

21

高等学校“十二五”规划教材

# C语言程序设计

◎主编 崔永君 彭静

CYUYANCHENG  
YUSHIJI



西安电子科技大学出版社  
<http://www.xduph.com>

## 内 容 简 介

计算机程序设计基础是高等学校计算机基础课程中的核心课程。本书总结了作者多年的教学和软件开发经验,以C程序设计语言为基础,注重程序设计与软件开发的基本概念、方法和思路,培养读者的基本编程能力、逻辑思维和抽象思维能力。

本书根据初学者的认知特点,循序渐进、紧贴教学、深入浅出地讲解了C语言的基本概念、数据类型、结构化程序设计的三种结构(顺序结构、分支结构、循环结构)、数组、函数、指针、结构体与共用体、编译预处理与位运算以及文件等相关知识。通过书中大量有着明确知识点的例题与习题,读者可体会并初步掌握程序的设计方法。

本书可作为高等院校计算机及理工类各专业、成人教育学院C语言程序设计课程的教材,也可作为广大计算机爱好者的自学教材。

### 图书在版编目(CIP)数据

C语言程序设计 / 崔永君, 彭静主编. —西安: 西安电子科技大学出版社, 2011.9

高等学校“十二五”规划教材

ISBN 978-7-5606-2626-0

I. ① C… II. ① 崔… ② 彭… III. ① C语言—程序设计—高等学校—教材 IV. ① TP312

中国版本图书馆CIP数据核字(2011)第130706号

策 划 杨丕勇

责任编辑 买永莲 杨丕勇

出版发行 西安电子科技大学出版社(西安市太白南路2号)

电 话 (029)88242885 88201467 邮 编 710071

网 址 www.xduph.com 电子邮箱 xdupfb001@163.com

经 销 新华书店

印刷单位 陕西光大印务有限责任公司

版 次 2011年9月第1版 2011年9月第1次印刷

开 本 787毫米×1092毫米 1/16 印 张 19

字 数 447千字

印 数 1~3000册

定 价 33.00元

ISBN 978-7-5606-2626-0/TP·1290

**XDUP 2918001-1**

\*\*\*如有印装问题可调换\*\*\*

本社图书封面为激光防伪覆膜, 谨防盗版。

# 前 言

“C 语言程序设计”是计算机及相关专业的一门程序设计启蒙课程，也是众多计算机后续课程的基础。本书以 C 程序设计零起点读者为主要对象，注重教材的可读性和可用性，由浅入深，强化知识点、算法、编程方法与技巧，很多例题后面给出了思考题，帮助读者了解什么是对的以及哪些是容易出错的，从而能够举一反三。本书还将程序测试、程序调试与排错、结构化与模块化程序设计方法等软件工程知识融入其中，同时，每章末的习题均以历年等级考试真题为主，具有代表性。

本书共 11 章。第 1 章从程序设计语言的发展着手，通过例题由浅入深地介绍了 C 程序设计的基本概貌；第 2 章采用有针对性的例题，介绍了 C 语言的基本数据类型、运算符与表达式；第 3~5 章详细讲解了面向过程的程序设计的三种基本结构：顺序、选择、循环；第 6 章介绍了一维数组、二维数组、字符数组和字符串的概念，并结合专业应用介绍了矩阵运算、数据表操作及杨辉三角形的求解问题；第 7 章从模块化程序设计的基本思想以及程序设计的易读性和可维护性出发，介绍了函数的基本概念，并介绍了多文件中函数和变量的处理；第 8 章从内存管理的角度对指针进行了较深入的分析；第 9 章介绍了结构体和共用体以及数据结构中线性链表的基本知识；第 10 章介绍了编译预处理与位运算；第 11 章介绍了文件的基本概念和常用操作。

参加本书编写工作的有彭静(第 1、2、3 章)、崔永君(第 4、5、7 章)、刘小明(第 8、9、11 章)、王芬琴(第 6、10 章)、张永花(附录 B 和 C)等。崔永君和彭静为主编，负责全书的策划、组织和定稿。王芬琴和刘小明为副主编。

本书承蒙兰州交通大学电信学院杜丽霞教授和张忠林教授仔细审阅，提出许多宝贵意见和建议，同时在本书编写和出版的过程中，兰州交通大学博文学院电信工程系计算机教研室的全体教师也给予了热情的帮助和支持，在此一并表示衷心的感谢。

鉴于作者水平有限，书中难免有疏漏和不妥之处，真诚地希望各位读者批评指正，以帮助我们进一步完善教材。

编 者

2011 年 7 月

# 目 录

<b>第 1 章 C 语言概述</b> .....	1
1.1 程序设计语言的发展 .....	1
1.2 C 语言的发展及其特点 .....	2
1.3 C 程序的基本结构与书写规则 .....	4
1.3.1 C 程序的基本结构 .....	4
1.3.2 C 程序的书写规则 .....	8
1.4 计算机运算基础(进位计数制、数值转换) .....	8
1.4.1 数的二进制、十进制、八进制和十六进制表示 .....	8
1.4.2 数制转换 .....	8
1.5 C 程序的编辑、编译和运行 .....	10
1.5.1 一般 C 程序的解题步骤 .....	10
1.5.2 在 Turbo C++环境中运行 C 程序的步骤 .....	11
习题一 .....	15
<b>第 2 章 基本数据类型、运算符和表达式</b> .....	17
2.1 字符集及词法约定 .....	17
2.1.1 C 语言的字符集 .....	17
2.1.2 词法约定 .....	18
2.2 C 语言的基本数据类型 .....	19
2.2.1 常量与变量 .....	20
2.2.2 整型数据 .....	21
2.2.3 实型数据 .....	24
2.2.4 字符型数据 .....	25
2.3 C 语言的运算符与表达式 .....	28
2.3.1 算术运算符与算术表达式 .....	28
2.3.2 关系及逻辑运算符 .....	33
2.3.3 赋值运算符与赋值表达式 .....	35
2.3.4 条件运算符与条件表达式 .....	39
2.3.5 逗号运算符与逗号表达式 .....	39
2.3.6 sizeof 运算符 .....	40
2.3.7 运算符与优先级小结 .....	41
2.4 常见错误 .....	42
习题二 .....	46

<b>第 3 章 顺序结构程序设计</b> .....	50
3.1 程序设计概述 .....	50
3.1.1 基本概念 .....	50
3.1.2 算法 .....	51
3.2 C 语句 .....	52
3.3 赋值语句 .....	53
3.4 数据输入输出在 C 语言中的实现 .....	54
3.5 格式化输入输出函数 .....	55
3.5.1 格式化输出函数 printf() .....	55
3.5.2 格式化输入函数 scanf() .....	57
3.6 字符输入输出函数 .....	62
3.6.1 字符输出函数 putchar() .....	62
3.6.2 字符输入函数 getchar() .....	63
3.7 顺序程序设计举例 .....	64
习题三 .....	65
<b>第 4 章 选择结构程序设计</b> .....	70
4.1 if 语句 .....	70
4.1.1 if 语句的三种基本形式 .....	70
4.1.2 if 语句的嵌套 .....	75
4.1.3 条件表达式 .....	77
4.2 switch 语句 .....	80
4.3 选择分支程序举例 .....	83
习题四 .....	89
<b>第 5 章 循环结构程序设计</b> .....	95
5.1 循环语句概述 .....	95
5.2 goto 语句 .....	96
5.3 while 语句 .....	96
5.4 do-while 语句 .....	98
5.5 for 语句 .....	100
5.6 三种循环语句的选用 .....	103
5.7 break 语句 .....	104
5.8 continue 语句 .....	106
5.9 循环的嵌套 .....	108
5.10 程序举例 .....	111
5.11 常见错误 .....	115
习题五 .....	117

<b>第 6 章 数组</b> .....	123
6.1 一维数组.....	123
6.1.1 一维数组的定义.....	123
6.1.2 一维数组元素的引用.....	124
6.1.3 一维数组的初始化.....	124
6.1.4 一维数组程序举例.....	125
6.2 二维数组.....	129
6.2.1 二维数组的定义.....	129
6.2.2 二维数组元素的引用.....	130
6.2.3 二维数组的初始化.....	130
6.2.4 二维数组程序举例.....	131
6.3 字符数组和字符串.....	135
6.3.1 字符数组的定义.....	135
6.3.2 字符串.....	135
6.3.3 字符数组的初始化.....	136
6.3.4 字符数组的引用.....	137
6.3.5 字符数组的输入与输出.....	137
6.3.6 字符串处理函数.....	139
6.3.7 字符数组程序举例.....	141
习题六.....	145
<b>第 7 章 函数</b> .....	148
7.1 函数的定义.....	148
7.2 函数的返回值与函数类型说明.....	149
7.2.1 函数的返回值.....	149
7.2.2 函数类型说明.....	151
7.3 函数的调用.....	152
7.3.1 函数的形式参数和实际参数.....	152
7.3.2 函数的调用.....	154
7.3.3 被调用函数的声明和函数原型.....	156
7.4 函数的嵌套调用与递归调用.....	157
7.4.1 函数的嵌套调用.....	157
7.4.2 函数的递归调用.....	160
7.5 变量的存储属性.....	166
7.5.1 局部变量.....	166
7.5.2 全局变量.....	167
7.5.3 动态存储变量.....	169
7.5.4 静态存储变量.....	169
7.6 多文件中函数和变量的处理.....	172

习题七.....	174
<b>第 8 章 指针</b> .....	<b>180</b>
8.1 指针变量的定义与引用.....	180
8.1.1 指针与指针变量.....	180
8.1.2 指针变量的定义.....	181
8.1.3 指针变量的赋值.....	182
8.1.4 指针变量的引用.....	182
8.2 指针运算符.....	184
8.2.1 指针运算符与指针表达式.....	184
8.2.2 指针变量作函数的参数.....	186
8.3 指针变量与数组.....	188
8.3.1 指针变量与一维数组.....	188
8.3.2 指针变量与二维数组.....	192
8.3.3 数组指针作函数的参数.....	194
8.4 字符指针.....	200
8.4.1 字符指针的定义和使用.....	200
8.4.2 字符指针用作函数参数.....	203
8.4.3 字符指针变量与字符数组.....	205
8.5 函数与指针.....	206
8.5.1 指针型函数.....	206
8.5.2 函数指针变量.....	207
8.6 指针数组和指向指针的指针变量.....	208
8.6.1 指针数组.....	208
8.6.2 指向指针的指针变量.....	210
8.6.3 main 函数的参数.....	213
习题八.....	214
<b>第 9 章 其他数据类型</b> .....	<b>221</b>
9.1 结构体.....	221
9.1.1 结构说明和结构变量定义.....	221
9.1.2 结构变量的使用.....	223
9.1.3 结构数组和结构指针.....	225
9.1.4 类型名定义.....	231
9.2 动态数据结构.....	232
9.2.1 动态分配内存.....	232
9.2.2 链表.....	233
9.2.3 堆栈.....	239
9.2.4 队列.....	243

9.3 共用体.....	245
9.4 枚举类型.....	247
习题九.....	248
<b>第 10 章 编译预处理与位运算.....</b>	<b>252</b>
10.1 位运算符与位运算.....	252
10.1.1 “按位与”运算符.....	252
10.1.2 “按位或”运算符.....	252
10.1.3 “按位异或”运算符.....	253
10.1.4 “按位取反”运算符.....	253
10.2 位移运算.....	254
10.2.1 左移运算.....	254
10.2.2 右移运算.....	255
10.2.3 与位运算有关的复合赋值运算符.....	255
10.3 位域.....	258
10.3.1 位域的定义和位域变量的说明.....	258
10.3.2 位域的使用.....	259
10.4 宏定义.....	259
10.4.1 无参宏定义.....	260
10.4.2 带参宏定义.....	263
10.5 文件包含.....	265
10.6 条件编译.....	266
习题十.....	268
<b>第 11 章 文件.....</b>	<b>270</b>
11.1 C 文件概述.....	270
11.2 文件的打开与关闭.....	270
11.3 文件的输入和输出.....	272
11.4 文件的定位.....	278
11.5 文件的错误检测及错误处理函数.....	279
习题十一.....	280
<b>附录 A 部分习题参考答案.....</b>	<b>282</b>
<b>附录 B 常用字符与 ASCII 代码对照表.....</b>	<b>286</b>
<b>附录 C Turbo C 常用库函数.....</b>	<b>288</b>
<b>参考文献.....</b>	<b>294</b>



# 第1章 C语言概述

## 1.1 程序设计语言的发展

自1946年世界上第一台电子计算机问世以来,计算机科学及其应用的发展十分迅猛,计算机被广泛地应用于人类生产、生活的各个领域,推动了社会的进步与发展。特别是随着因特网(Internet)的逐渐普及,传统的信息收集、传输及交换方式正被革命性地改变,人们已经难以摆脱对计算机的依赖,计算机将人类带入了一个新的时代——信息时代。

计算机是由硬件系统和软件系统两大部分构成的,硬件是物质基础,而软件可以说是计算机的灵魂。没有软件,计算机就是一台“裸机”,什么也不能干;有了软件,计算机才能灵动起来,成为一台真正的“电脑”。所有的软件,都是用计算机程序设计语言编写的。

计算机程序设计语言是人与计算机进行交流的有力工具。随着计算机技术的发展,计算机程序设计语言也不断发展,从机器语言到汇编语言,再到高级语言。

### 1. 机器语言

机器语言是第一代计算机语言。计算机发明之初,人们只能用计算机的语言去命令计算机工作,也就是只能写出一串串由“0”和“1”组成的指令序列(程序)交由计算机执行,这种语言就是机器语言。

使用机器语言是十分痛苦的,特别是在程序有错需要修改时更是如此。而且,由于每台计算机的指令系统往往各不相同,所以,在一台计算机上执行的程序要想在另一台计算机上执行,必须另编程序,这样造成了大量的重复性工作。但由于使用的是针对特定型号计算机的语言,故机器语言所编程序的运算效率是所有语言中最高的。

### 2. 汇编语言

为了减轻使用机器语言编程的困难,人们进行了一种有益的改进:用一些简洁的英文字母、符号串来替代一个特定指令的二进制串,比如,用“ADD”代表加法,“MOV”代表数据传递等。这样一来,人们很容易读懂并理解程序在干什么,纠错及维护也变得方便了,这种程序设计语言就是汇编语言,即第二代计算机语言。然而计算机是不认识这些符号的,这就需要有一个专门的程序,负责将这些符号翻译成二进制数的机器语言,这种翻译程序被称为汇编程序。

汇编语言同样十分依赖于机器硬件,移植性不好,但运算效率仍十分高。针对计算机特定硬件而编制的汇编语言程序能准确地发挥计算机硬件的功能和特长,程序精练而质量高,所以至今仍是一种常用且强有力的软件开发工具。



### 3. 高级语言

从最初与计算机交流的痛苦经历中人们意识到，应该设计一种这样的语言：接近于数学语言或人的自然语言，同时又不依赖于计算机硬件，编出的程序能在所有的机器上通用。经过努力，1954年，第一个完全脱离机器硬件的高级语言——FORTRAN 问世了。几十年来，共有几百种高级语言出现，有重要意义的有几十种，影响较大、使用较普遍的有 FORTRAN、ALGOL、COBOL、BASIC、LISP、NOBOL、PL /1、Pascal、C、PROLOG、Ada、C++、VC、VB、Delphi、JAVA 等。

20 世纪 60 年代中后期，软件越来越多，规模越来越大，而软件的生产基本上是各自为政，缺乏科学而规范的系统规划与测试以及评估标准，其恶果是耗费巨资建立起来的大批软件系统由于含有错误而无法使用，甚至带来巨大损失，软件给人的感觉是越来越不可靠，以致几乎没有不出错的软件。这一切极大地地震动了计算机界，史称“软件危机”。人们认识到：大型程序的编制不同于写小程序，它应该是一项新的技术，应该像处理工程一样处理软件研制的全过程。程序的设计应易于保证正确性，也便于验证正确性。为此，1969 年提出了结构化程序设计方法，1970 年，第一个结构化程序设计语言——Pascal 语言出现，标志着结构化程序设计时期的开始。

自 20 世纪 80 年代初开始，在软件设计思想上又产生了一次革命，其成果就是面向对象的程序设计方法的产生。在此之前的高级语言几乎都是面向过程的，程序的执行是流水线式的，即在一个模块被执行完成前，人们不能干别的事，也无法动态地改变程序的执行方向。这和人们日常处理事物的方式是不一致的，对人而言是希望发生一件事就处理一件事，也就是说，不能面向过程，而应是面向具体的应用功能，也就是对象(Object)。其方法就是软件的集成化，如同硬件的集成电路一样，生产一些通用的、封装紧密的功能模块(称之为软件集成块)，它与具体应用无关，但能相互组合，完成具体的应用功能，同时又能重复使用。对使用者来说，只关心它的接口(输入量、输出量)及能实现的功能，至于是如何实现的，使用者完全不用关心，C++、VB、Delphi 就是典型代表。

高级语言的下一个发展目标是面向应用，也就是说，只需要告诉程序你要干什么，程序就能自动生成算法并自动进行处理，这就是非过程化的程序设计语言。

## 1.2 C 语言的发展及其特点

C 语言是 1972 年由美国的 Dennis Ritchie 设计的，并首次在 DEC PDP-11 计算机上使用。它由早期的编程语言 BCPL (Basic Combind Programming Language)演变而来。1970 年，AT&T 贝尔实验室的 Ken Thompson 根据 BCPL 语言设计出较先进的并取名为 B 的语言最后导致了 C 语言的问世。

随着微型计算机的日益普及，出现了许多 C 语言版本，但由于没有统一的标准，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，美国国家标准协会(American National Standards Institute, ANSI)为 C 语言制定了一套 ANSI 标准，通常称之为 ANSI C，成为现行的 C 语言标准。

C语言之所以发展迅速，成为最受欢迎的语言之一，这源于C语言具有很多显著的特点，主要如下：

(1) 简洁、紧凑，使用方便、灵活。

ANSI C共有32个关键字，9种控制语句，程序书写自由，一般用小写字母表示，压缩了一切不必要的成分。C语言在表示方式上力求简单易行，如使用一对花括号“{ }”来表示复合语句的开始和结束，用赋值运算符(如+=、-=、\*=、/=等)表示进行相应运算并且将结果赋给左值(赋值号左边的变量)，等等。

(2) 运算符和数据结构丰富，表达式类型多样。

C语言共有34种运算符。在C语言中，把括号、赋值号、逗号、关系运算、逻辑运算等都作为运算符处理。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。

C语言提供了丰富的数据类型。C语言中的数据类型基本上可以分为两大类：一类是简单类型，如整型、实型、字符型等；另一类是在简单类型基础上按层次产生的各种构造类型，如数组类型、结构体类型和共用体类型等，此外还有指针类型。利用这些数据类型，C语言能够实现各种复杂的数据结构，如线性表、链表、栈、队列、树、图等。

C语言的表达式类型多样，既提高了编译效率和目标代码的质量，又提高了程序的可读性。

(3) C语言是结构化的程序设计语言。

结构化语言的特点是代码与数据分离，即程序的各个部分除了必要的信息交流外彼此独立，从而使程序之间很容易实现程序段的共享。C语言提供了顺序结构、选择(分支)结构和循环结构三种基本结构语句，并以函数作为模块，实现程序的模块化设计，符合现代编程风格。

(4) 语法限制不太严格，程序设计自由度大。

C语言编译系统语法检查不太严格。例如，在C语言中对数组下标越界不进行检查，变量类型使用灵活，整型数据和字符型数据及逻辑型数据可以通用。在语法上放宽限度，在程序设计上灵活自由，相应地，检查错误的任务也就转到了编程者身上。因此，这就要求编程者在编程时要自我约束，养成良好的编程习惯。

(5) 生成目标代码质量高，程序执行效率高。

C语言具有位(bit)操作的功能，可以直接对硬件进行操作，这使得C语言既具有高级语言的所有优点，又具有低级语言的许多功能，成为所谓的“中间语言”。C语言通过对位、字节和地址进行操作，可以对硬件进行编程，并可实现汇编语言的大部分功能，具有高效率的目标代码。用C语言编写的程序生成代码的效率比用汇编语言编写的仅低10%~20%。

(6) C语言编写的程序可移植性好。

与汇编语言相比，C语言编写的程序基本上不做或稍做修改就可以在其他工作平台上运行，从而使开发的软件独立于具体的计算机体系结构和软件环境，达到“共用”目标，提高了软件的生产效率，节省了用户的投资。

C语言还有其他的优点，读者可以在学习及实践中体会。当然，C语言也和其他语言一样，存在不足之处，如某些运算符优先顺序和习惯不完全一致，类型的转换比较随意等。尽管如此，C语言仍不愧为优秀的程序设计语言之一。



## 1.3 C 程序的基本结构与书写规则

任何一种程序设计语言都具有特定的语法规则和规定的表达方法。一个程序只有严格按照该语言规定的语法和表达方式编写，才能保证编写的程序在计算机中正确地被执行，同时也便于阅读和理解。

### 1.3.1 C 程序的基本结构

为了说明 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，从中可以了解一个 C 语言源程序的基本组成部分和书写格式。

**【例 1.1】** 输出一行信息。

源程序如下：

```
#include<stdio.h>
main()                /*主函数*/
{
    printf("Welcome to C!\n"); /*调用标准函数显示引号中的内容*/
    return 0;
}
```

这是一个最简单的 C 程序，其执行结果是在屏幕上显示一行信息：

```
Welcome to C!
```

程序中 / \*...\*/ 部分的内容表示对程序的说明(称为注释)，不参与程序的运行。注释文字可以是任意字符，如汉字、拼音、英文等。

C 语言程序是由函数组成的，任何 C 语言程序都有且只有一个主函数 `main()`，它是程序的入口，也是程序的出口。函数的内容(称为体)由一对花括号括起来。本程序只有一条输出语句，其中 `printf` 是函数名，圆括号中用双引号引起来的字符串是它的参数。该函数的功能是把这个字符串打印出来。圆括号后面的分号是语句的标志，任何语句都必须以分号结束。字符串中的“\n”是个控制字符，用于控制换行，它是一个不可显示的字符，即输出时看不见。

例 1.1 中用到了输出函数 `printf`，与之相应的是输入函数 `scanf`。这两个函数在第 3 章将详细介绍，这里先简单介绍一下它们的格式，以便下面使用。

`scanf` 和 `printf` 分别称为格式输入函数和格式输出函数，其意义是按指定的格式输入、输出值。因此，这两个函数括号中的参数都由以下两部分组成：

“格式控制串”，参数表

格式控制串是一个字符串，必须用双引号括起来，它表示输入、输出量的数据类型。各种类型的格式表示法可参阅第 2 章。在 `printf` 函数中的格式控制串内还可以出现非格式控制字符，在显示屏幕上将原样输出。参数表中给出了输入或输出的量。当有多个量时，用逗号间隔。例如：

```
printf("sine of %lf is %lf\n",x,s);
```

其中，%lf 为格式字符，表示按双精度浮点数处理；它在格式串中两次出现，对应了 x 和 s 两个变量。其余字符为非格式字符，按照原样输出在屏幕上。

### 【例 1.2】 输出多行信息。

源程序如下：

```
main()                /*主函数*/
{
    void proc();       /*函数声明*/
    int a=3;           /*指定 a 为整数，初始值为 3*/
    proc();            /*调用函数 proc，无返回*/
    printf("This is a sample of c program. \n"); /*调用标准函数，显示引号中的内容*/
    a = func();        /*调用函数 func，结果返回给 a*/
    printf("a=%d\n",a); /*显示 a 的值*/
}
void proc()           /*定义函数 proc，void 指定该函数不返回结果*/
{
    printf("Hello. \n"); /*调用标准函数，显示引号中的内容*/
}
int func()            /*定义函数 func，int 指定该函数返回一个整数*/
{
    return(2);        /*返回整数 2*/
}
```

本程序的执行过程是：

- (1) 程序从 main()处开始。
- (2) 变量 a 代表一个整数，并且初始值为 3。
- (3) 执行程序(函数)proc()；屏幕上显示 Hello，\n 为转义字符，代表换行。
- (4) 屏幕上显示 “This is a sample of c program。”。
- (5) 执行程序(函数)func()；将结果赋予 a，此时 a 的值为 2。
- (6) 在屏幕上输出 a 的值。

程序执行的结果是在屏幕显示三行信息：

```
Hello.
This is a sample of c program.
a=2
```

### 【例 1.3】 求两个数中的较大者。

源程序如下：

```
1  #include<stdio.h>
2  main()                /*主函数*/
3  {
```



```
4     int  a,b,c;                /*定义两个整数变量，以备后面程序使用*/
5     int  max(int,int);        /*函数声明*/
6     printf("Input an integer\n"); /*显示提示信息*/
7     scanf("%d",&a);          /*从键盘获得一个整数 a*/
8     printf("Input an other integer\n"); /*显示提示信息*/
9     scanf("%d",&b);          /*从键盘获得一个整数 b*/
10    c=max(a,b);              /*求 a 与 b 的最大值，并把它赋给变量 c*/
11    printf("max=%d\n",c);    /*显示程序运算结果*/
12    return 0;
13 }
14 int  max(int x, int y)
15 {
16     int z;
17     if(x>y)
18         z=x;
19     else
20         z=y;
21     return z;
22 }
```

该程序的运行结果为

```
Input an integer
8
Input an other integer
7
c=8
```

该程序的功能是从键盘输入两个数  $a$  和  $b$ ，求  $a$  和  $b$  的最大值  $c$ ，然后输出结果。`main()` 之前的一行称为预处理命令(详见后面)，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`，因此也称为头文件或首部文件。C 语言的头文件中包括了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。在本例中，使用了两个库函数：输入函数 `scanf`，输出函数 `printf`。`scanf` 和 `printf` 是标准输入、输出函数，其头文件为 `stdio.h`，在主函数前用 `include` 命令包含了 `stdio.h` 文件。

需要说明的是，C 语言规定对 `scanf` 和 `printf` 两个函数可以省去其头文件的包含命令，如例 1.2。在本例中也可以删去第一行的包含命令 `#include<stdio.h>`。同样，例 1.1 中使用了 `printf` 函数，也可以省略包含命令。

例 1.3 中的主函数体中又分为两部分，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 1.1 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 语言源程序结构中很重要的组成部分。例 1.3 中使用了三个变量 a、b 和 c,用来表示输入的自变量与计算的最大值。程序第 6~11 行为执行部分,或称为执行语句部分,用以完成程序的功能。第 6 行和第 8 行是输出语句,调用 printf 函数在显示器上输出提示字符串,要求输入自变量 a 与变量 b 的值。第 7 行与第 9 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 a 与变量 b 中。第 10 行是调用函数 `int max(int x, int y)` 计算 a 与 b 的最大值并把它送到变量 c 中,调用时将实际参数 a 和 b 的值分别传给 max 函数中的形式参数 x 和 y,经过执行 max 函数得到一个返回值(max 函数中变量 z 的值),把这个返回值赋给变量 c。第 11 行是用 printf 函数输出变量 c 的值(a 与 b 的最大值)。第 12 行是一个返回语句,这里是把整数 0 传送给运行该程序的操作系统环境,其作用是把程序成功执行完毕这一消息通知给操作系统,标志着程序的终止。

该例中除主函数 main 外,又定义了一个 max 函数。`int max(int x, int y)`称为函数头,max 为函数名,int 为函数返回值的类型,圆括号中的 x、y 称为参数,其前面的 int 规定了参数的类型。

max 函数体中的“`int z;`”语句定义了一个临时工作变量,它只在这个函数中使用,作用是接收 x、y 中的最大者,最后返回给调用函数。

从上面所举的几个例子可以看出 C 程序的基本结构特点如下:

- (1) 一个 C 语言源程序可以由一个或多个源文件组成。
- (2) 每个源文件可由一个或多个函数组成。C 程序的函数由编译系统提供的标准函数(如 printf、scanf 等)和用户自己定义的函数(如 proc、func 等)组成。

函数的基本形式是:

```
函数类型  函数名(形式参数)
形式参数说明;
{
    数据说明;
    语句;
}
```

其中:

① 函数头包括函数类型、函数名和形式参数(如 `int max(int x, int y)` 函数)。如果函数调用中无参数传递,则形式参数为空(如 `void proc()` 函数)。

② 形式参数说明指定函数调用传递参数的数据类型(如例 1.2 中语句“`int a,b,sum;`”)。函数体包括函数体内的数据说明和执行函数功能的语句,花括号“{”和“}”分别表示函数体的开始和结束。

(3) 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。程序从主函数开始执行,当执行到调用函数的语句时,程序将控制转移到调用函数中执行,执行结束后,再返回主函数中继续运行,直至程序执行结束。

(4) 源程序中可以有预处理命令(include 命令仅为其中的一种),预处理命令通常放在源文件或源程序的最前面。

(5) 每一个说明、每一个语句都必须以分号结尾,但预处理命令、函数头和花括号“}”



之后不能加分号。

(6) 标识符、关键字之间必须加至少一个空格以示间隔。若已有明显的间隔符，也可不再加空格。C 语言的关键字都以小写字母表示。C 语言中区分字母的大写与小写，例如，`else` 是关键字，`ELSE` 则不是。在 C 程序中，关键字不能用于其他目的，即不允许将关键字作为变量名或函数名来使用。

### 1.3.2 C 程序的书写规则

从便于阅读、理解及维护的角度出发，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可比高一层次的语句或说明缩进若干格后书写，以便结构更加清晰，增加程序的可读性。

在编程时应力求遵循上述规则，养成良好的编程习惯。

## 1.4 计算机运算基础(进位计数制、数值转换)

由于计算机内采用二进制表示数据，而 C 程序中经常使用八进制和十六进制来编制程序，因此程序员必须熟悉二进制、八进制和十六进制的运算及它们之间的转换规则。

### 1.4.1 数的二进制、十进制、八进制和十六进制表示

常用的数制有二进制、十进制、八进制和十六进制，它们有共性也有差别。

#### 1. 数码及进位法则

数码是构造一种数制所用的不同符号。各种进制的数码如下：

二进制：0, 1

十进制：0, 1, 2, 3, 4, 5, 6, 7, 8, 9

八进制：0, 1, 2, 3, 4, 5, 6, 7

十六进制：0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A(a), B(b), C(c), D(d), E(e), F(f)

#### 2. 位置计数法

数据中各个数字所处的位置决定着其大小(即权值)，同样的数字在不同位置上代表的权值是不同的。例如：

$$22.2 = 2 \times 10^1 + 2 \times 10^0 + 2 \times 10^{-1}$$

其中， $10^1$ 、 $10^0$ 、 $10^{-1}$  就是该位置的权值。

### 1.4.2 数制转换

日常习惯中使用的是十进制数，而计算机中用的却是二进制数，所以需要把十进制数转换成二进制数。但二进制数书写麻烦，因此通常也用八进制和十六进制表示，这样就存在各种数制之间的转换问题。



### 1. 将十进制数转换成二进制数

把十进制的整数和小数转换成二进制数时所用的方法是不同的,因此应该分别进行转换。

(1) 用余数法将十进制整数转换成二进制整数。把十进制整数不断地用2去除,将所得到的余数0或1依次记为 $K_0, K_1, K_2, \dots$ ,直到商是0为止,将最后一次所得的余数记为 $K_n$ ,则 $K_n K_{n-1} \dots K_1 K_0$ 即为该整数的二进制表示。在演算过程中可用竖式形式,也可用线图形式。

【例 1.4】  $(59)_{10} = ( )_2 = (K_n K_{n-1} \dots K_1 K_0)_2$ 。

竖式演算如下:

		余数	
2	59	1	----- $K_0$
2	29	1	----- $K_1$
2	14	0	----- $K_2$
2	7	1	----- $K_3$
2	3	1	----- $K_4$
2	1	1	----- $K_5$
	0		

因此,  $(59)_{10} = (111011)_2$ 。

(2) 用进位法将十进制小数转换成二进制小数。把十进制小数不断地用2去乘,将所得乘积的整数部分0或1依次记为 $K_1, K_2, K_3, \dots$ 。一般情况下,十进制小数并不一定都能用有限位的二进制小数表示,可根据精度要求,转换成一定位数即可。

注意:在把十进制数转换成二进制数时,熟记一些2的幂次的十进制及二进制值能加快转换速度。表 1.1 中列出了一些2的幂次所对应的十进制和二进制数。

表 1.1 2 的幂次所对应的十进制和二进制数

2 的幂次	十进制数	二进制数
$2^0$	1	1
$2^1$	2	10
$2^2$	4	100
$2^3$	8	1000
$2^4$	16	10000
$2^5$	32	100000
$2^6$	64	1000000
$2^7$	128	10000000
$2^8$	256	100000000
$2^9$	512	1000000000
$2^{10}$	1024	10000000000
$2^{-1}$	0.5	0.1
$2^{-2}$	0.25	0.01
$2^{-3}$	0.125	0.001
$2^{-4}$	0.0625	0.0001