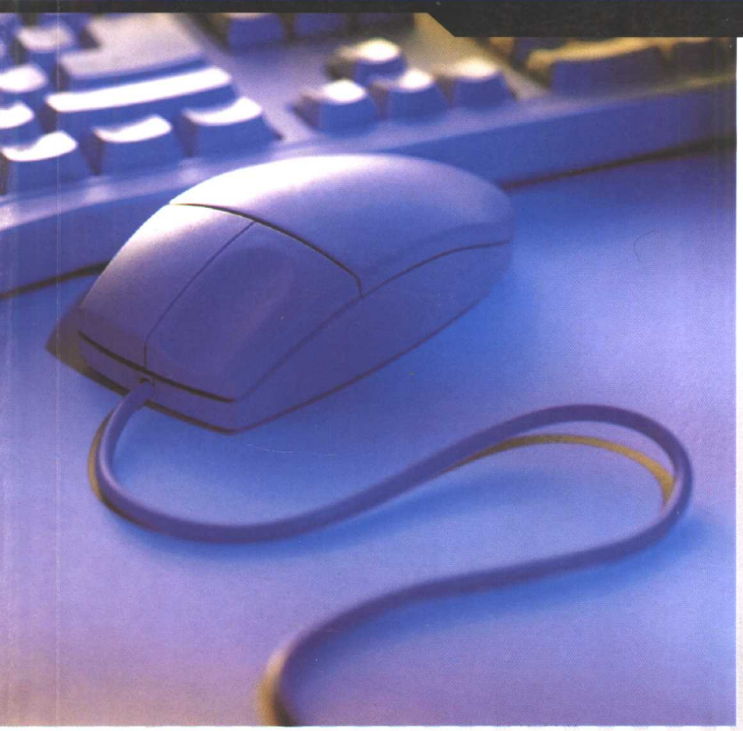
21世纪大学计算机基础教学 “1+X”改革系列教材

JISUANJICHENGXUSHEJIJICHU
C YUYANCHENGXUSHEJI

计算机程序设计基础
C语言程序设计



马德骏 张建宏 汤练兵 主编

科学出版社
www.sciencep.com

·21世纪大学计算机基础教学“1+X”改革系列教材·

计算机程序设计基础
——C语言程序设计

马德骏　张建宏　汤练兵　主编

科学出版社

北京

内 容 简 介

本书为高等院校非计算机专业初级计算机语言教材，主要面向非计算机专业初学程序设计的读者。全书共 11 章，前十章介绍了 C 语言的基本知识、基本算法和基本程序设计方法，第 11 章介绍了一些实用的示例。本书以 Turbo C 为主要对象，并在部分章节中适当兼顾介绍 VC++的面向过程部分的程序设计方法，为读者今后向面向对象程序设计语言平滑过渡打下基础。

本书通俗易懂，便于自学。除书中配有习题，本书还有配套的实验与习题教材，以帮助读者学习和掌握书中的各个知识点。

本书适用于大学本、专科非计算机专业学生，也可供高等职业技术学院、网络学院、成教学院学生以及计算机等级考试者、培训班学员、C 语言自学者使用。

图书在版编目（CIP）数据

计算机程序设计基础——C 语言程序设计/马德骏，张建宏，汤练兵主编. -北京：科学出版社，2006
（21 世纪大学计算机基础教学“1+X”改革系列教材）
ISBN 7-03-017959-5

Ⅰ.计… Ⅱ.①马…②张…③汤… Ⅲ.C 语言－程序设计 Ⅳ.TP312

中国版本图书馆 CIP 数据核字（2006）第 101723 号

责任编辑：王雨舸／责任校对：王望荣
责任印制：高 嵘／封面设计：曹 刚

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
湖北新华印务有限公司印刷
科学出版社发行 各地新华书店经销
*
2006 年 8 月第 一 版 开本：787×1092 1/16
2006 年 8 月第一次印刷 印张：15 1/4
印数：1~8 000 字数：374 000
定价：23.80 元
（如有印装质量问题，我社负责调换）

前　言

计算机文化、技术和应用水平是当今衡量一个人的知识水平和能力的重要标准之一。程序设计课程是计算机基本技能教育和能力培养的一个重要组成部分，它能培养学生利用计算机解决实际问题的能力，也能为学生在后继课程以及工作中应用计算机解决实际问题打下良好的基础。C 语言已成为大多数学校理工专业所选的程序设计课程的教学语言，也是在各种计算机证书考试中为大多数人所选择的程序设计语言。

《计算机程序设计基础——C 语言程序设计》是专为初学程序设计者编写的一本教材，全书共 11 章，前十章介绍了 C 语言的基本知识、基本算法和基本程序设计方法，内容分别为：C 语言程序设计基础知识，基本数据类型及其运算，顺序结构程序设计，选择结构程序设计，循环结构程序设计，数组，函数，指针，结构体、共用体和枚举，文件；第 11 章介绍了一些实用的示例。本教材力求通俗易懂，便于自学，同时也考虑到使教师在教学中易于拓宽。根据系统平台的发展和程序设计方法的发展，本教材以 Turbo C 为主要对象，并在部分章节中适当兼顾介绍 VC++的面向过程部分的程序设计方法，即“控制台应用程序”，目的是使读者在了解一般的 C 语言程序设计知识的同时，接触到字符界面和图形界面的两种不同开发环境，初步了解面向过程和面向对象开发方式上的差异，为读者今后向面向对象程序设计语言 VC++平滑过渡打下基础。在 VC++集成环境中可以弥补 TurboC 环境下编辑功能的不足，突破 Turbo C 环境下的汉字系统平台限制。对于 C 语言程序设计的学习，除了理论教学外，习题和实验环节是必不可少的，在与本教材配套的《计算机程序设计基础——C 语言程序设计实验与习题》一书中给出了大量精心设计的实验和习题及其参考答案，以配合对本教材各个知识点的学习和掌握。

本书主要针对计算机语言的初学者编写，适用于高等院校非计算机专业本、专科学生，也可供高等职业技术学院、网络学院、成教学院学生以及计算机等级考试者、培训班学员、C 语言自学者学习使用。

本书由马德骏、张建宏、汤练兵主编。第 1、2 章由张建宏编写，第 3、9 章由汤练兵编写，第 4、5、6、7 章由马德骏编写，第 8 章由陈志铭编写，第 10 章由杨朝阳编写，第 11 章由李捷编写。在制定本书编写大纲的过程中，马成前、王舜燕、汤英等同志给予了大力支持，提出了许多宝贵意见，在此表示诚挚的谢意！

由于水平和时间的限制，书中难免存在疏漏及不足之处，敬请读者和同行专家不吝赐教。

编　者

2006 年 6 月

目　录

第 1 章　C 语言程序设计基础知识

1.1　基 本 知 识

1.1.1　信息的表示

1. 数制

在日常生活中，十进制是人们习惯使用的数制，而在计算机中主要使用二进制，这是由计算机的物理特性所决定的。二进制具有数符少、容易表示、运算简单可靠、便于物理实现、节省设备等优点。所有计算机系统中的信息均以二进制编码形式表示、保存、传输和处理，由此可见二进制在计算机系统中的作用是非常重要的。以下简要介绍计算机系统中常见的几种不同数制系统的概念、表示方法、它们之间的转换以及运算等有关知识。

数制也称为计数制，是指用一组固定的符号和统一的规则来表示数值的方法。按进位的方法进行计数，称为进位计数制。

在进位计数制中有基数、数位、位权三要素。

基数是指在采用进位计数的数值系统中，如果用 R 个基本符号(即数码或数符)表示数值，则称其为基 R 数制(radix-R number system)，R 称为该数制的基数(radix)。计数规则为每个数位计满 R 就向高位进一，即逢 R 进一。

数位是指数码在一个数中所在的位置(记为 n，n 为整数)。

位权是指在某种进位计数制中，每个数位上的数码所代表的数值的大小，等于在这个数位上的数码乘以一个固定的数值，这个固定的数值就是这种进位计数制中该数位上的位权，第 n 位数的位权为 R^n。

一般地，任意一个有 n 位整数和 m 位小数的 R 进制数 N 可以表示为

$$N=(k_{n-1}k_{n-2}k_{n-3}\cdots k_1k_0k_{-1}k_{-2}\cdots k_{-m}) \tag{1-1}$$

或

$$N=\sum_{i=n-1}^{-m}k_i\times R^i \tag{1-2}$$

这里，R 为基数($R\geqslant 2$ 的整数)，R^i 为第 i 位数位的位权值。k_i ($i\in[-m, n-1]$)为该进位计数制使用的某个数码($0\leqslant k_i<R$)，即 k_i 值只能是 0 到 R−1 这些数字符号中的一个，如在十进制数制中，k_i 的取值只能是 0~9 之间的某一整数。

(1) 十进制数。十进制数制是进位计数制，有 0，1，2，3，…，9 这 10 个数字符号(数码)，位权为 10^i ($i\in[-m, n-1]$)。

在一个十进制数中，数字(数码)所在位置(数位)不同，则所代表的数值也不同。由式(1-2)，任何一个十进制数可以展开成为一个不同数位上的数字(数码)乘以该数位的位权值的和。例如，345 中的 3，处于百位上，权值为 10^2，即用 3×10^2 表示。以此类推，如将 234.56 写成按权展开式，即

$$234.56 = 2\times10^2 + 3\times10^1 + 4\times10^0 + 5\times10^{-1} + 6\times10^{-2}$$

在进行加、减法运算时，十进制遵循“逢十进一，借一当十”的规则。

(2) 二进制数。二进制也属于进位计数制。二进制仅有 0 和 1 两个数字符号，位权为 2^i ($i\in[-m, n-1]$)。例如，二进制数$(1110.1)_2$按式(1-2)展开表示为

$$1\times2^3 + 1\times2^2 + 1\times2^1 + 0\times2^0 + 1\times2^{-1}$$

进行加法、减法运算时，二进制遵循“逢二进一，借一当二”的规则。

例如，对于$(1110.1)_2$+$(1100.1)_2$和$(1110.1)_2$–$(1101.1)_2$，如同十进制加减法运算：首先，小数点对位，而后，逐位相加或相减。**注意**：要遵循二进制的规则。

$$\begin{array}{r} 1110.1 \\ +1100.1 \\ \hline 11011.0 \end{array} \qquad \begin{array}{r} 1110.1 \\ -1101.1 \\ \hline 0001.0 \end{array}$$

至于二进制乘法和除法，与十进制乘除法类似，这里不再叙述。

(3) 八进制数。八进制有 0, 1, 2, 3, 4, 5, 6, 7 这 8 个数字符号，权值为 8^i ($i\in[-m,n-1]$)。例如，八进制数$(203.1)_8$按式(1-2)展开表示为

$$2\times8^2 + 0\times8^1 + 3\times8^0 + 1\times8^{-1}$$

进行加法、减法运算时，八进制遵循“逢八进一，借一当八”的规则。

(4) 十六进制数。十六进制有 0，1，2，3，4，5，6，7，8，9，A，B，C，D，E，F 这 16 个数字符号，权值为 16^i ($i\in[-m,n-1]$)。例如，十六进制数$(2A3.1)_{16}$按式(1-2)展开表示为

$$2\times16^2 + A\times16^1 + 3\times16^0 + 1\times16^{-1}$$

注意：在十六进制中，A~F 这 5 个数字符号分别代表的数值是十进制的 10~15，且大小写等价。

在进行加法、减法运算时，十六进制遵循“逢十六进一，借一当十六”的规则。

(5) 不同数制间的转换。

1) 二进制、八进制、十六进制数转换为十进制数。转换时，通常按位权法进行，位权值按十进制运算法则计算，并将各位上的数码(按十进制值)乘以该位的权值再进行累加所得到的数就是相应的十进制数。如：

$$(1110.01)_2 = 1\times2^3 + 1\times2^2 + 1\times2^1 + 0\times2^0 + 0\times2^{-1} + 1\times2^{-2} = (14.25)_{10}$$

$$(125.11)_8 = 1\times8^2 + 2\times8^1 + 5\times8^0 + 1\times8^{-1} + 1\times8^{-2} = (85.140625)_{10}$$

$$(b1.2)_{16} = 11\times16^1 + 1\times16^0 + 2\times16^{-1} = (177.125)_{10}$$

2) 二进制、八进制、十六进制数之间的转换。

• **二进制数转换成八进制或十六进制数**　因为二进制、八进制、十六进制之间有内在的对应关系：2^3=8，2^4=16。3 个二进制位所能表示的最大整数为 7，最小非负整数为 0；4 个二进制位则相应为 15 和 0。由于有这种对应关系的存在，因此在将一个二进制数转换为八进制数时，对其整数部分从小数点开始向左三位一组，进行划分，高位不够三位时，左边补 0；对其小数部分从小数点开始向右三位一组，低位不够三位时，右边补 0；然后，再将各组二进制数分别转换成八进制数即可。将一个二进制数转换为十六进制数时，也采用上述方法，只是分组时是 4 个二进制位一组即可。如：

$$(1010111.01111)_2 = (001\,010\,111.011\,110)_2 = (127.36)_8$$

$$(1011111.01111)_2 = (0101\,1111.0111\,1000)_2 = (5F.78)_{16}$$

• **八进制数、十六进制数转换成二进制数**　将八进制数转换成二进制数时，只需将八进

制数的各个数字分别用 3 个二进制数位来表示；将十六进制数转换成二进制数时，只需将十六进制数的各个数字分别用 4 个二进制数位来表示即可。如：

$$(32.12)_8 = (011\ 010.001\ 010)_2 = (011010.001010)_2$$

$$(9A.0C)_{16} = (1001\ 1010.0000\ 1100)_2 = (10111010.00001100)_2$$

说明：在将各个数字转换为三位(或四位)二进制数位表示的过程中，若某个数字变换后不够三位(四位)，则要补 0 以凑够三位(四位)。例如，将八进制数$(2)_8$转换成二进制数时，应为$(010)_2$，而不能是$(10)_2$。若使用不当，则有可能产生错误。

·八进制数、十六进制数间的转换 通常，它们之间的转换是利用二进制数作为过渡进行变换的，即先将一种数据转换成二进制数，再将该二进制数转换成另一种数据。如：

$$(A5)_{16} = (1010\ 0101)_2 = (010\ 100\ 101)_2 = (245)_8$$

3) 十进制数转换成二进制、八进制、十六进制数。

·十进制数转换成二进制数 将十进制数转换成二进制数时，采用分别对整数部分和小数部分进行转换的方法。

对于整数部分，采用“除二取余”的方法：将该十进制数整数部分除以 2，所得余数作为二进制数整数部分的最低位，也就是小数点左边第一位(个位)；再将商作为被除数除以 2，所得余数作为小数点左边第二位。以此类推，可以得到一系列的余数，按其先后次序，自小数点起，从右向左排列，就构成了二进制数的整数部分。对于小数部分，采用“乘二取整”的方法：将该十进制数小数部分乘以 2，取积的整数部分，作为二进制数小数点后的第一位；再将积的小数部分乘以 2，取其整数部分，作为二进制数小数点后第二位。以此类推，就可得到二进制数的小数部分。将整数部分和小数部分合并，即得到对应的二进制数。

例如，求$(28.125)_{10}$所对应的二进制数，处理过程如下：

		余数			
2	28	0……最低位	$0.125\times2=0.25$	……0	最高位
2	14	0	$0.25\times2=0.5$	……0	
2	7	1	$0.5\times2=1.0$	……1	最低位
2	3	1			
	1 ……	最高位			

整数部分：$(28)_{10}=(11100)_2$　　小数部分：$(0.125)_{10}=(0.001)_2$

合并整数部分和小数部分后得到：$(28.125)_{10}=(11100.001)_2$

·十进制数转换成八进制、十六进制数 原则上与十进制数转换成二进制数的方法一样，只是除法运算时是除以 8 或 16 取余，乘法运算时是乘以 8 或 16 取整。建议实际转换时，先转换成二进制数，再由二进制数转换成八进制或十六进制数，这样做简单一些。

例如，求$(10.25)_{10}$所对应的八进制数和十六进制数。

$$(10.25)_{10}=(1010.01)_2=(001010.010)_2=(12.2)_8=(1010.0100)_2=(A.4)_{16}$$

说明：在进行小数部分转换过程中二进制数位分组时，低位不足三位(转换成八进制)或四位(转换成十六进制)，应补 0 凑够三位(四位)后再转换。例如，将二进制数$(.01)_2$转换成十六进制数时，要先写成$(.0100)_2$，再转换成$(.4)_{16}$。若不补 0 凑成四位二进制位而直接转换，则可能会产生错误。

2. 数据的编码

计算机中的数据分为两大类：数值型数据和非数值型数据。数值型数据是指可以进行算术运算的数据，如$(256)_{10}$、$(0111.101)_2$等都是数值型数据；非数值型数据通常不参加算术运算，如字符串“你好，世界”、“2002.6.15”等，就是典型的非数值型数据。

(1) 数值型数据的表示。在计算机中表示一个数值型数据，首先需要解决数的长度、符号和小数点位置的表示等问题。通常，在计算机中，使用固定长度的二进制位数来表示数值型数据，如：用 8 个、16 个、32 个二进制位来表示一个数值型数据。在计算机中，数值型数据的小数点位置总是隐含的，以便节省存储空间。隐含的小数点位置可以是固定的，也可以是可变的。前者称为定点数(fixed point number)，后者称为浮点数(floating point number)。

• 定点数 在计算机中，整数是按定点数格式存放的。定点数有两种表示方法，即定点整数和定点小数(纯小数)。由于在 Turbo C 中，主要用到定点整数，故在此只介绍定点整数的表示方法，对定点小数感兴趣的读者可阅读其他有关资料，在此不再赘述。

有符(signed)定点整数格式如图 1-1 所示。其中，最左边的一位二进制位是符号位，若该位为 0，则表示该数为正数；若为 1，则表示为负数。其余为数值部分，小数点隐含在最右边。无符(unsigned)定点整数格式如图 1-2 所示。其中，全部二进制位用来表示数值部分，小数点隐含在最右边。

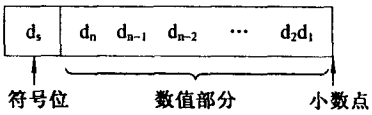


图 1-1 有符定点整数格式

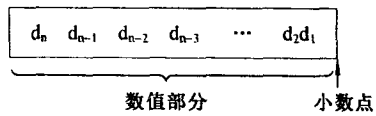


图 1-2 无符定点整数格式

• 浮点数 在计算机中，浮点数的格式如图 1-3 所示。其中，浮点数由阶码和尾数两部分组成，阶码是整数，阶符和阶码的位数 n 合起来确定浮点数的表示范围以及小数点的位置；尾数是小数，其位数 m 确定浮点数的精度(有效数字的位数)；尾数的符号 d_s 是浮点数的符号。

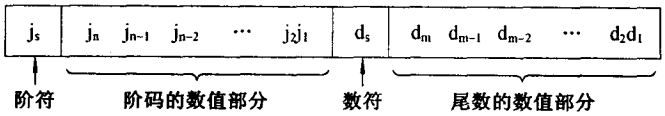


图 1-3 浮点数格式

在解决了数据表示方法的问题以后，还有一个值得考虑的就是数据的编码问题。所谓编码问题就是数据在计算机内如何表示的问题，如：符号位如何表示？它对运算有何种影响等？常用的编码方法有原码、反码和补码方法，采用何种编码方法对于计算机的运算效率有很大影响。为了区分数据的书面表示和计算机内的编码表示，将前者称为数据的真值，后者称为数据的机器码。

• 原码表示法 所谓定点数原码表示就是用最高位充当符号位以表示数的符号，0 表示正数，1 表示负数，其余的部分用二进制数形式来表示数的绝对值。假定用八位二进制位来表示一个整数 a 的原码，如：

$$a_{真值}=(+51)_{10}=(+00110011)_2 \quad 则 \quad [a]_{原}=00110011$$

$$a_{真值}=(-126)_{10}=(-01111110)_2 \quad 则 \quad [a]_{原}=11111110$$

原码表示法在八位二进制位中表示范围：$-127 \leqslant a \leqslant +127$。

原码表示法中，0 有两种编码形式：

$$[+0]_{原}=00000000 \qquad [-0]_{原}=10000000$$

• 反码表示法 我们可以通过一个二进制整数的原码得到其反码。即：当 $a \geqslant 0$ 时，$[a]_{原}=[a]_{反}$；当 $a<0$ 时，则保持$[a]_{原}$的符号位不变，其余各个二进制位逐位取反，即 0 变 1，1 变 0。

假定用八位二进制位来表示一个二进制整数 a 的反码，如：

$$[a]_{原}=00110011 \quad 则 \quad [a]_{反}=00110011$$

$$[a]_{原}=10110011 \quad 则 \quad [a]_{反}=11001100$$

反码表示法在八位二进制位中表示范围：$-127 \leqslant a \leqslant +127$。

反码表示法中，0 有两种编码形式：

$$[+0]_{反}=00000000 \quad [-0]_{反}=11111111$$

·补码表示法 类似地，我们可以通过一个二进制整数的反码得到其补码。即：当 $a \geqslant 0$ 时，$[a]_{补}=[a]_{反}$；当 $a<0$ 时，则 $[a]_{补}=[a]_{反}+1$。如：

$$[a]_{反}=00110011 \quad 则 \quad [a]_{补}=00110011$$

$$[a]_{反}=10110011 \quad 则 \quad [a]_{补}=10110100$$

补码表示法在八位二进制位中表示范围：$-128 \leqslant a \leqslant +127$。

补码表示法中，0 的编码形式唯一：$[0]_{补}=00000000$。补码表示法中，在原码编码中-0 的编码用来表示整数-128，即 $[-128]_{补}=10000000$。

注意：在补码表示法中，0 的表示是唯一的。

在上述介绍中，我们是以八位二进制位为例，同样，我们可以将其应用推广到十六位，三十二位等二进制位中去，这里不再叙述。

·补码的加、减法 补码的加法公式：

$$[a+b]_{补}=[a]_{补}+[b]_{补}$$

补码的减法公式：

$$[a-b]_{补}=[a]_{补}-[b]_{补}=[a]_{补}+[-b]_{补}$$

例 1.1 $a=[11]_{10}=[00001011]_{原}$，$b=[5]_{10}=[00000101]_{原}$，运用补码运算规则，求 a+b。

因为 $[a]_{补}=00001011$，$[b]_{补}=00000101$，则

$$\begin{array}{r} 00001011 \\ +00000101 \\ \hline 00010000 \end{array} \quad \text{正数的补码}\rightarrow$$

由于补码运算的结果仍然是补码，而上述结果最高位为 0，是正数的补码，正数的原码、反码、补码相同。故

$$[a+b]_{补}=[a]_{补}+[b]_{补}=[00010000]_{补} \quad [a+b]_{反}=[00010000]_{反}$$

$$[a+b]_{原}=[00010000]_{原} \quad [a+b]_{10}=[16]_{10}$$

例 1.2 $-a=[-11]_{10}=[10001011]_{原}$，$b=[5]_{10}=[00000101]_{原}$，运用补码运算规则，求 b-a。

因为 $[-a]_{补}=11110101$，$[b]_{补}=00000101$，则

$$\begin{array}{r} 00000101 \\ +11110101 \\ \hline 11111010 \end{array} \quad \text{负数的补码}\rightarrow$$

由于补码运算的结果仍然是补码，而上述结果最高位为 1，是负数的补码，负数的补码减去 1 得到其反码，进而对其反码取反得到其原码。故

$$[b-a]_{补}=[b]_{补}+[-a]_{补}=[11111010]_{补}$$

$$[b-a]_{反}=[11111010]_{补}-1=[11111001]_{反} \quad [b-a]_{原}=[10000110]_{原}$$

$$[b-a]_{10}=[-6]_{10}$$

在 Turbo C 中，有符整数是按有符定点整数格式表示的，无符整数是按无符定点整数格

式表示的。

(2) 字符型数据的表示。在计算机所使用的数据中，还有另一种类型的数据，即字符型数据，它包括字母、文字、符号、数字等。由于计算机内部所有的信息都是以二进制形式存放，所以字符型数据必须进行处理并以二进制形式表示，才能为计算机存储并操作。为了便于识别和统一使用字符数据，对于任意计算机可以识别的字符，都使用一套特定编码规则，使得每个字符与一个唯一的编码一一对应，即一个编码表示一个字符。目前，在计算机中流行各种不同的编码规则，如对于英文字符、符号等字符有 ASCII 码、BCD 码，对于中文有 GB2312 国标码等各种不同的编码规则，而且这些编码规则一般是国家标准或国际标准，为国家或国际上所承认并且执行。

·ASCII 码 ASCII 码(American standard code for information interchange，美国信息交换标准编码)是计算机系统中广泛使用的一种字符编码。该编码已经国际标准化组织所采纳，成为国际间通用的信息交换标准编码。目前国际上流行的是 ASCII 编码的七位版本，即用一个字节的低七位表示一个字符，高位充零。7 个二进制位可表示 128 种状态，故可用来表示 128 个不同的字符。在 ASCII 编码的七位版本中有 33 个通用控制字符、95 个可打印显示的字符(其中 10 个数字、52 个大小写英文字母、33 个标点符号和运算符号)。ASCII 码字符集见附录 I。

·国家标准汉字编码 GB2312—80(国家标准汉字编码)是常用的汉字编码标准，它收录了 6763 个常用汉字，同时根据这些汉字使用频率的高低，又将它们分成两部分：一部分称为一级汉字共 3755 个，即最常用的汉字；另一部分称为二级汉字共 3008 个，为次常用的汉字。GB2312—80 还收录了一些数字符号、图形符号、外文字母等。

GB2312—80 规定用连续的两个字节来表示一个汉字，并且只用各个字节的低七位，最高位未定义，这样的编码方式就有可能与 ASCII 码字符编码产生冲突。因为就单个字节来说，两种编码规则都只用到一个字节的低七位，ASCII 码规定高位充零，而国标码对高位未定义，因此，对单个字节而言，不能确定它到底是一个 ASCII 码字符还是一个汉字的一部分(低字节或高字节)。例如，对于机器内存中连续两个字节，它们的低七位内容分别为 0110000 和 0100001，它是两个 ASCII 码字符，还是一个汉字，无法判断。于是有很多解决这类问题的方案应运而生，变形国标码就是其中之一，并且得到了广泛的应用。变形国标码的主要特点是将国标码编码的各个字节的最高位置 1，以达到区别于 ASCII 编码的目的。

如上述两个低七位内容为 0110000 和 0100001 的字节，如果它们的最高位均为 1，即 10110000 和 10100001(变形国标码)，表示一个汉字“啊”；如果最高位均为 0，即 00110000 和 00100001(ASCII 码)，表示为两个 ASCII 码字符“0”和“!”。

由于计算机中各种信息都是以二进制形式存在，有的是数值，有的是 ASCII 码字符，有的是汉字或是其他信息，如何区分它们呢？这实际上取决于我们(或者程序)按照何种规则判读它们。因此，应注意两点：① 任何信息在计算机中均以二进制形式表示和保存；② 计算机中二进制数据表示的究竟是数值、字符、图形或其他信息，仅与处理这些二进制数据的规则有关。

1.1.2 计算机系统的基本组成

一个完整的计算机系统是由硬件系统和软件系统两大部分组成。硬件系统是构成计算机系统的各种物理设备的总称，是计算机系统的物质基础，它由运算器、控制器、存储器、输入设备和输出设备组成；软件系统是为运行、管理和维护计算机而编制的程序和各种文档的

总和。计算机系统的组成如图 1-4 所示。

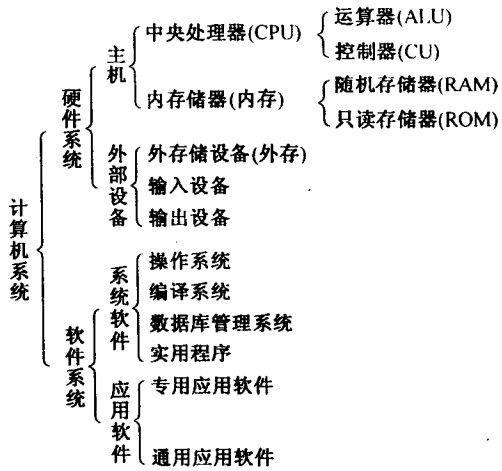


图 1-4　计算机系统的组成

1. 硬件系统

计算机硬件主要由中央处理器、主存储器、辅助存储器、输入设备、输出设备等组成。

(1) 中央处理器(CPU)。CPU 的作用是不断地从内存中取指令并执行指令。由机器码组成的这种指令序列，称为可执行程序。CPU 严格按照程序(即由内存中取来的指令序列)工作，CPU 的指令控制部件负责解释并执行各条指令，在内部进行算术运算、逻辑运算，对外部按指令要求与内存、接口设备交换信息并协调整个计算机系统的工作。

(2) 主存储器(main memory)。主存储器又称为内存储器，简称内存，用来存放当前运行的程序和数据。根据其工作方式和读写功能的不同，内存可分为：只读存储器和随机存储器。

1) 只读存储器(ROM)。ROM(read only memory)中存放的信息在一般情况下只能被读出，不能被改写，通常用来存放那些永久不变的信息，如计算机加电自检程序、启动引导程序等。它的内容是由厂家出厂前写入的，一旦写好，用户就不能随便更改。

2) 随机存储器(RAM)。与 ROM 不同，RAM(random access memory)中的信息有两个特点：一是随机存取信息，即信息不仅可以随时读取，而且还可以随时写入；二是具有挥发性，当电源关闭后，保存的信息会消失，且不可恢复。

(3) 辅助存储器(auxiliary memory)。主存储器速度快但价格昂贵，因而容量受到限制；且 RAM 中的信息不能长期保存，断电后即“挥发”，所以计算机采用了大容量的辅助存储器，如磁带、磁盘、光盘等。辅助存储器只能与主存储器交换信息，是主存储器的扩充，它与主存储器一起构成计算机存储体系。

在存储器中，用来存放一个二进制代码位的存储器最小单位是存储位(bit)，其中只能存放一个二进制数 0 或 1。由若干个连续的存储位组成一个存储单元，规定 8 个二进制位组成一个字节(byte)存储单元，进而由许多个存储单元组成存储器。在计算机中，为了有效地区分不同的存储单元，必须将它们按字节逐一编号，该编号称为存储单元(字节)的地址。通过地址，可以访问各个存储单元。计算机存储器中的随机存储器所包含的存储单元(字节)的总数通常称为该存储器的存储容量。容量越大，能存储的信息越多。存储容量常用字节数来表示，如 64KB、128MB 等。存储容量的换算关系如下：

1byte=8bits

1KB=1024bytes=2^{10}bytes　　1MB=1024KB=2^{20}bytes

1GB=1024MB=2^{30}bytes　　1TB=1024GB=2^{40}bytes

(4) 输入/输出设备。输入设备是向计算机输入数据、信息的设备的总称。它将计算机程序、文本信息、多媒体信息以及各种数据转换成计算机能处理的数据形式并输送到计算机。常见的输入设备有键盘、鼠标、扫描仪等。输出设备是能将计算机处理好的信息转换成文本、图形、多媒体等形式并输出的设备。常见的输出设备有显示器、打印机、绘图仪等。

2. 软件系统

计算机只有配备了软件系统才能进行工作。一台计算机能否发挥其应有的作用，实现硬件系统所能完成的信息处理功能，取决于软件系统的优良与否。

软件一般指计算机运行所需的各种程序、数据以及相关的文档。软件系统由系统软件和应用软件两大部分组成。系统软件是用来对计算机进行管理、控制和维护，以及支持应用程序的运行的软件的集合。应用软件是在系统软件的支持下为解决各类实际问题而设计开发的软件(程序)。

(1) 系统软件。系统软件用于管理计算机资源，分配和协调计算机各部件工作，提高计算机的使用效率，方便用户使用计算机。系统软件包括以下四大类：

1) 操作系统：用于管理计算机系统资源的程序的集合。它的主要功能是处理机管理、存储器管理、文件管理、设备管理、作业管理。操作系统是计算机和用户的接口。常见的操作系统有 DOS、Windows、Unix 等。

2) 编译系统：指各种程序设计语言机器处理程序。计算机程序设计语言分为低级语言和高级语言，低级语言包括：机器语言、汇编语言；高级语言包括：Basic、Fortran、C 等。

3) 实用程序：为系统维护和运行提供便捷而有效的服务性程序。这类程序称为实用程序(ultility)。实用程序种类很多，通常包括诊断程序、调试程序、文本编辑程序等。

4) 数据库管理系统：实现有组织地、动态地存储大量相关数据，方便用户和应用程序访问的计算机软件、硬件资源组成的系统。它包括数据库和数据库管理软件。用于对于大量数据的存储、整理、查询、维护和建立各种报表以及数据资源的多用户、多应用程序的共享等。常用的数据库管理系统有 FoxPro、Oracle 等。

(2) 应用软件。应用软件是用户或软件开发人员在系统软件的支持下，为解决各类实际问题而设计、开发的软件，它包括通用应用软件(即软件包)和专用应用软件(即用户应用软件)。通用应用软件是指由软件公司专业人员为解决通用性问题而设计的软件，以供用户选择使用。这类软件很多，如 Office 2003(办公自动化)、Access(数据库)、SAS(统计分析系统)等。专用应用软件是指用户为了解决特定问题自己或委托他人研制开发的软件，如工资管理系统等。

3. 计算机语言

计算机语言是人类为了有效地与计算机进行信息的传递、沟通，并且使计算机能按照人类的意志进行工作而开发出的一种语言。人类使用它描述解决问题一系列步骤，计算机能够识别并执行它，以达到解决问题的目的。

(1) 机器语言。计算机能够理解并执行的由“0”和“1”组成的二进制指令，即机器指令。机器语言就是这样的机器指令以及相关规则的集合。用机器语言所编写的解决某一实际问题的一系列指令就称为机器语言程序。显然，一方面由于机器指令由“0”和“1”组成，用它编制出来的程序机器可以直接识别并执行，执行的效率比较高；但另一方面，正是由于

机器指令的难记、难写、难懂、难调试、难移植等特点，使得用机器语言编制程序成为一件非常冗长、繁琐甚至是“痛苦”的工作。在计算机问世初期，人们只能使用机器语言编制程序。

(2) 汇编语言。为了克服机器语言的缺点，计算机工作者采用了助记符的方法，使每条机器指令与一个符号一一对应，便于记忆和书写。如用 ADD 表示加法操作，用 SUB 表示减法操作等。这些特定的符号和一些相应的格式和规则，构成了汇编语言。

汇编语言是面向机器的语言，但机器并不能直接执行，需要使用“编译”程序将它翻译成为机器指令后，计算机方可执行。

(3) 高级语言。虽然使用汇编语言编写程序比机器语言方便，但是仍然很“麻烦”，除了要记忆众多的助记符和规则外，还需要了解计算机硬件有关知识，这对于一般计算机用户是一件困难的事情。于是计算机工作者设计出面向应用的计算机语言，即高级语言，它更接近人类的自然语言，便于学习、理解和使用。

显而易见，用高级语言编写的程序也不能直接地被计算机执行，只有经过“编译”或“解释”程序处理后才能在计算机上执行。

4. 翻译方式

上述多次提及使用编译程序或解释程序对高级语言程序(或汇编语言程序)进行“翻译”处理，使它成为机器指令程序，才能提交计算机执行。**注意**：其核心是将高级语言(或汇编语言)转换成“机器指令程序”。下面分别介绍这两种处理方式。

(1) 解释方式。使用解释程序(interpreter)将高级语言程序的语句逐条“翻译”成机器指令并逐条提交计算机执行，直至程序结束(图 1-5)，当高级语言程序中语句出现错误时立即终止执行，并给出错误信息以便修改后重新解释执行。

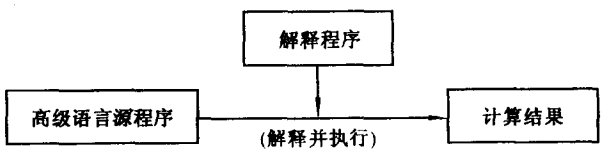


图 1-5 解释方式

(2) 编译方式。分两步进行：首先通过编译程序(compiler)对整个高级语言程序进行编译，它包括翻译和查错(词法分析、语法和语义分析、生成和优化目标程序)，出现错误时，停止编译，报告错误，不生成目标程序，待修改源程序后，再进行编译，直到最终得到正确的目标程序；然后使用链接程序(linker)对目标程序进行链接，得到可执行的程序，这时才能将可执行程序提交计算机执行(图 1-6)。

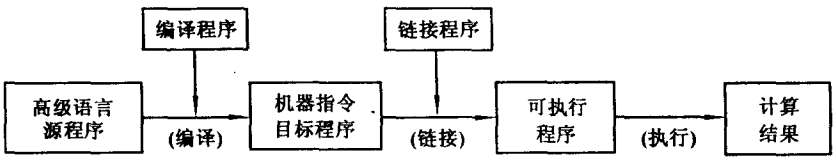


图 1-6 编译方式

解释方式占用内存空间少，但运行效率低；编译方式占用内存空间多，运行效率高。我们将要学习的 Turbo C 是以编译方式工作的。