新世纪高等院校计算机应用技术系列教材

C语言

C Programming and Exercises

程序设计

主　编　王声决　罗坚

副主编　徐文胜　李雪斌　傅清平

主　审　黄明和　聂承启

中国铁道出版社

CHINA RAILWAY PUBLISHING HOUSE

C语言程序设计

主　编　王声决　罗　坚
副主编　徐文胜　李雪斌　傅清平
主　审　黄明和　聂承启

中国铁道出版社
2003·北京

（京）新登字 063 号

内 容 简 介

本书根据教育部提出的非计算机专业计算机基础教学三个层次的要求，参照《全国计算机等级考试大纲（2002 年版）》和《全国高等学校计算机等级考试（江西考区）考试大纲》组织编写。

全书共分 8 章，分别介绍了 C 程序的基本构成与 Turbo C 的使用；基本数据类型、运算符和表达式、基本输入与输出操作；算法的概念和结构化程序设计的三种方法；函数的使用、变量的存储类型；指针和数组的使用、动态内存分配与动态数组和字符串函数；结构和联合的使用、链表及其操作；文件的类型与操作以及面向对象的概念和 C++的基本语法现象。

本书适合作为高等院校的教科书，也可作为广大编程爱好者的自学读物。

图书在版编目（CIP）数据

C 语言程序设计/王声决，罗坚编著. —北京：中国铁道出版社，2002.12

ISBN 7-113-05047-6

Ⅰ.C… Ⅱ.①王…②罗… Ⅲ.C 语言—程序设计 Ⅳ.TP312

中国版本图书馆 CIP 数据核字(2002)第 100138 号

书　　名：C 语言程序设计

作　　者：王声决 罗坚 徐文胜 李雪斌 傅清平

出版发行：中国铁道出版社（100054，北京市宣武区右安门西街 8 号）

策划编辑：严晓舟 马 建 魏 春

责任编辑：苏 茜 彭立辉

封面设计：孙天昭

印　　刷：北京化工印刷厂

开　　本：787×1092 1/16 印张：20.5 字数：485 千

版　　本：2002 年 12 月第 1 版 2003 年 8 月第 2 次印刷

印　　数：4001～7000 册

书　　号：ISBN 7-113-05047-6/TP.838

定　　价：29.00 元

前　言

C 语言是一种非常出色的程序设计语言，它精练、灵活、应用领域广泛，虽然走过了 30 个春秋，至今依然在计算机专业教学和计算机应用程序设计中起着重要作用。

本书根据教育部提出的非计算机专业计算机基础教学三个层次的要求，参照《全国计算机等级考试大纲（2002 年版）》和《全国高等学校计算机等级考试（江西考区）考试大纲》组织编写。作者长期从事高等学校 C 语言课程教学，亲身感受到学生在学习过程中遇到的各种困难。为了使学生能够在 C 语言的学习过程中始终保持强烈的学习兴趣，领悟到程序设计的奥妙，掌握并使用 C 语言解决本专业的实际问题，作者对书中的内容和写作方法作了精心考虑，使得本书具有以下一些特色：

1. 系统全面。内容安排由浅入深，循序渐进。全书围绕结构化程序设计方法，全面展开 C 语言教学内容，示例丰富，习题难易适中，既有助于语法理解的内容，又有提高学习兴趣和实践编程能力的例子。书中同时引用了部分数据结构的算法实例，有利于读者深入学习计算机相关课程。作为 C 语言的发展，本书最后一章介绍 C++的基本内容和面向对象程序设计思想。
2. 实践性强。本书从一开始就强调学习 C 语言最好的方法是上机编写程序，这个观点始终贯穿全书。书中既介绍了使用 Turbo C 调试程序的技术，又介绍了如何在 MS Visual C++ 环境下调试 C 程序。
3. 通俗易懂。在写作方式上既注意到概念的严谨，又考虑到语言叙述的通俗易懂，对一些难以理解的算法和容易混淆的概念使用了图解，使得本书不仅适用于课堂讲授，也便于自学。

全书共分 8 章。第 1 章介绍了 C 程序的基本构成与 Turbo C 的使用。第 2 章介绍了基本数据类型、运算符和表达式、基本输入与输出操作。第 3 章介绍了算法的概念和结构化程序设计的三种方法。第 4 章介绍了函数的使用、变量的存储类型。第 5 章介绍了指针和数组的使用、动态内存分配与动态数组和字符串函数。第 6 章介绍了结构和联合的使用、链表及其操作。第 7 章介绍了文件的类型与操作。第 8 章介绍了面向对象的概念和 C++的基本语法现象。

本书第 1 章、第 2 章、附录 A 和附录 B 由王声决编写，第 3 章由傅清平编写，第 4 章和附录 C 由李雪斌编写，第 5 章、第 6 章由徐文胜编写，第 7 章、第 8 章和附录 D 由罗坚编写。全书由黄明和教授、聂承启教授担任主审。

在本书的编写过程中，杨印根、敖小玲、刘申之、汪浩、李建元、吴克捷、熊刚、王昌晶、傅玲莉、聂伟强等教师对本书提出了许多宝贵的意见，桂训泉、尹红、郭奇峰、王祖勤、张建平、李渊姗、黄惠、胡敏、陈兰芳、崔仙翠、程瑞芬等同志对本书的成稿与编排工作提供了很大帮助，在此一并表示衷心感谢。

由于编者水平有限，书中难免存在错误与不足，恳请读者批评指正。

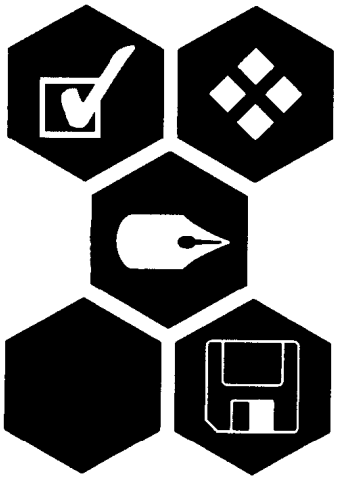
编　者

2002 年 10 月于江西师范大学

目　录

1

简单的 C 程序设计



学习新的程序设计语言的最佳途径是尽早地用它编写程序 、进行程序的调试，解决实际问题。本章从最简单的第一个程序开始，逐步介绍了 C 语言基于函数的程序结构，变量与常量、算术运算、循环结构、基本输入输出标准函数，使读者了解一个 C 语言程序的基本框架和它的书写格式。至于有关语法规则细节，读者先不必深究，学到有关章节时自然会理解。通过介绍 Turbo C 集成环境的使用，要求读者掌握一个 C 程序的编写、编译、连接、调试、直到成功运行的全过程，并能够动手上机操作。由于所举例子并没有用到 C 语言的所有特性，不可能完整地表达出使用 C 语言编程的特点，所以本章最后简要阐述了 C 语言的主要特点，为读者学习以下各个章节起一个点睛的作用，相信读者通过后面各章节的学习，能够真正理解到这些特点。

1.1 几个简单的 C 程序

例1.1：在 DOS 屏幕上显示“hello,world！”

```
/*              第一个 C 语言程序举例                */
/*   包含有关标准库的信息  */
#include <stdio.h>
/*   定义名为main 的函数，它不接收实参值*/
main()
{
/*   main 的语句括在花括号中  */
/*   main 函数调用库函数 printf 原样打印字符序列，\n 代表换行  */
printf("word，hello! \n");
}
```

程序分析：

1. 注解：夹在“ /* ”与“ */ ”之间的字符序列，用于解释该程序是做什么的，目的是为了使程序更易于理解和提起记忆，这些注解在编译时会被自动忽略掉。它们可以在程序中自由地使用，可以出现在程序中的任何位置。字符序列中可以使用空格、制表符或换行符。读者应重视使用注解，养成良好的编程习惯。
2. main 函数：函数是 C 语言的基本单位，每一个 C 程序，不论大小，都是由一个或多个函数组成的。函数是一个单独的程序模块，完成指定的功能。在本例中具有两个函数，一个是名字为 main 的函数。一般而言，可以给函数任意命名，但 main 是一个特殊的函数名。一个 C 程序不论由多少个文件组成，都有一个且只能有一个 main 函数，通常称为主函数。任何一个 C 程序都从它开始运行。main 函数常常还要调用其他函数来协助其完成某些工作，被调用的函数有些是由程序人员自己编写的，有些则由系统函数库提供。本例中的 printf 函数就是由系统函数库提供的。函数中的语句用一对花括号{ }括起来。本例中的 main 函数只包含一条语句 printf("world，hello！\n");，语句最后有一个分号。
3. 函数的调用：当调用一个函数时，先要给出这个函数的名字，然后考虑与被调函数的数据交换。在函数之间进行数据交换的一种方法是让调用函数向被调用函数提供

一串叫做实参（变元）的值。函数名后面的一对圆括号用于把这一串实参（实参表）括起来。在本例子中，main 函数不要求任何实参，故用空实参表（）表示。main 函数用“world,hello!\n”作实参调用 printf 函数。

4. printf 是一个用于打印的格式化输出库函数，在本例中，它用于在 DOS 屏幕上照原样显示双引号内的字符序列 “world，hello！\n”（不包括双引号）。用双引号括住的字符序列叫做字符串。本例中仅使用字符串作为 printf 的实参。
5. 字符串中的字符序列 \n 表示换行符，在显示时它用于指示从下一行的左边开始显示字符。

注意，\n 只表示一个字符。除此之外，C 语言中还有：表示制表符的\t 、表示回退符的\b、表示双引号的 \"、表示反斜杠符本身的 \\。

6. 文件包含命令：#include <stdio.h>。

这里的 #include 称为文件包含命令，其意义是把尖括号 < > 内指定的文件包含到本程序中，成为本程序的一部分。被包含的文件通常是由系统提供的，其扩展名为“.h”。因此也称为头文件或首部文件。如果使用了系统提供的库函数，一般应在文件的开始用 #include 命令，将被调用的库函数信息包含到本文件中。本例中的 #include <stdio.h> 是因为调用了标准输入输出库中的 printf 。需要说明的是，C 语言规定对 scanf 和 printf 这两个函数可以省去对其头文件的包含命令。所以在本例中也可以删去第二行的包含命令#include <stdio.h>。

例1.2：编写程序，计算 t 的值

$$t=1+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\frac{1}{5}$$

这个程序本身仍只由一个名为 main 的函数组成。它要比上面一个例子长，增加了一些新的内容，包括变量说明、算术表达式以及格式输出。该程序如下：

```
#include <stdio.h>
main( )
{
/*  定义整型变量        */
int sum;
   /*  计算累加和的语句    */
sum = 1 + 1/2 + 1/3 + 1/4 + 1/5;
   /*  按整型数输出计算结果  */
printf("The sum is %d\n", sum);
}
```

程序分析：

1. 变量说明： 本例子的主函数体中分为两部分：一部分是说明部分，另一部分是执行部分。在说明部分说明了函数所用到的变量的类型，通常放在函数开始处的可执行语句之前。上一个例子没有使用任何变量，因此没有说明部分。C 语言规定，程序中所有用到的变量都必须先说明，后使用，否则将会出错。说明语句由一个类型名与若干所要说明的变量名组成，int sum ;中的 int 是类型名，sum 是变量名。

类型 int 表示所列变量为整型变量（整数不包含小数部分）。float 型表示所列变量为浮点变量（浮点数可以有小数部分）。除 int 与 float 之外，C 语言还提供了其他一些基本数据类型，包括：char（字符型）、short（短整型）、long（ 长整型）、double（ 双精度浮点型）。另外，还有由这些基本类型构成的数组、结构与联合类型、指向这些类型的指针类型以及返回这些类型的函数，将在后面适当的章节分别介绍它们。

2. 赋值表达式：与 Basic、Pascal、Fortran 语言一样，数值的计算使用赋值表达式实现。在 C 中“=”称为赋值号，含义不同于数学中的等号，而是将其右边表达式的值赋给左边的变量。
3. 算术表达式：书写的数学计算式在 C 语言中要写成合法的 C 语言算术表达式。在 C 中，算术运算符包括“+”、“-”、“*”、“/”以及取模运算符“%”。“/”整数除法要截取掉结果中的小数部分。表达式“x % y”的结果是“x”除以“y”的余数，要求“x”和“y”均为整数，余数的符号与被除数相同。顺便提醒的是，C 语言中的乘号要写成“*”，“1.0/2.0”表示浮点数的 1 除以浮点数的 2，而“1/2”表示整型数 1 除以整型数 2，它们的结果是不一样的。进一步的内容在下一章中介绍。
4. 格式输出：这个例子使用了 printf 函数更多的功能。Printf 是一个通用格式化输出函数，下一章将做详细介绍。本例中 printf 函数具有两个实参，第一个实参是要打印的字符串，其中百分号“%”指示用第二个实参 sum 对其进行替换，“d”指示按整型数打印 sum 的值。
5. 本例运行以后，在 DOS 屏幕上显示如下：

```
The sum is 1
```

显然，结果是不精确的。为什么呢？原因在于 1/2、1/3、1/4 和 1/5 在 C 中计算结果都是 0。为了得到更加精确的计算结果，必须用浮点数代替上面的整型数。

例1.3：修改后的程序

```
#include <stdio.h>
main( )
{
/*  定义单精度浮点型变量  */
  float sum;
   /*计算累加和语句 */
  sum = 1.0 + 1.0/2.0 + 1.0/3.0 + 1.0/4.0 + 1.0/5.0;
   /*  按浮点型数输出计算结果  */
  printf("The sum is %f\n", sum);
   }
```

程序分析：

1. 上一例不精确是因为 1/2 按整数除法，截取后结果为 0 。1.0/2.0 是两个浮点数的除法，不作截取处理。在 C 语言程序中，浮点数最好写成带小数点，即使该浮点数取的是整数值，因为这样使程序比较清晰。

2. %f 对应于单精度的浮点数 sum 。表示按单精度浮点数格式打印计算结果。其他还有几种指定打印宽度的格式串:

%8d 打印十进制整数，至少 8 个字符宽。

%8f 打印浮点数，至少 8 个字符宽。

% .2f 打印浮点数，小数点后有两位小数。

%8.2f 打印浮点数，至少 8 个字符宽，小数点后有 2 位小数。

此外，printf 函数还可以识别如下格式说明：表示八进制数的 %o 、表示十六进制数的 %x 、表示字符的 %c 、表示字符串的%s 以及表示百分号 % 本身的 %%。

printf 函数第 1 个实参中的各个 % 分别对应于第 2 个、第 3 个…第 *n* 个实参，它们在数目和类型上都必须匹配，否则将出现错误。

例1.4：计算 t 的值的另一个 C 程序

对于一个特定任务，可以用多种方法来编写程序。上面的程序直接按照算式写法表达，语句很长，可以想象，如果累加 100 项，累加 1000 项就很难写了。下面是另一版本的 C 程序，完成同样的计算任务。

```
#include <stdio.h>
main( )
{
int i;
float sum ;
sum = 1.0;
for( i=2; i<6 ; i++ )
  sum = sum + 1.0/i;
printf("The sum is %f\n", sum);
}
```

程序分析：

1. for 循环语句：本例的特点是累加计算，所以可以采用循环语句来实现。for 是一种循环语句，for 后面的圆括号内共包含三个部分，它们之间用分号隔开。第一部分 i = 2 是初始化部分，仅在进入循环前执行一次。第二部分是用于控制循环的条件测试部分 i<6 ，这个条件要进行求值。如果所求得的值为真，那么就执行循环体（本例循环体中只包含一条语句 sum = sum + 1.0/i;），然后再执行第三部分 i++(i=i+1)，加步长，并再次对条件求值。一旦求得的条件值为假，那么就终止循环的执行。for 循环语句的循环体可以是单条语句，也可以是用花括号括住的一组语句。
2. 条件测试：用于控制循环执行次数的条件测试 i<6 在 C 中称作条件表达式。如果 i 小于 6，其结果是“真”（True），否则是“假”（False），除小于 < 外，还有大于 > 、小于等于 <= 、大于等于 >= 、等于 = 和不等于!= 等运算符，例如 i 不等于 10 写作 i != 10。

关于循环结构的用法将在第 3 章介绍。

3. 如果某个算术运算符的运算分量都是整数类型，那么就执行整数运算。如果某个算术运算符的运算分量有一个是浮点运算分量和一个是整数类型分量，那么这个整数类型分量在开始运算之前会被转换成浮点类型。所以 1.0/i 不会影响结果的精度。

例1.5：扩充上一个程序功能的例子。

$$t=1+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\frac{1}{5}+\cdots+\frac{1}{m}$$

编写 C 程序，计算 t 的值，其中 m 由键盘输入

C 程序如下：

```
#include <stdio.h>
/*  函数声明   */
float func(int x);
/*          主函数          */
    main()
{  /*    说明部分，定义变量    */
   int m;
   float c;
      /*  提示串 */
   printf("\n Please enter 1 integer number:");
      /*    输入变量 m 的值    */
   scanf("%d",&m);
     /*调用 func 函数，将得到的值赋给 c  */
   c = func(m);
     /*输出 c 值*/
   printf("\n the sum =  %f\n",c);
   return 0;
}
/*定义 func 函数，函数值为浮点型，形参 x 为整型数据 */
    float func(int x)
{  /*func 函数中的说明部分，定义本函数中用到的变量 i,sum */
   /*  i 为整型，sum 为浮点型，同时 sum 赋初值 1.0          */
   int i;
   float sum=1.0;
   for(i=2;i<=x;i++)
   sum = sum + 1.0/i;
      /*  将 sum 的值返回，通过 func 带回调用处    */
   return sum;
    }
```

程序分析：

1. 由功能划分确定函数划分本程序进一步表达了 C 语言基于函数的基本结构。上例中程序的执行过程是，首先在屏幕上显示提示串，请用户输入一个整数，回车后计算出累加和并在屏幕上显示。程序需要处理三件事情：从键盘接受输入 m；计算累

加和（其功能是接受 main 传递的 m，计算出一加二分之一，加三分之一，一直加到 m 分之一的和，并把它返回给 main）；输出计算结果。所以程序除了一个主函数 main 以外，还包含计算累加和的函数 func，处理键盘输入的函数 scanf 和处理输出的函数 printf ，输入和输出这两个函数都是系统库函数，由 Turbo C 系统库提供，用户只要按规定使用而不必自己编写，但累加和计算函数 func 是一个由用户自己编写的自定义函数。

2. 键盘输入函数 scanf ：本程序中的 scanf 语句的作用是从键盘上输入一个整型数给整型变量 m。%d 的含义与前面介绍的 printf 的用法一致，表示按十进制整型数输入，&m 的含义是将输入的数输入给变量 m，注意不要漏写“&”，其含义下一章将作详细介绍。

3. 自定义函数 func：

函数定义的一般形式为：

```
                    /* 首部 */
返回值类型    函数名（形参类型 形参一，形参类型 形参二，… ）
{          /*  函数体  */
    说明部分
    执行部分
}
```

对照上面的例子：

```
返回值类型     函数名(形参类型 形参 )              /* 首部 */
float       func(    int     m   )
{                                  /*  函数体  */
    …


}
```

一个函数名后面必须跟一对圆扩号，函数参数可以有也可以没有，例如前面介绍到的主函数 main，没有用到参数，写成 main()（也可以使用参数，将在第 5 章中介绍它的用法）。

函数体一般包括两部分：

说明部分：

在此定义函数内部所要使用到的变量，对照上面的例子，

```
int i;
float sum=1.0;
```

另外，对被调用的函数也需要进行说明。

执行部分：

由若干条语句组成。对照上面的例子，

```
for(i=2;i<=m;i++)
    sum = sum + 1.0/i;
return sum;
```

func 函数计算得到的值由 return 语句返回给 main 函数。关键字 return 可以后跟任何表达式，函数不一定都返回一个值。不含表达式的 return 语句用于控制返回调用者（但不返