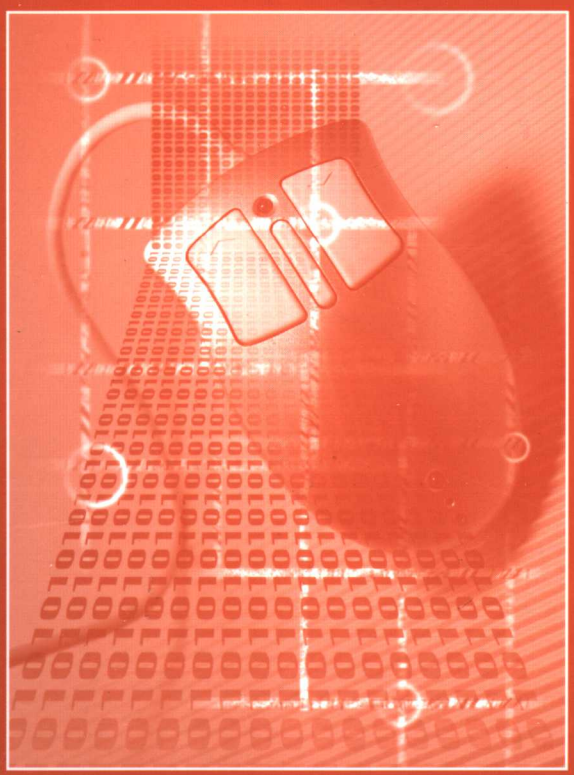
世纪英才
高职高专
计算机系列教材

C语言程序设计



赵海廷　主编
宋海民　李成海　刘成　编著

人民邮电出版社
POSTS & TELECOM PRESS

世纪英才高职高专计算机系列教材

C语言程序设计

赵海廷　主编

宋海民　李成海　刘　成　编著

人民邮电出版社

图书在版编目（CIP）数据

C 语言程序设计 / 赵海廷主编；宋海民，李成海，刘成编著. —北京：人民邮电出版社，2005.2
（世纪英才高职高专计算机系列教材）
ISBN 7-115-12926-6

I. C… II. ①赵… ②宋… ③李… ④刘… III. C 语言—程序设计—高等学校：技术学校—教材 Ⅳ.TP312

中国版本图书馆 CIP 数据核字（2004）第 125582 号

内 容 提 要

本书是按照高职高专计算机应用专业《C 语言程序设计教学大纲》的要求而编写的，系统地介绍了数据类型、变量、数组、运算符、表达式、典型语句、函数、指针、结构体以及文件等内容。本书在编写过程中按照循序渐进、重点突出和难点分散的原则组织内容，并列举了大量的例题和习题。

本书可作为高职高专计算机专业及相关专业的 C 语言程序设计教材，也可作为广大工程技术人员继续教育或自学的教材，还可作为 C 语言培训教材。

世纪英才高职高专计算机系列教材
C 语言程序设计

◆ 主　　编　赵海廷
编　　著　宋海民　李成海　刘　成
责任编辑　刘　朋

◆ 人民邮电出版社出版发行　北京市崇文区夕照寺街 14 号
邮编　100061　　电子函件　315@ptpress.com.cn
网址　http://www. ptpress.com.cn
读者热线　010-67129264
北京朝阳展望印刷厂印刷
新华书店总店北京发行所经销

◆ 开本：787×1092　1/16
印张：19
字数：472 千字　　2005 年 2 月第 1 版
印数：1- 5 000 册　　2005 年 2 月北京第 1 次印刷

ISBN 7-115-12926-6/TP・4349

定价：25.00 元

前　言

C 语言是近年来在国内外得到迅速推广和使用的一种程序设计语言。C 语言以功能丰富、表达能力强、使用方便灵活、目标程序效率高和可移植性强而著称，既具有高级语言的优点，又具有低级语言的许多特点，因此特别适合于编写系统软件。近年来，许多原来用汇编语言编写的软件现在大多都应用 C 语言编写了。而学习和使用 C 语言比学习和使用汇编语言容易得多。

现在，C 语言为广大计算机应用人员所喜爱。不少高等院校不仅在计算机专业开设了 C 语言课程，而且在非计算机专业也开设了 C 语言课程。除此之外，正在工作岗位上的大量工程技术人员也都渴望能尽快地掌握 C 语言，以便于用 C 语言进行相应的程序设计。

由于 C 语言涉及的概念较复杂，规则繁多，使用灵活，不少初学者深感学习困难。在中国，众多的计算机用户并不熟悉英语，因此用英文进行提示的软件不能迅速地得到推广。基于以上原因，用汉字进行提示成为软件迅速普及和推广的先决条件之一。Windows 支持下的汉化 Turbo C 正好圆满地解决了这个问题。作者针对初学者的实际问题，特别是在职工程技术人员的特点，总结多年从事继续工程教育、学历教育、职业技术教育及 C 语言培训的教学经验，编写了《C 语言程序设计》。作者在编写本书时特别注意初学者可能在如下方面出现困难：

① 结构化程序设计语言所固有的变量的存在性和可见性概念是非结构化程序设计语言所没有的。而以非结构化程序设计语言作为第一程序设计语言者，对变量的存在性和可见性往往不太容易理解。本书从概念的引入到举例讲解，乃至多次深入强化训练，使读者能较容易地突破变量的存在性和可见性这一关。

② C 语言具有多达 40 余个运算符，且表达式的构成灵活，运算符优先级和结合性较繁杂，这些困难成为初学者掌握 C 语言的又一障碍。本书力求多举例子，详细分析，使读者能从中领悟其特点，掌握其特性，以帮助初学者闯过表达式这一关。

③ C 语言程序以函数为基本模块，因此函数间的数据传递是学习 C 语言的又一难点。本书以函数间数据的传值方法和传址方法为重点，以帮助初学者解决数据传递这一难点。

④ 指针是 C 语言的一大特色，指针的灵活使用使 C 语言更接近于硬件系统，而免去了不同机种对硬件的束缚。本书详尽地讨论了指针的基本特性和指针的应用方法，以帮助读者尽快闯过指针这一关。

⑤ 结构体是 C 语言构造的数据类型的一种，结构体的灵活应用使 C 语言程序更具特色。本书提供了较多的例子来说明结构体的应用，使读者能尽快掌握结构体。跨过这一关将使读者的编程能力和编程技巧有一个大幅度的提高。

本书具有如下特点：

① 本书是针对计算机应用及其相关专业而编写的，兼顾了其他相近专业与众多的在职工程技术人员和准备参加自学考试的广大读者的需要，同时本书对于通过“C 语言程序设计二级考试”也有帮助。

② 针对 C 语言概念繁杂的特点，本书把 C 语言的难点分散、重点突出，每章既有概念

引入，又有对前面章节难点的再应用，使读者阅读起来循序渐进。

③ 本书把介绍的重点放在语言的使用上，即如何正确运用 C 语言编写程序。书中所举的例子主要是帮助读者如何正确使用 C 语言，而不是把重点放在算法的设计上。对于较难的例题，先分析后讲述程序的设计，并在难于理解之处加以解释或注释。

④ 本书是引导读者进入 C 语言殿堂的钥匙，换句话说，本书只介绍了 C 语言的基础部分，不可能面面俱到地全面讲述。当读者弄懂了本书的内容之后，在编写大型软件或复杂程序时还会遇到一些问题，此时只要查阅有关参考资料是不难解决的。

⑤ 本书中的所有例题和习题的答案是用“Turbo C 2.0”或“VC++ 6.0”编写的，并且已调试通过。由于程序设计语言是实践性很强的课程，故建议读者尽量多上机，以尽快掌握 C 语言的编写方法和提高调试程序的能力。

本教材是按 52～60 教学时数编写的，具体学时分配如下：第 1 章为 4 学时，第 2 章为 3 学时，第 3 章为 3 学时，第 4 章为 6～8 学时，第 5 章为 4 学时，第 6 章为 4 学时，第 7 章为 4～5 学时，第 8 章为 8～10 学时，第 9 章为 2 学时，第 10 章为 6～8 学时，第 11 章为 3 学时，第 12 章为 5～6 学时。

上机实验安排如下：

实验一为 C 语言程序设计基础；实验二为基本数据类型与变量的存储属性；实验三为数组；实验四为运算符和表达式；实验五为顺序、选择与循环结构程序设计；实验六为函数；实验七为指针；实验八为结构体；实验九为联合体、枚举；实验十为文件。每个实验要保证两学时。若想通过“C 语言程序设计二级考试”，则需要增加 20 学时的考前辅导及上机操作训练。

本书在编写过程中，相关领导以及计算机技术教研室的同仁也都曾给予了大力支持和帮助，在此一并深致谢意。欢迎广大读者通过电子邮件 zh-H-T@126.com 与作者联系。

由于作者学识水平有限及经验不足，书中难免存在疏漏之处，恳请广大同行和读者批评指正。

作者

武汉科技大学中南分校

目　录

第1章　C语言程序设计基础

1.1　基 础 知 识

1.1.1　进位计数制及不同数制之间的转换

1. 二进制数

进位计数制是一种计数的方法，最常用的是十进制计数法。一个任意的十进制数都可以表示为 $a_na_{n-1}\cdots a_0.b_1b_2\cdots b_m$，其含义是 $a_n\times10^n+a_{n-1}\times10^{n-1}+\cdots+a_0\times10^0+b_1\times10^{-1}+b_2\times10^{-2}+\cdots+b_m\times10^{-m}$，其中 a_i 和 b_j 是 0、1、2、3、4、5、6、7、8、9 中的一个（i=0,1,…,n；j=1,2,…,m）。

十进制数的基数为 10，即其数码的个数为 10 个，且遵循逢十进一的规则。上式中对应于每位数字的 10^k（k 取整数）称为该位数字的权，所以每位数字乘以其权所得到的乘积之和就是该数的值。例如：$12389.4567=1\times10^4+2\times10^3+3\times10^2+8\times10^1+9\times10^0+4\times10^{-1}+5\times10^{-2}+6\times10^{-3}+7\times10^{-4}$。

十进制数是人们最熟悉、最常用的一种数制，但它不是惟一的数制。例如计时用的分、秒就是按 60 进制计数的。

基数为 r 的 r 进制数的值可以表示为 $a_n\times r^n+a_{n-1}\times r^{n-1}+\cdots+a_0\times r^0+b_1\times r^{-1}+b_2\times r^{-2}+\cdots+b_m\times r^{-m}$，其中 a_i、b_j 可以是 0、1……r−1 中的任一个数码，r^k 则是各位数的权。

计算机中为了便于存储及满足计算机的物理实现，采用了二进制数。二进制数的基数为 2，只有 0、1 两个数码，遵循逢二进一的规则，它的各位权是以 2^k 表示的，因此二进制数 $a_na_{n-1}\cdots.b_1b_2\cdots b_m$ 的值是 $a_n\times2^n+a_{n-1}\times2^{n-1}+\cdots+a_0\times2^0+b_1\times2^{-1}+b_2\times2^{-2}+\cdots+b_m\times2^{-m}$，其中 a_i、b_j 为 0、1 两个数码中的一个。

例如：$101101.01_2=1\times2^5+0\times2^4+1\times2^3+1\times2^2+0\times2^1+1\times2^0+0\times2^{-1}+1\times2^{-2}=45.25_{10}$，其中数的下标表示该数的基数 r。二进制数 101101.01 与十进制数 45.25 是等值的。

n 位二进制数可以表示 2^n 个数，例如 3 位二进制数可以表示 8 个八进制数，如表 1.1 所示。

表 1.1　　3 位二进制数表示的八进制数

二进制数	000	001	010	011	100	101	110	111
对应的八进制数	0	1	2	3	4	5	6	7

4 位二进制数可以表示十进制数 0～15，共 16 个数，如表 1.2 所示。

表 1.2　　4 位二进制数可以表示的十进制数

二进制数	0000	0001	0010	0011	0100	0101	0110	0111
对应的十进制数	0	1	2	3	4	5	6	7
二进制数	1000	1001	1010	1011	1100	1101	1110	1111
对应的十进制数	8	9	10	11	12	13	14	15

为了便于人们阅读与书写，计算机领域经常使用八进制数或十六进制数来表示二进制数。它们的基数和数码如表 1.3 所示。

表 1.3　　十六、十、八及二进制数的基数和数码

进位计数制	基　数	数　码
十六进制数	16	0，1，2，3，4，5，6，7，8，9，A，B，C，D，E，F
十进制数	10	0，1，2，3，4，5，6，7，8，9
八进制数	8	0，1，2，3，4，5，6，7
二进制数	2	0，1

从表中可以看出，23_{10} 可以表示为 17_{16}、27_8 以及 10111_2，1.375_{10} 可以表示为 1.6_{16}、1.3_8 及 1.011_2 等。在计算机中，通常用数字后面跟一个英文字母来表示该数的数制，其中十进制数一般用 D（Decimal）来表示，二进制数用 B（Binary）来表示，八进制数用 O（Octal）来表示，十六进制数用 H（Hexadecimal）来表示，例如 117D、1110101B、0075H 等。当然也可以用这些字母的小写形式，本书中用大写英文字母作为后缀表示该数的数制。

2. 二进制数和十进制数之间的转换

二进制数转换为十进制数的方法是按权展开求和，也就是将各位的二进制数码乘以其对应位的权值后求和。

【例 1.1】　将二进制数 1011100.10111B 转换为对应的十进制数。

$$1011100.10111B=1\times2^6+1\times2^4+1\times2^3+1\times2^2+1\times2^{-1}+1\times2^{-3}+1\times2^{-4}+1\times2^{-5}=92.71875D$$

十进制数转换为二进制数的方法有多种，通常采用的是除 2 取余法，也就是把要转换的十进制数整数部分不断地除以 2，记下余数，直到商为零时为止。

【例 1.2】　将十进制数 N=117D 转换为对应的二进制数。

117/2=58　　$r_0=1$
58/2=29　　$r_1=0$
29/2=14　　$r_2=1$
14/2=7　　$r_3=0$
7/2=3　　$r_4=1$
3/2=1　　$r_5=1$
1/2=0　　$r_6=1$

所以 N=117D=1110101B。需要注意的是转换后的二进制数要按“倒序”方式书写。

对于被转换的十进制数的小数部分，则应不断乘以 2，记下其整数部分，直到小数部分为零或者达到某种精度要求时为止。

【例 1.3】　将十进制数 N=0.8125D 转换为二进制数。

0.8125×2=1.625　　i_1=1

0.625×2=1.25　　i_2=1

0.25×2=0.5　　i_3=0

0.5×2=1.0　　i_4=1

所以 N=0.8125D=0.1101B。需要注意的是，转换后的二进制数要按“顺序”方式书写。

十进制数的小数部分并不一定能完全转化为二进制小数，这与计算机内部表示小数的方法有关。如小数 0.3，无论是多长的字长都不能将其完全表示出来。

3．十六进制数和十进制数之间的转换

众所周知，在计算机内部数的运算和存储都是采用二进制的。但是，二进制数对于人的阅读、书写和记忆都很不方便。十进制数是人们最熟悉的一种计数制，但它与二进制数之间没有直接的对应关系。为了便于人们对二进制数的描述，应该有一种易于与二进制数相互转换的数制。显然，使用 2^n 作为基数的数制是能够适合人们的这种需求的，常用的有八进制数和十六进制数，以下主要介绍十六进制数。

十六进制数有 16 个数码，它们是 0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F，需要用四位二进制数表示，其中 A 表示十进制数的 10，B 表示十进制数的 11……F 表示十进制数的 15。十六进制数中各位的权是 16^k。它与二进制数和十进制数的对应关系如表 1.4 所示。

表 1.4　　二进制数、十进制数和十六进制数的对应关系

二进制数	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
十进制数	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
十六进制数	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

十六进制数转换为十进制数和二进制数转换为十进制数一样，也是按权展开求和，只是各个位的权值是 16^k。

【例 1.4】　将十六进制数 N=F3CH 转换为对应的十进制数。

$$F3CH=15\times16^2+3\times16^1+12\times16^0=3840+48+12=3900D$$

与十进制数转换为二进制数类似，十进制数转换为十六进制数就是把要转换的十进制数的整数部分不断除以 16，记下余数，直到商为零时为止。

【例 1.5】　将十进制数 N=48956D 转换为对应的十六进制数。

48956/16=3059　　r_0=12=C

3059/16=191　　r_1=3=3

191/16=11　　r_2=15=F

11/16=0　　r_3=11=B

所以 N=48956D=BF3CH。

对于要转换的十进制数的小数部分，则应不断乘以 16，记下其正数部分，直到结果的小数部分为零或者达到某种精度要求时为止。

【例 1.6】　将十进制数 N=0.625D 转换为对应的十六进制数。

0.625×16=10.0　　i_0=10=A

所以 N=0.625D=0.AH。

4．十六进制数和二进制数之间的相互转换

由于十六进制数的基数是 2 的 4 次幂，所以这两种数制之间的转换是比较容易的。把一个二进制数转换为十六进制数时，只要从小数点开始，向左和向右以每四位为一组，就可以将其转换为十六进制数。

【例 1.7】 将二进制数 10110111111.10101B 转换为对应的十六进制数。

0101 1011 1111.1010 1000

5 B F . A 8

所以 10110111111.10101B=5BF.A8H。

反之，十六进制数转换为二进制数就是把一位十六进制数用 4 位二进制数表示，即可得到对应的二进制数。

【例 1.8】 将十六进制数 3F5D.A7H 转换为对应的二进制数。

3 F 5 D .A 7

0011 1111 0101 1101.1010 0111

所以 3F5D.A7H=11111101011101.10100111B。

1.1.2 计算机中数和字符的表示方式

计算机中的数据信息可以划分为两类：数值数据和非数值数据。数值数据有确定的值，它表示数的大小，能在数轴上找到它们的确切位置。非数值数据一般用来表示符号或文字，它们没有值的含义。

1．数的机器码表示法

二进制数具有运算简单、便于物理实现和节省资源等优点，所以被计算机普遍采用。二进制数和十进制数一样都有正负之分，在计算机中常采用数的符号和数值一起编码的方法来表示数据。常用的有原码、反码和补码表示方法。这几种表示方法都将数据的符号数值化，通常正号用“0”表示，负号用“1”表示。把带有正、负号的数称为真值，而把数据在计算机内的机器码称为机器数。

原码表示法是一种比较直观的表示方法，其符号位表示该数的符号，而数值部分仍然保持其真值的特征。

对于正数 $x=+0.x_1x_2\cdots x_n$，则$[x]_{原}=0.x_1x_2\cdots x_n$；对于负数 $x=-0.x_1x_2\cdots x_n$，则$[x]_{原}=1.x_1x_2\cdots x_n$。

若 x 的原码形式为 $x_0.x_1x_2\cdots x_n$，其中 x_0 为符号位，则原码表示的定义是：

$$[x]_{原}=\begin{cases} x & 0\leqslant x<1 \\ 1-x \text{ 或 } 1+|x| & -1<x\leqslant 0 \end{cases}$$

式中，$[x]_{原}$是机器数，x 是真值。

【例 1.9】 x=+0.1010，则$[x]_{原}=0.1010$

y=−0.1010，则$[y]_{原}=1.1010$

原码表示法有两个特点：一是零的表示有“+0”和“−0”之分，它的原码表示形式分别是$[+0]_{原}=0.000\cdots000$，$[-0]_{原}=1.000\ldots000$；二是符号位 x_0 的取值由下式决定。

$$x_0=\begin{cases}0 & 0\leqslant x<1\\ 1 & -1<x\leqslant 0\end{cases}$$

反码也是计算机中的一种机器数。在反码表示法中，符号位的表示方法与原码相同，而在其数值部分，正数的反码与正数的原码相同，负数的反码是其原码的按位取反值。

若 x 的反码形式为 $x_0.x_1x_2\cdots x_n$，其中 x_0 是符号位，则反码表示法的定义是：

$$[x]_{反}=\begin{cases}x & 0\leqslant x<1\\ (2-2^{-n})-|x| & -1<x\leqslant 0\end{cases}$$

式中，n 代表小数点后的位数。

对于正数 $x=+0.x_1x_2\cdots x_n$ 则 $[x]_{反}=0.x_1x_2\cdots x_n$；对于负数 $x=-0.x_1x_2\cdots x_n$ 则 $[x]_{反}=1.x_1x_2\cdots x_n$。

【例 1.10】 x=+0.1101，则$[x]_{反}$=0.1101

x=−0.1101，则$[x]_{反}$=1.0010

对于 0，其反码也有“+0”和“−0”之分，它们的反码表示形式分别为：$[+0]_{反}=0.00\cdots0$，$[-0]_{反}=1.11\cdots1$。

补码是计算机中使用得非常普遍的编码，它主要解决了将减法转换为加法这样的问题。例如：假设现在的标准时间是 6 点整，而有一只手表为 11 点，为了校准时间，可以将时针往逆时针方向拨 5 个小时，也可以将时针往顺时针方向拨 7 个小时。这两种方法都可以校准手表的时间。若将逆时针方向拨动时针称为减操作的话，则顺时针方向拨动时针就应该是加操作了。我们将其记为“−5”和“+7”，在校准时间上“−5”和“+7”是等效的。这种关系记作：

$$-5=+7(\text{mod}12)$$

其含义是−5 与+7 对于模 12 来说是互补的，或者说以 12 为模时−5 的补码为+7。

模也称为模数，是一个计量单位。记作 mod 或 M。同理，在模为 12 时，−2 的补码是 10，−4 的补码是 8，等等。

由此可见，负数的补码可以用模加上该数得到。

【例 1.11】 以 10 为模计算 7−2。

以 10 为模时−2 的补码是 8，即 7−2=5 和 7+8=15 是等价的。只是后者的结果多了一个模值。在模为 10 的情况下，10 与 0 是等价的。因此，只要将上述结果中的十位的 1 作零处理或丢掉，就可以得到正确的结果。

在计算机中使用补码运算。对于二进制纯小数$\pm0.x_1x_2\cdots x_n$，可用模 2 获得补码。下面以模为 2 来讨论补码的表示方法。

确定模以后，数 x 对 2 的补码定义为$[x]_{补}=2+x \pmod 2$。若 x>0，则将模 2 舍去，因为正数的补码就是其本身，形式上与原码相同。

【例 1.12】 若 x=0.1010，则$[x]_{补}$=2+0.1010=0.1010

若 x=−0.1010，则$[x]_{补}$=2+(−0.1010)=1.0110

一般而言，若数 x 的补码形式为 $x_0.x_1x_2\cdots x_n$，其中 x_0 是符号位，则有：

$$[x]_{补}=\begin{cases}x & 0\leqslant x<1\\ 2+x=2-|x| & -1\leqslant x<0\end{cases}$$

式中，$[x]_{补}$表示 x 的补码，即机器数，x 为真值，2 为模。

对于正数：$x=+0.x_1x_2\cdots x_n$，则$[x]_{补}=0.x_1x_2\cdots x_n$。

对于负数：$x=-0.x_1x_2\cdots x_n$，则$[x]_{补}=10.00\cdots0-0.x_1x_2\cdots x_n$。

对于0：在补码的定义下只有一种表示形式，即$[+0]_{补}=[-0]_{补}=0.00\cdots0$。

采用补码表示法做减法比采用原码法方便得多，不论参加运算的数是正数还是负数，都可以用加法实现。但是，根据补码的定义，求负数的补码时，要从模中减去负数的绝对值，这就需要做一次减法运算。显然，这样做不太方便。

比较求负数的反码和补码的公式$[x]_{反}=2-2^{-n}-|x|$和$[x]_{补}=2-|x|$，可得$[x]_{补}=[x]_{反}+2^{-n}$。

这就是通过反码求得补码的公式。这个公式说明，求一个负数的补码时，其方法是符号位置1，其余各个位0变1，1变0，然后在最末位（2^{-n}）上加1。数值位各位取反的逻辑实现很方便，从触发器的反相端输出即可得到。这样就避免了求补码或反码时做减法带来的麻烦。

从上面的讨论不难发现，正数的原码、反码和补码的符号位均为0，数值部分就是其机器数。负数的原码的符号位为1，数值部分为其机器数；负数的反码的符号位为1，数值部分将各个位按位取反；负数的补码的符号位为1，数值部分是其反码再在最末位加1。

【例1.13】 x=0.101101，求x的原码、反码和补码。

根据定义有：$[x]_{原}=0.101101$，$[x]_{反}=0.10110$，$[x]_{补}=0.101101$。

【例1.14】 x=−0.101101，求x的原码、反码和补码。

根据定义有：$[x]_{原}=1.101101$，$[x]_{反}=1.010010$，$[x]_{补}=1.010011$。

【例1.15】 将下列机器数转换为真值。

$[x]_{原}=0.101110$，则$x=0.101110$

$[x]_{原}=1.101010$，则$x=-0.101010$

$[x]_{反}=1.101010$，则$x=-0.010101$

$[x]_{反}=0.110110$，则$x=0.110110$

$[x]_{补}=0.110110$，则$x=0.110110$

$[x]_{补}=1.110110$，则$x=-0.001010$

2. 无符号整数

在某些情况下，若要处理的数据全是正数，那么再保留符号位已没有意义了。此时，可以把最高位的“符号位”作为数值来处理，这样的数称为无符号整数。16位有符号数的取值范围为−32768～32767，而16位无符号数的取值范围是0～65535；8位有符号数的表示范围为−128～127，而8位无符号数的表示范围是0～255。

在计算机中最常用的无符号数是表示地址的数据。此外，双精度数的低位字也是无符号数。在某些情况下，有符号数在计算机中用补码表示，它与无符号数在处理上是有差别的，在处理数据时要注意它们之间的差别。

3. 字符的表示法

计算机中处理的信息并不全是数，有时需要处理字符或字符串，例如从键盘输入的信息或打印机输出的信息都是字符方式，因此计算机必须能表示字符。

西文字符包括：

① 英文字母：A、B……X、Y、Z，a、b……x、y、z。

② 数字符号：0、1、2……8、9。

③ 专用字符：+、−、*、/等。

④ 非打印字符：BEL（Bell，响铃）、LF（Line Feed，换行）、CR（Cariage Return，回车）等。

这些字符在机器中必须用二进制数表示。一般计算机中最常用的是美国信息交换标准代码（American Standard Code for Information Interchange，ASCII），这种代码用一个字节（8位二进制）来表示一个字符。ASCII 码采用 7 位二进制代码来对字符编码，它包括 32 个标点及符号、10 个阿拉伯数字、52 个英文大小写字母和 34 个控制字符，共 128 个。例如，阿拉伯数字 0～9 的 ASCII 码分别是 30H～39H，英文大写字母 A～Z 的 ASCII 码是 41H～5AH。并不是所有的 ASCII 码都可以显示或打印，如回车、换行等。控制字符只完成一个控制动作。

在计算机内部，每个 ASCII 码都占用一个字节的存储空间，通常其最高位为逻辑“0”，低 7 位作为字符的二进制编码。有时，在实际操作中把最高位当作奇偶校验位，用来检验代码在存储和传送过程中是否发生错误。

4．汉字的表示法

西文是拼音文字，通常用有限的字母（如英文为 26 个字母，俄文为 32 个字母）就可以拼写出全部西文信息。因此，仅需要对有限个数的字母进行编码，就可以将全部西文信息输入计算机。而汉字信息则不一样，汉字是象形文字，一个汉字就是一个方块图形，计算机要对汉字信息进行处理，就必须对数目繁多的汉字进行编码，建立起一个上万汉字的编码表。因此，汉字的编码远比西文的编码复杂。

汉字编码有内码和外码之分。外码又称汉字输入编码，用以实现汉字的输入方式。目前我国公布的汉字编码有上百种，其编码的方法可以按照汉字的字形、字音和音形结合分为三类。常用的输入方法有区位码、国标码、首尾码、拼音码、五笔字型码、自然码、智能 ABC 码、清华紫光和郑码等。内码是对计算机系统内部进行汉字信息的存储、交换、检索等操作的编码。汉字内码采用 2 字节表示，没有重码，它要求与国标码有简单的对应关系。

所谓的国标码（又称为交换码）是《国家标准信息交换用汉字编码基本字符集》的简称。它是我国国家标准局于 1981 年，为适应计算机对汉字信息交换的处理而颁布的国家标准，编号为 GB2312-80。该标准按 94×94 的二维代码表形式，收集了 6763 个汉字和 682 个字符、序号、数字、拉丁字母、希腊字母和汉语拼音符号等，共计 7445 个图形字符。该标准最多可包含 8836 个图形字符，适应了一般汉字处理、汉字通信等系统之间的信息交换。应该指出的是，汉字的输入编码和内码是两个不同的概念，不可混为一谈。

区位码：用每个汉字在二维代码表中的行、列位置来表示的代码，行号称为区号，列号称为位号，区位码是汉字的输入码。例如，汉字“中”位于第 54 区 48 位，其区位码为 5448。

国标码：国标码=区位码+32，即区号和位号各加 32 以后所得到的双七位二进制编码。国标码用于不同汉字系统之间汉字的传输和交换。

机内码：英文 DOS 的机内码是 ASCII 码，国标码是双七位二进制编码，把它们用作内码时将会与 ASCII 码相混淆。为此可利用 ASCII 码的最高位为“0”这一特点，把两字节国标码的每一个字节的最高位置“1”，以示区别。这样，形成了汉字的另一种编码方法，即汉字的机内码。简单地说，机内码=国标码+128，或者机内码=区位码+160。