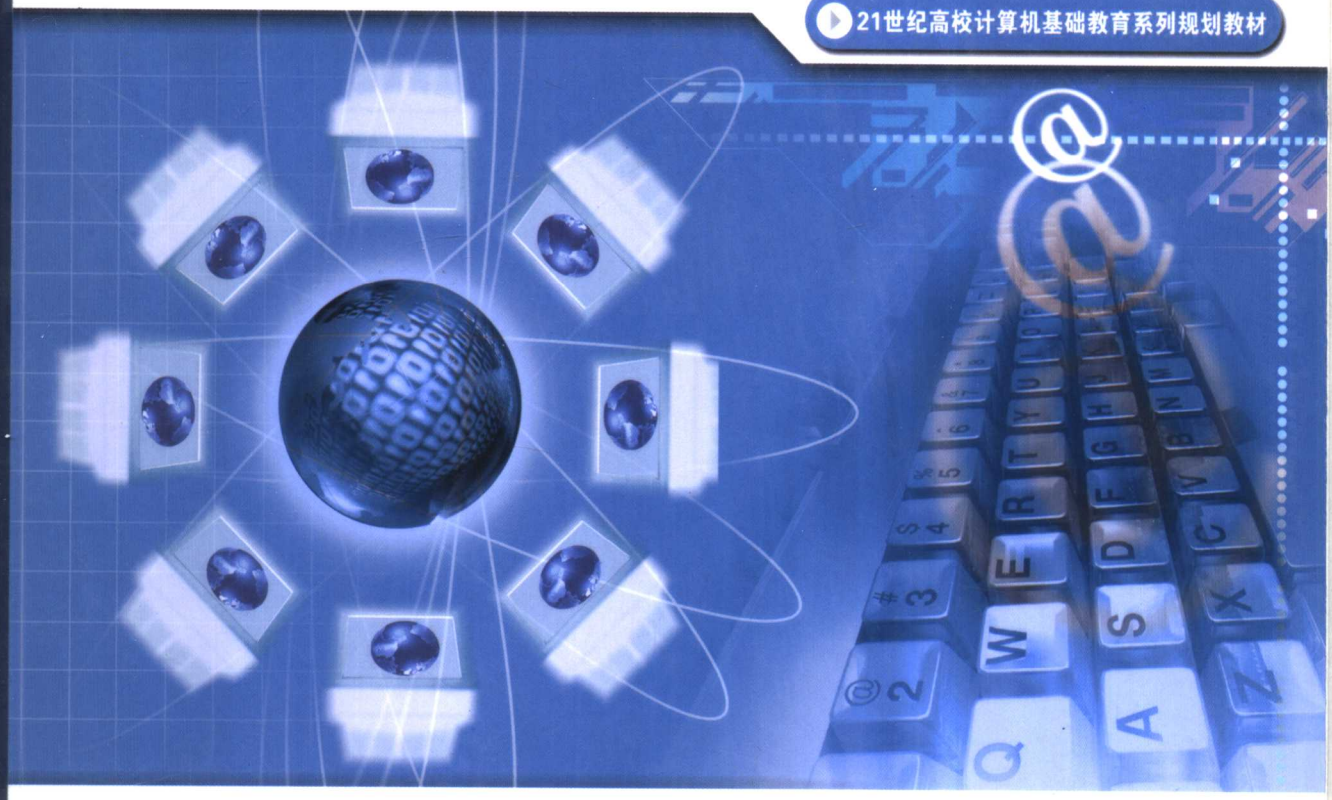
21世纪高校计算机基础教育系列规划教材



C程序设计教程

罗　坚　王声决　主　编
徐文胜　李雪斌　傅清平　副主编

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

21世纪高校计算机基础教育系列规划教材

C程序设计教程

主　编　罗　坚　王声决

副主编　徐文胜　李雪斌　傅清平

主　审　聂承启

中国铁道出版社

CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

C 语言是程序员的入门语言，也是许多大学的计算机专业和计算机公共基础课的首选编程语言。本书以 C 语言的初学者作为主要对象，根据 C 语言程序设计课程的特点，系统地介绍了 C 语言的语法规则及编程实例。全书叙述严谨、实例丰富、难易适中、重点突出，希望通过对本书的学习，能逐步提高读者的计算机应用能力，为进一步学习计算机语言奠定扎实的基础。

本书主要内容包括 C 程序设计入门，数据类型、运算符和表达式，算法与程序设计基础，函数，指针类型与数组类型，结构类型与联合类型，文件，并适当补充了面向对象程序设计 C++的内容。为了避免学习过程中的枯燥乏味，书中还精选了一些富有实用性及趣味性的实例，增强了全书的可读性，使读者能在轻松的气氛中愉快地进行学习。

本书适合作为高校计算机专业及相关专业的 C 语言程序设计教程，也可作为广大编程爱好者的自学读物，同时也是参加各类计算机等级考试备考的一本不可多得的辅导书。

图书在版编目（CIP）数据

C 程序设计教程 / 罗坚，王声决主编. —北京：中国铁道出版社，2007.3
（21 世纪高校计算机基础教育系列规划教材）
ISBN 978-7-113-07731-0

Ⅰ.C… Ⅱ.①罗…②王… 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（2007）第 041358 号

书　　名：C 程序设计教程
作　　者：罗　坚　王声决　等
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　秦绪好
责任编辑：杨　勇
特邀编辑：李振华
封面制作：白　雪
责任校对：王清飞
印　　刷：中国铁道出版社印刷厂
开　　本：787×1092　1/16　**印张：**23　**字数：**538 千
版　　本：2007 年 5 月第 1 版　2007 年 5 月第 1 次印刷
印　　数：1~5 000 册
书　　号：ISBN 978-7-113-07731-0/TP・2096
定　　价：30.00 元

序言

PREFACE

21 世纪是一个新的充满机遇与挑战的时代。在这个世纪之初的几年里，我国高等教育体制正经历着一场平稳而深刻的改革，这场改革正在对传统的普通高等教育的培养目标与当前社会实际需求不相适应的现状进行历史的反思和变革的尝试。

20 世纪末的几年里，高等职业教育的迅速崛起，是关系到高等教育体制变革的一件大事。在短短的几年里，普通中专教育和普通高专教育全面转轨，以高等职业教育为主导的各种形式的应用型人才培养发展到了与普通高等教育等量齐观的地步，其来势之迅猛，令人深思。

无论是正在平稳变革中的普通高等教育，还是迅速发展着的高职教育，都向我们提出了一个同样不可回避的重要问题：中国的高等教育为谁服务，是仅为教育发展自身，还是为包括教育在内的整个社会？当然，答案当然是后者。

既然教育必须为社会服务，它就必须按社会不同领域的需求来完成自己的教育过程。也就是说，教育资源必须按照社会形成的各个行业和岗位的需要实施配置，这就是我们长期以来一直关注的教育目的问题。

普通高等教育走应用型人才培养的道路和走研究型人才培养的道路，不同层次的大学可根据自身的实际情况各取所需，因为这始终是一个理性运行的社会状态下的高等教育正常发展的途径。

高等职业教育的崛起，既是高等教育体制变革的结果，也是高等教育体制变革的一个阶段性表征。它的进一步发展，必将极大地推进中国教育体制变革的进程。作为一种应用型人才培养的教育，它从专科层次起步，进而应用本科教育、应用硕士教育、应用博士教育……当应用型人才培养的渠道贯通之时，也许就是我们迎接中国教育体制变革的成功之日。

正是瞄准这种应用性人才培养目标，根据实际教学的需要，我们组织编写了这套**“21 世纪高校计算机基础教育系列规划教材”**。编写这套系列教材的基本宗旨是：突出应用技术，面向实际应用。在选材上，根据实际应用的需要决定内容的取舍，坚决舍弃那些现在用不到，将来也用不到的内容。在叙述方法上，采取“问题——方法——应用”的模式，这种从实际问题到找出解决问题方法，再到实际运用的“实际——构思——实践”的教学过程符合人们掌握（或认识）一门新技术的认识规律，实践证明已取得了很好的效果。

本套系列教材适用于大学本科生和高等职业学院的学生作为相应课程的教科书，同时适合作为参加各省举办的高等学校在校生的计算机等级考试的辅导教材，也可作为全国计算机等级考试的教材或参考书。

参加本系列教材策划和编写工作的有在计算机界享有盛誉的 10 多位专家和一批有丰富教学经验的老师。在此对于他们的智慧、奉献和劳动表示深切的谢意。中国铁道出版社以很高的热情和效率组织了系列教材的出版工作。在组织编写出版的过程中，得到了江西省计算机等级考试专家组和各高等院校老师的热情鼓励和支持，对此谨表衷心感谢。

本系列教材如有不足之处，请各位专家、老师和广大读者不吝指正。

聂承启

2007 年 3 月

前言

FOREWORD

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，虽然走过了三十多个春秋，至今依然在计算机专业教学和计算机应用程序设计领域中起着重要作用。

本书根据教育部提出的非计算机专业计算机基础教学的要求，参照《全国计算机等级考试考试大纲（2004 年版）》和《全国高等学校计算机等级考试（江西考区）考试大纲与样题（第四版）》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，既有便于语法理解的内容，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 语言的延伸，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 Microsoft 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，对于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，也适用于自学。

全书共分 8 章。第 1 章介绍了 C 程序的基本构成和 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/输出操作，第 3 章介绍了算法的概念和结构化程序设计的 3 种方法，第 4 章介绍了函数的使用和变量的存储类型，第 5 章介绍了指针与数组的使用、动态内存分配、动态数组和字符串函数，第 6 章介绍了结构类型和联合类型的使用、链表及其操作，第 7 章介绍了文件的类型与常用操作，第 8 章介绍了面向对象的概念和 C++程序设计。

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 章和第 6 章由徐文胜编写，第 7 章、第 8 章和附录 D 由罗坚编写。

全书最后由罗坚修改定稿。

为了配合教学，我们还编写了配套实验用书《C 程序设计实验教程》，该书除了为《C 程序设计教程》提供了全部的习题解答之外，还根据教学内容安排了同步的上机实验，提供了全真考试的模拟训练，并增加了全国计算机等级考试二级公共基础知识的专门介绍，已由中国铁道出版社出版。

在本书的编写过程中，得到了全国高等学校计算机等级考试（江西考区）专家组全体专家的悉心指导，在此表示诚挚的谢意。

由于编者水平有限，书中疏漏和不足之处在所难免，敬请广大读者批评指正。

编 者

2007 年 3 月

目 录

CONTENTS

第 1 章　C 程序设计入门 1
1.1　几个简单的 C 程序 1
1.2　C 语言常用符号 8
1.2.1　C 语言的关键字 8
1.2.2　标识符 8
1.2.3　其他符号 9
1.3　C 语言程序的上机调试步骤 9
1.4　Turbo C 集成开发环境（IDE） 10
1.4.1　Turbo C 2.0 的安装 10
1.4.2　Turbo C 2.0 集成化操作界面 10
1.4.3　Turbo C 2.0 简单操作 11
1.5　C 语言的概况 20
1.5.1　C 语言的发展过程 20
1.5.2　C 语言的特点、用途 20
1.5.3　学习建议 21
习题 21
第 2 章　数据类型、运算符和表达式 23
2.1　常用的进位制 23
2.1.1　二进制、八进制和十六进制数 23
2.1.2　十进制、二进制、八进制和十六进制数之间的换算 24
2.2　数与字符在计算机内存中的表示方法 25
2.2.1　机器数和真值 25
2.2.2　原码、反码和补码 25
2.2.3　定点数和浮点数 26
2.2.4　字符编码 27
2.3　常量 27
2.3.1　整型常量 27
2.3.2　实型常量 28
2.3.3　字符常量 28
2.3.4　字符串常量 30
2.3.5　符号常量 30
2.4　变量 31
2.4.1　整型变量 31
2.4.2　实型变量 34
2.4.3　字符型变量 36

2.5 常用运算符与表达式 37
2.5.1 算术运算符与算术表达式 38
2.5.2 赋值运算符和赋值表达式 39
2.5.3 强制类型转换运算符 40
2.5.4 加一、减一运算符 40
2.5.5 逗号运算符和逗号表达式 41
2.5.6 位运算 41
2.6 基本输入/输出操作的实现 43
2.6.1 字符的输入和输出 43
2.6.2 有格式的输入和输出 44
习题 50
第3章 算法与程序设计基础 56
3.1 算法概述 56
3.1.1 算法的概念 56
3.1.2 算法的特性 57
3.2 算法的常用表示方法 58
3.2.1 自然语言表示法 58
3.2.2 流程图 59
3.2.3 N-S 结构流程图 60
3.2.4 伪代码表示法 61
3.2.5 用计算机语言表示算法 62
3.3 结构化程序设计方法 62
3.4 C 语句概述 65
3.5 选择结构程序设计 68
3.5.1 关系运算符和关系表达式 68
3.5.2 逻辑运算符和逻辑表达式 69
3.5.3 if 语句 71
3.5.4 if 语句的嵌套 75
3.5.5 条件运算符和条件表达式 77
3.5.6 switch 语句 78
3.5.7 选择结构程序设计举例 81
3.6 循环程序设计 87
3.6.1 goto 语句以及用 goto 语句构成的循环 87
3.6.2 while 语句 88
3.6.3 do…while 语句 89
3.6.4 for 语句 91
3.6.5 循环的嵌套 94
3.6.6 break 语句 96

3.6.7 continue 语句 97
3.6.8 循环程序设计举例 98
3.7 综合程序应用举例 101
习题 110
第 4 章 函数 **114**
4.1 函数概述 114
4.2 函数的定义 116
4.3 函数的调用与返回值 117
4.3.1 实参与形参的区分 118
4.3.2 函数的调用 119
4.3.3 对被调用函数的原型声明 120
4.3.4 函数的返回语句与返回值 122
4.4 函数的参数传递方式 125
4.4.1 值传递方式 125
4.4.2 地址传递方式 127
4.5 函数的嵌套与递归 130
4.5.1 函数的嵌套调用 130
4.5.2 函数的递归调用 132
4.6 变量的作用域 136
4.6.1 局部变量 136
4.6.2 全局变量 136
4.6.3 分程序 139
4.7 变量的生存期 140
4.7.1 自动变量 140
4.7.2 静态变量 141
4.7.3 外部变量 142
4.7.4 寄存器变量 144
4.8 内部函数和外部函数 145
4.8.1 内部函数 145
4.8.2 外部函数 145
4.9 编译预处理命令 147
4.9.1 宏定义 147
4.9.2 文件包含 149
4.9.3 条件编译 151
习题 152
第 5 章 指针类型与数组类型 **160**
5.1 数据类型的定义 160
5.2 指针类型的定义与使用 161

5.2.1 指针与指针类型的定义 161
5.2.2 指针常量与变量 162
5.2.3 指针参数 165
5.2.4 函数指针 167
5.2.5 void 指针 169
5.3 数组类型的定义与使用 170
5.3.1 数组与数组类型的定义 170
5.3.2 一维数组变量与基本操作 171
5.3.3 数组参数 175
5.3.4 二维数组与二级指针 177
5.3.5 动态数组 183
5.3.6 字符数组与字符串 187
5.3.7 main()函数的数组参数 194
习题 195
第6章 结构类型与联合类型 198
6.1 结构与联合类型的概述 198
6.2 结构类型 200
6.2.1 结构类型与结构变量 200
6.2.2 结构变量的基本操作 203
6.2.3 结构指针与结构数组 205
6.3 链表及其操作 212
6.3.1 链表及其实现 212
6.3.2 链表的基本操作 216
6.3.3 链表的应用 220
6.4 位域与联合类型 224
6.5 枚举类型 228
习题 230
第7章 文件 233
7.1 文件概述 233
7.1.1 文件的概念 233
7.1.2 文件的分类 233
7.1.3 文件缓冲区 235
7.1.4 文件类型指针 236
7.1.5 文件的操作流程 237
7.2 文件的打开和关闭 239
7.2.1 打开文件的函数 240
7.2.2 关闭文件的函数 241
7.3 文件的顺序读/写 241

7.3.1 文本文件的顺序读/写 241
7.3.2 二进制文件的顺序读/写 260
7.4 文件的定位与随机读/写 264
7.4.1 rewind()函数 265
7.4.2 fseek()函数 265
7.4.3 ftell()函数 267
7.5 文件状态检查函数 269
7.5.1 文件读/写结束检查函数 feof() 269
7.5.2 文件出错检查函数 ferror() 272
7.5.3 文件出错复位函数 clearerr() 272
习题 273
第 8 章 面向对象技术与 C++ 279
8.1 C++的起源和特点 279
8.2 简单的 C++程序 280
8.3 C++程序的开发过程 282
8.4 C++的输入和输出 283
8.4.1 用 cout 输出 283
8.4.2 用 cin 输入 283
8.4.3 I/O 流类库操纵符简介 284
8.5 设置函数参数的默认值 286
8.6 内联函数 288
8.7 重载函数 289
8.8 变量的引用 292
8.8.1 引用的概念 292
8.8.2 引用作函数参数 293
8.9 面向对象的基础知识 295
8.9.1 面向对象的概念 295
8.9.2 面向对象程序设计的优点 296
8.9.3 面向对象系统的特性 297
8.10 类和对象 298
8.10.1 类的定义 298
8.10.2 对象的定义 300
8.10.3 对象的成员表示 301
8.11 构造函数 302
8.12 析构函数 306
8.13 继承与派生类 308
8.13.1 继承与派生类的概念 308
8.13.2 派生类的定义格式 309

8.13.3 公有派生类 310
8.13.4 私有派生类 311
8.13.5 保护成员 311
8.13.6 派生类的构造函数 312
习题 314
附录 A 常用字符与 ASCII 码对照表 319
附录 B Visual C++集成环境下调试标准 C 程序的方法 320
附录 C 常用库函数介绍 325
附录 D C 语言编译错误信息 345
参考文献 354

第 1 章 C 程序设计入门

编写程序和调试程序是设计程序的必经之路。本章从第一个最简单的程序开始，逐步介绍 C 语言基于函数的程序结构，引入了常量与变量、算术运算、循环结构、选择结构和基本输入/输出标准函数等概念，帮助读者了解 C 语言程序的基本框架及书写格式。除此之外，还介绍了 Turbo C 集成开发环境（IDE）的使用方法，详述了 C 程序从编辑、编译、连接、调试，直到成功运行这一完整的上机实践过程，希望读者能够尝试着上机操作，调试例题，体会不用纸笔而用计算机来求解问题的快乐。

限于篇幅，本章的示例不可能用到 C 语言的所有特性，没有完整地表达出使用 C 语言编程的特点；对于例题中所涉及的语法规则，读者初学时不必深究，相信通过后续内容的学习，一定能够真正理解这些特点。

1.1 几个简单的 C 程序

【例 1.1】要求在命令提示符窗口中显示“hello,world”这一行文字。

```
/*第一个 C 语言程序举例*/
#include<stdio.h>          /*包含有关标准库的信息*/
main()                     /*定义 main()函数,没有实参值，语句体在花括号中*/
{
   printf("hello, world\n");
/*调用库函数 printf()在命令提示符窗口显示 hello, world 这行文字*/
}
```

程序分析：

（1）注释：程序代码中位于“/*”与“*/”之间的字符序列称为注释，用来解释程序或者语句的作用。被注释的内容可以是一行文字或者连续的多行文字，使用注释能增强程序的可阅读性，使程序易于理解。注释可以在程序中自由地使用，但在程序编译时被自动忽略。读者应重视使用注释，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的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。本例中的另一个函数就是由系统函数库提供的名为 printf 的函数。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("hello,world\n");，语句最后有一个分号。

（3）函数的调用：当要调用一个函数时，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

叫做实参又称（又称变元）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“hello,world \n”作为实参调用 printf()函数。

（4）printf()函数是一个格式化输出库函数，在本例中，它用于在命令提示符窗口照原样显示双引号内的字符序列“hello,world\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()函数的实参。

（5）显示内容中的字符序列'\n'是一个换行符，用于控制从下一行的最左边位置开始显示其后续的字符。'\n'只表示一个字符，注意不能把它看成\和 n 这两个符号。

具有这种特征的字符称为转义字符，除'\n'之外，还有表示制表符的'\t'、表示退格符的'\b'、表示双引号的'\"'、表示反斜杠符本身的'\\'，详见第 2 章“2.3.3 字符常量”中的介绍。

（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”，常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。本例中使用#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：如果一个程序只使用了 scanf()或 printf()这样的库函数，此时对头文件的包含命令可以省略不写，因此本例中也可以删去#include <stdio.h>这一行。有关函数使用的详细内容将在第 4 章中介绍。

【例 1.2】数值计算：编写程序，计算 t 的值。

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

这个程序仍只由一个名为 main 的函数组成，与例 1.1 相比，本例增加了一些新的内容，包括变量说明、算术表达式和格式输出。程序如下：

```
#include<stdio.h>
main()
{
   int sum;                              /*说明整型变量 sum*/
   sum=1+1/2+1/3+1/4+1/5;                /*计算累加和*/
   printf("The sum is %d\n",sum);        /*按整型数格式显示计算结果*/
}
```

程序分析：

（1）变量说明：本例的主函数体中包含两部分，一是说明部分，二是执行部分。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定义（有时也称说明），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;语句中的 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）和 double（双精度浮点型）等。另外，还有由这些基本类型构成的数组、结构与联合类型和指向这些类型的指针类型以及返回这些类型的函数，具体内容在后面的章节中介绍。

（2）赋值表达式：与 Basic，Pascal，Fortran 语言一样，C 语言中数学值的计算使用赋值表达式实现。在 C 语言中，“=”称为赋值号，含义不同于数学中的等号，而是将其右边的表达式的值赋给左边的变量。

（3）算术表达式：编程时需把传统的数学式子转换为 C 语言中等价的表达式。C 语言中的算术运算符包括加法+、减法-、乘法*、除法/和求模运算符%。求模运算只适用于整数，表达式 $x\%y$ 代表 x 除以 y 所得的余数，这里 x 和 y 均为整数，余数的符号与被除数 x 相同；整数中的除法与实数中的除法结果不同：整数中的除法结果仍为整数，只保留商的整数部分而抛弃小数部分，即截尾取整，而实数除法与数学中介绍的一致。因此，5%2 的结果为 1，5/2 的结果为 2，5.0/2.0 的结果为 2.5，更详细的内容将在第 2 章中介绍。

（4）格式输出：本例使用了 printf()函数更多的功能。printf()函数是一个通用格式化输出函数，将在第 2 章中详细介绍。本例中 printf()函数具有两个变元，第一个变元是要打印的字符串，其中百分号“%”指示用第二个变元 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，在命令提示符窗口中显示如下：

```
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 语言程序中计算结果都是 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()
{
   float sum;                                    /*定义单精度浮点型变量*/
   sum=1.0+1.0/2.0+1.0/3.0+1.0/4.0+1.0/5.0;      /*求累加和*/
   printf("The sum is %f\n",sum);                /*按浮点型数格式显示计算结果*/
}
```

程序分析：

（1）例 1.2 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的形式，即使该浮点数取的是整数值，因为这样可以使程序更加清晰。

（2）“%f”对应于单精度的浮点数 sum。表示按单精度浮点数格式显示计算结果。其他几种指定显示宽度的格式串如下：

“%8d”显示十进制整数，至少 8 个字符宽；

“%8f”显示浮点数，至少 8 个字符宽；

“%.2f”显示浮点数，小数点后有两位小数；

“%8.2f”显示浮点数，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”和表示百分号本身的“%%”。

printf()函数第1个变元中的各个%分别对应于第2个、第3个、……、第 n 个变元，它们在数目和类型上都必须匹配，否则将出现错误。有关数据类型、运算符和表达式的内容将在第2章中介绍。

【例1.4】例1.3的另一种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1 000项，就很困难了。

下面是例1.3的另一种编程方法，完成的是同样的计算任务。

```
#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（即重复计算）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，对 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，就终止循环的执行。for循环语句的循环体可以是单条语句，也可以是包含在花括号内的一组语句。

（2）条件测试：用于控制循环执行次数的条件测试（i<6）在C语言中称做关系表达式。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（!=）等关系运算符，在C语言中数学里的 a=3 应写为 a==3，数学里的 $b\neq10$ 应写为 b!=10。

关于循环结构的用法将在第3章中介绍。

（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，另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，1.0/i 不会影响结果的精度。

【例1.5】扩充例1.3功能的又一个C程序。从键盘上输入数列的项数 m，此处要求 m>5，要求编程计算下列数列的和。

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

```
#include<stdio.h>
float func(int x);          /*函数声明*/
                            /*主函数*/
main()
{
   int m;                   /*说明部分，定义变量 m 和 c*/
   float c;
                            /*提示用户从键盘输入一个整数，该数表示求和的项数值*/
   printf("\n Please enter an integer number m (m>5):");
                            /*由键盘输入变量 m 的值*/
   scanf("%d",&m);
                            /*如果输入的项数值大于 5，则计算累加和*/
   if(m>5)
  {
      c=func(m);            /*调用函数 func()计算累加和，将得到的值赋给 c*/
      printf("\n The sum=%f\n",c);   /*显示结果*/
  }
   return 0;
}
                            /*定义函数 func()，函数值为浮点型，参数 x 为整型*/
  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*/
      int i;                /*i 为整型*/
      float sum=1.0;        /*sum 为浮点型，同时给 sum 赋初值 1.0*/
      for(i=2;i<=x;i++)
        sum=sum+1.0/i;
      return sum;           /*返回 sum 的值，通过 func()带回调用处*/
  }
```

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。本例中程序的执行过程是，首先在屏幕上显示提示字符串，请用户输入一个整数，按回车键后计算出累加和并在屏幕上显示。程序需要处理三件事情：从键盘接收输入的 m；如果 m>5 则计算累加和（其功能是接收 main()传递的 m，计算出 1 加 1/2，加 1/3，一直加到 1/m 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()，输入函数和输出函数都是系统库函数，由 Turbo C 系统库提供，用户只需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 语句中的用法一致，表示按十进制整型数输入，&m 的含义是将输入的数赋给变量 m，注意不要漏写“&”，其含义将在第 2 章中进行详细介绍。

（3）if选择语句：在求解过程中，往往需要根据输入的数据作出逻辑判断，对不同的结果作出不同的处理。例如本题如果输入的m大于5，则作累加和计算，否则不作计算。

简单if语句的格式如下：

```
if(表达式)
{
   语句或语句组
}
```

if语句的执行过程是，如果if处的表达式的值不为0，则表示结果为真，将执行花括号内的语句；如果表达式的值为0，则花括号内的语句将不执行。有关选择结构程序设计的内容将在第3章中介绍。

（4）自定义函数func()。

用户自定义函数的一般定义形式为：

```
函数返回值类型   函数名(形参类型  形参1,形参类型  形参2,… )    /*函数首部*/
{                                                            /*函数体*/
    函数的说明部分
    函数的执行部分
}
```

对照上述的定义格式，自定义函数fun()的形式为：

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

自定义函数必须先定义后使用。在定义一个自定义函数时，函数名的后面必须跟上一对圆括号，这是函数结构的特有标志。圆括号中的形参可以是一个或多个，也可以一个都没有，当没有形参时，此时的圆括号也必须加上，不能省略不写。

函数体一般包括说明部分和执行部分。

说明部分：在此定义函数内部所要使用到的变量。

对照上面的例子为：

```
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。

对照上面的例子为：

```
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语句用于使控制返回调用者（但