



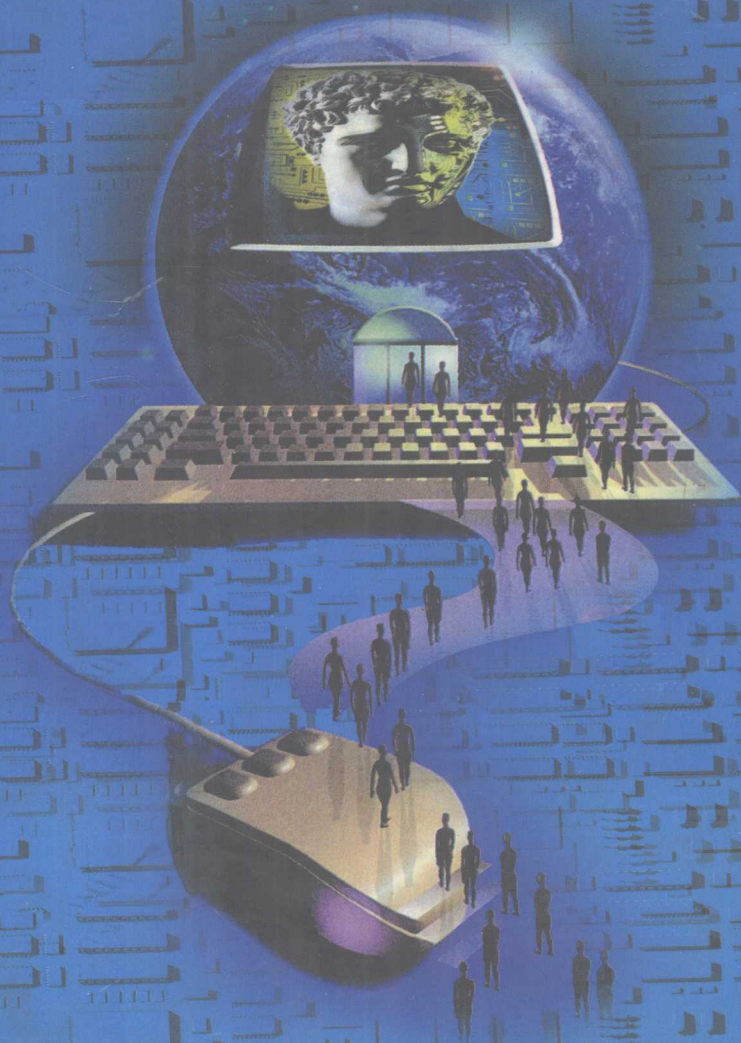
普通高校系列教材·信息技术

# C 语言程序设计

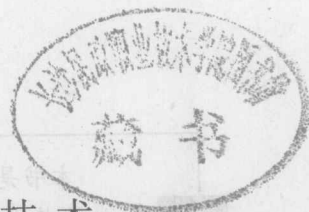
普通高校教材（信息技术）编委会组编

主编：蒋劲柏

主审：虞焰智



南京大学出版社



普通高校系列教材·信息技术

# C 语言程序设计

蒋劲柏 主 编

虞焰智 主 审



A0184219

南京大学出版社

内  
容  
简  
介

本书是全国普通高等学校系列教材中的一本,介绍了C语言程序设计的基本知识。

C语言是一种中级语言,同时具有高级语言和低级语言的一些良好特性。因此,它受到了广大编程人员的欢迎,目前已成为最主流的编程语言之一。

本书面向的是C语言的初级入门者,介绍了C语言的基本知识和初步程序设计方法。在讲述时,力求语言深入浅出,通俗易懂。对于大部分的知识点,都配有一定的实例,以求读者能够更好地掌握。

本书适合普通高校用作相关课程的教材。

图书在版编目(CIP)数据

C语言程序设计/蒋劲柏主编.—南京:南京大学出版社,2001.11

ISBN 7-305-02473-2

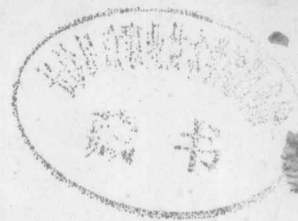
I. C... II. 蒋... III. C语言—程序设计—高等学校—教材  
IV. TP312

中国版本图书馆 CIP 数据核字(2001)第 082877 号

丛 书 名 普通高校系列教材·信息技术  
书 名 C语言程序设计  
主 编 蒋劲柏  
出版发行 南京大学出版社  
社 址 南京市汉口路 22 号 邮编 210093  
电 话 025-3596923 025-3592317 传真 025-3303347  
网 址 www.njupress.com  
电子函件 nupress1@public1.ptt.js.cn  
经 销 全国新华书店  
印 刷 合肥学苑印刷厂印刷  
开 本 787×1092 1/16 印张:16.25 字数:410 千  
版 次 2001 年 11 月第 1 版 2002 年 5 月第 2 次印刷  
ISBN 7-305-02473-2/TP·227  
定 价 23.00 元

\* 版权所有,侵权必究。

\* 凡购买南大版图书,如有印装质量问题,请与所购图书销售部门联系调换。



## 普通高校教材(信息技术)编委会

主任:孙钟秀 中国科学院院士

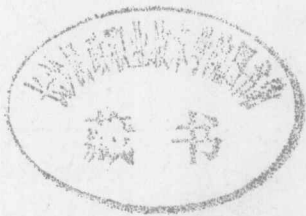
副主任:张福炎 南京大学教授

陈国良 中国科学技术大学教授

钱洲胜 中国计算机函授学院院长

### 委员(按姓氏笔划排序):

|     |              |       |           |
|-----|--------------|-------|-----------|
| 王佩珠 | 西安交通大学       | 王 文 兰 | 桂林电子工业学院  |
| 王蔚韬 | 重庆建筑大学       | 伍良富   | 成都电子科技大学  |
| 成松林 | 东南大学         | 刘存书   | 郑州信息工程大学  |
| 朱大奇 | 安徽工业大学       | 朱宝长   | 西安电子科技大学  |
| 孙德文 | 上海交通大学       | 杜象元   | 上海交通大学    |
| 李茂青 | 厦门大学         | 李学干   | 西安电子科技大学  |
| 杨来利 | 兰州大学         | 何淑兰   | 北京科技大学    |
| 张凤祥 | 华中科技大学       | 张绍林   | 河北行政学院    |
| 张维勇 | 合肥工业大学       | 张民坤   | 云南工业大学    |
| 张景书 | 哈尔滨工程大学      | 邹华跃   | 中国计算机函授学院 |
| 赵良全 | 新疆大学         | 皇甫正贤  | 东南大学      |
| 洪志全 | 成都理工学院       | 姚君遗   | 合肥工业大学    |
| 高 平 | 浙江大学         | 陶世群   | 山西大学      |
| 曹翊旺 | 湖南省计算机高等专科学校 | 梁文康   | 山东大学      |
| 韩国强 | 华南理工大学       | 舒 洪   | 南昌大学      |
| 葛 燕 | 中国科学技术大学     | 解世耀   | 辽宁大学      |
| 谭耀铭 | 南京大学         | 黎庆国   | 合肥工业大学    |



## 出版前言

近些年来我国的高等教育事业有了长足的发展,高校招生人数年年递增,越来越多的年轻人有机会接受正规的高等教育。这一举措无疑对我国的社会进步和经济发展有着重要的意义。但是人们也深刻地认识到,高等教育质量的好坏是一个不容忽视的关键性问题,而保证教育质量的一个重要环节就是抓好教材建设,但是教材内容陈旧,教学手段落后的现象一直存在着,尤其像计算机技术这样的新兴领域发展迅猛、知识更新日新月异、教学内容落后于新技术新知识的矛盾显得尤为突出。基于上述两方面考虑,在南京大学出版社的鼎力相助下,一个以组编高校信息、电子类专业教材为主要任务的教材编委会成立了。

针对我国高等教育的现状和信息、电子技术的发展趋势,编委会组织部分高校的专家教授进行了深入的专题研讨。大家一致认为,在当前情况下组编一套紧跟新技术发展、符合高校教学需要、满足大学生求知欲望的系列教材势在必行,这将有助于规范教学体系、更新教学内容、把握教学质量,培养合格人才。专家们还对教材的结构、内容、体例及配套服务等方面提出了具体要求。为了能使这套教材逐步完善,并促进全国各地高校教学质量的提高,编委会决定在教材之外认真做好三件事:第一,为每本教材配备一本供学生使用的学习参考书,其主体内容为学习方法指导、习题分析与解答、典型题解或课程设计、模拟测试卷及解答、实验指导书;第二,定期对教材内容进行修订,及时补充新技术新知识,并根据具体情况组编新的教材;第三,有计划地组织各地高校教师进行教学交流与研讨,通过这种途径来提高偏远地区的师资水平。我们相信,通过各方面的大力支持和大家的不懈努力,这套教材会逐步被广大师生所接受,并在使用过程中得到完善、充实。

本套教材所涉及到的源程序、素材、效果图、电子教案等,请到 <http://ccccc.net.cn/~book> 下载。

大家都知道,组编这样一套系列教材是个牵涉面很广的大工程。这个工程不仅在起步阶段需要得到各级教育主管部门、各高等院校、出版社的大力支持和协助,而且在使用过程中也离不开各位专家、教授、学生的热心呵护和指导,因此,殷切期待所有的能人志士关心我们,帮助我们,向我们提出好的建议或意见,为我们指出教材中的不足之处。

最后,感谢所有为本套系列教材出版付出辛勤劳动的同志们。

普通高校教材(信息技术)编委会

2001年10月

## 编 者 的 话

本书是普通高等学校系列教材中的一本,介绍了C语言程序设计的基本知识。

C语言是一种介于高级语言和低级语言之间的中级语言,既具有高级语言人性化的特点,又具有与低级语言相似的操纵硬件的能力。因此,自推出之日起,就受到了广大编程人员的欢迎,目前已成为最主流的编程语言之一。

本书介绍了C语言程序设计的基础知识,包括C语言的语法结构、数据类型和初步程序设计等内容。全书共分10章:

第1章介绍了C语言的起源、发展、特点、风格、C语言程序开发的实质及C语言程序开发环境。强调C语言程序表现为函数的集合体,指明了编写正确而有效的函数和充分利用库函数是C语言程序设计的最基本技能。

第2章介绍了C语言的词法、数据和表达式的概念。这是C程序设计最基础的内容,对其中的各个基本概念要仔细地理解,特别是数据类型及变量的概念更要认真体会。

第3章介绍了简单的C语言程序设计知识。这一章介绍的是C语言的语法结构,也就是C语言使用的各种语句。其中特别介绍了两个最常使用的函数:标准输入函数和标准输出函数,读者应熟练掌握。

第4章是第3章的延伸,介绍了C语言语句中的流程控制语句。需要强调的是,对于三种流程的使用,是编写程序必须掌握的技巧。

第5章介绍的是一种重要的数据结构——数组。数组是一种常用的数据结构,需要认真学习。

第6章介绍的是指针知识。指针是C语言中最富特色的部分,也是较难掌握的部分,需要花较多的精力重点掌握。可以说,学好指针也就学好了C语言。

第7章介绍了函数的定义、说明及调用规则,同时对库函数的使用进行了归纳性的总结。函数间的数据传递是本章的难点。

第8章介绍了三种非基本数据类型:结构、联合与枚举。其中结构类型需要熟练掌握,在其基础上可以构造出链表等复杂的数据结构。

第9章介绍的是编译预处理与分别编译。

第10章介绍的是文件的一些基本知识。

阅读本书,读者可以初步掌握C语言的相关知识,能使用C语言进行一些简单的程序设计。

本书面向的是C语言的初级入门者,在讲述时我们注意到了这一点,力求语言深入浅出,通俗易懂,对于大部分的知识点,都配有一定的实例,以求读者能够更好地掌握。

本书适合高等学校用作相关课程的教材。

编 者

2001年11月

# 目 录

|                    |      |
|--------------------|------|
| 第1章 概述             | (1)  |
| 1.1 C语言的起源和发展      | (1)  |
| 1.1.1 C语言的起源       | (1)  |
| 1.1.2 C语言的标准化      | (2)  |
| 1.2 C语言的特点         | (2)  |
| 1.3 C语言程序设计概貌      | (4)  |
| 1.3.1 程序设计的本质      | (4)  |
| 1.3.2 C语言程序设计的本质   | (5)  |
| 1.3.3 C语言程序设计初步    | (8)  |
| 1.4 C语言程序上机环境      | (10) |
| 1.4.1 C语言集成开发环境的安装 | (10) |
| 1.4.2 C语言集成开发环境的配置 | (11) |
| 1.4.3 C语言程序上机步骤    | (11) |
| 1.4.4 集成开发环境简介     | (12) |
| 习题                 | (18) |
| 第2章 词法、数据和表达式      | (19) |
| 2.1 C语言的词法         | (19) |
| 2.1.1 字符集          | (19) |
| 2.1.2 保留字          | (21) |
| 2.1.3 标识符          | (22) |
| 2.2 C语言的数据类型       | (22) |
| 2.2.1 什么是数据类型      | (22) |
| 2.2.2 C语言的数据类型     | (23) |
| 2.2.3 数据类型的长度      | (24) |
| 2.3 常量             | (24) |
| 2.3.1 整型常量         | (25) |
| 2.3.2 实型常量         | (25) |
| 2.3.3 字符常量         | (26) |
| 2.3.4 字符串常量        | (26) |
| 2.3.5 符号常量         | (27) |
| 2.4 变量             | (27) |
| 2.4.1 变量的数据类型和定义   | (27) |
| 2.4.2 变量的存储类型与作用域  | (30) |
| 2.4.3 变量的作用范围      | (31) |
| 2.4.4 变量的初始化       | (32) |

|                             |      |
|-----------------------------|------|
| 2.5 运算符 .....               | (33) |
| 2.5.1 算术运算符 .....           | (34) |
| 2.5.2 关系运算符 .....           | (34) |
| 2.5.3 逻辑运算符 .....           | (35) |
| 2.5.4 增1减1运算符 .....         | (35) |
| 2.5.5 赋值运算符 .....           | (36) |
| 2.5.6 算术自反赋值运算符 .....       | (36) |
| 2.5.7 逗号运算符 .....           | (37) |
| 2.5.8 条件运算符 .....           | (37) |
| 2.5.9 长度运算符 .....           | (38) |
| 2.5.10 位运算符 .....           | (38) |
| 2.5.11 运算符的优先级和结合性 .....    | (39) |
| 2.6 表达式 .....               | (40) |
| 2.6.1 算术表达式 .....           | (41) |
| 2.6.2 关系表达式 .....           | (41) |
| 2.6.3 逻辑表达式 .....           | (42) |
| 2.6.4 赋值表达式 .....           | (42) |
| 2.6.5 逗号表达式 .....           | (43) |
| 2.6.6 条件表达式 .....           | (43) |
| 习题 .....                    | (43) |
| 第3章 简单C语言程序设计 .....         | (48) |
| 3.1 表达式语句 .....             | (48) |
| 3.2 空语句 .....               | (49) |
| 3.3 复合语句 .....              | (49) |
| 3.4 控制语句 .....              | (51) |
| 3.5 函数调用语句 .....            | (51) |
| 3.6 标准输入输出函数 .....          | (52) |
| 3.6.1 标准输出函数 printf() ..... | (52) |
| 3.6.2 标准输入函数 scanf() .....  | (56) |
| 习题 .....                    | (58) |
| 第4章 程序流程的控制 .....           | (60) |
| 4.1 程序的三种基本结构 .....         | (60) |
| 4.1.1 顺序结构 .....            | (60) |
| 4.1.2 选择结构 .....            | (60) |
| 4.1.3 循环结构 .....            | (61) |
| 4.2 选择结构流程控制语句 .....        | (61) |
| 4.2.1 if 语句 .....           | (61) |
| 4.2.2 switch 语句 .....       | (64) |

|                         |       |
|-------------------------|-------|
| 4.3 循环结构流程控制语句 .....    | (70)  |
| 4.3.1 while 语句 .....    | (70)  |
| 4.3.2 do-while 语句 ..... | (73)  |
| 4.3.3 for 语句 .....      | (75)  |
| 4.3.4 三种循环语句的嵌套 .....   | (78)  |
| 4.3.5 改变循环结构的三种语句 ..... | (80)  |
| 习题 .....                | (83)  |
| 第5章 数组 .....            | (90)  |
| 5.1 一维数组 .....          | (90)  |
| 5.1.1 一维数组的定义和使用 .....  | (91)  |
| 5.1.2 一维数组的初始化 .....    | (91)  |
| 5.1.3 一维数组的应用举例 .....   | (92)  |
| 5.2 字符数组与字符串 .....      | (94)  |
| 5.2.1 字符数组与字符串的概念 ..... | (95)  |
| 5.2.2 字符串函数 .....       | (96)  |
| 5.2.3 字符串数组的应用举例 .....  | (100) |
| 5.3 多维数组 .....          | (101) |
| 5.3.1 多维数组的定义 .....     | (102) |
| 5.3.2 多维数组的初始化 .....    | (102) |
| 5.3.3 多维数组的引用 .....     | (104) |
| 习题 .....                | (106) |
| 第6章 指针 .....            | (108) |
| 6.1 指针与指针变量 .....       | (108) |
| 6.1.1 指针与指针变量的概念 .....  | (108) |
| 6.1.2 指针变量的使用 .....     | (109) |
| 6.2 指针的运算 .....         | (113) |
| 6.3 指针与数组 .....         | (116) |
| 6.3.1 指针与数组的关系 .....    | (116) |
| 6.3.2 指向多维数组的指针 .....   | (117) |
| 6.4 字符数组与指针 .....       | (119) |
| 6.5 指针数组 .....          | (120) |
| 6.6 多级指针 .....          | (122) |
| 习题 .....                | (123) |
| 第7章 函数 .....            | (126) |
| 7.1 函数的概念 .....         | (126) |
| 7.1.1 函数的分类和定义 .....    | (127) |
| 7.1.2 函数的调用 .....       | (129) |
| 7.2 函数间的数据传递 .....      | (131) |

|        |             |       |
|--------|-------------|-------|
| 7.2.1  | 形参与实参的结合    | (131) |
| 7.2.2  | 函数返回值       | (133) |
| 7.2.3  | 全局变量        | (137) |
| 7.3    | 函数的参数       | (138) |
| 7.3.1  | 数组作为函数参数    | (138) |
| 7.3.2  | 命令行参数       | (139) |
| 7.4    | 函数与变量       | (141) |
| 7.4.1  | 变量的作用域      | (141) |
| 7.4.2  | 变量的生存期      | (145) |
| 7.5    | 指向函数的指针     | (146) |
| 7.6    | 库函数         | (147) |
| 7.7    | 函数的递归调用     | (148) |
|        | 习题          | (151) |
| 第8章    | 结构、联合与枚举    | (157) |
| 8.1    | 结构体类型       | (157) |
| 8.1.1  | 结构体类型概述     | (157) |
| 8.1.2  | 结构体类型变量     | (159) |
| 8.1.3  | 结构体类型与数组    | (165) |
| 8.1.4  | 结构体类型与指针    | (170) |
| 8.1.5  | 结构体类型与函数    | (172) |
| 8.1.6  | 结构体类型的应用    | (173) |
| 8.2    | 联合          | (180) |
| 8.3    | 枚举          | (183) |
| 8.4    | 用户定义类型      | (185) |
|        | 习题          | (186) |
| 第9章    | 编译预处理与分别编译  | (190) |
| 9.1    | 编译预处理       | (190) |
| 9.1.1  | 宏定义         | (190) |
| 9.1.2  | 文件包含        | (197) |
| 9.1.3  | 条件编译        | (199) |
| 9.2    | 分别编译        | (203) |
|        | 习题          | (205) |
| 第10章   | 文件          | (208) |
| 10.1   | C文件系统概述     | (208) |
| 10.1.1 | 文件的概念       | (208) |
| 10.1.2 | 文件的分类       | (209) |
| 10.1.3 | 文件类型指针      | (211) |
| 10.2   | 标准设备文件的输入输出 | (211) |

|        |                                |       |
|--------|--------------------------------|-------|
| 10.2.1 | 标准设备文件函数 .....                 | (212) |
| 10.2.2 | 标准设备文件的换向 .....                | (215) |
| 10.3   | 一般文件的操作 .....                  | (216) |
| 10.3.1 | 文件的打开和关闭 .....                 | (216) |
| 10.3.2 | 文件的字符输入输出函数 .....              | (217) |
| 10.3.3 | 文件的字符串输入输出函数 .....             | (219) |
| 10.3.4 | 文件的格式化输入输出函数 .....             | (221) |
| 10.3.5 | 文件的数据块输入输出函数 .....             | (223) |
| 10.4   | 文件的定位操作 .....                  | (226) |
| 10.4.1 | 取得文件指针的当前位置函数 ftell( ) .....   | (226) |
| 10.4.2 | 改变文件指针的当前位置函数 fseek( ) .....   | (226) |
| 10.4.3 | 置文件指针于文件开头位置函数 rewind( ) ..... | (228) |
| 10.5   | 文件的错误检测 .....                  | (228) |
| 10.5.1 | 文件读写错误检测函数 ferror( ) .....     | (228) |
| 10.5.2 | 清除文件错误标志函数 clearerr( ) .....   | (229) |
| 10.5.3 | 文件结束检测函数 feof( ) .....         | (229) |
| 习题     | .....                          | (229) |
| 附录一    | ASCII 代码表 .....                | (232) |
| 附录二    | C 语言常用系统函数按字母顺序索引汇总表 .....     | (233) |
| 附录三    | C 语言编译错误信息 .....               | (237) |

# 第 1 章

## 概 述

6342356

C 语言是一种结构化语言,它具有结构紧凑、高效、易于移植、模块化等特点,因此广泛应用于系统程序设计、科学计算、控制工程、图形处理、数据分析、文字编辑以及人工智能等领域。本章将对 C 语言作一个简单的介绍,让读者对 C 语言有一个大概的认识。请读者务必仔细阅读这一章,从而对 C 语言有一个整体的把握,在以后的学习中,这种认识可以帮助读者顺利地学习 C 语言。另外,在学习完所有知识以后,可以将所学的东西与本章的介绍相对应,这样可以帮助你纠正一些错误的认识。

### 1.1 C 语言的起源和发展

#### 1.1.1 C 语言的起源

在计算机发展早期,程序(包括计算机操作系统等软件)都是由汇编语言编写的。汇编语言比较接近硬件,可以对硬件直接进行控制,执行速度比较快。汇编语言的优点是比较明显的,但它的缺点也同样明显。它是一种低级语言,编写起来比较复杂,调试也难,可读性和移植性都很差,这极大地限制了它的发展。

随着计算机技术的发展,出现了高级语言。这些语言在人机界面上对汇编语言进行了改进,以一种近似于人工语言的语言来编写程序。20 世纪 60 年代初期,ALGOL 60、FORTRAN、COBOL 语言甚为流行,用它们编制的程序开创了最初的软件产业。但是这些语言在对汇编语言的缺点进行改正的同时,也丢弃了它的优点,对硬件的控制变得复杂。同时,由于发展的不完善,这些语言数据结构简单,编程结构复杂,难以掌握,也在一定程度上限制了程序设计的发展。

1970 年,美国贝尔(Bell)实验室的肯·汤普逊(Ken Thompson)对一种比较接近硬件的 BCPL 语言(由 ALGOL 语言演化而来)进行简化,从而衍生出了一种新的语言,他将之称为 B 语言。与其他高级语言相比,B 语言更接近于硬件。B 语言同样具有过于简单、功能不强和生成的解释代码运行速度慢的弱点。

针对这一情况,1972年,贝尔实验室的布朗·W·卡尼汉(Brian W·Kernighan)和丹尼斯·M·利奇(Dennis M·Ritchie)两人(简称 K&R)对 B 语言进行了充实和完善。在保持 B 语言接近硬件特点的同时,提供了丰富的数据类型,并加进了许多在当时来说较为先进的程序设计思想。或许是因为 BCPL 的第二个字母是“C”,或许是因为紧跟在“B”之后的字母是“C”,K&R 将这种新型的语言命名为 C 语言。

### 1.1.2 C 语言的标准化

由于 C 语言既具有汇编语言接近硬件的特性,又具有高级语言人性化的特点,所以 C 语言一问世,就受到了普遍的欢迎,学习和使用 C 语言的人越来越多。随着应用的不断深入和扩展,出现了很多版本的 C 语言,这给 C 语言的进一步推广应用带来了阻碍,因为一个版本的 C 程序不能在另一版本的 C 语言环境中使用。为了解决各种版本 C 程序之间的交流问题,制订一个统一的 C 语言标准的要求日趋迫切。1983 年,美国国家标准协会 ANSI 在继承和发展的基础上制定了新的 C 语言标准,称为 ANSI C 标准。本书介绍的 Turbo C 符合 ANSI C 标准。

## 1.2 C 语言的特点

从特性上来说,C 语言属于一种中级语言,一方面具有高级语言人性化的特点,另一方面又具有低级语言直接面向硬件的特点,在学习时应注意这两种特性的结合。

具体来讲,C 语言具有以下几点显著特征:

### 1. C 语言是一种结构化程序设计语言

结构化程序设计是 20 世纪 70 年代以来盛行的一种程序设计方法,它的具体思路是:将一个程序分成若干模块,各个模块应具有独立性、抽象性和可维护性;程序的逻辑结构限定必须由顺序、选择和循环三种基本结构组成,限制 GOTO 语句的使用。C 语言中使用函数串结构,适应结构化程序设计的要求;提供了多种程序控制语句,可以实现多种程序流程。

虽然 C 语言本质是一种结构化程序设计语言,但为了某些场合下的方便,C 语言还是提供了一条非结构化语句——GOTO 语句,在学习时要加以注意。

### 2. C 语言提供了丰富的数据类型

数据类型的丰富程度直接影响了高级语言功能的强弱。丰富的数据类型将使程序结构简单,数据描述结构化,从而适应更广泛的应用领域。

C 语言的基本数据类型有字符以及多种不同长度的整数、浮点数数据类型。以整型为例,又有无符号整型、短整型、普通整型和长整型之分。

C 语言的基本数据类型还有一个特点,它们是一种非强类型,意思是指数据类型及它的值在运算时不一定要求严格一致。比如说,字符型的变量可以作为整型变量直接进行运算;不存在逻辑型数值,以整数数值是否为零作为逻辑型数值的真和假,等等。

在基本数据类型的基础上,可以构造出数组、结构和联合等导出数据类型。这些导出数据类型可以用于描述一些较为复杂的数据。

C 语言还提供了一种特别的数据类型——指针型,并提供了一些相关操作。使用指针类型,用户可以直接作面向计算机硬件性能的描述,这一功能在某些场合是必需的。

指针可以说是 C 语言中最富特色的一个亮点,它将指针、地址计算、数组和结构类型的存取处理功能统一在一起,提供了极大的灵活性。利用指针,还可以实现像汇编语言一样进行二进制运算和寄存器存取的功能。指针直接体现了 C 语言面向硬件的特点。

这里需要特别指出的是,指针是提高 C 语言描述能力的重要手段之一,这是由于其提供了极大的灵活性,但优点有时即是缺点,正是由于其极大的灵活性,给读者的学习造成了极大的麻烦。从某种意义上来说,学好指针就意味着学好了 C 语言,因此,读者需要对指针的学习引起特别的重视。

### 3.C 语言提供了编译预处理功能

C 语言提供了三种预处理功能:宏定义、文件包含和条件编译。

宏定义命令的作用是给一些常用的对象重新命名,以便引用。进行宏定义以后,在程序中就可以用宏名来引用这些对象,在编译时,这些宏名会自动替换为原来的内容,这个替换的过程称为宏展开。宏定义可以有两种:无参数的和有参数的。

文件包含命令可以将另一个 C 语言源程序的全部内容包含进来,通常可以把经常用到的、带公用性的一些函数或符号等集合在一起形成一个源文件,然后用此命令将这个源文件包含进来,这样可以避免在每个新程序中都要重新键入这些内容并保证代码的一致性。

为了解决程序的可移植性问题,C 语言提供了条件编译命令,它能使同一源程序在不同的编译环境下生成不同的目标代码文件。

### 4.C 语言可以对变量进行存储类型的定义和说明

存储类型指明了变量存储在计算机中的什么位置。在 C 语言中,可以借助存储类型符来指明变量是静态变量或非静态变量、局部变量或是非局部变量、是否是寄存器变量等。不同存储类型的变量具有不同的特性,这种特性的不同可以帮助程序设计者进行一些特别的控制。

特别是在 C 语言中,还可以对函数进行构造类型和存储类型的说明,它指出了函数返回值的构造类型和存储类型。

### 5.C 语言提供了丰富的运算功能

运算符的种类和运算的多样性决定了一种语言处理数据的能力,C 语言提供了 34 种运算符和 15 个等级的运算优先级顺序,在此基础上可以构造出各种各样的表达式,能够很好地满足各种数据处理的要求。

### 6. 简洁、高效的程序编制

C 语言的语法、词法都比较简洁,这为输入程序和阅读程序带来了方便。举个简单的例子,在另一种高级语言——PASCAL 中,使用 BEGIN 和 END 作为复合语句的界定符,而在 C

beijing end

语言中,复合语句的界定符使用的是“{”和“}”,这使 C 语言程序的书写更为简洁。

C 语言提供了十大类语句,使用这些语句,可以设计出各种需要的功能模块和复合语句的界定符块。

## 7. 丰富的库函数

C 语言本身提供了丰富的库函数,这些库函数实现了一些常用的功能,比如求字符串的长度、判断字符的大小写等,用户可以直接使用这些功能模块而不需自己编制。

一些功能相近的库函数被放在一起形成了库文件,使用库函数,只需使用文件包含命令将库文件包含进来即可。这种方式也提供了一种思路,可以将自己编制的一些实用程序模块保存在一个文件中,下次需要这些模块的时候,可以像使用库函数一样,将这个文件包含在程序中即可。通过这种方法,用户实际上就是拥有了自己的软件包。

与其他高级语言相比较而言,C 语言是一种比较灵活的语言,在学习时要注意这一特性。在刚开始的时候,读者可能会为它的这一特性烦恼不已,但当掌握了它以后,就会喜欢上它的。

# 1.3 C 语言程序设计概貌

## 1.3.1 程序设计的本质

作为一种工具,最理想的情况就是人能够和计算机直接交流,也就是要求计算机能够直接理解人类的语言。但这一点实现起来太难了,计算机说到底只是一种机器,它无法做到这一点。人工智能、自然语言的理解等领域的发展可能会逐步地改善这种情况,但那只是将来的情况,目前的计算机还是太“笨”了。

所以,要和计算机打交道,只能是使用一种特别的方式来让计算机了解用户的意图,即使用计算机语言。这种情况很好理解,与英国人打交道要使用英语,与德国人打交道要使用德语,与计算机打交道就要使用计算机语言。如同人类语言分为很多种类一样,计算机语言也有机器语言、汇编语言和高级语言之分。这些语言的功能是一致的,即都是一种和计算机打交道的工具,区别仅仅在于使用的语法、词法不同而已。高级语言具有和人类语言更相近的语法习惯(人性化),易于学习,易于理解,因此得到了广泛的应用。这也是今天多数人在学习高级语言,而不是汇编语言或是机器语言的原因。

需要指出的是,计算机能够直接处理的语言只有机器语言,汇编语言和高级语言必须先经过一定的处理,转换成机器语言后,才能被计算机所接受并进行处理,因此从执行的效率来讲,汇编语言和高级语言都不如机器语言。

所以,程序设计的所有工作简单地讲实际上就只有两步:了解程序设计任务的具体要求,并使用人类语言进行描述;将人类自然语言的描述转换成计算机语言的描述,并将这种描述输入到计算机中。这就是程序设计的本质。这如同请一个外国人为你进行工作一样,首先你自己要知道工作的具体内容,然后你要将这些内容翻译成对方的语言告诉他,他才可

以为你进行工作。

日常生活中所讲的程序设计往往指的是第二步,即将人类自然语言的描述转换成计算机语言的描述,并将这种描述输入到计算机中。请注意,这并不是程序设计的全部。在某种意义上来说,第一步比第二步更为重要。第二步的工作更像是一个“翻译”的工作,只要你在这种语言上花费的时间够多,你就可以担当这个角色。但是“翻译”这个过程中所使用的工具并不是最重要的,重要的是“翻译”的内容,也就是你第一步所要确定的东西。一个合格的程序设计者绝对不仅仅只是一个代码编写者,他更应该关注的是如何描述出问题的全部,并将之细化成容易编制代码的部分,以及如何高效率地实现代码的编制。前一个问题称为软件设计,计算机教育中有一门课程叫做软件工程,其中讲述的就是这方面的内容。后一个问题属于算法研究的部分,在数据结构这门课程中将会介绍到各种各样的算法。

也许有人会很奇怪,介绍 C 语言程序设计,怎么扯到软件工程和数据结构上去了,这是因为语言并不是程序设计的全部,正如英文字母、单词、词组并不是英文的全部一样,需要让大家有一个正确的认识。在这本书里,并不涉及到软件工程和数据结构这两门课程的高深内容,但是一些最基本的东西还是需要告诉大家。

### 1.3.2 C 语言程序设计的本质

曾经提到过,C 语言具有人性化的特点。现已知道,所谓语言的人性化,实际上指的就是如何用接近人类语言的方式来描述某个问题。这种接近包括两个方面:思考问题方式上的接近和语言描述上的接近。下面以一个具体的实例来看看在 C 语言中,如何用一种接近人类语言的方式来描述问题。在这个过程中,请大家注意在 C 语言这种特定语言中是如何体现程序设计的本质的。

假设有一人,称之为甲,现在他需要处理一大批几何图形的求面积的任务,这些图形包括长方形、三角形、圆形。为了提高工作的效率,为这个人提供了三个助手,助手乙专门计算长方形的面积,助手丙专门计算三角形的面积,助手丁专门计算圆形的面积。可以采取的做法是:①由甲读取数据,如长方形的长和宽,三角形的底和高,圆形的半径等,并根据数据类型不同分发给助手乙、丙、丁三人,②乙、丙、丁三人专门负责计算,并将结果返回给甲。这个过程可用图 1-1 表示。

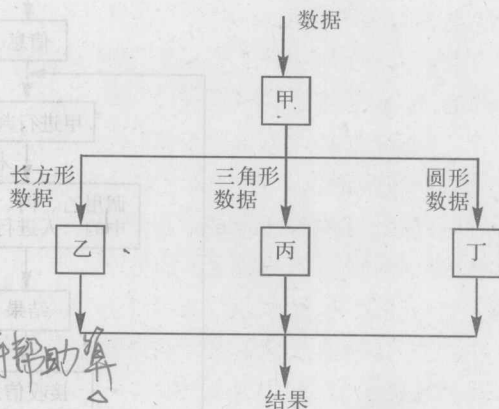


图 1-1 自然语言的工作过程描述

这是日常生活中的处理流程,在 C 语言中,大概的思路也是这样的,它将模仿这一流程,并使用计算机语言将这一流程描述下来。

①它首先建立一个部件来模仿甲做的工作,在 C 语言中,将这种部件称为函数,也就是说,建立一个函数模仿甲,如同现实世界中每样物品都有一个名称一样,为这个函数取一个名字,叫 `main()`;然后再建立三个函数,分别模仿乙、丙、丁三人所做的工作,分别将这三个函数命

名为 `fun1()`, `fun2()`, `fun3()`。在这里,请读者不要太过于关注函数是如何实现功能上的模仿的,你现在只需建立这个印象:这四个函数能够分别完成甲、乙、丙、丁的工作就可以了,随着讲解的深入,你会慢慢知道函数是如何具体实现的。于是,在 C 语言中,整个事件的处理过程可以如图 1-2 所示。

将图 1-1 和图 1-2 进行对比,会发现两者结构上是相似的,仅仅是名称上有所区别,在这一点上体现了 C 语言在思考问题方式上和人类思维的接近。

图 1-2 中所示的并不是最终的 C 语言程序,还必须使用 C 语言的语法写出具体的源程序才算完成设计任务。C 语言的语法和自然语言中的英语比较接近,以上述例子中的甲为例,看看 C 语言的源程序是如何描述问题的。

先用自然语言描述一下甲的工作,流程可能是这样的:甲先接收一定的信息,判断需要计算的是长方形还是三角形,或者是圆形。在这里,可以使用不同的数字(1,2,3)代表不同的几何图形(长方形、三角形、圆形),即读到数据 1,就认为处理的是长方形的数据,2 则是三角形,3 则是圆形,并使用数字 0 表示全部工作结束;然后接收相应的数据(长方形的长和宽、三角形的底和高、圆形的半径),将这些数据分别传给乙、丙、丁;乙、丙、丁三人根据传给他们

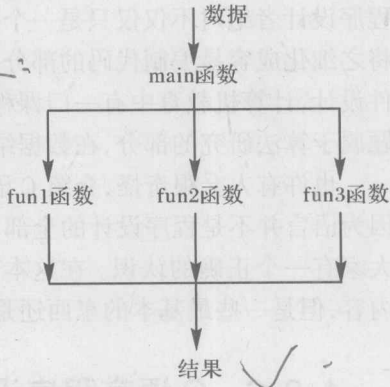


图 1-2 C 语言的工作过程描述

的数据进行计算,得出结果并将结果返回给甲,甲再接受信息进行判断……如此周而复始,直到所有的数据都处理完为止(接收到数据 0)。以上的流程可以用图 1-3 概略表示。

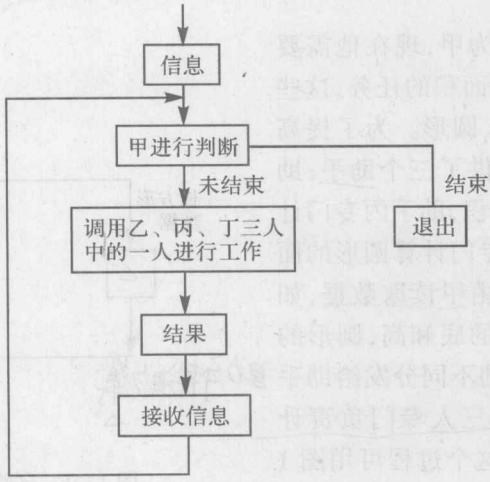


图 1-3 程序流程示意图

以上的过程可以简单地看作是程序设计的第一步:了解程序设计任务的具体要求,并使用人类语言进行描述。需要注意的是,在这里由于仅仅是一个例子,所以这一步并不是完整的,还有乙、丙、丁三人的工作没有进行描述。