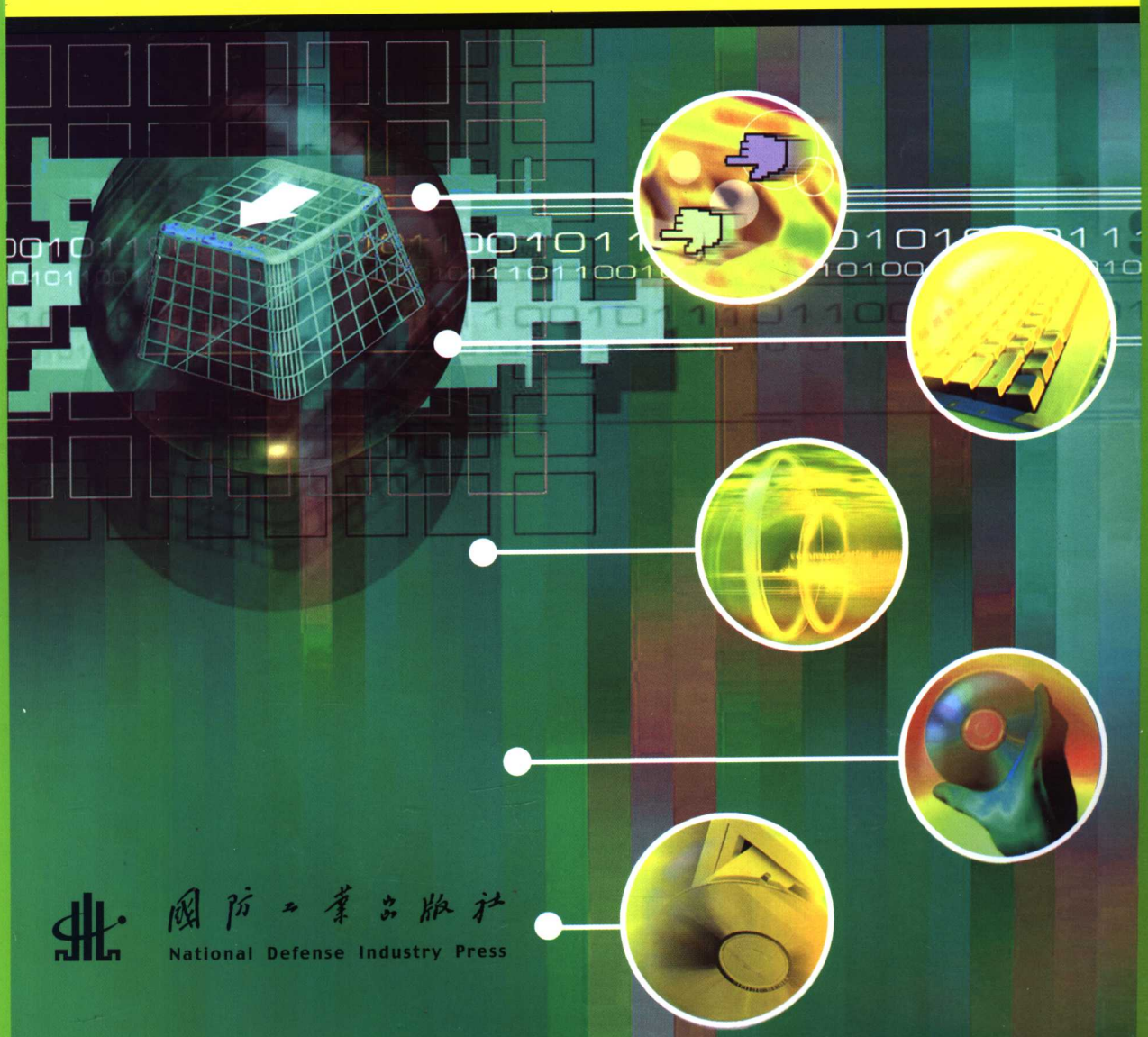
21世纪高等院校规划教材

C语言程序设计基础

主　编　梁成升
副主编　杜素芳　陈　曦　韩利娟　郝世选



国防工业出版社
National Defense Industry Press

21 世纪高等院校规划教材

C 语言程序设计基础

主编　梁成升

副主编　杜素芳　陈　曦　韩利娟　郝世选

国防工业出版社

·北京·

图书在版编目(CIP)数据

C 语言程序设计基础/梁成升主编. —北京:国防工业出版社,2006.8
21 世纪高等院校规划教材
ISBN 7-118-04705-8

Ⅰ.C... Ⅱ.梁... 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(2006)第 090964 号

※

国防工业出版社出版发行
(北京市海淀区紫竹院南路 23 号 邮政编码 100044)
腾飞胶印厂印刷
新华书店经售
*
开本 787×1092 1/16 印张 20½ 字数 471 千字
2006 年 8 月第 1 版第 1 次印刷 印数 1—4000 册 定价 33.00 元

(本书如有印装错误,我社负责调换)

国防书店:(010)68428422 发行邮购:(010)68414474
发行传真:(010)68411535 发行业务:(010)68472764

前　言

C语言是一种通用的程序设计语言，也是许多高校为学生开设的第一门程序设计课程。本书充分考虑程序设计入门的教学特色，理论上做到必须、够用，注重联系实际，突出实用性；语言组织上通俗易懂，做到在内容的编排上尽量符合初学者的要求。

多年来，编者一直从事C语言的教学工作，编者结合多年教学、科研和应用经验，结合C语言程序设计的特点，对全书内容做了精心安排，分解难点，由浅至深，力求用简明易懂的语言和丰富的例题清晰地解释C语言中各个复杂概念，使读者能轻松掌握该门语言。本书主要特点可归纳如下：

(1) 按照循序渐进的原则，逐步介绍C语言中的基本概念和语法规则。

(2) 通过典型的例题分析，着重强调了利用C语言进行程序设计的方法。

(3) 本书的重点是C语言的使用，没有深奥的理论和算法，书中例题都给出了比较详细的解释。因此，特别适合于初学者和自学者使用。

(4) 在文字叙述上力求条理清晰、简洁，以利于读者阅读。

(5) 每章的最后都附有上机练习和一定量的习题，习题包括填空题、选择题和编程题，这些练习和习题对于读者巩固已学习的内容很有帮助。

编者认为，要学好C语言，除了掌握C语言的基本理论之外，还必须加强实践。本书中的所有例题都在微机上调试通过，希望读者边学习边上机实践，这样不仅可以加快学习进度，也能提高学习效率。

参加本书编写的有杜素芳、陈曦、韩利娟、郝世选等，最后由梁成升修改定稿。

由于编者水平和经验有限，编写时间仓促，缺点和错误在所难免，希望广大读者批评指正。

编　者

2006年7月

目　录

第 1 章　程序设计基础

本章着重介绍了程序设计中所涉及的基础知识：从逻辑结构的角度讲述计算机的工作原理；数据在计算机中的表示形式；计算机语言的概念；程序的概念；程序设计的灵魂——算法的概念及其表示形式；最后将讲解编制程序所必需的步骤，并介绍有关编译程序和链接程序的一些知识。

1.1　计算机的逻辑结构

从 1946 年世界上第一台电子计算机 ENIAC 诞生之日起，计算机系统就一直采用由冯·诺伊曼提出的“程序内存式”逻辑结构进行工作。这种逻辑结构可简述如下：计算机要处理的数据、程序以及指令都先放入存储器中进行暂时存放，然后依次输送至中央处理器 CPU 进行运算，计算的结果再放回至存储器中，最后输送至输出设备显示或打印，如图 1.1 所示。

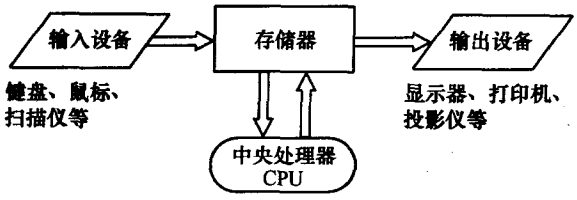


图1.1　计算机逻辑结构简图

这里，输入设备是指键盘、鼠标、扫描仪等，输出设备是指显示器、打印机等。中央处理器 CPU（Central Processing Unit）是对信息或数据进行处理和运算的部件。存储器是计算机的数据记忆部件，好比一个数据的仓库，所有输入数据、待计算的数据以及要输出的数据都存放在存储器中。存储器有主存储器（简称主存，也称内存储器）和辅助存储器（简称辅存，也称外存储器）之分。主存储器中存放当前在计算机上运行的程序和数据，但它具有易失性，即只有计算机在开机状态下，主存储器中的内容才能保留，关闭或重启计算机其中的内容就都会被清空。辅助存储器中存放主存数据的副本和当前不再运行的程序和数据，特点是其中的数据能永久性地保存，即计算机关闭后其中的内容仍然被保留。

从实物形态上看，辅存是指硬盘、软盘、光盘以及平时我们经常使用的移动硬盘和 MP3 闪存等；主存指的是电脑主机箱中的内存条。一个完整的“程序内存式”计算机逻辑结构如图 1.2 所示。

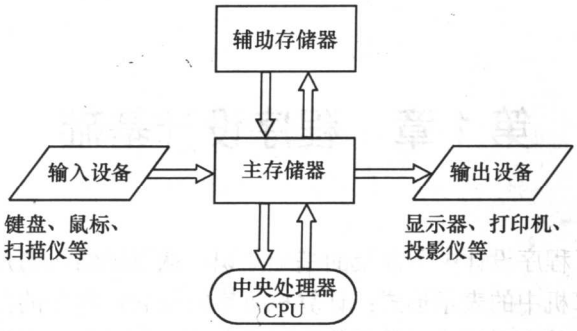


图1.2　计算机逻辑结构完整图

比如，我们要使用 Photoshop 图像处理软件（简称 PS）对一张存放在软盘中的图片进行编辑。首先，启动 PS 程序，这个过程是将硬盘（外存储器）中的 PS 程序调入内存（内存储器）。然后将软盘(外存储器)中的图片也载入内存。之后通过鼠标和键盘（输入设备）输入的编辑指令，对内存中的这张图片进行编辑，在此过程中 CPU 参与了图像数据的运算。编辑成功后需要进行保存，也就是将存放在内存中已编辑完毕的图片转移至软盘上。由此看来，内存作为主存储器，在整个过程中起到数据传输与存放的枢纽作用。

图 1.3 所示是电脑主机箱中的主存储器：内存条。内存条安装在电脑主机板中的内存插槽中，如图 1.4 所示。

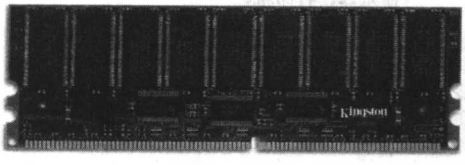


图1.3　内存条

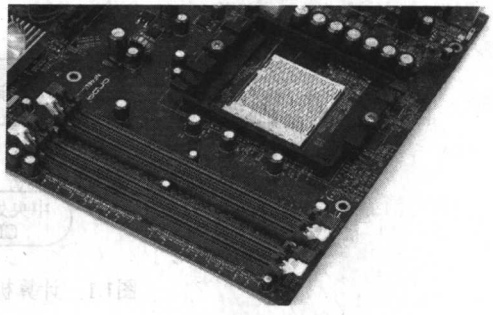


图1.4　主板内存插槽

主存储器是计算机内部用于暂时存放数据的硬件设备，由大量的存储单元构成。每个存储单元可以存放 8 位二进制数据，也就是 8 个由 0 或 1 组成的数字串。这样的存储单元称为一个字节，即主存储器是以字节为基本单位进行数据存储的。这些存储单元是连续排列的，并且被编了号，这个号就是存储单元的地址。计算机的其他硬件（比如 CPU）将依据这些地址信息访问对应存储单元中数据。

如果将一栋楼中的每个房间比作存储单元，则房间的门牌号就好比存储器单元的地址。这些地址信息从 0 开始顺序排列，最大地址数值取决于存储器的容量。

C 语言中有很多令初学者感到困惑的地方都牵涉到内存知识。比如：整型变量的取值范围、不同类型变量间赋值时的类型转换规则、数据的溢出、函数参数的值传递及地址传递、数组地址的分配、指针变量的理解等。一旦建立起变量与地址间的对应关系，对后续知识，特别是对指针的学习是大有好处的。

1.2 数字信息量的度量单位

在计算机中，一切数据都表示为二进制代码，即由数字 0 和 1 组成的字符串。信息的容量也就是指包含 0 或 1 字符的个数。这里列出常用的单位：

bit（比特）：1 个 0 或 1 字符。

Byte（字节）：8 个 1 或 0 组成的字符串。

由于数字信息量通常较大，用比特或字节来表述会很繁琐，因此就有了 KB、MB 和 GB。

$1KB=2^{10}$ Byte=1024 Byte

$1MB=2^{10}KB=1024KB$

$1GB=2^{10}MB=1024MB$

1.3 计算机中数据的表示和运算

数据形态多种多样，如文字数据、图形数据、声音数据等。但不管数据表象多么千差万别，只要一归入计算机处理，都全部地统一为二进制数据了，即用数字 0 和 1 来表示。

1. 采用二进制的原因

计算机内部用电子器件的状态来表示数字信息。一种数制有多少种不同的数字，电子器件就需要多少种不同的状态。十进制有 0～9 十个数字，因此十进制数需要由若干个具有 10 种不同稳定状态的电子元件来表示，而二进制只有数字 0 和 1，因此只需要由若干个具有 2 种稳定状态的电子元件就可以表示了。显然，稳定状态越少，电子元件的制造就越简单。如开关的接通与断开，晶体管的导通与截止，都可以由 0 和 1 两个数字表示。因此用二进制表示数据，数字的状态少，使得计算机元件制作简单，工作性能可靠。

2. 二进制运算规则简单

二进制求和的规则有：

0+0=0

0+1=1+0=1

1+1=10

二进制求积的规则有：

0×0=0

0×1=1×0=0

1×1=1

1.3.1 二进制、八进制和十六进制

前面已经提到二进制是计算机中数据的唯一表示形式，但由于二进制数的阅读与书写比较复杂，为了方便，在阅读与书写时通常用十六进制或八进制来表示。下面我们先

来了解这几种进制的计数规则。

1. 十进制的计数规则

加减法时，逢 10 进 1，借 1 当 10。

每一数位有 10 种状态，即 0、1、2、3、4、5、6、7、8、9，因此 10 被称为十进制的基数。我们可以这样解析一个十进制数：

$1234.56=1\times10^3+2\times10^2+3\times10^1+4\times10^0+5\times10^{-1}+6\times10^{-2}$，其中 10^k 被称为相应数位的权值。因此说，一个十进制数的值就是各个数位的数值与其权值的乘积和。

2. 二进制的计数规则

加减法时，逢 2 进 1，借 1 当 2。

每一数位有两种状态：0、1，因此 2 被称为二进制数的基数。

为了区别不同进制的数，通常将数括起来，并把数制的基数写在右下角。

我们可以这样解析一个二进制数（将它转换成十进制数）：

$(1011.11)_2=1\times2^3+0\times2^2+1\times2^1+1\times2^0+1\times2^{-1}+1\times2^{-2}=(11.75)_{10}$

3. 八进制的计数规则

加减法时，逢 8 进 1，借 1 当 8。

每一数位有 8 种状态：0、1、2、3、4、5、6、7，因此 8 被称为八进制数的基数。我们可以这样解析一个八进制数（将它转换成十进制数）：

$(711)_8=7\times8^2+1\times8^1+1\times8^0=(457)_{10}$

4. 十六进制的计数规则

加减法时，逢 16 进 1，借 1 当 16。

每一数位有 16 种状态：0、1、2、3、4、5、6、7、8、9、a、b、c、d、e、f。其中，a/A、b/B、c/C、d/D、e/E、f/F 分别代表十进制中数值 10、11、12、13、14、15，因此 16 被称为十六进制数的基数。我们可以这样解析一个十六进制数：

$(fdea)16=15\times16^3+13\times16^2+14\times16^1+10\times16^0=(65002)_{10}$

1.3.2 二进制的算数运算

1. 二进制的加法

二进制数的加法与十进制数的加法规则相同，只是逢 2 进 1。

例如，10101+11001=101110。图 1.5 中，数右下角的 1 表示进位。

2. 二进制的减法

二进制数的减法运算与十进制数的减法类同，只是借 1 当 2。

例如，11001-10101=100。图 1.6 中，数右上角的 1 表示借位。

$$\begin{array}{cccccc} & 1 & 0 & 1 & 0 & 1 \\ +\ 1 & 1 & 1 & 0 & 0_1 & 1 \\ \hline 1 & 0 & 1 & 1 & 1 & 0 \end{array}$$

图1.5 二进制的加法

$$\begin{array}{cccccc} & 1 & 1^1 & 0 & 0 & 1 \\ - & 1 & 0 & 1 & 0 & 1 \\ \hline & 0 & 0 & 1 & 0 & 0 \end{array}$$

图1.6 二进制的减法

3. 二进制的乘法

十进制数的乘法规则同样适用于二进制数，只不过二进制数的数位相乘只有两个结果：1 或 0。例如，1101×101=1000001，如图 1.7 所示。

4. 二进制的除法

十进制数的除法规则同样适用于二进制数，只不过二进制数的数位相除只有两个结果：1 或 0。例如，1001÷11=11，如图 1.8 所示。

```
                1  1  0  1
                   1  0  1
-------------------------------
                1  1  0  1
          0  0  0  0
       1  1  0  1
-------------------------------
1  0  0  0  0  0  1
```

图1.7　二进制的乘法

```
          1  1
      ------------
1  1  | 1  0  0  1
         1  1
      ------------
            1  1
            1  1
      ------------
               0
```

图1.8　二进制的除法

1.3.3　二进制、八进制、十六进制和十进制之间的转换

1. 十进制转换成二进制

把十进制的整数部分和小数部分转换成二进制时所用的方法是不同的，因此应该分别进行转换。

(1) 用“除 2 取余法”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，将每次所得到的余数(0 或 1)依次记为 $K_0, K_1, K_2, \cdots$，直到商为 0 止，将最后一次得到的余数记为 K_n，则 $K_n \cdots K_2 K_1 K_0$ 即为该整数的二进制表示。例如，将$(46)_{10}$转换成二进制数，如图 1.9 所示。

```
2 | 4 6
2 | 2 3    余数 K0=0
2 | 1 1    余数 K1=1
2 |  5     余数 K2=1
2 |  2     余数 K3=1
2 |  1     余数 K4=0
  |  0     余数 K5=1
```

图1.9　除2取余法

注意：第一次得到的余数为二进制数的最低位，最后一次得到的余数为二进制数的最高位。所以$(46)_{10}=(K_5K_4K_3K_2K_1)_2=(101110)_2$。

(2) 用“乘 2 取整法”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}，…，直到乘积为 0 止。一般说来，这个转换过程永远不会遇到乘积为 0 的情况。对于无法精确表示的小数，计算机只能用有限位的二进制小数表示，有限位越多，这个二进制小数就越接近实际的十进制小数。例如，将$(0.46)_{10}$转换成二进制小数，如图 1.10 所示。

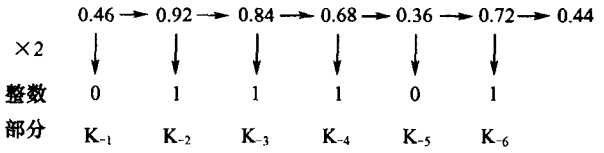


图1.10　乘2取整法

在取 6 位小数时有：$(0.46)_{10}=(0.K_{-1}K_{-2}K_{-3}K_{-4}K_{-5}K_{-6})_2=(0.011101)_2$

(3) 在把十进制转换成二进制时，熟记一些十进制数所对应 2 的次幂数能加快转换的进度。表 1.1 中列出了一些常用的 2 的幂次。在转换时，可尽量将所要转换的十进制数组合成若干个 2 的幂次的和，这样就能快速地写出对应的二进制数了。比如：

$(1256.75)_{10}$=1024+128+64+32+8+0.5+0.25

$=2^{10}+2^7+2^6+2^5+2^3+2^{-1}+2^{-2}$

$=(10000000000+10000000+1000000+100000+1000+0.1+0.01)_2$

$=(10011101000.11)_2$

如表 1.1 所示。

表1.1　常用的2的幂次

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

2. 八进制、十六进制和二进制的转换

八进制、十六进制是在二进制的基础上演变而来的，它们的产生是为了使二进制数据表述更简洁。三者有着简单的对应关系：由于2^4=16，因此连续 4 位二进制数相当于 1 位十六进制数。同理，由于2^3=8，因此连续 3 位二进制数相当于 1 位八进制数。

二进制转换为八进制和十六进制数的方法如下：

(1) 将二进制数从右向左每 3 位分一组，然后将每一组二进制数转换为一个八进制