学习

冯博琴 等 编著

C语言学习指南

机械工业出版社

指南

C 语言学习指南

冯博琴　王建仁　梁　力　顾　刚　郑庆华　编著

机械工业出版社

本书是C语言学习方法的指导书，对初学者及钻研C语言的读者均有所裨益。全书分四个部分共13章：第一部分是第1～8章，主要包括C语言基础部分的难点、重点内容；第二部分是第9、10章，介绍C语言的进一步知识，包括图形、屏幕和混合编程；第三部分是第11、12章，包括上机操作及调试技巧；第四部分是第13章，介绍了几个综合应用的例子。每一章围绕一个主题，分成三个层次：①主要知识点介绍；②易犯的错误和释疑；③运用技巧和体会。

本书内容丰富、概念清晰、实用性强，可与其他C语言教材配合，适用于高校师生和计算机培训班使用，也可供自学者参考。

图书在版编目(CIP)数据

C语言学习指南/马博琴等编著．—北京：机械工业出版社，1996.7
ISBN 7-111-05141-6

Ⅰ.C… Ⅱ.马… Ⅲ.C语言-指南 Ⅳ.TP312C-62

中国版本图书馆CIP数据核字(96)第04826号

出 版 人：马九荣(北京市百万庄南街1号 邮政编码100037)
责任编辑：蒋有彩 版式设计：王 颖 责任校对：肖新民
封面设计：姚 毅 责任印制：路 琳
机械工业出版社印刷厂印刷·新华书店北京发行所发行
1996年7月第1版第1次印刷
787mm×1092mm 1/16·16.25印张·392千字
0 001—4 000册
定价：19.80元

前　言

从本世纪的60年代开始，一场新的信息革命悄然来临，它把人类带进信息化社会。与这个社会相适应的社会技术是信息技术，它的核心是计算机技术。由于这项技术在人类发展史上大大改变了人类创造物质财富和精神财富的方式、方法、过程和结果，也改变了社会结构和人类自身的生活方式、习俗等，因此引起各个部门空前的重视。

21世纪将是一个信息化社会，这个社会对人才素质和知识结构都会提出更高的要求。对于高等教育的各个学科，计算机的作用已不仅仅是一种工具，而是各学科本身的重要组成部分。加强计算机基础教育不仅是为了提高计算机本身的水平，而且将为提高其他学科的教育水平打好基础。由此可见，加强计算机基础教育既是文化基础教育、人才素质教育，又是强有力的技术基础教育。加强这种教育不仅是信息化社会需要，也是各学科发展的需要。计算机教育水平的高低已成为评价学校教学质量的重要指标。学生本人的计算机应用能力反映了个人素质，影响他的竞争能力。因此，目前理、工、文、管、农、林、医等各类学校都在努力使计算机教育上一新台阶。

目前计算机程序设计语言仍是计算机基础教育的最基本内容之一，因为大多数用户还是利用程序设计语言作为编程序的手段来利用计算机的。根据我们教学体会，教学和教材都存在着这样问题：学生大多数是计算机的初学者，对计算机的解题过程和程序设计非常陌生；学时又十分紧张；教材编写受格式、篇幅限制，学生学习的困难和问题颇多，不少人是带着许多问题结束这些课程学习的。

作者求学时，读过樊映川先生编写的《高等数学学习方法指导》，受益匪浅。受此启发，觉得有必要跳出教材，站得稍高些，对程序设计语言的难、重、热点进行深入一点的讨论，因此组织了几位长期从事此项工作的教师编写了《FORTRAN语言学习指南》和《C语言学习指南》。

这些书在内容组织和写法上不同于教材。总体结构上包括三部分，即语言的基本内容、上机指导和进一步知识。所选择的内容基本上是该语言的重点、难点和热点。每一章围绕一个主题，一般都包含知识概要、易犯的错误和释疑、以及有关的应用技巧或体会。

作者希望本书对以下两种人有所帮助：一种是正在学习这门语言的读者，不论他用何种教材，本书都能为他解惑，成为他的辅导老师；另一种是想提高编程能力的读者，因为各章都有这方面的技巧、经验和体会，同时还包含一些新内容，如编程风格、图形处理和混合编程等。这些在教材中一般"无暇"顾及，但在应用中是热点，学生很感兴趣。

《C语言学习指南》由冯博琴主编，王建仁(第1～4、11、12章)，梁力(第6、7、13章)，顾刚(第5、8章)和郑庆华(第9、10章)执笔。由于作者水平有限，取材和写法未必合适，疏漏和谬误之处在所难免，希望同行和读者指正。

冯博琴

1995年7月于西安交通大学

目　录

第1章 C语言基础

1.1 高级语言组成概述及C语言特点

计算机是一种能按照程序规定的步骤自动地高速处理数据的工具。程序是由人编写的，编写计算机程序要用计算机语言。计算机语言有机器语言、汇编语言、高级语言等几类。其中高级语言是使用最方便的一类，它接近于人们的自然语言，用高级语言编程序符合人们的习惯，能比较自然地表达各种概念，编出的程序能在不同的计算机上运行。目前使用的高级语言有多种，最常用的有BASIC、FORTRAN、PASCAL、C语言等。这些语言各有特点，但基本内容和概念是相似的。

1.1.1 高级语言组成概述

计算机是用来处理数据的。实际中的数据形式多种多样，例如一般的数值或一段文字等。这些不同形式的数据在计算机存储器中存储时，所占用的存储空间是不同的，比如，整数"15"与字符串"I am a student"所占用的存储空间就不一样。为了有效地存储和处理这些不同形式的数据，计算机语言首先将所要处理的数据划分成若干类别，对不同类别的数据规定了不同的存储形式和表示方法。同一类别数据的存储形式和表示方法是相同的。这些数据类别在计算机语言中称为数据类型，有多少类数据就有多少种数据类型。所以，对数据类型的划分和规定是计算机处理数据的基础，也是计算机语言的基础。

计算机在执行程序处理数据过程中，经常要从输入设备(如键盘)上接收初始数据。这些输入的数据存储到存储器的何处？又如何取出来参加运算？此外，对于程序处理过程中产生的中间结果和最终结果也有同样问题。比如，任给三角形三边长a、b、c，要求计算机根据公式

$$P=(a+b+c)/2$$

$$s=\sqrt{p(p-a)(p-b)(p-c)}$$

计算三角形面积S。从键盘上必须输入三条边长的具体值，比如3、4、5，计算机要计算出(3+4+5)/2，并且要将计算结果6存储起来，再用于计算面积。这些数据存储到哪里呢？这就必须要求计算机语言提供使用存储器的方法和手段。对该问题，计算机语言是通过提供变量这一概念实现的。在计算机语言和程序中，变量是存储单元的代表或称存储单元的名。要想使用存储器，就需要设置变量，变量设置得越多，可以使用的存储空间也就越大。这样，存储器使用问题就变成了对变量的引用问题。由于不同类型的数据占用的存储空间多少不一样，所以，设置变量时，还必须指明变量的类型。

由于不同类型数据的存储方法不同，表示事物的特性不同，所以，能参与的运算就不一样。例如，两个数能进行相加运算，而两个字符串就不能象数那样相加。为了能有效地处理数据，计算机语言在确定了数据类型、各类数据的表示方法和存储方式之后，还应该确定各类数据都能进行哪些运算以及运算规则。这一内容在计算机语言中是通过设置运算符实现的。设置的运算符越多，该语言所能完成的运算就越多。将数据与运算符连接起来的式子称为表达式，

如 2.5+6.2 就是一个表达式。每个表达式都能计算出一个值，该值称为表达式的值，表达式是计算机语言中处理数据的基本手段。

实际中，绝大多数问题都是要经过许多步运算才能得出结果，计算一个表达式需要另一个表达式的计算结果，所以，计算机语言还必须提供控制计算顺序的手段和方法。这些对计算机语言的要求，是通过设置各种控制语句实现的。

最后，计算机执行程序时，需要从输入设备（如键盘、磁盘）上输入数据，计算结果要送到输出设备（如显示器）上输出，所以，计算机语言还应提供输入输出手段，这些要求在语言中是通过输入输出语句实现的。

从以上讨论可以看出，数据类型规定了计算机语言可以表示数据的种类和处理方法，变量是使用存储单元的手段，变量名是存储单元的代号，运算符是各类数据运算规则和方法的具体体现，而表达式则是运算的具体表现形式，各种控制语句对运算过程进行控制，输入输出语句则提供了计算机同外界打交道的手段。实际中，需要计算机解决的问题各式各样，不同时期、不同的人对这些问题的不同处理方法就使得不同的计算机语言得以出现。尽管如此，各种语言的基本组成是一致的，只是语言的具体规则有些不同，这些规则就是各种语言的语法。要编写程序必须熟悉所用语言的语法。同时，由于各种语言组成方面的相似性，使得我们在掌握了一种语言之后，可以比较容易地学会其它语言。

1.1.2 C 语言的特点

与其它高级语言相比，C 语言具有以下特点：

1）将汇编语言和高级语言的特点集于一体，既有面向硬件系统、便于直接控制硬件等汇编语言所具有的功能，又具有高级语言的面向用户、易学易记、程序容易书写和阅读等优点。

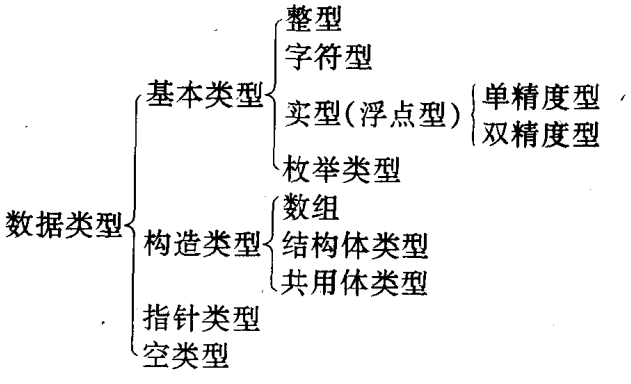
2）数据类型多样，运算符丰富，具有结构化的控制语句和模块化的程序结构。

3）移植性很强。C 语言本身很少，关键字只有 32 个，它将所有与外部设备有关的控制部分都抛给了库函数，而 C 语言编译程序本身仅处理一些与硬件关系不密切的有关数据类型及程序的流程控制问题，使它从一个计算机系统到另一个计算机系统的移植改写容易实现。

4）语法限制不严格，程序设计自由度大。这使程序设计者可以充分发挥才智，设计各类精巧程序，但对不熟练的开发者来说，也增加了出错的可能。

1.2 基本数据类型

1.2.1 C 语言可以使用的数据类型



在程序中,每种数据类型都有常量和变量之分。常量是指在程序运行过程中其值不能被改变的量。常量从其本身就可确定其数据类型,如 12 为整型,12.5 是实型。变量是存储单元的符号名,存储单元中存几,变量的值就是几,由于存储单元中的值可以改变,所以变量的值是可以改变的。变量必须先定义,后使用。程序中是通过定义变量取得存储空间的,定义一个变量要做两件事:一是说明它的名;二是说明它的类型。

变量名同以后将要出现的数组名、函数名、类型名、文件名等在计算机语言中统称为标识符。标识符是一个由字母(包括大写和小写)、数字和下划线三类字符组成的字符串,而且还必须符合以下规则:

1)开头的必须是字母或下划线。

2)字母的大小写是有区别的。

3)称为保留字的部分标识符系统已使用,用户不能再用。

至于标识符中的字符个数,各种 C 语言系统的规定不同,Turbo C 语言中规定为 1～31 个字符。

1.2.2 整数类型

整型常量可以用十进制、八进制和十六进制表示。例如

25　　十进制数

025(即 21)　八进制数　　特征是以 0 开头,各数字位为 0～7

0x25(即 37)　十六进制数　　特征是以 0x 开头,各数字位为 0～9 或 A～F(a～f)

整型又可细分为不同长短的整数类型,在不同的计算机上规定不完全一样,PC 机中的规定及取值范围见表 1－1。

表 1－1　PC 机中各种整型数占据的存储空间长度及取值范围

数据类型	长度(位)	取　值　范　围
int	16	$-2^{15}\sim 2^{15}-1$　即 －32768～32767
long int 或 long	32	$-2^{31}\sim 2^{31}-1$　即 －2147483648～2147483647
unsigned int	16	$0\sim 2^{16}-1$　即 0～65535
unsigned long	32	$0\sim 2^{32}-1$　即 0～4294967295

在 PC 机中,short 型与 int 型是一样的。

整型变量的定义方法为:

int a; 定义 a 为整型变量,代表的存储空间为 16 位。

long b; 定义 b 为长整型变量,代表的存储空间为 32 位。

1.2.3 实数类型

实数类型简称实型,也称浮点型。实型常量有两种写法:

1)小数形式　如 78.5,78.0,0.78,78 等。

2)指数形式　如 1.5e3, 1e－5 等。这里的字母 e 大小写均可。

实型又可分为 float 型和 double 型两种,占用的空间及取值范围见表 1－2。

表 1-2 PC 机中实型数据占据的存储空间及取值范围

数据类型	长度(位)	取值范围
float	32	$-10^{38}\sim10^{38}$
double	64	$-10^{308}\sim10^{308}$

1.2.4 字符类型

字符类型简称字符型。字符常量由一对单引号将字符括起来，表示的是该字符在 ASCII 码表中的代码值。如'a'表示 97;'A'表示 65。

ASCII 码表中的控制字符不可显示，用转义字符表示，如

'\0' 表示 null(空)　　即 0
'\t' 表示 tab(制表)　　即 9
'\n' 表示换行　　即 10
'\r' 表示回车　　即 13

一个字符数据占用 8 位即 1 字节的存储空间，类型符为 char。

例如，语句:char c;定义了一个字符变量 c，取值范围为 -128～+127。

字符型数据的存储方法与整型数据是一致的，只是空间小，所以在许多场合，两种类型是可以通用的。

1.3 运算符

1.3.1 算术运算符

C 语言中的算术运算符有 +，-，*，/，%。

前四种运算为加、减、乘、除，与日常中的用法相同。% 为取模运算符或称求余运算符，仅针对两个整型数运算，结果为两个整型数相除之后的余数。

如　　10%3=1　　10%5=0

两个整型数求余运算结果为 0 时，表示能整除，程序中经常用到。

对于前四种运算，类型不同的数据可混合进行，数据类型与运算结果的关系为：

1)同类型运算，运算结果仍保持原数据类型。

如　　2*5=10　　6.4/2.0=3.2

实际中，应特别注意的是两个整型数进行除法运算。例如 1/2 的结果为 0，而不是 0.5，即两个整数相除只取整数部分。

2)类型不同的数据混合运算，先将精度低的类型向精度高的类型转换后，再做运算，这样可保证运算结果不损失精度。

如　　6.4/2→6.4/2.0=3.2

1.3.2 自增运算符++和自减运算符--

++和--是 C 中特有的运算符。以++为例，n++和++n 都使变量 n 自增 1，最终结果与 n=n+1 一样，但处理过程却有区别。++n 表示 n 先增 1，然后再进到具体的式子中计算，而 n++表示 n 本身先进入式子中计算，最后再增 1，比较下面两组句子：

语句	结果	语句	结果
n=1;	n为1	n=1;	n为1
n++;	n为2	++n;	n为2
m=5*(n++);	m为10,n为3	m=5*(++n);	m为15, n为3

n--与--n与此类似。

有效地使用++与--可简化程序,提高程序的运行速度。

1.3.3 赋值运算符"="

赋值运算符"="的功能是给变量赋值,赋值表达式的格式为:

变量=表达式

计算过程是将表达式的运算结果赋给变量,变量的值即为赋值表达式的计算结果。具体赋值过程是先判断表达式运算结果的类型与变量的类型是否相同,如果相同,直接赋值,否则,先将表达式运算结果的类型转换成变量的类型,然后赋值。

尽管赋值表达式在后面加上分号后就是一个赋值语句,而且与其它语言中的相应语句相同,但C语言将它作为一个表达式,就使赋值操作可以出现在程序中任何允许表达式出现的地方,这一点也是导致C程序紧凑、简洁的一个原因。

1.3.4 运算符的优先级和结合性

在C语言中,运算符的优先级和结合性规定了表达式运算操作的处理顺序。在对表达式求值时,先考虑运算符的优先级,如果相邻两个运算的优先级相同,则按结合性处理。例如下面表达式的计算过程为:

2+3*4-5

↓ *优先级最高

2+12-5

↓ +,-优先级相同,考虑结合性,从左至右

14-5

↓

9

又如,表达式x=y=1,两个"="运算符优先级相同,按从右到左的结合性处理,先算y=1,将1赋给y,整个赋值表达式的值也为1,原式变为x=1,执行后,x、y的值均为1。

由于C语言运算符种类繁多,优先级和结合方向不一,初学编程时,如还不能完全掌握,可通过加括号的方法规定运算顺序。

1.4 输入输出及程序的基本格式

1.4.1 输入输出

在C语言中,最基本的输入操作是调用格式输入函数scanf(),最基本的输出操作是调用格式输出函数printf(),其一般格式分别为:

scanf(控制串,地址表达式表)

printf(控制串,表达式表)

输入输出是控制串与后面的表达式表配合起来完成的。控制串中有两类字符:一类为普

通字符;另一类为以%开头的格式转换符。例如下面的语句

```
c=33;
printf("c10=%d, c8=%o, c16=%x, cc=%c\n",c,c,c,c);
```

的执行的结果为:

```
c10=33, c8=41, c16=21, cc=!
```

格式控制串中的%d、%o、%x、%c都是格式转换符,其余是普通字符。printf()的执行过程为:从控制串左边开始,依次扫描每个字符,遇到普通字符原样输出,遇到格式转换符,将后面与之对应的表达式的值转换成相应格式输出。上列的33,41,21和!分别是将33转化成十进制、八进制、十六进制整数和字符形式输出的结果,遇到格式串中的转义字符'\n',光标回到下一行首。

对于下面的语句序列

```
int c1,c2;
scanf("%d,%o",&c1,&c2);
printf("c1=%d, c2=%d\n",c1,c2);
```

输入 33,33

输出 c1=33,c2=27

从上面可以看出,输入的两个数表面上相同,均为33,但转换符不同,%d意思是将十进制数33存入c1,%o意思是将八进制数33存入c2。输出时,均按十进制输出,故c1为33,c2为27。

可见,C语言这种借用格式转换符输入输出数据的方法,可使输入输出灵活多样。

进一步,由于C语言常用于编写系统程序,而在系统程序中,字符输入输出操作甚多,所以C语言又提供了专用于字符输入输出的函数getchar()和putchar(),它们与scanf和printf的关系是:

```
c=getchar()    等价于 scanf("%c",&c)
putchar(c)     等价于 printf("%c",c)
```

所不同的是程序中如果使用这两个函数,应在程序前加

```
#include   "stdio.h"
```

1.4.2 程序的基本格式

一个C程序由一个或多个函数构成,至少必须存在一个main函数,main函数的基本形式如下:

```
main()
{ 变量说明语句;
  执行语句;
}
```

每条语句以分号作为结束标志,一行可写多个语句,一条语句也可分作几行写,但对初学者来说,为清楚起见,最好一行写一个语句。

1.4.3 注释

程序中的注释部分用/*和*/括起来,可加在程序中的任何地方,以加强程序的可读性。

1.5 编程技巧

本节通过先举例后总结说明方式，提出一些编写 C 程序的方法和注意事项。

例 1.1 求方程 $ax^2+bx+c=0$ 的根。a、b、c 的值由键盘输入，且 $a\neq0$、$b^2-4ac\geqslant0$。下面程序利用求根公式 $x1=\frac{-b+\sqrt{b^2-4ac}}{2a}$ 和 $x2=\frac{-b-\sqrt{b^2-4ac}}{2a}$ 解方程。

程序如下：

```
#include "math.h"
main()
  { float a, b, c, d, x1, x2;
    printf("Input a, b, c: ");                    /* 输出提示信息 */
    scanf("%f%f%f", &a, &b, &c);
    printf("a = %f, b = %f, c = %f \n", a, b, c);  /* 显示输入结果 */
    d = sqrt(b * b - 4 * a * c);
    x1 = ( - b + d)/(2 * a);
    x2 = ( - b - d)/(2 * a);
    printf("x1 = %f, x2 = %f \n", x1, x2);
  }
```

运行情况如下：

```
Input a, b, c:      1  -2  -3
x1 = 3.000000,      x2 = -1.000000
```

方法与问题说明：

1)在输入语句前，应增加输出提示信息的语句。通过键盘输入数据时，要用到 scanf 等函数。计算机执行 scanf 函数时，等待用户从键盘上键入数据，但这仅仅是等待，并没有任何提示。本例在 scanf 之前增加了 printf 语句，意思是在屏幕上显示与输入数据有关的提示信息。这样，在程序运行时，当操作者看到提示信息时，计算机刚好在执行 scanf 等待输入，于是，操作者可根据提示信息输入应该输入的数据。所以，建议读者在编程序时，在键盘输入语句之前增加输出提示信息的语句。上例中第 4 行就是为此而加的。

2)在输入语句之后，增加回声显示语句。正确输入数据是得出正确结果的前提。但是，由于 C 语言语法很灵活，输入语句和输入数据中的许多错误，编译程序都不指出来。为了及早发现输入语句和输入过程中的错误，最好在刚输入完数据后就显示接收数据的变量值，以同输入数据对比，相同则正确，否则错误。上例中的第 6 行就是为此而加的。根据该语句所起的作用，称其为回声显示语句。这种技术能及时查出输入语句和输入过程中的所有错误。

上例中的 4、5、6 三行，可称为输入数据的三步曲，即提示输入、具体输入、回声显示。

3)库函数的使用方法。上面程序在进行开平方运算时，使用了开平方函数 sqrt，它是 C 语言中的一个库函数。根据 C 语言规定，程序中如果使用了库函数，在程序前面就必须用预处理命令 #include 将与该类库函数对应的说明文件包含到程序中来。开方函数 sqrt 是数学类函数，数学类函数对应的说明文件名为 math.h。math 是英文单词 mathematics 的前四个字

母，h 是 head 的第一个字母。程序的第一行 #include "math.h"就是为此而写的。C 语言中库函数的设置、用法及对应的说明文件参见有关书籍。

在 C 语言中，pow(x, y)是指数函数，该函数计算 x^y。

例 1.2 任给整型变量 a, b 各为一值，如 a=8, b=6，要求计算 a * b，并按下列格式输出：

```
a * b = 8 * 6 = 48
```

```
main()
  { int a, b, c;
    printf("Input a, b: ");          /* 输出提示信息 */
    scanf("%d%d", &a, &b);
    c = a * b;
    printf("a * b = %d * %d = %d \n", a, b, c);
  }
```

运行过程为

```
Input a, b: 8 6
a * b = 8 * 6 = 48
```

第二次运行为

```
Input a, b: 10 5
a * b = 10 * 5 = 50
```

方法与问题说明：

' 该题计算过程很简单，主要问题集中在输出格式上。当需要按规定格式输出结果时，我们可仔细分析该格式，将格式中的数据分作两类：一类为每次输出都不变的字符；另一类为变化的数据。上例中 a * b =、*、= 为不变的字符，其余为变化的数据。在 printf 语句中安排输出格式时，即在格式串中，将不变的字符直接写上，对于变化的数据则用 %d 等转换符代替，在输出项表中，依次写上要输出值的变量名或表达式。这样，在执行 printf 语句时，顺序扫描格式串，不变的字符原封不动显示出来，碰见 %d 等转换符时，就依次输出一个变量或表达式的值。从最终输出结果看，好象是用变量值替换掉了 %d 等转换符。

例 1.3 某单位发工资以元为单位，不发零钱。编一程序，能对任给的工资额，计算出应发各种面值钞票的张数，并要求总张数最少。

程序如下：

```
main()
  { int m;
    int n100, n50, n10, n5, n2, n1;
    printf("Input m: ");              /* 输出提示信息 */
    scanf("%d", &m);
    printf("m = %d \n", m);           /* 显示输入结果 */
    n100 = m/100;    m = m%100;
    n50 = m/50;      m = m%50;
    n10 = m/10;      m = m%10;
    n5 = m/5;        m = m%5;
```

```
    n2 = m/2;
    n1 = m%2;
    printf("100: %d\n 50: %d\n", n100, n50);
    printf(" 10: %d\n 5: %d\n", n10, n5);
    printf(" 2: %d\n 1: %d\n", n2, n1);
}
```

如某人工资为347元,则运行过程为

```
input m:347
100:3
50:0
10:4
5:1
2:1
1:0
```

方法与问题说明:

1)整型变量的定义及整型数据的除法、求余运算。在数学上,整型数和实型数都是数值,使用上没有区别。许多初学者也将此概念搬到了计算机语言中,认为整型数是实型数的特殊情况,实型数可以代替整型数,忽视了整型数和实型数在存储和计算方面的区别。例如,上例中,一直采取了整型数据的"/"运算和"%"运算,如果将m定义为实型,则上述程序就完全错了。所以,实型数是代替不了整型数的。实际上,应根据具体问题确定采用整数运算还是实数运算,或者定义整型变量还是实型变量。

2)变量的命名。从语法上讲,一个变量名只要符合命名规则,就是一个合法的变量名。但在实际编程序时,给变量命名除了要符合语法规则外,还要便于阅读,最好是见名知义,即见了一个变量,就可知道该变量代表什么、在程序中起什么使用。上面程序在记录各种钞票的张数时,使用了变量n100、n50等,使人一看就知道每个变量的含义,达到了见名知义的效果。如果将这些变量用a,b,c等代替,就使程序难于阅读。

例1.4 说明字符与整数的关系。

```
main()
{ char c1, c2;
  c1 = 'A';
  c2 = 65;
  printf("%d, %c\n", c1, c1);
  printf("%d, %c\n", c2, c2);
}
```

运行结果:

```
65, A
65, A
```

方法与问题说明:

字符型数据在内存中是以ASCII码形式存储的。字符'A'的ASCII码为65,所以将'A'赋