HZ BOOKS

高等院校计算机教材系列

C语言程序设计

刘振安 主编

为教师提供教学课件

机械工业出版社
China Machine Press

高等院校计算机教材系列

C语言程序设计

刘振安 主编

机械工业出版社
China Machine Press

本书主要讲授C语言的面向过程程序设计方法，并介绍常用的逻辑求解、查找、冒泡排序、蒙特卡罗法、迭代、递推和递归等算法，以便培养解决实际问题的能力。

本书将程序设计归纳为三种典型结构，并结合三种典型结构，介绍C语言编程的核心问题，同时利用Visual C++集成环境，进行编程和调试训练，提供完整的多文件编程实例，提高编程和程序测试能力，从而为设计实用程序打下良好基础。

本书注重理论联系实际，概念清楚，实用性强，易于教学，适合作为高等院校的教材，也可以作为培训班教材、自学教材及工程技术人员的参考书。

图书在版编目（CIP）数据

C语言程序设计/刘振安主编．-北京：机械工业出版社，2007.1
（高等院校计算机教材系列）
ISBN 7-111-20078-0

Ⅰ.C…　Ⅱ.刘…　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123101号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）
责任编辑：朱　劼
北京牛山世兴印刷厂印刷·新华书店北京发行所发行
2007年1月第1版第1次印刷
184mm×260mm·18.25印张
定价：29.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换
本社购书热线：（010）68326294

前　言

计算机科学发展的每一步几乎都在软件设计和程序设计语言中得到充分体现。软件是一个发展的概念，随着软件开发规模的扩大和开发方式的变化，人们开始将程序设计语言作为一门科学来对待。程序设计方法和技术的发展不仅直接导致了一大批风格各异的程序设计语言的诞生，而且对计算机理论、硬件、软件以及计算机应用技术等方面都产生了深远的影响。

需要强调的是，从软件专业的角度讲，能胜任较低层次的软件开发是软件开发能力的基础。在开始学习编程时，不应该把重点放在熟悉可视化编程工具等高层次的技巧上，而应该放在学会直接采用这种语言进行“手工”编程的基本功上。

另外，大多数程序设计语言教材都是把重点放在基本词法、语法和简单的程序设计上，读者学完之后，却很难编出实用的程序。有些公司的程序员竟然将几千行的程序放在一个文件中，造成调试和维护的不便。因此，本书将解决实际问题的能力和多文件编程方法均作为训练的重点之一。

本书是在作者自 1994 年以来开设的几门课程的基础上，通过合理组合与取舍内容编写而成的，力求反映学科发展，展现它们的最新特征。

本书的主要特点如下：

1）本书把重点放在程序设计方法上，将结构化程序设计与 C 语言的函数设计融合起来，透彻介绍 C 语言的核心问题。

2）本书将程序设计归纳为三种典型结构，并提出具体的设计思想。

3）本书用单独的一章讲述结构化设计实例，不仅总结结构化程序的知识，介绍软件测试和调试的基础知识，还以一个完整的例子说明如何使用多文件结构模式，设计实用程序及如何选择测试用例进行测试。通过这一章的强化训练，读者可进一步掌握多文件结构，大大提高实际编程能力。

4）本书注重理论联系实际，每一章均给出实验和习题，并注重编程环境和调试技能的训练。

5）每章不仅给出精选的典型例题，还给出错误分析，通过正反对比，使读者更好地理解课文内容。

6）本书不按照知识范畴，而是按内容的难度设计章节结构并进行教学重点划分，从而保证每一章均不超前引用后面章节的知识，从而大大降低学习的台阶和难度。

7）本书给出常用的典型算法，例如逻辑求解、查找、冒泡排序、蒙特卡罗法、迭代、递推和递归等算法，以便培养学生的学习兴趣和解决实际问题的能力。

8）本书各章既互相衔接，又具有一定的独立性，方便教师根据自己的教学需要选择教学内容。

本书可以与机械工业出版社出版的《C 程序设计课程设计》一书配合使用，这将能起到事半功倍的效果。

本书共分 11 章。第 1 章是 C 语言程序设计基础，重在介绍主函数和解题的完整过程，并进行集成环境下的程序编辑、编译和运行的基本功训练。第 2 章介绍基本数据类型和表达式，同时

介绍程序调试基础知识。第3章将阐述C语言的控制结构。第4章探讨一维数组和指针。第5章是计算机解题实例，重在训练求解逻辑问题的能力并训练枚举法的解题思路。第6章探讨函数与多文件编程，这也是全书的重点，该章将结构化程序设计与C语言的函数设计融合起来，并将程序设计归纳为三种典型结构，提出具体的设计思想。第7章将介绍函数参数、函数指针和多维数组，重点讨论函数参数传递问题，并借助函数指针和多维数组，深入讨论函数的设计和调用问题。第8章是常用算法实例，介绍常用的查找、冒泡排序、蒙特卡罗法、迭代、递推和递归等算法。第9章主要介绍结构类型和链表，结构是必讲内容，其他为可选内容。第10章将讨论文件的相关知识。第11章是结构化程序设计实例，目的是总结结构化程序的知识，介绍软件测试和调试，并以一个完整的例子说明如何进行多文件编程。该章涉及内容较多，可以根据实际教学情况取舍。最后附录包含了一些必要的知识。

中国科学技术大学软件学院院长陈国良院士，原安徽大学副校长、计算机系程慧霞教授及南京大学计算机系陈本林教授在百忙之中审阅了书稿并提出许多宝贵意见，许多使用过原教材的院校老师也提出很多宝贵意见，特此表示感谢。写作中还参考了大量资料，有的收入参考文献之中，还有些没有收入其中，在此对这些作者表示感谢。

刘燕君、周军、潘剑锋、王文涛、孙忱等参加了本书的部分编写工作，本书在定稿之前，曾为许多软件工程硕士及本科生讲授，他们也提出了一些好的建议并验证了书中的程序及实验，特此感谢。

由于我们才疏学浅，不妥之处在所难免，敬请各位读者不吝赐教，给予指正为盼。作者的联系方式为：zaliu@ ustc. edu. cn。

刘振安

于中国科学技术大学

目　录

第1章　C语言程序设计基础

本章将简要介绍C语言的特点，并通过简单而典型的C语言实例，引入C语言的主函数以及实现输入和输出的基本方法，从而建立C语言程序设计的基本概念。

1.1　C语言特点

C语言是20世纪70年代初期美国贝尔（Bell）实验室的Dennis M. Ritchie设计的一种程序设计语言，正式发表于1978年。

1970年，Ken Thompson在早期编程语言BCPL的基础上开发了一种新的语言，取名叫B。Dennis M. Ritchie在B的基础上，于1971年开发了第一个C编译程序，1972年开始使用（主要是在贝尔实验室内部使用）。以后，C语言又经过多次改进，直到1975年用C语言编写的UNIX操作系统第6版公诸于世后，C语言才举世瞩目。目前，C语言的应用领域已不再限于系统软件的开发，而成为当前最流行的程序设计语言之一。

1978年，Brian Kernighan和Dennis M. Ritchie在《C程序设计语言》（*The C Programming Language*）[⊖]一书中对C语言进行了详尽的描述。随着微型计算机的日益普及，大量的C语言工具相继问世。然而这些工具没有统一的标准，并有不一致的现象。为了改变这种情况，ANSI（美国国家标准协会）于1983年成立了一个专门委员会，为C语言制定了ANSI标准。当时比较流行的是TURBOC，它不仅能够满足ANSI标准，还提供了一个集成开发环境，同时也按传统方式提供了命令行编译程序版本以满足不同用户的需要。随着Windows编程的兴起，Borland C和Microsoft C受到用户的欢迎。我国目前流行的是兼容C语言的Microsoft Visual C++ 6.0及Boralnd C++集成环境。

C语言是一种通用的程序设计语言。C语言的通用性和无限制性使得它对于许多程序设计者来说都显得更加通俗，更加有效。目前，C语言已应用于各个方面的程序设计，无论设计系统软件（操作系统、编译系统等）或应用软件（图形处理），还是用于数据处理（如企业管理）或数值计算等方面都可以很方便地使用C语言。

C语言具有如下特点：

1）C语言吸取了汇编语言的精华，使C语言相对于高级语言来讲是“低级”语言（汇编语言是一种面向机器的程序设计语言，尽管它的编程与高级语言相比要麻烦得多，但由于它具有描述准确和目标程序质量高的优点，所以汇编语言仍然有很强的生命力）。

- C语言提供了对位、字节以及地址的操作，使程序可以直接对内存及指定寄存器进行操作。
- C语言吸取了宏汇编技术中的某些灵活的处理方法，提供#define和#include预处理命令。目前ANSI新标准采用了C++的const类型限定符，提高了C语言的可靠性。
- C语言能很方便地与汇编语言连接。在C程序中引用汇编程序与引用C语言函数一样，这为某些特殊功能程序的设计提供了方便。

⊖ 该书已由机械工业出版社引进出版，书号是7-111-12806-0。

2）C语言继承和发扬了高级语言的长处，使C语言相对于汇编语言来讲又是“高级”语言。

- C语言吸取了ALGOL的分程序结构思想。C程序中，可用一对花括号“{ }”把一串语句括起来而成为复合句（分程序），在括号内可定义变量。它还继承了PASCAL的数据类型，提供了相当完备的数据结构。
- C语言吸取了FORTRAN语言的模块结构思想。C程序的每一个函数都是独立的，可以单独编译。对设计一个大的程序来说，这样有利于分工编程和调试。
- C程序中的任何函数都允许递归，这样就可以方便地实现某些算法。

3）C语言的规模适中、语言简洁，其编译程序简单、紧凑。C语言在表示上比较简洁（比如用一对花括号{ }代替Begin-End语句，运算符尽量缩写等），C语言本身没有提供输入和输出工具以及并行操作，它的许多成分都是通过显式函数调用来完成的，而且运行时所需要的支持少，占用的存储空间也小。

4）C语言的可移植性好，这是指程序可以容易地从一个环境不加或稍加改动搬到另一个完全不同的环境下去运行。汇编程序因依赖机器硬件，所以根本不可移植；而一些高级语言，如FORTRAN等编译的程序也是不可移植的。

5）C语言生成的代码质量高，在代码效率方面可以和汇编语言相媲美。

6）C语言很容易被其他领域接受。单片微机领域首先使用C语言作为开发工具，随着嵌入式技术和DSP的发展，它们也将C语言作为自己的开发工具。由于C语言的固有特点，尽管C++发展很快，但仍然替代不了C语言在很多领域中的应用。

C语言的优点很多，但也有不足之处。例如，运算符优先级太多，不便记忆；有些约定与常规约定不同；类型检验较弱，转换比较随便，不太安全等。但由于上述突出的优点，C语言仍不失为一个实用的通用程序设计语言，学习和使用它的人越来越多。

1.2 C程序的主函数

用C语言编写的程序称为C语言源程序，简称C程序。C程序一般由一个或若干个函数组成，在组成一个程序的若干函数中必须有一个且只能有一个名为main的函数（称为主函数），运行C程序时总是从main函数开始执行。一个函数在其名字之后一定要有一对圆括号，圆括号中是否有参数由编程者决定，目前只介绍无参数的main函数，包含多个函数的C程序结构将在函数一节介绍。C程序所在的文件以后缀“.c”作为文件扩展名。

1.2.1 简单的C程序

C程序至少要有一个主函数，下面是一个简单的标准C程序。

【例1.1】打印字符串。

```
/* 只有主函数的示例程序
   功能:打印字符串* /
#include <stdio.h>                          /* 包含头文件 * /
int main()                                  /* 主函数* /
{
    printf ("Hello! How are you?");         /* 打印字符串* /
    return 0;                               /* 主函数main的返回值* /
}
```

1. *注释语句*

例1.1是一个完整的C程序，“/*”和“*/”之间的内容是语句注释，可以注释一行或者

多行。在编译程序时，注释不参加，所以注释不产生目标代码。所谓目标代码，就是程序可以执行的代码。一定要配对使用“/*”和“*/”以标注不参加编译的内容，否则就会出错。目前，大多使用C++的编译环境（例如 Visual C++ 6.0），C++环境提供一个行注释符“//”，其作用是告诉编译器，从“//”号开始向右的内容均不参加编译。例如：

```
#include <stdio.h>          // 包含头文件
```

由于我们编写C程序时主要使用C++编译器，并且初学者比较容易漏掉“*/”号，因此为了减少错误，熟悉C++的习惯并为以后学习提供方便，本书将在行注释中使用符号“//”作为注释符。

2. 主函数和库函数

在例1.1的程序中有一对花括号“{ }”，可以把它看作程序体括号，也可以用它括起任何一组语句构成一个复合句（或称分程序）。注意，在一个函数中至少应有一对花括号。C程序的一般函数或主函数“main()”之后应有一个“{”，在函数的最后应使用“}”。在一个C程序或一个函数中，“{”和“}”必须成对出现。语句

```
printf ("Hello ! How are you ?");
```

是一个函数调用语句，它调用名为 printf 的库函数，括号内由双引号括起来的部分是函数的参数，即要打印的内容。printf 是标准输出函数，对应于输出设备终端显示器，上述语句表示要在终端输出字符串“Hello ! How are you ?”。语句最后的分号“;”是语句结束标志。也就是说，C语言必须用“;”号作为语句的结束标志。

C函数分为两类：系统本身提供的库函数和自定义函数。所谓自定义函数，就是程序设计人员根据需要自行设计的一段完成一个特定的功能的函数。C语言中的主函数也称为函数，并且将其定义为“int main()”。必须记住，一个C语言程序必须有一个主函数“main()”，而且一个可执行的C语言程序总是从“main()”函数开始执行的。这里使用“int main()”，要求 main 返回一个整数值。但程序有时确实无需返回值，所以用语句“return 0;”返回一个为0的值以满足“int main()”的要求。

库函数又称标准函数，例如标准输出函数 printf。标准函数定义在相应的头文件（头文件的后缀是.h）中。如果要使用这些标准函数，必须先在主函数之前使用#include 语句包含头文件，然后在需要调用它们的地方直接写上函数名，带上参数即可调用库函数。例如，printf 函数的头文件是 stdio.h，先使用如下语句：

```
#include <stdio.h>
```

然后就可以在程序中使用 printf 库函数完成输出功能。

C语言有非常丰富的库函数，应该尽量利用它们。例如，求一个整数的绝对值的库函数 abs 是在头文件 math.h 中定义的，使用时需要先包含 math.h。

【例1.2】使用库函数求一个整数绝对值的例子。

```
#include <stdio.h>
#include <math.h>
int main()
{
    int a;                                  //声明一个整数变量
    printf("请输入一个整数:");
    scanf("%d",&a);                         //读入输入值
    printf("%d 的绝对值的是%d\n", a, abs(a));
    return 0;
```

```
}
```

语句“int a;”用来声明一个整型变量，在程序执行过程中，变量 a 的值可以改变，以存储从键盘输入的整数值。

假如要求 -82 的绝对值，程序运行结果如下：

```
请输入一个整数：-82 <CR>
-82 的绝对值的是 82
```

如果编译系统支持中文，则可以在程序中使用汉字。在以后的例题中，有时为了方便学习，还将采用汉字。读者如在西文操作系统下工作，换成相应的西文语言即可（中文并不影响程序的正确性）。

这里用符号“<CR>”代表按下回车键。意思是在输入 -82 之后，按一下回车键以表示输入完毕，以便通知程序将这个输入值取走并赋给变量 a。一般来讲，程序要求输入时，均要按回车符确认输入完毕，即在按回车键之前，还可以修改输入，一旦按了回车键，就无法修改了。以后只在需要强调的地方使用“<CR>”加以提示，否则不再给出这个符号。

scanf 用来读取用户从键盘输入的值。scanf 和 printf 的使用方法将在 1.3 节介绍。

3. 文件包含语句

#include 语句是文件包含语句，它的作用是把一个指定文件的内容包含进来。书写时，可以使用引号也可以用尖括号。例如：

```
#include "filename"
```

或者

```
#include <filename>
```

都是在程序中把文件 filename 的内容（引号或尖括号是必需的）包含进来。

还要注意，文件名用双引号还是尖括号括起来，其含义并不一样。使用尖括号时，C 编译系统将首先在 C 语言系统指定的目录中寻找包含文件，如果没有找到，就到当前工作目录中去寻找，这是引用系统提供的包含文件所采用的方法。自定义的包含文件一般都放在自己指定的目录中，所以在引用它们时，采用双引号以通知 C 编译器到用户当前的目录下和系统指定的目录下寻找包含文件。例如，用户自定义的包含文件为 myfile.h，引用形式为

```
#include "myfile.h"
```

在程序设计中，文件包含语句是非常有用的。一般 C 系统中带有大量的 .h 文件，用户可根据不同的需要将相应的 .h 文件包含起来。在例 1.2 中，因为要用到 C 语言提供的数学运算库，所以要用

```
#include <math.h>
```

语句。而标准输入输出是定义在库函数 stdio.h 中的，所以要用

```
#include <stdio.h>
```

语句。一般的 C 程序都离不开这条语句，初学 C 语言的读者也最容易遗漏这条语句。

1.2.2 程序语句

下面介绍几种不同的程序语句，但要注意，一条完整的 C 语言的程序语句必须以分号“;”结束。

1. 声明语句

声明语句用来声明变量的类型和初值。“int main()”将主函数声明为整型，即 main 返回整数值。如果将例 1.2 中的语句“int a;”改为“int a=5;”，则声明一个整型变量 a 并将 5 赋给该变量。

2. 表达式语句

表达式语句由一个表达式构成，用以描述算术运算、逻辑运算或产生某种特定动作。最典型的用法是由赋值表达式构成一个赋值语句。例如，“a=3”是一个赋值表达式，而“a=3;”就是一个赋值语句。可以看到，一个表达式的最后加一个分号就构成了一个程序语句。一个程序语句最后必须出现分号，分号是程序语句中不可缺少的一部分。又例如：

```
i=i+1
```

是表达式，不是语句。而

```
i=i+1;
```

是语句，作用是使 i 的值加 1。由此可见，任何表达式都可以加上分号而成为语句。

3. 程序控制语句

程序控制语句用来描述语句的执行条件与执行顺序。C 语言的控制语句参见表 1-1。

表 1-1 C 语言的控制语句及其描述

语句	作用	语句	作用
if( ) ~else~	条件语句	for( ) ~	循环语句
while( ) ~	循环语句	do ~ while( )	循环语句
continue	结束本次循环语句	break	中止循环或 switch 语句
switch	多分支选择语句	goto	转移语句
return	从函数返回语句		

在 9 种控制语句中，括号表示其中包含一个条件，~表示内嵌语句。例如，if( ) ~else~ 的一条具体语句可写成：

```
if ( x > y ) z = x; else z=y;
```

程序控制语句详细的使用方法在第 3 章详细介绍。

4. 复合语句

C 语句又分为简单语句和复合语句两种类型。在 C 语言中，在表达式“x=1”和“printf ("%d\n", x)”等的后面加上分号，即变成“x=1;”和“printf ("%d\n", x);”，就构成了简单语句，分号是语句的终结符。

用花括号“{”和“}”把一些说明和语句组合在一起，使它们在语法上等价于一个简单语句，这样的语句称为复合语句（或称分程序）。例如，在下面的语句中

```
if (a>=0)                          //1
{                                  //2
   printf ("输入为: %d\n", a);     //3
   return a;                       //4
}                                  //5
else                               //6
{                                  //7
   printf("输入为: %d\n",a);       //8
   return(-a);                     //9
```

```
}                                    //10
```

当条件 a > =0 成立时，执行 if 后的复合语句，否则执行 else 之后的复合语句。一个复合语句结尾处的“}”后不能带分号（参见语句 5 和 10），否则会导致错误。另外，也不能遗漏复合语句的最后一条语句与“}”之间的分号（参见语句 4 和 9）。复合语句可由若干条语句组成，这些语句可以是简单语句，也可以是复合语句，从而使 C 语言的语句形成一种层次结构。原则上可以不断地扩大这种层次，复合语句在程序中是一种十分重要的结构。

5. *函数调用语句*

函数调用语句是由一次函数调用加一个分号而构成的一个语句。例如：

```
printf ( "How are you?" );
```

6. *空语句*

只有一个分号的语句称为空语句，即

```
;
```

空语句什么事情也不做。

1.2.3 大小写字母的使用

C 语言严格区分大小写字母，如变量 B 和 b 是完全不同的两个变量。C 语言习惯使用小写字母，而且以下划线“_”开头的标识符一般由系统内部使用，用户最好不要用它作为标识符的第一个字符。另外，习惯上把自定义的标识符用大写字母表示。

1.2.4 程序的书写格式

C 语言的格式很自由，一行也可以写几条语句。不过，C 语言的适当格式对于充分理解这种语言非常重要。一个格式适当的程序和一个不适当格式的程序就像一封打印得很漂亮的信和一封打印得非常凌乱的信，给人的印象是大不一样的。程序的书写格式应该使源代码易于理解，特别是容易被输入这些程序的程序员所理解，这有助于调试复杂程序和修改以前输入的代码。

前面给出的程序就是按此原则书写的。由此可见，使用具有缩进格式和必要的空行的书写风格，可使源代码具有层次性和逻辑性，以增加程序的可读性和可操作性。

一般来讲，每次缩进 5 个字符的位置，并按程序特性设置空行。在本书中，为了节省篇幅，有意识地减少空行及缩进字符的数量。读者在输入程序时，不要模仿，应注意养成良好的书写风格。

在书写程序语句时，一般应遵循如下规则：

1）括号紧跟在函数名的后面，但在 for 和 while 后面，应用一个空格与左括号隔开后续语句，以增加可读性。

2）在数学运算符的左右两侧应各留一个空格以与表达式区别。

3）在表示参数时，逗号后面留一个空格。

4）在 if、for、do…while 和 while 语句中，应合理使用缩进、一对花括号和空行。

5）在 if…else 等语句及其嵌套语句中，书写格式要易于查找错误并提高可读性。

1.2.5 简单 C 程序的基本结构模式

从例 1.2 可以总结出只有主函数的简单 C 程序的基本结构模式如下：

```
文件包含部分                         //包含头文件等
```

```
int main()                    //主程序
{                             //主程序开始
    声明语句部分              //声明程序中所要使用的变量
    执行语句部分              //程序的可执行语句
    return 0;                 //返回值
}                             //主程序结束
```

声明部分目前只涉及变量，不管以后增加何种声明，这些声明必须放在可执行语句之前，即所有声明放在一起。这种结构已经完全能解决第 1 ~ 5 章的程序设计问题，所以应该牢牢掌握并熟练使用。为了减少学习难度，本书到第 6 章再增加主函数之前的内容。

1.3 基本的输入与输出

输入与输出不是 C 语言的一部分，而是以标准函数形式提供的。程序在引用库函数的每一个源程序文件的开头含有如下一行：

```
#include <stdio.h>
```

文件 stdio.h 定义了 I/O 库所用的某些宏和变量，使用#include 可把 stdio.h 文件包含进来，一起编译。虽然有的 C 编译器使用 scanf 和 printf 函数，不需要包含 stdio.h，但建议养成使用这条语句的习惯。Scanf 和 Printf 是 C 语言标准库提供的两个很有用的格式化输入输出标准函数，为程序员实现各种格式化输入输出提供了方便。下面简要介绍格式化输入输出函数 scanf 和 printf，以方便目前的学习，详细的使用方法参见第 2 章。

格式化输出函数 printf 的格式如下：

```
printf (控制字符串, 参数1, 参数2, …);
```

printf 的功能是按照控制字符串对参数进行转换，按格式要求在标准输出设备上输出。控制字符串中包含两种字符：一种是普通字符，将原样输出；另一种是格式符，它是以“%”开头紧跟格式字符，说明其对应参数的输出格式。最常用的 4 种格式符如下：

- d—将参数按十进制整数形式输出。
- c—将参数看作单个字符输出。
- s—将参数所指出的字符串输出，字符串以空字符为终止符。
- f—将参数按浮点数形式输出。

例如，对于下面的语句

```
x1 =1234;
x2 =2345;
printf (" \nx1 = %d x2 = %d \n",x1,x2);
```

其中，“\nx1 = %d x2 = % d\n”是控制字符串，x1、x2 是参数。控制字符串中的 x1 = 和 x2 = 都是普通字符，按原样输出。% d 是格式说明符，依次说明 x1、x2 均按十进制（整型）形式输出。\ n 是换行符，第一个 \ n 是先换行再输出 x1 和 x2。第二个 \ n 是在输出 x1 和 x2 之后，换到下一行。上述语句执行后的输出结果为：

```
x1 =1234 x2 =2345
```

格式化输入函数 scanf 的格式与 printf 类似：

```
scanf (控制字符串, 参数1, 参数2, …);
```

scanf 函数实现从标准输入设备（终端）上按控制字符所规定的格式输入数值或字符，并将输入

内容存入参数所指定的单元中。printf 中的参数是给出要输出值的变量名，而 scanf 中的参数是要给出接收数据的变量地址，这一点大家要特别注意。例如，语句

```
scanf ("%d%d", &x, &y);
```

该语句用于将从终端输入二个十进制数分别赋给 x 和 y。这里 &x 和 &y 表示 x 及 y 的地址。输入的二个十进制数之间用空格分隔。

【例 1.3】编写一个程序，根据输入正方形的边长，求它的周长。

```
#include <stdio.h>
void main()
{
    int a;
    printf("请输入边长:");
    scanf("%d", &a);
    printf("边长=%d周长=%d\n", a, 4*a);
}
```

下面是输入 15 时的运行实例：

```
请输入边长：15
边长=15 周长=60
```

1.4 初学者最容易出现的错误

编程中总难免会有错误，关键是要知道如何检查错误和排除错误。严格来说，错误并无规律，而且错误的种类是千奇百怪的，有时排除的错误也是不值一提的。这项工作确实没有什么技巧，但它又是程序设计中不可或缺的！因此，我们必须对此有正确的认识。尽管还没有真正接触 C 语言的语句，但本节会及早提醒读者在书写 C 语句时注意最容易出现的语法规则错误。因为往往最容易的地方也是最容易出错的地方。

1. 忘记主程序 main() 的返回类型和包含 stdio.h 文件

注意，C 语言提供的标准主函数是 int 型，需要“return 0;”与 main 配合。尽管有些语言环境不需要使用语句“#include <stdio.h>”，但建议要养成使用这条语句的良好习惯，这对以后学习C++ 很有好处。注意，不能在包含库文件和自定义的函数名称后面使用分号，例如“int main();”和“#include <stdio.h>;”都是错误的。这是一个常见错误，就是专业的软件程序员，有时也会犯此类错误。

2. 遗漏分号和花括号，或者增加分号和花括号

初学者容易在 printf 语句后遗漏结束符“;”，而在#include 语句中又误用“;”号作为结束符。花括号是成对出现的，遗漏花括号会改变程序的运行方式。一般来讲，人们更容易忘记作为程序结束的右花括号。

3. 在 scanf 语句中忘记使用“&”号或多了“\n”号

在 scanf 语句中的变量前面应加上 & 号。例如，下面的语句错将 &x 写为 x，且格式符中多了“\n”号。

```
scanf ("%d\n", x); //错误语句
```

4. 程序中的注释符号使用不当

程序中的注释符号以/* 开始，以 */结束。若将结束符号错写成/* 或遗漏，而后面又有注释，就可能会使许多行的程序变成注释，影响运行结果。下面是一个假设的例子：