普通高等教育“十二五”规划教材

21世纪面向计算思维丛书

卫春芳 张 威 主编

C语言程序设计

C YUYAN CHENGXU SHEJI

科学出版社

普通高等教育“十二五”规划教材
21世纪面向计算思维丛书

C语言程序设计

卫春芳　张　威　主编

科学出版社
北京

内 容 简 介

本书全面、系统地介绍C语言的基本概念、基本语法、数据类型、程序结构及计算机高级语言程序设计的方法和常规算法。本书既考虑到国家计算机二级考试大纲主要内容，又结合具体的程序设计综合要求。全书运用计算思维的方法设计程序，以程序案例为导向，拓宽学生的思维，引导学生自主思考，逐步掌握程序设计的一般规律和方法。本书内容深入浅出，配合典型例证，通俗易懂，实用性强，蕴含了作者丰富的教学经验和编程心得。

全书共12章。第1章～第3章介绍程序设计与C语言的基础知识；第4章～第5章介绍分支、循环两种程序设计结构及常用的基本算法；第6章～第9章介绍数组、函数、指针和字符串，由浅入深地介绍C语言的语法，并通过经典算法示例来逐步讲解程序设计；第10章～第12章主要介绍编译预处理和动态分配、结构体、文件等介绍。

本书可作为普通本科高等学校、普通高等专科学校的计算机教材，也可作为计算机培训和计算机等级考试辅导教学用书，还可作为科技人员或程序开发人员的参考用书。

图书在版编目(CIP)数据

C语言程序设计/卫春芳，张威主编. —北京：科学出版社，2016.2

(21世纪面向计算思维丛书)

普通高等教育“十二五”规划教材

ISBN 978-7-03-047303-5

Ⅰ.①C… Ⅱ.①卫… ②张… Ⅲ.①C语言-程序设计-高等学校-教材

Ⅳ.①TP312

中国版本图书馆CIP数据核字(2016)第021073号

责任编辑：王雨舸/责任校对：董艳辉

责任印制：彭 超/封面设计：蓝 正

科学出版社 出版

北京东黄城根北街16号

邮政编码：100717

http://www.sciencep.com

武汉市新华印刷有限责任公司印刷

科学出版社发行 各地新华书店经销

*

开本：787×1092 1/16

2016年2月第 一 版 印张：21 1/4

2016年2月第一次印刷 字数：490 500

定价：39.90元

(如有印装质量问题，我社负责调换)

前　言

C语言作为一门最通用的语言，在过去很流行，将来依然会如此，它是许多计算机专业和非计算机专业学生学习程序设计语言的首选。

本书全面、系统地介绍C语言的基本概念、基本语法、数据类型、程序结构及高级语言程序设计的方法和常规算法，既考虑到国家计算机二级考试大纲主要内容，又结合具体的程序设计综合要求。本书根据初学者的特点，由浅入深，循序渐进，旨在帮助学生掌握C语言程序设计的基本方法，理解领会C语言的特点和本质，提高学生运用C解决实际问题的综合能力。各章中精选有配合相关知识点的相应案例程序，每个程序都给出完整的注释、运行结果和分析说明，案例程序由浅入深，强化了知识点、算法、编程方法与技巧。全书运用计算思维的方法设计程序，以程序案例为导向，拓宽学生的思维，引导学生自主思考，逐步掌握程序设计的一般规律和方法。

本书内容深入浅出，配合典型案例，通俗易懂，实用性强，蕴含作者丰富的教学经验和编程心得。本书以C99标准为主线，示例程序都可在Visual C++ 6.0环境下编译运行，每一章后面均附有习题。

全书共12章。第1章～第3章介绍程序设计与C语言的基础知识；第4章～第5章介绍分支、循环两种程序设计结构及常用的基本算法；第6章～第9章介绍数组、函数、指针和字符串，由浅入深地介绍C语言的语法，并通过经典算法示例来逐步讲解程序设计；第10章～第12章主要介绍编译预处理和动态分配、结构体、文件等。

本书由卫春芳、张威任主编，孙军、曹芝兰任副主编。第3、4、8、12章由卫春芳编写，第1、2、7章由张威编写，第9、10、11章由孙军编写，第5、6章由曹芝兰编写，最后由卫春芳、张威统稿、定稿。

因时间仓促，加之编者水平有限，书中难免存在不妥之处，恳请广大读者批评指正。

编　者

2016年1月

目　录

第 1 章　程序设计基础

1.1　程序和程序设计语言

1.1.1　程序

自从电子计算机发明以来，人类社会发生了翻天覆地的变化。计算机似乎是万能的，它以不可思议的方式改变着人们的工作、生活和娱乐。每个人都需要掌握好计算机以更好地融入社会，然而计算机为什么如此神通广大，这是我们每个人心中的疑问。

计算机的“万能”实际上依赖于其上运行的五花八门、各种各样的程序。计算机的核心——CPU 可以执行各种基本指令，程序实际上就是这些指令序列的集合。计算机在微观上执行的是最基本的指令操作(算术运算和逻辑运算)，当运行一个程序时，计算机就会自动地执行程序中的指令序列，从而实现更宏观的功能。

1.1.2　程序设计语言

计算机程序是由程序员编写实现的，然而计算机的工作方式决定了程序员不可能用人类语言对计算机发号施令。这样，程序设计语言就被发明出来了，程序员通过它来编写程序，它是程序员和计算机进行交流的桥梁，所以又被形象地称为计算机语言。

跟人类语言类似，程序设计语言借助于各种字符符号和相关的语法来构成实现，学习程序设计语言无非就是掌握读和写的能力。然而程序设计语言的语法要求十分严谨，不允许有丝毫的差错，一点错误都将导致程序不能运行。

对于非计算机专业人士，是否有学习程序设计语言的必要呢？其实，通过学习程序设计语言可以更好地理解计算机的思维和工作方式。举例来说，鸡兔同笼问题：鸡兔共 35 只，共有 94 只脚，问各有多少只鸡和兔？

大部分人会这样解：设有 x 只鸡，则有$(35-x)$只兔，求解一元一次方程

$$2x+4(35-x)=94$$

就能得到问题的结果。然而计算机倾向于这样解决问题：列举鸡兔 35 只的所有组合数，计算哪种组合的脚的数量是 94 只。

由于人类大脑的思维方式是归纳和演绎，所以人更倾向于用公式和定理来解决问题，而计算机是电路思维，特点是速度快，只借助基本的算术运算和逻辑运算也能解决问题。

就像数学之于自然科学的基础地位，程序设计语言对于计算机科学来说也是非常基础和重要的。美国总统奥巴马在谈到美国的教育时曾说“所有人都应更早地学习如何编程”。而在英国，中学生必须学习两个或两个以上的编程语言以及相关的科目。为了更好

地掌握计算机这个重要的工具，学习程序设计语言是极其必要的。

1.1.3 程序设计语言的发展历史

程序设计语言种类繁多，自出现以来，已有上千种的程序设计语言被发明出来，其发展大致经历了以下三个阶段。

1. 机器语言

计算机 CPU 执行的是指令，这些指令以二进制形式存储在存储器中。一个指令通常由操作码和地址码组成，其中操作码指明了指令的操作性质及功能，地址码则给出了操作数或操作数的地址。这样一条指令就表示为一串二进制，例如，将地址 16 的存储单元中的数据存储到编号 1 的寄存器可能表现为如下形式：

0001000100000010000

这种二进制代码是计算机可以直接识别和执行的，被称为机器指令。机器指令的集合就是该计算机的机器语言。

机器语言的好处是速度快，资源占用少。但其缺点也很明显：编程人员需要熟记所用计算机的全部指令代码及其含义；编程序时，需要处理每条指令和每条数据的存储分配和输入输出，记住各种工作单元的状态；编出的程序全是 0 和 1 的指令代码，不直观，易出错；最后，由于不同型号计算机的机器语言是有差别的，机器语言编写的程序通用性弱，可移植性差。所以，除了计算机生产厂家的专业人员外，一般程序员已经不再去学习机器语言了。

2. 汇编语言

机器语言难以记忆，人们想到用字符符号来帮助记忆和理解指令，从而产生了汇编语言。汇编语言也叫符号语言，它用助记符代替机器指令的操作码，用地址符号或标号代替指令或操作数的地址。例如，用 ADD 表示“加”，SUB 表示“减”，AND 表示“与”，MOV 表示“传送”等。

举个例子，要将寄存器 BX 的内容送到寄存器 AX 中，对应的机器指令和汇编指令分别如下：

```
1000100111011000        机器指令
mov ax,bx               汇编指令
```

汇编语言相较于机器语言，方便记忆和学习，但其仍是在硬件层次上进行操作的语言，因此增加了编程的复杂性，专业性较强，编写的程序不够直观。机器语言和汇编语言都需要根据特定的计算机来设计应用，是面向机器的语言，也称为低级语言。

计算机能直接识别和执行的是二进制形式的机器指令，汇编指令由于使用了字符和符号，不能被计算机直接执行。因此，汇编语言书写的源程序需要翻译成二进制形式的目标程序才能执行，这个过程称为编译，实现编译处理的程序称为编译器。

学习汇编语言需要较强的专业知识，学习难度较高。但由于它面向机器，是计算机底

层设计程序员必须了解的语言，通常应用在编写驱动程序、嵌入式操作系统和实时运行程序等场合。

3. 高级语言

与面向机器的低级语言相比，人们更需要一种面向用户的高级语言。高级语言使用接近自然语言和数学语言的方式来书写程序，更加符合人们的思维习惯，方便理解，阅读简单。例如，有如下部分程序代码：

```
s=pi*r*r;
printf("area=%d\n",s);
```

相信大部分人都能猜出这段代码的功能：求圆的面积并打印。而且这段高级语言的代码没有表现出与底层硬件的相关性，因此高级语言是不依赖于具体机器的，可以在不同的计算机平台上通用（或少量修改）。

显然，高级语言书写的程序也不能被计算机直接执行，因为它不是二进制形式的，代码中包含各种字符和符号，也需要进行“翻译”。与汇编语言类似，高级语言也需要借助编译程序（编译器）来将高级语言书写的源程序转换为机器语言形式的目标程序，然后才能被计算机执行。高级语言的一条语句通常会对应成多条机器指令，编写各种高级语言程序需要使用各自语言的编译器软件。

自从有了高级语言，编程就不再是少数专业人士独享的技能。高级语言的学习者不需要深入理解计算机的内部结构和工作原理，学习编程更多是对逻辑思维的训练和表达。任何人都可以通过掌握高级语言来为自己的工作、学习及研究服务。

1.2　C语言的发展及特点

C语言的发明人是肯·汤普森（Ken Thompson）和丹尼斯·里奇（Dennis M. Ritchie），两人也是鼎鼎大名的UNIX系统的发明人。肯·汤普逊和丹尼斯·里奇因为“研究发展了通用的操作系统理论，尤其是实现了UNIX操作系统”一起获得了1983年的图灵奖。C语言是为实现UNIX操作系统而设计的一种工作语言，也可以说C语言是UNIX操作系统的“副产品”。

1970年，美国贝尔实验室的肯·汤普森以BCPL语言为基础，设计出很简单且很接近硬件的B语言。第一版的UNIX就是基于B语言来开发的。B语言在进行系统编程时不够强大，所以肯·汤普森和丹尼斯·里奇对其进行了改造，并于1971年共同发明了C语言。

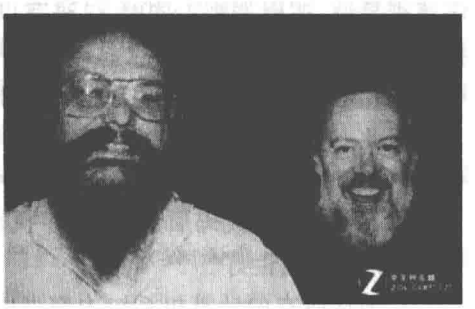

图1-1　肯·汤普森（左）和丹尼斯·里奇（右）

出于对UNIX系统移植到其他类型计算机的需要，C语言有很好的可移植性，可以使用在任意架构的处理器上，只要那种架构的处理器具有对应的C语言编译器和库，然后将C源代码

编译、连接成目标二进制文件之后即可运行。

C语言诞生至今仍被广泛使用，为了使C语言健康地发展，人们成立了C标准委员会，建立C语言的标准。1989年，ANSI(American National Standards Institute，美国国家标准学会)发布了第一个完整的C语言标准，简称“C89”，也称为“ANSI C”。1999年，在做了一些必要的修正和完善后，ISO(International Organization for Standardization，国际标准化组织)发布了新的C语言标准，简称“C99”。在2011年12月8日，ISO又正式发布了新的标准，简称为“C11”。

C89是目前使用最广泛的标准，得到了几乎所有主流编译器的支持，所以本书对C语言语法的介绍主要以C89标准为主。

C语言的诞生是现代程序语言革命的起点，是程序设计语言发展史中的一个里程碑。自C语言出现后，在C语言基础上扩充及衍生出的C++、Java和C#等面向对象语言相继诞生，并在不同领域获得极大成功。至今，C语言仍旧在系统编程、嵌入式编程等领域占据着统治地位。

作为一门高级语言，C语言有着如下的优点：

(1) 简洁紧凑、方便灵活。

C语言一共只有32个关键字，9种控制语句，程序书写形式自由，主要使用小写字母表示，压缩了许多不必要的成分，相比较其他计算机语言，源程序书写起来较短，因而减少了输入工作量。

(2) 运算符丰富。

运算符是程序设计语言中最基本的功能单位。C语言的运算符包含的范围很广泛，共有34种运算符。C语言的运算类型极其丰富，表达式类型多样化。因此，灵活使用这些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语言的数据类型有：整型、实型、字符型、数组类型、指针类型、结构体类型、共用体类型等。数据类型多意味着数据表达能力强，可以实现各种复杂的数据结构并进行运算。尤其是C语言中引入了指针概念，使得程序的效率更高。

(4) C语言是结构化语言。

结构化语言只使用顺序、选择和循环三种控制结构，采用结构化的编程方式可以使程序清晰易读、逻辑严密。同时，C语言以函数为程序的模块单位，便于实现程序的模块化，提高代码利用率。

(5) 允许直接访问物理地址，可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，能够像汇编语言一样对位(bit)、字节和地址进行操作，可以进行系统软件的编写。

(6) 生成目标代码质量高，程序执行效率高。

(7) 可移植性好。

几乎在所有的计算机系统中都可以使用C语言。

C语言除了以上的优点外，也有一些比较明显的缺点。由于C语言使用灵活，设计自由度高，就导致其语法限制不太严格，从而影响了程序的安全性，例如，对数组下标越界不

作检查等。同时，使用灵活也导致C语言比其他高级语言更难掌握。

1.3　C和C++

早期的计算机编程采用的是面向过程的方法，通过设计一个算法就可以解决当时的问题。C语言采用的结构化编程方法就是面向过程的，遵从自顶向下的原则。随着计算机应用水平的提高，计算机被用于解决越来越复杂的问题，这时面向过程的方法就显得力不从心了，面向对象的方法应运而生。C++是在C语言的基础上开发的一种通用编程语言，它进一步扩充和完善了C语言，是一种面向对象的程序设计语言。

可以认为，C是C++的子集。C++中的过程化控制及其他相关功能与C是基本相同且兼容的。此外，C++拓展了面向对象设计的内容，这就是面向过程的C所没有的了。所以，在早期，C++也被称为“C with Class”。从C到C++，从面向过程到面向对象，这体现了随着计算机解决的问题规模变化编程思想的演变。

1. 面向过程

C语言是结构化语言，其重点在于算法与数据结构。C程序的设计主要考虑如何通过一个过程，对输入进行运算处理从而得到输出，这其实是很实际的一种思考方式。若问题规模比较大，就采用自顶向下、逐步求精的设计方法，将问题逐层细化，分成不同的模块解决。例如，做菜一般有三个步骤：

(1) 准备材料若干。

(2) 按照一定的流程对材料进行操作。

(3) 完成后起锅装盘。

在这个过程中，材料就相当于程序要处理的数据，操作流程就相当于解题的算法。因此可以很形象地总结一个编程的公式

程序＝数据＋算法

2. 面向对象

随着计算机解决问题的规模和复杂度提升，面向过程的方法就无法应付自如了。从现实中进行观察，当问题规模变大后，人们就会进行职能的细分，从而产生出各种属性不一的对象，管理的着眼点也从细枝末节提升到对每个岗位的控制。面向对象的方法主要考虑的是如何构造一个对象模型，让这个模型能够契合与之对应的问题域，这样就可以通过获取对象的状态信息得到输出或实现过程控制。

同样以做菜为例，现在要做很多的菜，要开饭店，做菜的规模变大后，就会采用如下的方法：

(1) 按照功能需求设置各种不同的工种岗位，例如：行政总厨、厨师长、主厨、厨师、打荷、配菜、墩工、厨房杂工等。

(2) 明确各个岗位的职能。

行政总厨：负责厨房的生产、管理工作，为饭店管理部门及时提供各种有关信息。

厨师长:负责人员安排和日常工作。

主厨:大师傅,负责走大菜。

厨师:负责走平常菜。

打荷:负责给主厨和厨师端碟子和辅助工作。

配菜:把原材料准备好,并按步骤装在不同的码兜里。

墩工:将各种材料准备成半成品。

厨房杂工:打扫收拾,洗碗。

(3) 每个岗位都在自己的职能范围内行事,向上级负责,差遣下级。

(4) 厨房的整个工作就在这些不同岗位的人的互动中实现完成。

在这里,每一个工种就是一个对象,工种的职能就属于对象的属性,解决问题就是抽象出这些对象,并通过这些对象的互动实现问题的求解。面向对象的编程也可以总结一个很形象的公式

程序=对象+对象+…+对象

面向过程和面向对象并不是截然对立的两种方法,在面向对象的设计中也会用到面向过程,因为面向过程是最基础的设计方法,每个对象的属性实现都是用面向过程的方法来实现的。随着编程的经验逐渐积累,会对这两种程序设计的方法有更深入的体会和理解。

从学习的角度来说,C++包含C,C比较基础,而C++更先进,内容也更多。学好C语言会对C++的学习有很好的促进作用。

1.4 简单的C程序介绍

为了学习C语言的语法,有必要先了解一下C语言程序的一般写法,下面通过几个例题来了解C语言的程序。

例 1.1 在屏幕上输出一句话。

【程序代码】

```
#include <stdio.h>                  //预编译文件包含指令
int main( )                         //主函数首部
{                                   //函数体开始标志
    printf("Hello everyone! \n");   //输出函数输出指定字符信息
    return 0;                       //函数返回值0
}                                   //函数体结束标志
```

【运行结果】

```
Hello everyone!
Press any key to continue
```

【程序说明】

(1) 函数是C程序的基本组成单位。

在C程序中,几乎所有的代码都写在函数中,以函数的形式将代码封装起来可以实

现代码的重复利用，并可使程序的结构更清晰。函数要定义后才可使用，定义函数一般分为函数首部和函数体两部分。在本例中，main( )即为一个函数，观察这个函数的组成。

函数首部：

```
int          main          ( )
 ↓            ↓             ↓
函数类型     函数名      函数参数
```

int 表示该函数返回一个整数；main 为该函数的名称，即主函数；( )里是函数参数，本函数没有参数，所以空着不写。

函数体：

```
{                                       //函数体以{开始
    printf("Hello everyone! \n");       //输出函数输出指定字符信息
    return 0;                           //函数返回值 0
}                                       //函数体以}结束
```

接着函数首部后面，以{ }括起来的代码就是函数体，调用一个函数就是执行该函数的函数体。这里，main 函数的函数体中调用了另外一个函数 printf( )，不同的函数之间是可以相互调用的。由于 main 函数需要返回一个整数，return 0 表示返回 0 值。执行完 return 0，main 函数就执行结束了并返回。

main 函数比较特殊，C 程序的执行就是从 main 函数开始执行，main 函数结束整个程序就结束。所以，main 函数是程序的入口，每个程序都必须有且仅有一个 main 函数。

(2) printf( )也是一个函数，执行该函数就是执行其函数体代码，该函数同样应该先定义才能被调用。但在本程序中并未看到 printf 函数的定义语句，因为 printf 函数是一个系统提供的标准库函数，已经被定义好了可以直接使用，只需把 stdio. h 这个文件包含进来就可以使用了。

(3) #include <stdio. h>的意思就是将文件 stdio. h 包含到源文件里，在编译时将该文件里的代码替换到这句话的位置，这样后面就可以使用 stdio. h 文件里面定义或声明的函数了。

stdio 即“标准输入输出”，所以不仅 printf 函数，其他系统提供的输入输出库函数也可以通过包含 stdio. h 文件来使用。输入输出操作是每个程序最基本的要素，无法想象一个程序会没有输出。可以认为每个 C 程序都至少有一条 printf 函数构成的语句，这样 #include <stdio. h>也就成为每个程序必有的第 1 句话。

(4) printf 函数的用法。

printf( )中括号用来填写函数的参数，在这里只有一个参数，即“Hello everyone! \n”，它是一个字符串常量，两边的双引号是用来表示字符串常量的界限符，并不是字符串的一部分。printf 函数可以有更多的参数，用法比较多，此处只是输出一串字符，故只有一个字符串常量参数。

需要注意的是，printf( )的输出结果是显示在命令行界面的，不同于在文本编辑软件中打字，其格式控制(例如换行)也是通过代码字符实现的，此处字符串结尾的'\n'表示的就是换行。所以，如果要实现多行字符的打印，就需要在每个换行的位置添加'\n'。当有

多句 printf( )时，其结果也是严格按照参数中的代码字符的含义控制显示输出，不要想当然的认为会以多行显示。例如：

```
printf("Hello");
printf("everyone! ");
printf("\n");
```

使用这三句的效果与使用一句：

```
printf("Hello everyone! \n");
```

的效果是等同的。

(5) 在输出的最后还有一句“press any key to continue”，对照程序可以发现这句话不是程序中代码的功能，它是编译器软件在程序运行结束时自动停顿的一个提示，按任意键将关闭输出窗口，其作用是为了便于观察程序的输出结果。

(6) 每行代码后面以//开头的一句话，它是行注释的标记符号，可以用来在每一行结束的位置添加说明性文字。注释不属于代码的一部分，它仅仅是为了增强程序的可读性，方便理解程序的功能。

语法小结

通过对例 1.1 的分析，可以总结一个写 C 程序的基本套路，即：

```
#include <stdio.h>
int main( )
{
    ……
    return 0;
}
```

这是所有 C 程序都有的固定写法，在{ }中的省略号部分填写具体的功能代码。

例 1.2 求两个整数之和。

【程序代码】

```
#include <stdio.h>
int main( )
{
    int a,b,sum;                    //定义 3 个整型变量 a,b,sum
    a=5;                            //对变量 a 赋值
    b=9;                            //对变量 b 赋值
    sum=a+b;                        //计算 a+b,将结果赋值给变量 sum
    printf("sum=%d\n",sum);         //输出变量 sum 的值
    return 0;
}
```

【运行结果】

```
sum=14
Press any key to continue
```

【程序说明】

(1) 在本例中，出现了变量的简单用法，所谓变量是其值可以变化的量。我们知道计算机中的数据都存放在内存中，所有变量的值也都存放于内存之中。因而变量实质就是内存中的一块存储单元，变量名与这块内存单元做了映射，对变量赋值即是将数据写入对应的存储单元。

用来表示变量的内存单元要通过变量定义的方式分配出来，所以我们看到了这样一句：

```
int a,b,sum;
```

其中 int 表示整型类型，不同的数据类型用不同的字符表示，数据的类型决定了存储空间的大小及数据存储的格式。在 int 后接着写变量名就是定义了变量，多个变量可以同时定义，变量名之间以逗号分隔。

注意：定义多个变量时的分隔符不要写成分号，分号是 C 程序一条语句结束的标志，即每条语句以分号结尾。此处是一条定义语句同时定义 3 个变量，变量之间的分隔符若写成分号就变成了多条语句而非一条语句。

(2) 定义好变量后，就可以对变量使用运算符进行运算操作。此处的加法和赋值运算符与数学上的含义相同，读起来非常容易，使用数学语言表达是高级语言的一个优点。需要注意的是变量一定要先定义才能使用，直接使用而不定义是错误的。

(3) 本例中的 printf 函数除了输出字符串外还要输出变量的值。函数的参数也是以逗号分隔的，第二个参数 sum 是变量名，它与前面第一个参数(双引号括起来的 sum=%d\n)中的%d 对应，sum 变量的值将出现在%d 的位置，如下所示：

```
printf("sum=%d/n",sum);
```

sum 的值将替换%d 出现在字符串中

%d 叫格式字符，用来规定 printf 函数后面参数中的变量(或表达式)以指定格式输出显示，%d 对应的是十进制整数格式。C 语言中的数据类型很多，不同的类型会使用不同的格式字符。

语法小结

变量必须先定义再使用，输出变量的值需要搭配格式字符来使用，格式字符和变量的类型要相符。

例 1.3　求两个整数的最大值。

【程序代码】

```
#include <stdio.h>
int main( )
{
    int max(int x,int y);          //对 max 函数的声明
    int a,b,c;                     //定义 3 个整型变量 a,b,c
    scanf("%d%d",&a,&b);           //输入变量 a,b 的值
    c=max(a,b);                    //调用 max 函数求得 a,b 最大值赋值给 c
```

```
    printf("最大值是%d\n",c);   //输出变量 c 的值
    return 0;
}
/*************************
  求两整数最大值函数
  输入:两个整数
  输出:两个整数的最大值
  *************************/
int max(int x,int y) //max 函数的首部,函数返回整型值,有两个整型形参 x 和 y
{
    int z;           //定义变量 z 用来存放最大值
    if(x>y) z=x;     //如果 x>y,将 x 赋值给 z
    else  z=y;       //反之(即 x<=y),将 y 赋值给 z
    return z;        //将 z 的值作为函数值返回到调用 max 函数的位置
}
```

【运行结果】

```
7 10
最大值是10
Press any key to continue_
```

【程序说明】

(1) 在 max 函数前出现的一段文字是块注释,块注释以/*开始,以*/结束,可以是多行的注释文字。块注释单独书写,不能和代码混杂一块,它和行注释一样不属于代码的一部分。

(2) 本例中增加了变量的输入操作,使用的是 scanf 函数。scanf 函数和 printf 函数一样是标准的库函数,只要包含了头文件 stdio.h 就可以直接使用。变量的输入同样需要格式字符搭配使用,若同时有多个变量输入,格式字符和变量之间按顺序一一对应。在输入函数中,字符串参数的后面是变量的地址,即在每个变量名加一个 & 符号,不能省略。

```
scanf("%d%d",&a,&b);
```

以整型格式接收变量 a 的值

当程序执行到 scanf 函数时会停下来,这时需要用户在屏幕输入变量的值,当有多个变量值需要输入时,每个值之间以空格间隔,输入完成后按回车键结束。

(3) 本例中的 max 函数是自定义函数,需要按照函数的定义格式分函数首部和函数体将代码完整写出。max 函数的最大值功能用到了选择结构 if…else…语句,这里先不详细解释,可先根据注释理解其工作流程。

对 max 函数的调用出现在 main 函数中,即调用语句在函数定义前面,为保证调用语句的书写正确需要知道函数的首部信息。main 函数中的第 1 句是 max 函数首部构成的单独一句,有了这句话,后面的 max 调用语句就能检查其正确与否。这样的由函数首部