



新世纪高职高专实用规划教材

● 计算机系列

# C语言程序设计教程

C YUYAN CHENGXU SHEJI JIAOCHENG

高福成 潘旭华 赵玉刚 编著



清华大学出版社

新世纪高职高专实用规划教材 计算机系列

# C 语言程序设计教程

高福成 潘旭华 赵玉刚 编著

清华大学出版社

北 京

## 内 容 简 介

C 语言是目前广泛流行的通用程序设计语言,是全国许多高校普遍开设的一门计算机基础课程,也是许多计算机专业人员和计算机爱好者学习程序设计语言的首选。

本书以程序设计为主线,以程序设计的需要带动语言知识的学习,系统介绍了 C 语言程序设计的基本知识、基本数据类型和数据运算、程序控制结构、数组、指针、函数、结合和联合、文件、编译预处理、图形和图像处理技术等,并通过丰富的程序设计实例,加强程序设计思维方法和实际编程的训练。每章都安排了上机指导和大量的练习题,帮助读者提高实际操作水平,自我检测 C 语言知识的掌握程度,以及阅读程序、调试程序和编制程序的实际能力。

本书可作为大专院校“C 语言程序设计”课程的教材,全国计算机等级考试(C 语言)培训班教材,也可供从事程序开发的工程技术人员参考。

版权所有,翻印必究。举报电话:010-62782989 13901104297 13801310933

本书封面贴有清华大学出版社激光防伪标签,无标签者不得销售。

### 图书在版编目(CIP)数据

C 语言程序设计教程/高福成,潘旭华,赵玉刚编著. —北京:清华大学出版社,2004.9

(新世纪高职高专实用规划教材 计算机系列)

ISBN 7-302-09417-9

I. C… II ①高…②潘…③赵… III C 语言—程序设计—高等学校:技术学校—教材 IV.TP312

中国版本图书馆 CIP 数据核字(2004)第 090623 号

出 版 者:清华大学出版社 地 址:北京清华大学学研大厦

<http://www.tup.com.cn> 邮 编:100084

社 总 机:010-62770175 客户服务:010-62776969

组稿编辑:王景先

文稿编辑:张彦青

封面设计:陈刘源

印 刷 者:北京市清华园胶印厂

装 订 者:三河市金元装订厂

发 行 者:新华书店总店北京发行所

开 本:185×260 印张:28.25 字数:672 千字

版 次:2004 年 9 月第 1 版 2004 年 9 月第 1 次印刷

书 号:ISBN 7-302-09417-9/TP·6574

印 数:1~5000

定 价:36.00 元

---

本书如存在文字不清、漏印以及缺页、倒页、脱页等印装质量问题,请与清华大学出版社出版部联系调换。联系电话:(010)62770175-3103 或(010)62795704

# 《新世纪高职高专实用规划教材》序

## 编写目的

目前,随着教育的不断深入,高等职业教育发展迅速,进入到一个新的历史阶段。学校规模之大,数量之众,专业设置之广,办学条件之好和招生人数之多,都大大超过了历史上任何一个时期。然而,作为高职院校核心建设项目之一的教材建设,却远远滞后于高等职业教育发展的步伐,以至于许多高职院校的学生缺乏适用的教材,这势必影响高职院校的教育质量,也不利于高职教育的进一步发展。

目前,高职教材建设面临着新的契机和挑战:

(1) 高等职业教育发展迅猛,相应教材在编写、出版等环节需要在保证质量之前提下加快步伐,跟上节奏。

(2) 新型人才的需求,对教材提出了更高要求,科学性、先进性和实用性充分体现。

(3) 高职高专教育自身的特点是强调学生的实践能力和动手能力,教材的取材和内容设置必须满足不断发展的教学需求,突出理论和实践的紧密结合。

(4) 新教材应充分考虑一线教师的教学需要和教学安排。

有鉴于此,清华大学出版社在相关主管部门的大力支持下,组织近百所高等职业技术学院的优秀教师以及相关行业的工程师,推出了一系列切合当前教育改革需要的高质量的面向就业的职业技术实用型教材。

## 系列教材

本系列教材主要涵盖以下领域:

- 计算机基础及其应用
- 计算机网络
- 计算机图形图像处理与多媒体
- 电子商务
- 计算机编程
- 电子电工
- 机械
- 数控技术及模具设计
- 土木建筑
- 经济与管理
- 金融与保险

另外,系列教材还包括大学英语、大学语文、高等数学、大学物理、大学生心理健康等基础教材。所有教材都有相关的配套用书,如实训教材、辅导教材、习题集等。

## 教材特点

为了完善高等职业技术教育的教材体系，全面提高学生的动手能力、实践能力和职业技术素质，特意聘请有实践经验的高级工程师参与系列教材的编写，采用了一线工程技术人员与在校教师联合编写的模式，使课堂教学与实际操作紧密结合。本系列丛书的特点如下：

- (1) 打破以往教科书的编写套路，在兼顾基础知识的同时，强调实用性和可操作性。
- (2) 突出概念和应用，相关课程配有上机指导及习题，帮助读者对所学内容进行总结和提高。
- (3) 设计了“注意”、“提示”、“技巧”等带有醒目标记的特色段落，让读者更容易得到有益的提示与应用技巧。
- (4) 增加了全新的、实用的内容和知识点，并采取由浅入深、循序渐进、层次清楚、步骤详尽的写作方式，突出实践技能和动手能力。

## 读者定位

本系列教材针对职业教育，主要面向高职高专院校，同时也适用于同等学历的职业教育和继续教育。本丛书以三年高职为主，同时也适用于两年制高职。

本系列教材的编写和出版是高职教育办学体制和动作体制改革下的产物，在后期的推广使用过程中将紧紧跟随职业技术教育发展的步伐，不断吸取新型办学模式，课程改革的思路和方法，为促进职业培训和继续教育的社会需求奉献自己的一份力。

我们希望，通过本系列教材的编写和推广应用，不仅有利于提高职业技术教育的整体水平，而且有助于加快改进职业技术教育的办学模式、课程体系和教学培训方法，形成具有特色的职业技术教育的新体系。

教材编委会

# 新世纪高职高专实用规划教材

## · 计算机系列编委会

顾问 吴文虎

主编 边奠英

副主编 刘 璟 李兰友

委员 (以姓氏笔画为序)

王景先 王温君 刘光然 许洪杰

曲建民 迟丽华 李 平 汪 洋

林章波 张 炜 张 玲 赵家俊

高福成 傅连仲 韩 劼 喻 梅

詹青龙 魏则荣

# 前 言

程序设计课程包含两个任务,一个是学会一种语言,第二个是具有一定的设计程序能力。相对来说,学习语言比较容易,因为它是一种工具,是一种技能,通过边学边用达到熟能生巧并不困难。而程序设计能力的培养需要花较大的力气,因为它需要较强的逻辑思维能力,要理解和掌握一批常用的算法,这对初学者来说比较陌生,比较抽象。算法的严密性需要我们周密思考,稍有不慎就会出错,如果对算法未能了如指掌,查错也很困难;算法的多样性和多变性则使人眼花缭乱,如果对它望而生畏、望而却步,就不能达到锻炼自己、提高自己的目的。由于程序本身就是算法的一种描述形式,学习程序设计的重点应放在掌握和使用算法上,通过读懂现成的程序来理解算法,通过自行编制程序来掌握算法,是学习程序设计的基本方法。

本书以程序设计为主线,以程序设计的需要带动语言知识的学习。书中内容按两条线索安排:一条是数据类型,从基本数据类型(第2章)、数组(第4章)到指针类型(第5章)和复合数据类型(第7章);另一条是程序控制结构,从最简单的顺序结构(第1章)到选择结构、循环结构(第3章),再到模块化结构(第6章)。对圈套规模的程序设计特别有用的文件操作、编译预处理、图形和图像处理则分别安排在第8章、第9章和第10章。为了强化程序设计的训练,每章都安排了大量的程序设计实例,通过这些实例介绍常用算法和编程技巧,配备的上机指导则可以帮助读者加强实际操作训练。每章都有量大面广的练习题,包括选择题、填空题、程序改错题和编程题,便于进行语言知识、阅读程序、调试程序和编制程序实际能力的自我检测。

书中附录部分给出了各章上机练习题和习题的参考答案,以方便读者自学。

本书由高福成主编,参加编写的有潘旭华、赵玉刚、梁静毅、李军、王桂荣、曲立敏、王梦倩、边佳、毛莹、高菡、杨玉兰等老师。由于作者水平所限,疏漏和错误之处难以避免,恳请使用本书的读者提出宝贵意见。

在本书编写及出版过程中,得到天津大学边奠英教授的悉心指导,清华大学出版社第三事来部王景先编辑也为本书的出版付出了辛勤的劳动,编著者一并表示真诚的感谢。

编著者  
2004年6月

---

# 目 录

---

|                                                 |    |
|-------------------------------------------------|----|
| <b>第 1 章 C 程序设计的初步知识</b> .....                  | 1  |
| 1.1 C 语言的产生和发展 .....                            | 1  |
| 1.2 C 程序的基本结构 .....                             | 2  |
| 1.2.1 一个简单的 C 程序 .....                          | 2  |
| 1.2.2 C 程序的基本结构 .....                           | 3  |
| 1.3 C 程序的书写风格 .....                             | 6  |
| 1.4 简单的 C 程序设计 .....                            | 6  |
| 1.4.1 赋值语句的简单使用 .....                           | 6  |
| 1.4.2 格式输入输出函数的<br>简单使用 .....                   | 7  |
| 1.4.3 库函数和标题文件 .....                            | 7  |
| 1.4.4 简单程序设计举例 .....                            | 8  |
| 1.5 C 程序的开发过程 .....                             | 9  |
| 1.6 Turbo C 2.0 集成环境的使用 .....                   | 11 |
| 1.7 上机指导 .....                                  | 15 |
| 1.8 习题一 .....                                   | 19 |
| <b>第 2 章 基本数据类型和数据运算</b> .....                  | 21 |
| 2.1 基本数据类型 .....                                | 21 |
| 2.2 常量及其类型 .....                                | 23 |
| 2.3 变量及其类型 .....                                | 26 |
| 2.4 符号常量 .....                                  | 30 |
| 2.5 算术运算 .....                                  | 32 |
| 2.5.1 算术运算符 .....                               | 32 |
| 2.5.2 算术表达式 .....                               | 33 |
| 2.6 赋值运算 .....                                  | 37 |
| 2.7 逗号运算 .....                                  | 39 |
| 2.8 关系运算和逻辑运算 .....                             | 41 |
| 2.9 测试数据长度运算 .....                              | 45 |
| 2.10 位操作 .....                                  | 46 |
| 2.10.1 位逻辑运算 .....                              | 46 |
| 2.10.2 移位运算 .....                               | 48 |
| 2.10.3 位操作运算符组成的<br>复合赋值运算符 .....               | 49 |
| 2.11 常用数学函数 .....                               | 50 |
| 2.12 不同类型数据的输入输出 .....                          | 51 |
| 2.12.1 printf() 函数 .....                        | 51 |
| 2.12.2 scanf() 函数 .....                         | 55 |
| 2.12.3 单字符输入输出函数<br>(getchar()、putchar()) ..... | 57 |
| 2.12.4 单字符输入函数<br>(getche()和 getch()) .....     | 59 |
| 2.13 上机指导 .....                                 | 60 |
| 2.14 习题二 .....                                  | 63 |
| <b>第 3 章 程序流程控制</b> .....                       | 71 |
| 3.1 结构化程序设计的概念 .....                            | 71 |
| 3.2 选择结构程序设计 .....                              | 72 |
| 3.2.1 用 if-else 语句构成二<br>分支选择结构 .....           | 72 |
| 3.2.2 用 if-else if-else 语句构成<br>多分支选择结构 .....   | 77 |
| 3.2.3 用条件表达式实现的<br>选择结构 .....                   | 79 |
| 3.2.4 用 switch 语句构成多<br>分支选择结构 .....            | 80 |
| 3.2.5 用 switch 和 break 构成<br>多分支选择结构 .....      | 83 |
| 3.3 循环结构程序设计 .....                              | 85 |
| 3.3.1 用 for 语句构成循环结构 .....                      | 85 |
| 3.3.2 用 while 语句构成循环结构 .....                    | 89 |
| 3.3.3 用 do-while 语句构成<br>循环结构 .....             | 91 |
| 3.3.4 循环的嵌套 .....                               | 93 |

|                                      |            |                                       |            |
|--------------------------------------|------------|---------------------------------------|------------|
| 3.4 转移控制语句 .....                     | 95         | 6.3.1 虚实结合的过程和特点 .....                | 201        |
| 3.4.1 break 语句 .....                 | 95         | 6.3.2 变量的传递 .....                     | 202        |
| 3.4.2 continue 语句在循环结构<br>中的作用 ..... | 96         | 6.3.3 数组(含字符串)的传递 .....               | 205        |
| 3.4.3 goto 语句和标号 .....               | 98         | 6.4 用函数返回值传递数据 .....                  | 209        |
| 3.5 上机指导 .....                       | 99         | 6.5 用全局变量传递数据 .....                   | 211        |
| 3.6 习题三 .....                        | 105        | 6.6 用函数指针调用函数 .....                   | 212        |
| <b>第 4 章 数组和字符串</b> .....            | <b>117</b> | 6.7 变量的存储类型对函数<br>调用的影响 .....         | 213        |
| 4.1 数组的概念 .....                      | 117        | 6.8 函数的存储类型对函数<br>调用的影响 .....         | 216        |
| 4.2 数组的定义和初始化 .....                  | 119        | 6.9 函数的递归调用 .....                     | 216        |
| 4.2.1 数组的定义和存储 .....                 | 119        | 6.10 main()函数的参数和返回值 .....            | 218        |
| 4.2.2 数组的初始化 .....                   | 121        | 6.11 上机指导 .....                       | 220        |
| 4.2.3 通过初始化定义隐含<br>尺寸数组 .....        | 124        | 6.12 习题六 .....                        | 226        |
| 4.3 数组的基本操作 .....                    | 125        | <b>第 7 章 复合数据类型</b> .....             | <b>238</b> |
| 4.4 数组在数值计算中的应用 .....                | 131        | 7.1 结构类型 .....                        | 238        |
| 4.5 数组在字符串处理中的应用 .....               | 136        | 7.1.1 结构类型的定义和<br>存储模式 .....          | 238        |
| 4.5.1 字符串处理函数 .....                  | 136        | 7.1.2 结构变量、结构数组<br>和结构指针的定义 .....     | 240        |
| 4.5.2 字符型数组在文字<br>处理中的应用 .....       | 141        | 7.1.3 访问结构变量和结构<br>数组的成员 .....        | 242        |
| 4.6 上机指导 .....                       | 144        | 7.1.4 结构变量、结构数组和<br>结构指针的初始化和赋值 ..... | 244        |
| 4.7 习题四 .....                        | 147        | 7.1.5 结构类型的数据在函数<br>间的传递 .....        | 247        |
| <b>第 5 章 指针</b> .....                | <b>156</b> | 7.1.6 用递归结构处理链表 .....                 | 253        |
| 5.1 地址、指针和指针变量 .....                 | 156        | 7.2 联合类型 .....                        | 256        |
| 5.2 指针的运算 .....                      | 160        | 7.3 位段结构类型 .....                      | 259        |
| 5.3 指针的定义及用指针访问变量 .....              | 163        | 7.4 枚举类型 .....                        | 262        |
| 5.4 用指针访问一维数组 .....                  | 166        | 7.5 类型定义 typedef .....                | 263        |
| 5.5 用指针访问二维数组 .....                  | 168        | 7.6 上机指导 .....                        | 265        |
| 5.6 用指针处理字符串 .....                   | 171        | 7.7 习题七 .....                         | 268        |
| 5.7 二级指针 .....                       | 177        | <b>第 8 章 文件</b> .....                 | <b>280</b> |
| 5.8 用指针进行内存动态分配 .....                | 178        | 8.1 文件概述 .....                        | 280        |
| 5.9 上机指导 .....                       | 180        | 8.2 文件的打开与关闭 .....                    | 282        |
| 5.10 习题五 .....                       | 183        | 8.3 文件的读写操作 .....                     | 284        |
| <b>第 6 章 函数</b> .....                | <b>194</b> |                                       |            |
| 6.1 C 程序的模块结构 .....                  | 194        |                                       |            |
| 6.2 函数的定义和调用 .....                   | 196        |                                       |            |
| 6.3 用参数传递数据 .....                    | 201        |                                       |            |

|                                  |            |                                              |            |
|----------------------------------|------------|----------------------------------------------|------------|
| 8.3.1 文本文件读写函数.....              | 285        | 10.1.3 文本方式下拉式<br>菜单的设计.....                 | 347        |
| 8.3.2 二进制文件读写函数.....             | 290        | 10.2 图形方式.....                               | 352        |
| 8.4 文件检测函数.....                  | 292        | 10.2.1 图形方式的设置.....                          | 352        |
| 8.5 文件的顺序存取和随机存取.....            | 294        | 10.2.2 图形屏幕函数.....                           | 355        |
| 8.6 上机指导.....                    | 304        | 10.2.3 屏幕绘图程序设计.....                         | 362        |
| 8.7 习题八.....                     | 307        | 10.3 动画制作.....                               | 371        |
| <b>第9章 编译预处理.....</b>            | <b>320</b> | <b>附录1 C语言运算符集.....</b>                      | <b>377</b> |
| 9.1 编译预处理命令.....                 | 320        | <b>附录2 ASCII 代码表.....</b>                    | <b>378</b> |
| 9.2 宏定义.....                     | 320        | <b>附录3 Turbo C 2.0 常用库函数<br/>及其标题文件.....</b> | <b>379</b> |
| 9.3 文件包含.....                    | 326        | <b>附录4 Turbo c 2.0 编译错误信息.....</b>           | <b>387</b> |
| 9.4 条件编译.....                    | 328        | <b>附录5 上机指导参考答案.....</b>                     | <b>390</b> |
| 9.5 上机指导.....                    | 331        | <b>附录6 习题参考答案.....</b>                       | <b>403</b> |
| 9.6 习题九.....                     | 333        |                                              |            |
| <b>第10章 Turbo C 图形和图像处理.....</b> | <b>338</b> |                                              |            |
| 10.1 文本方式.....                   | 338        |                                              |            |
| 10.1.1 文本方式的基本设置.....            | 339        |                                              |            |
| 10.1.2 字符屏幕函数.....               | 342        |                                              |            |

---

# 第 1 章 C 程序设计的初步知识

---

**教学提示：**本章主要介绍 C 语言和 C 程序设计的初步知识，这些知识在后续章节中还会出现，学习完这些内容后，读者对 C 程序将会有有一个初步的了解。

**教学目标：**熟悉 C 程序的基本结构和书写风格；掌握 C 语言关键字和标识符的命名方法，了解 C 编译系统提供的标题文件的功能，学会用赋值语句和格式输入/输出函数编制简单的 C 程序，掌握 Turbo C 集成环境的使用。

## 1.1 C 语言的产生和发展

C 语言是一种面向过程的通用程序设计语言，它以表达简明、使用灵活、结构化的流程控制、丰富的数据结构和操作符集合、良好的程序可移植性和高效率的目标代码为特征。C 语言不仅是一种高级语言，还兼有低级语言的功能，因此既可用于编写系统程序，也可用于编写不同领域的应用程序。

C 语言的祖先是 ALGOL60(ALGOriThm Language)。ALGOL60 具有可读性和可移植性好的特点，但它不能直接对硬件进行操作，不宜用来编写系统程序。人们开始考虑一种集高级语言和低级语言功能于一身的语言，以便可以用它编写可读性和可移植性都比较好的系统程序。

1963 年，英国的剑桥大学和伦敦大学首先将 ALGOL60 发展成 CPL，1967 年，剑桥大学的 Martin Richards 将 CPL 改制成 BCPL，1970 年，美国贝尔实验室的 Ken Thompson 将 BCPL 修改成 B 语言，并用 B 语言开发了第一个高级语言 UNIX 操作系统，在 DEC 公司的 PDP-7 小型机上运行。

1972 年，Ken Thompson 与在 UNIX 系统上的亲密合作者 Dennis M. Richie 将 B 语言修改设计成 C 语言。C 语言既保持了 BCPL 和 B 语言的精炼和接近于硬件的特点，也克服了它们过于简单、数据无类型等缺点。1973 年，Ken Thompson 和 Dennis M. Richie 又合作将 1969 年用汇编语言编写的 UNIX 操作系统改用 C 语言编写，其中 C 语言代码占 90% 以上，只保留了少量汇编语言代码。这样就使得 UNIX 操作系统向其他类型的机器上移植变得相当简单。到 70 年代中期，UNIX 操作系统和 C 语言作为软件设计师的良好工具开始风靡世界。

1978 年，以 UNIX 第 7 版中的 C 编译程序为基础，Brain W. Kernighan 和 Dennis M. Richie 合著了影响深远的名著《The C Programming Language》。这本书中介绍的 C 语言成为后来广泛使用的 C 语言版本的基础，称为 K&R C 语言。其后的十几年中，适用于不同机种和不同操作系统的 C 编译系统相继问世，从而把 C 语言的应用推向了更加广泛普及的阶段。1983 年美国国家标准局(ANSI)制定了 C 语言标准。这个标准不断完善，并从 1987 年开始实施，称为 ANSI C。1988 年，K&R 修改了他们的经典著作《The C Programming Language》，

按 ANSI C 标准重新编写了该书。现在一般称 ANSI C 为新标准或现代 C, K&R C 为旧标准或传统 C。此后陆续出现的各种 C 语言版本, 如 Microsoft C 5.0、9.0, Turbo C 2.0、3.0, Quick C 等都是与 ANSI C 兼容的版本。它们的语法和语句功能是一致的, 差异表现在各自的标准函数库中收纳的函数种类、格式和功能上, 尤其是图形函数库的差异更大一些。

C 语言是一种面向过程的语言, 意思是用 C 语言编程时, 必须按照算法的实现过程逐条语句编写, 通知计算机一步一步怎么做。进入 80 年代后, 面向对象的程序设计概念日益普及。所谓面向对象, 是通过类和对象把程序所涉及的数据结构和对它施行的操作有机地组织成模块, 将数据本身和对数据的处理细节进行最大限度的封装, 从而使开发出来的软件易重用、易修改、易测试、易维护、易扩充。正如其他传统的程序设计语言都在发展自己面向对象的新版本一样, C 语言在发展的同时, 也朝着支持面向对象程序设计 (OOP—Object Oriented programming) 的方向迈出了步伐。1986 年, 美国 AT&T 的贝尔研究所的 Bjarne Stroustrup 推出了 C 语言的超集 C++ 语言, 也叫“带类的 C”。在软件开发领域, C 和 C++ 一直是最有生命力的程序设计语言, 大多数优秀的程序员都是 C/C++/Visual C++ 的高手, 大多数成功的软件也是用 C 或 C++ 语言编写的。

随着信息时代的到来, 由于设计 Internet 上的 Web 浏览器的需要, 1994 年出现了 Java 语言, 它不仅支持 OOP, 而且具有软硬件平台无关性的特点, 适合于程序员进行网络开发。Java 脱胎于 C++, 被称为 C++ 的衍生语言。2000 年, Microsoft 推出了其下一代计算计划——Microsoft.NET, 它是一个具有公共语言子集的开发平台, 实现了多种语言及其类库的无缝集成, 使应用程序的开发更容易、更简单, 并可能成为下一代网络通信标准。C# 是专为这一平台推出的全新语言, 它也派生于 C 和 C++, 并具有语法简洁、面向对象、与 Web 紧密结合、卓越的安全性能、灵活性和兼容性俱佳等特点, 成为 .NET 平台一流的网络编程工具。

C、C++、Visual C++、C# 的发展轨迹反映了人类信息处理能力的不断深化。

## 1.2 C 程序的基本结构

C 程序的基本结构指的是一个 C 程序的基本组成成分。本节通过一个程序实例来说明 C 程序的基本结构。

### 1.2.1 一个简单的 C 程序

下面是一个简单的 C 程序, 它只有一个 main() 函数。

【例 1.1】编制程序根据给定的半径  $r$ , 计算圆的面积  $a$ 。

先定义三个变量分别表示半径、面积和周长, 然后指定半径  $r$  的值, 由计算公式计算面积和周长并存放在变量  $a$  和  $c$  中, 最后输出半径  $r$ 、面积  $a$  和周长  $c$  的值。程序如下:

```
#define PI 3.14159          /*宏定义*/
main()                     /* 函数名 */
{                           /* 函数体开始 */
    float r,a,c;           /* r:半径; a:面积; c:周长*/
```

```
    r=2.5;                /* 给定 r 的值 */
    a=PI*r*r;             /* 计算面积 a */
    c=2*PI*r;             /* 计算周长 c */
    printf("r=%f,a=%f\n",r,a); /* 输出 r,a */
}
```

程序第1行用#define 定义符号常数PI, 它代表圆周率; 第2行是函数名main(), C语言规定: 一个程序由若干个函数组成, 至少有一个用main()命名的主函数; 第3行至第9行为函数体, 是程序的变量定义和执行部分, 其中左花括号表示函数体的开始, 右花括号表示函数体的结束。每行右边“/\*”和“\*/”之间的文字称为注释, 它对程序的执行没有影响, 只对程序进行某些必要的说明, 以帮助阅读程序。程序运行结果如下:

```
r=2.500000, a=19.634956
```

### 1.2.2 C 程序的基本结构

C 程序的基本结构是函数, 一个C程序由一个或多个C函数组成, 一个C函数由若干条C语句构成, 一条C语句由若干基本单词形成。

#### 1. C 函数

C 函数是完成某个整体功能的最小单位, 它是相对独立的模块。一个简单的C程序可能只有一个主函数, 而复杂的C程序则可能包含一个主函数和任意多个其他函数。所有的C函数都具有的相同结构是: 函数名、形式参数和函数体。

##### (1) 函数名

函数名是标识和调用函数的依据。在C程序中, 主函数有固定的名称main, 其他函数则可以根据标识符的命名方法任意取名。主函数通常包括了整个程序的轮廓, 由它再调用其他函数。

##### (2) 形式参数

形式参数是函数调用时进行数据传递的主要途径。形式参数简称形参、虚参或哑元, 必须放在函数名后面的一对圆括号中。当形式参数超过一个时, 相互之间用逗号隔开。有的函数也可能没有形式参数, 但圆括号不能省略。

##### (3) 函数体

函数体用来描述函数的功能。函数体要用花括号括起来, 主要有两大部分, 第一部分是本函数内部用到的局部变量类型定义(C语言中, 所有的变量都要先定义后使用), 第二部分是完成本函数功能的语句序列。

图1.1描述了C程序的一般格式, 除了main()函数外, 函数fun1()到funn()均为用户自行命名的函数。程序执行时, 无论各个函数的书写位置如何, 总是先执行main()函数, 由main()函数调用其他函数, 最后终止于main()函数(有关函数的详细介绍见第6章)。

```
    全局变量说明
main()
{  局部变量说明
    语句序列
}
fun1(形式参数表)
{  局部变量说明
    语句序列
}
fun2(形式参数表)
{  局部变量说明
    语句序列
}
.....
funn(形式参数表)
{  局部变量说明
    语句序列
}
```

图 1.1 C 程序的一般格式

## 2. C 语句

C 语句是完成某种程序功能(如赋值、输入、输出等)的最小单位,所有的 C 语句都以分号结尾。C 语句可分为表达式语句、复合语句和空语句三类。

### (1) 表达式语句

任何一个 C 表达式的末尾加上分号后,就构成表达式语句。如

```
i=0;
x=x+1;
```

等。

### (2) 复合语句

复合语句是将一组 C 语句用花括号括起来,它在语法上被视为一条语句。如

```
while (i<10)
{  sum=sum+i;
    i++;
}
```

复合语句通常用在条件分支或循环语句中。有时为了数据隐藏的目的,用复合语句形成一个代码块,块中定义的局部变量不会对程序的其他部分产生副作用。

### (3) 空语句

空语句指的是只有一个分号的语句。如

```
for(i=0; i<1000; i++)
;
```

由一条 for 语句(循环语句)和一条空语句组成。空语句用作循环语句的循环体,表示什么也不做。事实上,这个循环的功能是延迟一小段时间。有时,空语句被用作转向点。

### 3. 基本单词

一个 C 语句由若干个基本单词组成。C 语言共有五种基本单词,即关键字(亦称保留字)、标识符、常数、操作符和分隔符。例如,语句

```
float r, a, c;
```

中, float 是关键字(代表数据类型), r、a 和 c 是标识符(表示变量);又如,语句

```
a=PI*r*r, c=2*PI*r;
```

中, “=”、“\*”是操作符, “2”是常数, “PI”是符号常数。

#### (1) 关键字

关键字是 C 语言中有特定意义和用途且不得作为它用的字符序列, ANSI C 标准规定的关键字有 32 个, 如表 1.1 所示。

表 1.1 ANSI C 关键字

|          |        |         |       |          |          |
|----------|--------|---------|-------|----------|----------|
| auto     | break  | case    | char  | const    | continue |
| default  | do     | double  | else  | enum     | extern   |
| float    | for    | goto    | if    | int      | long     |
| register | return | signed  | short | sizeof   | static   |
| struct   | switch | typedef | union | unsigned | void     |
| volatile | while  |         |       |          |          |

注意: 所有的 C 关键字都必须小写。

#### (2) 标识符

标识符是用来表示变量名、数组名、函数名、指针名、结构名、联合名、枚举常数名、用户定义的数据类型名及语句标号等用途的字符序列, 可由 1~32 个字符组成, 第一个字符必须是字母或下划线, 后面的字符可以是字母、数字或下划线。例如

AB, Ab, aB, ab, A\_b, \_ab, ab\_, s2d, W\_length

等都是正确的标识符, 而

A+B, A'B, A.B, 2abc,  $\alpha$ ,  $\beta$ , d%等是错误的标识符。

**注意:** 标识符不能与 C 关键字相同, 而且区分大小写。例如 AB 和 Ab 是两个不同的标识符, ELSE 可以作为标识符, 它不会与 C 关键字 else 混淆。

顺便指出, 在 C 编译系统的库函数中, 经常使用以下划线“\_”打头的函数名或变量名, 所以在程序中也应尽量避免使用以“\_”打头的标识符, 免得与库函数冲突。

#### (3) 常数

C 语言中的常数包括数值常数(如 123, -23.5, 1.2E4 等)、字符常数(如 'a', 'B', 'c' 等)、字符串常数(如 "xyz", "good morning" 等)、符号常数以及枚举常数(枚举将在第 7 章将介绍)。

#### (4) 操作符

操作符包括各种运算符(如+、-、\*、/),有特定意义的标点符号(如花括号、方括号、圆括号、逗号)等。

#### (5) 分隔符

分隔符用来分隔相邻的标识符、关键字和常数。最常用的分隔符是空格,此外还可以用制表符、换行符、换页符等作为分隔符。

## 1.3 C 程序的书写风格

C 语言采用自由的程序书写风格。

(1) 每个函数在整个程序文件中的位置任意,主函数不一定出现在程序的开始处,但不管主函数位于程序中的何处,程序运行时总是从主函数开始。

(2) 每个程序行中的语句数量任意,既允许一行内写几条语句,也允许一条语句分几行书写,但每条语句都必须以分号“;”结束。有时还可以在程序的适当地方(如两个函数之间)加进一个或多个空行,使程序结构更加清晰。

(3) 注释的位置任意,注释可以出现在程序的任何地方,既可以独占一行或几行,也可以出现在某语句的开头或结尾处。如果注释占有几行,则每一行都要以“/\*”开始,以“\*/”结束,“\*”和“/”之间不能有空格。注释不是 C 语句,它对程序的编译和运行没有影响,使用注释的惟一目的是增加程序的可读性。

(4) 尽管 C 程序的书写几乎没有限制,但为使程序清晰易读,通常每行写一条语句。不同结构层次的语句从不同的位置开始,即按缩进格式书写成阶梯形状,可以用 Tab 键或空格键调整各行的起始位置。

## 1.4 简单的 C 程序设计

一个程序通常具备三个功能,即输入数据、数据运算和输出结果。在 C 语言中,数据运算主要是由赋值语句完成的,数据的输入、输出则需要调用 C 编译系统提供的输入/输出函数完成。本节简要介绍赋值语句和格式输入、输出函数的使用,以便能进行最简单的程序设计。

### 1.4.1 赋值语句的简单使用

赋值语句具有计算和存储两大功能,其一般格式为

$v=e;$

其中,  $v$  是变量名,  $e$  是表达式。例如,  $x=a+b$ , 其功能是先计算表达式  $a+b$  的值, 再将该值转换成和  $x$  相同的数据类型, 然后将该值保存在变量  $x$  中。

### 1.4.2 格式输入输出函数的简单使用

输入指的是通过输入设备将原始数据送入计算机,以便计算机对它们进行处理;输出指的是将保存在内存中的计算结果送到输出设备上,以便人能阅读。C 语言不提供输入输出语句,C 程序中的输入和输出主要是通过 C 编译系统提供的输入输出函数实现。其中用得最多的是格式输出函数 `printf()` 和格式输入函数 `scanf()`。本节对它们做一简单介绍,详细介绍见第 2 章。

#### 1. 格式输出函数 `printf()`

格式输出函数 `printf()` 用来按指定的格式输出数据,是内存与显示器之间进行数据交换的主要手段,也是用户获得程序运行结果的主要途径,其一般调用形式为

```
printf("格式控制字符串",输出项目清单);
```

其中, `printf` 是函数名,其后括号中的内容是该函数的参数:格式控制字符串用双引号括起来,用来规定输出格式,例如, `%d` 用来输出十进制整数, `%f` 用来输出实数;输出项目清单中包含零个或多个输出项,它们可以是常数、变量或表达式,当有多个输出项时,相互之间用逗号隔开。例如

```
printf("%f,%f",x,y);
```

用来按实数形式输出变量 `x` 和 `y` 的值。

#### 2. 格式输入函数 `scanf()`

格式输入函数 `scanf()` 用来按指定的格式接收输入数据,是内存与键盘之间进行数据交换的主要手段,也是用户为程序运行提供原始数据的主要途径,其一般调用形式为:

```
scanf("格式控制字符串",输入项目清单);
```

其中, `scanf` 是函数名,其后括号中的内容是该函数的参数:格式控制字符串用双引号括起来,用来规定输入格式,其用法和 `printf()` 相同;输入项目清单中至少包含一个输入项,且必须是变量的地址(变量地址的表示形式是在变量名前面加一个“&”),当有多个输入项时,相互之间用逗号隔开。例如

```
scanf("%f%d",&a,&b);
```

用来接收从键盘输入的一个实数和一个十进制整数,并分别存放在变量 `a` 和 `b` 中。

### 1.4.3 库函数和标题文件

上节介绍的 `printf()` 和 `scanf()` 是 Turbo C 编译系统提供的库函数。库函数不是 C 语言的组成部分,而是由 C 编译系统提供的一些非常有用的功能函数,例如各种输入输出函数、数学函数、字符串处理函数等,可供用户在自己的程序中直接调用。这些库函数的说明、类型定义和宏定义都分门别类地保存在相应的标题文件(也称头文件)中,而对应的子程序则存放在运行库(.lib)中,当需要使用系统提供的库函数时,只要在程序开始用

```
#include <标题文件>
```

或

```
#include "标题文件"
```

就可以调用其中定义的库函数。`printf()` 和 `scanf()` 是在标题文件 `stdio.h` 中定义的,因此,在