包含若干个其他函数。`main()` 函数可以调用其他函数, 其他函数可以相互调用, 但其他函数不能调用主函数。

②C 语言的函数库十分丰富, 在程序中既可以调用系统提供的库函数, 也可以调用用户自己编写的函数。

- (3) 一个函数由函数首部和函数体组成, 函数体又包括声明部分和执行部分。

①函数首部。即函数的第 1 行，包括函数名、函数类型、函数属性、函数参数（形式参数）名、参数类型。

②函数体。即函数首部下面的花括号内的部分。如果在一个函数中包括有多层花括号，则最外层的一对花括号是函数体的范围。

函数体一般包括以下两部分。

声明部分。声明部分包括定义在本函数中所用到的变量，如例 1.3 中在 `main()` 函数中定义变量“`int a, b, c;`”；对本函数所调用函数进行声明，如例 1.3 中在 `main()` 函数中对 `max()` 函数的声明“`int max(int x, int y);`”。

执行部分。执行部分由若干个语句组成，指定在函数中所进行的操作。

在某些情况下也可以没有声明部分（例如例 1.1），甚至可以既无声明部分也无执行部分。如：

```
void dump()  
{}
```

是一个空函数，什么也不做，但这是合法的。

(4) 一个 C 语言程序总是从 `main()` 函数开始执行，与 `main()` 函数在程序中的位置无关。

(5) C 语言程序书写格式自由，一行内可以写几个语句，一个语句也可以分写在多行上。

(6) 每个语句和数据声明的最后必须有一个分号。分号是 C 语句的必要组成部分。

(7) C 语言本身没有输入输出语句（一般输入输出操作是通过库函数完成的）。

(8) 程序应当包含注释。

1.5 运行 C 程序的步骤与方法

1.5.1 运行 C 程序的步骤

为了使计算机能按照人的意志进行工作，必须根据问题的要求，编写出相应的程序。所谓程序，就是一组计算机能识别和执行的指令，每一条指令使计算机执行特定的操作。用高级语言编写的程序称为“源程序”，而计算机只能识别和执行由 0 和 1 组成的二进制指令，不能识别和执行用高级语言编写的指令。为了使计算机能执行高级语言源程序，必须先用一种称为“编译程序”的软件，把源程序翻译成二进制形式的“目标程序”，然后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。

编写好 C 源程序后，如何上机运行呢？通常需经过如下几个步骤：

(1) 上机输入与编辑源程序（如 `xx.c` 或 `xx.cpp`）。

(2) 对源程序进行编译，先用 C 编译系统提供的“预处理器”对程序中的预处理指令进行编译预处理，得到目标程序（如 `xx.obj`）。

(3) 将目标程序与库函数连接，得到可执行的目标程序（如 `xx.exe`）。

(4) 运行可执行程序，得到运行结果。

1.5.2 C 语言的版本及运行环境

为了编译、连接和运行 C 程序，必须要有相应的 C 编译系统。目前使用的大多数 C 编译系统都是集成开发环境（integrated development environment, IDE），IDE 把程序的编辑、编译、连接和运行等操作全部集中在一个界面上进行，功能丰富，使用方便。

C 程序编译器很多，目前比较流行的是 DEV-C++，其既可以编译 C++ 程序，又可以编译 C 程序。

1. 进入 DEV-C++ 集成开发环境

单击“开始”→“所有程序”→“DEV-C++”命令，进入启动界面，如图 1-1 所示。

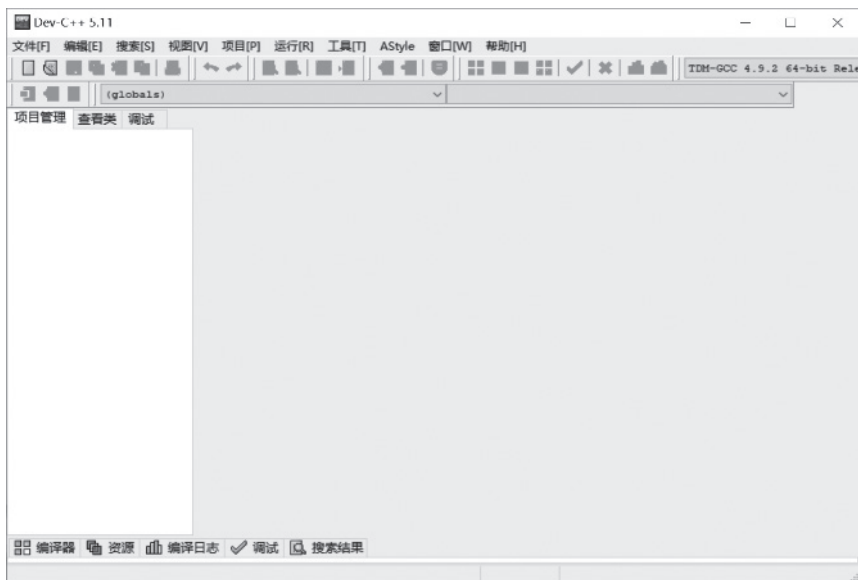


图 1-1 DEV-C++ 启动界面

2. 新建 C/C++ 源程序

单击“文件”→“新建”→“源代码(S)”命令。新建源代码对话框如下图 1-2 所示。

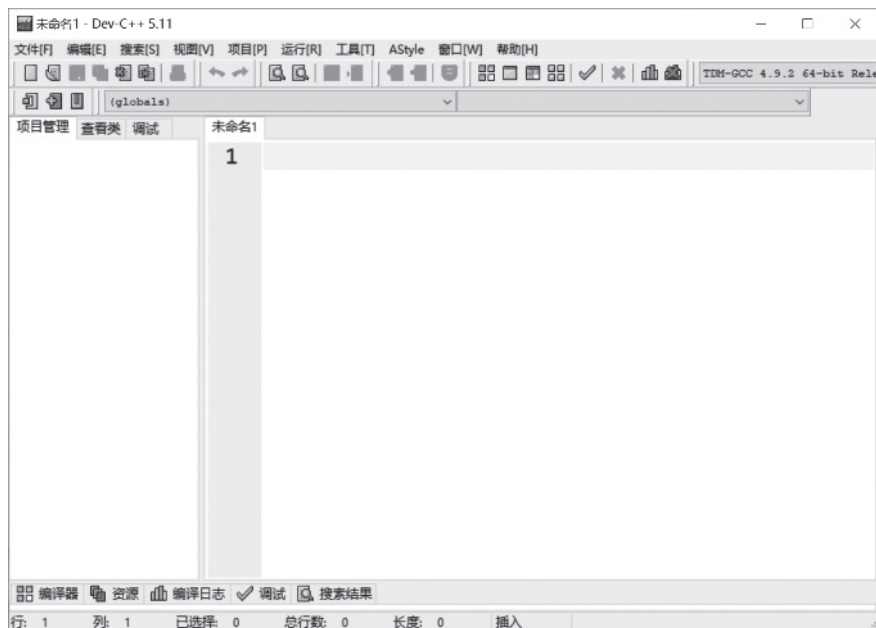


图 1-2 新建源代码对话框

3. 输入源程序

逐行输入源程序代码，关键字之间用一个或多个空格间隔，换行时按【Enter】键。在输入源程序时，还要注意区分大小写字母，如图 1-3 所示。

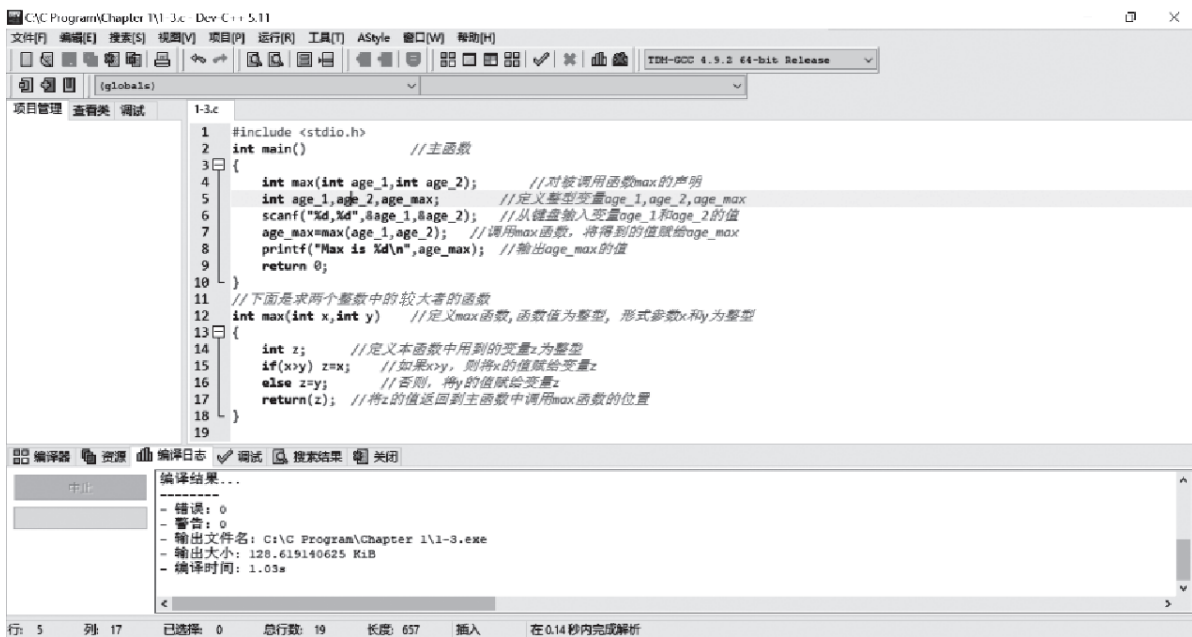


图 1-3 程序编辑界面

输入完成后，经检查无误，可以单击工具栏中的“保存”按钮或“文件”菜单中的“保存”命令保存源程序，保存路径可以自己任意选择。

如果想编辑已保存过的源程序，可以单击工具栏中的“打开”按钮或“文件”菜单中的“打开”命令，打开已经保存过的源程序。进行编辑后，还可以单击“文件”菜单中的“另存为”命令，将其保存为一个新的文件。

4. 编译运行程序

(1) 源程序输入完成以后，单击“运行”菜单中的“编译 [F9]”命令，对源程序进行编译。如有错误，会在程序编辑界面下面的窗格中显示，如图 1-4 所示。错误必须改正，警告（warning）不影响运行。单击向上的滚动条，查看错误原因，对源文件进行修改，然后再次编译。

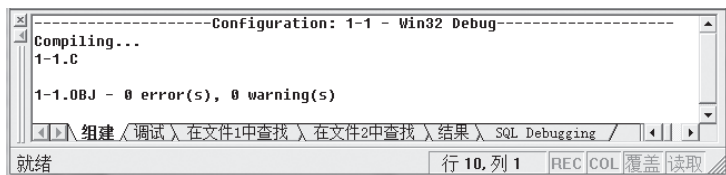


图 1-4 显示编译错误

(2) 源程序编译成功后，单击“运行”菜单中的“运行 [F10]”命令，使程序和系统提供的资源建立连接。

(3) 查看程序运行结果，当提示输入数据时，输入“17, 21”，按【Enter】键，程序运行结果如图 1-5 所示，屏幕最后显示 Press any key to continue, 通知用户“按任何一键以便继续”；如结果不正确，则需要调试程序。

5. 关闭工作空间

如果已完成对一个程序的操作，不需要再进行处理，可单击“文件”菜单中的“关闭工作空间”命令，结束对该程序的操作。

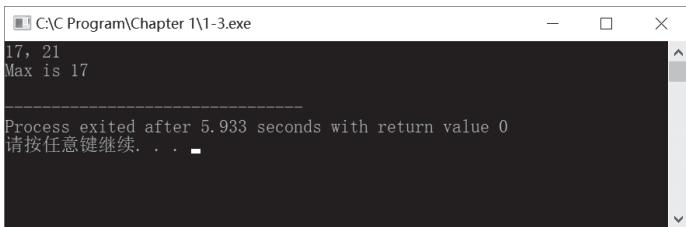


图 1-5 程序运行界面

1.6 程序设计的任务

如果只是编写和运行一个很简单的程序，上面介绍的步骤就够了。但是实际上要处理的问题比上面见到的例子复杂得多，需要考虑和处理的问题也复杂得多。程序设计是指从确定任务到得到结果、写出文档的全过程。

从确定问题到最后完成任务，一般要经历以下几个工作阶段：

(1) 问题分析。对于接手的任务要进行认真的分析，研究所给定的条件，分析最后应达到的目标，找出解决问题的规律，选择解题的方法。在此过程中可以忽略一些次要的因素，使问题抽象化，例如用数学式子表示问题的内在特性。

(2) 设计算法。即设计出解题的方法和具体步骤。例如要解个方程式，就要选择用什么方法求解，并且把求解的每一个步骤清晰无误地写出来。一般用流程图来表示解题的步骤。

(3) 编写程序。根据得到的算法，用一种高级语言编写出源程序。

(4) 对源程序进行编辑、编译和连接，得到可执行程序。

(5) 运行程序，分析结果。运行可执行程序，得到运行结果。能得到运行结果并不意味着程序正确，要对结果进行分析，看它是否合理。例如把“b-a;”错写为“a=b;”，程序不存在语法错误，能通过编译，但运行结果显然与预期不符，因此要对程序进行调试(debug)。调试的过程就是通过上机发现和排除程序中故障的过程，经过调试，得到了正确的结果。但是工作不应到此结束，不要只看到某一次结果是正确的，就认为程序没有问题。例如，求 $c=b/a$ ，当 $a=4$ ， $b=2$ 时，求出 c 的值为 0.5，是正确的，但是当 $a=0$ ， $b=2$ 时，就无法求出 c 的值。说明程序对某些数据能得到正确结果，对另外些数据却得不到正确结果，程序还有漏洞，因此，还要对程序进行测试(test)。所谓测试，就是设计多组测试数据，检查程序对不同数据的运行情况，从中尽量发现程序中存在的漏洞，并修改程序，使之能适用于各种情况。作为正式产品使用的程序，是必须经过严格测试的。

(6) 编写程序文档。许多程序是提供给别人使用的，如同正式的产品应当提供产品说明书一样，正式提供给用户使用的程序，必须向用户提供程序说明书(也称为程序文档)。程序文档内容应包括程序名称、程序功能、运行环境、程序的装入和启动、需要输入的数据，以及使用注意事项等。



案例分析与实现

1. 案例分析

以下两个程序主要涉及 C 语言的程序结构。通过这两个程序，掌握以下知识：

- (1) C 语言程序的基本结构。
- (2) 输入/输出函数。
- (3) C 语言的书写特点。

2. 案例实现过程

程序 1:

```
#include <stdio.h>
int main()
{
    printf("这是我的第一个 C 程序\n");
}
```

程序 2:

```
#include <stdio.h>
int main( )
{
    int score;
    printf(" 请输入考试成绩: \n");
    scanf("%d", &score);
    if(score>=60)
        printf(" 及格 \n");
    else
        printf(" 不及格 \n");
}
```

3. 案例执行结果

程序 1 运行结果如图 1-6 所示；程序 2 运行结果如图 1-7 所示。

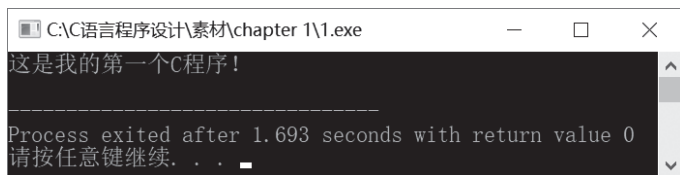


图 1-6 程序 1 运行结果

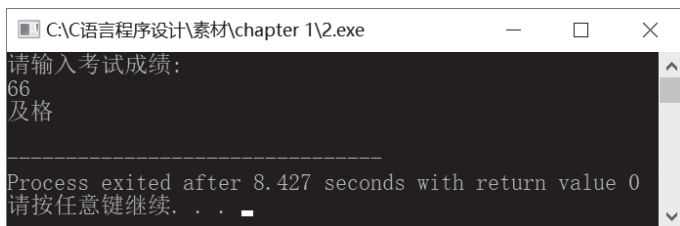


图 1-7 程序 2 运行结果

小 结

本章所介绍的主要内容是 C 语言的程序设计基础知识。虽然本章的内容在初学者看起来比较繁杂，学起来比较枯燥，但本章的内容是学好 C 语言的基础，是每个 C 语言程序员必须熟练掌握的。现在我们一起回忆一下本章中有哪些内容值得我们特别留意和必须深刻领会。

1. 计算机是由程序控制的，要使计算机按照人们的意图工作，必须用计算机语言编写程序。
2. 机器语言和汇编语言依赖于具体计算机，属于低级语言，难学难用，无通用性。高级语言接近人类自然语言和数学语言，容易学习和推广，不依赖于具体计算机，通用性强。
3. C 语言是一种使用广泛的计算机语言，语言简洁紧凑，使用方便灵活，功能很强，既有高级语言的优点，又具有低级语言的功能，既可用于编写系统软件，又可用于编写应用软件。掌握 C 语言程序设计是程序设计人员的一项基本功。
4. 一个 C 语言程序是由一个或多个函数构成的，必须有一个 main 函数。程序由 main 函数开始执行。在函数体内可以包括若干语句，语句以分号结束。一行内可以写多个语句，一个语句可以分写为

5. 上机运行一个 C 程序必须经过 4 个步骤：编辑、编译、连接、执行。要熟练掌握上机技巧。

6. 程序设计的任务应当包括: ①问题分析; ②设计算法; ③编写程序; ④对源程序进行编辑、编译和连接; ⑤运行程序、分析结果; ⑥编写程序文档。

习 题

一、选择题。

1. C 语言中主函数的个数为 () 个。

- A. 1 B. 2 C. 无穷个 D. 任意个

2. 以下关于 C 语言描述错误的是 ()。

- A. 一个 C 程序总是从 `main` 函数开始执行
- B. 每个语句和数据声明的最后必须有一个分号
- C. C 语言的块式注释是以 `/*` 开始并以 `*/` 结束的
- D. 一个 C 程序可以包含多个 `main` 函数

3. C 语言源程序文件后缀为 ()。

- A. .EXE B. .OBJ C. .c D. .ASM

4. C语言是由()组成的。

- A. 子程序 B. 主程序与子程序 C. 函数 D. 过程

5. C 语言属于 () 语言。

- A. 机器语言 B. 汇编语言 C. 高级语言 D. 面向对象语言

二、编程题。

1. 编写程序，输出“Hello World!”。

2. 编写程序，将个人的基本信息（姓名、性别、籍贯、住址）打印到控制台上输出。各条信息分别输出一行。

3. 编写程序, 输入两个数, 比较并输出其中较小的那个数。