



高等教育“十二五”规划教材

# C语言程序设计教程

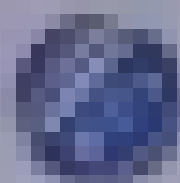
C YUYAN CHENGXU SHEJI JIAOCHENG

主 编 田 静

副主编 张 爽 赵微巍



北京交通大学出版社  
<http://www.bjtu.cn>



清华大学“十二五”规划教材

# C语言程序设计教程

清华大学出版社 清华大学出版社

清华大学出版社  
清华大学出版社

清华大学出版社

高等教育“十二五”规划教材

# C 语言程序设计教程

主 编 田 静  
副主编 张 爽 赵微巍

北京交通大学出版社

· 北京 ·

## 内 容 简 介

本书以突出实践应用能力为出发点,采用流行的“任务驱动+案例+项目实训”的模式,从大量实例入手,由浅入深地对C程序设计内容进行了全面讲述。全书共分为11章,包括C程序设计概述,C程序基础知识,顺序结构程序设计,选择结构程序设计,循环结构程序设计,数组应用,函数应用,指针应用,结构体、共同体和枚举类型,文件,综合应用实例。本书是由一直从事C程序设计教学的一线教师根据实践教学和应用研究体会编写而成。内容通俗易懂,实例非常丰富、典型而全面,目标明确,实用性强,理论适度,深入浅出,注重理论与实践结合,形式新颖,使读者通过实例能够轻松愉快地全面掌握C程序设计的方法和应用。

本书适合作为高等院校计算机专业的教材,也适合广大计算机爱好者及参加培训、考试、考级的人员作为自学和参考用书。

版权所有,侵权必究。

## 图书在版编目(CIP)数据

C语言程序设计教程 / 田静主编. —北京:北京交通大学出版社,2014.7

(高等教育“十二五”规划教材)

ISBN 978-7-5121-1939-0

I. ① C… II. ① 田… III. ① C语言-程序设计-高等学校-教材 IV. ① TP312

中国版本图书馆CIP数据核字(2014)第126516号

责任编辑:田秀青

出版发行:北京交通大学出版社

电话:010-51686414

地 址:北京市海淀区高粱桥斜街44号 邮编:100044

印 刷 者:北京艺堂印刷有限公司

经 销:全国新华书店

开 本:185×260 印张:15.75 字数:393千字

版 次:2014年8月第1版 2014年8月第1次印刷

书 号:ISBN 978-7-5121-1939-0/TP·787

印 数:1~2 000册 定价:35.00元

本书如有质量问题,请向北京交通大学出版社质监组反映。对您的意见和批评,我们表示欢迎和感谢。

投诉电话:010-51686043, 51686008; 传真:010-62225406; E-mail: press@bjtu.edu.cn。

# 前 言

本书以突出实践应用能力为出发点,采用流行的“任务驱动+案例+项目实训”的模式,从大量实例入手,由浅入深地对C程序设计内容进行了全面讲述。全书包括以下内容:C程序设计概述,C程序基础知识,顺序结构程序设计,选择结构程序设计,循环结构程序设计,数组应用,函数应用,指针应用,结构体、共同体和枚举类型,文件,综合应用实例及附录。每章包括“学习目标”、“实例”、“技能训练”、“任务小结”、“习题”、“项目实训”等模块。学习目标部分明确学习任务,给出每一章应该掌握的内容及达到的目的,以便进行针对性学习;应用实例给出分析与设计算法、准确的程序、经常出现的问题和解题技巧,以实例带动知识学习;技能训练进一步针对所学知识强化其方法和要点的掌握;任务小结归纳要点和难点,并总结相关的编程技巧;习题帮助学生对所学内容进行细化和巩固;项目实训给出有针对性、较综合的实践应用题目,配有考核要点和方法,可分阶段对所学知识进行实践与检验。

本书适合作为高等院校计算机专业的教材,也适合广大计算机爱好者及参加培训、考试、考级的人员作为自学和参考用书。本书具有以下特色。

## 1. 案例丰富

本书具有较大的信息量,结合知识点介绍大量典型案例,每个实例有程序解析,做到举一反三,为读者对知识的学习和巩固提供保障,为开发实际应用项目打好基础。

## 2. “任务驱动+案例+项目实训”模式

教材紧密突出计算机专业教学的特点,淡化理论,加强实训。目标明确,针对性强,从实例入手,由浅入深地对C程序设计内容进行详细的讲述。在内容的讲解上,以每章的重点内容为主线,注重案例分析。最后配有项目实训,做到学以致用。

## 3. “渐进式”的讲解思路

采用由浅入深的渐进式设计思路,力求每一章讲解全面,注重程序设计方法的讲解,规范程序设计风格,引导读者理解必备的理论知识,又快速掌握专业的程序设计思想和逻辑思维方法。书中介绍的实例由易到难,对内容进行了提高扩展,丰富的内容和实例可适应不同层次读者学习。从基础学习到灵活运用再到综合应用,学习思路科学。

本书由主编田静和副主编张爽、赵微巍完成大纲讨论及全书内容的编写,唐红杰、杨雅媛参编完成校稿和文字图片的整理工作,全书由田静负责统稿和定稿。本书在编写过程中,参考了相关的文献资料,在此对相关作者表示衷心的感谢。同时,对本书出版付出努力的北京交通大学出版社的有关同志表示诚挚的感谢。

由于编者水平有限,书中难免存在缺点和错误,敬请有关专家和广大读者批评指正。

编 者

2013年5月

# 目 录

|                          |      |
|--------------------------|------|
| 第1章 C程序设计概述 .....        | (1)  |
| 1.1 C语言概述 .....          | (1)  |
| 1.1.1 C语言的由来 .....       | (1)  |
| 1.1.2 C语言的特点 .....       | (1)  |
| 1.2 C程序的结构 .....         | (2)  |
| 1.2.1 C程序的结构分析 .....     | (2)  |
| 1.2.2 C程序的格式要求 .....     | (3)  |
| 1.2.3 C程序的结构特点 .....     | (4)  |
| 1.3 C程序的调试与运行 .....      | (4)  |
| 1.3.1 C程序的实现过程 .....     | (4)  |
| 1.3.2 Turbo C的操作过程 ..... | (4)  |
| 本章小结 .....               | (5)  |
| 习题 .....                 | (6)  |
| 项目实训 .....               | (6)  |
| 第2章 C程序基础知识 .....        | (7)  |
| 2.1 标识符、常量和变量 .....      | (7)  |
| 2.1.1 标识符 .....          | (7)  |
| 2.1.2 常量 .....           | (8)  |
| 2.1.3 变量 .....           | (8)  |
| 2.2 数据类型 .....           | (9)  |
| 2.2.1 整型数据 .....         | (9)  |
| 2.2.2 实型数据 .....         | (11) |
| 2.2.3 字符型数据 .....        | (12) |
| 2.3 运算符和表达式 .....        | (13) |
| 2.3.1 算术运算符及其表达式 .....   | (14) |
| 2.3.2 赋值运算符及其表达式 .....   | (15) |
| 2.3.3 自加和自减运算 .....      | (15) |
| 2.3.4 关系运算符及其表达式 .....   | (16) |
| 2.3.5 逻辑运算符及其表达式 .....   | (17) |
| 2.3.6 逗号运算符及其表达式 .....   | (18) |
| 技能训练 .....               | (18) |
| 本章小结 .....               | (19) |
| 习题 .....                 | (19) |
| 项目实训 .....               | (21) |

|                                 |      |
|---------------------------------|------|
| 第3章 顺序结构程序设计 .....              | (22) |
| 3.1 结构化程序设计的基础 .....            | (22) |
| 3.1.1 算法与数据结构 .....             | (23) |
| 3.1.2 结构化程序设计方法 .....           | (24) |
| 3.2 顺序结构程序设计简介 .....            | (25) |
| 3.3 格式输出函数 printf() .....       | (26) |
| 3.3.1 printf()函数的使用 .....       | (26) |
| 3.3.2 printf()函数的附加格式字符 .....   | (28) |
| 3.4 格式输入函数 scanf() .....        | (30) |
| 3.5 字符输出/输入函数 .....             | (32) |
| 3.5.1 字符输出函数 putchar() .....    | (32) |
| 3.5.2 字符输入函数 getchar() .....    | (33) |
| 技能训练 .....                      | (34) |
| 本章小结 .....                      | (35) |
| 习题 .....                        | (35) |
| 项目实训 .....                      | (37) |
| 第4章 选择结构程序设计 .....              | (38) |
| 4.1 选择结构程序设计简介 .....            | (38) |
| 4.2 if 语句 .....                 | (39) |
| 4.2.1 单分支 if 语句 .....           | (40) |
| 4.2.2 双分支 if 语句 .....           | (40) |
| 4.2.3 嵌套 if 语句 .....            | (41) |
| 4.2.4 多分支 if...else if 语句 ..... | (44) |
| 4.3 switch 语句 .....             | (45) |
| 4.4 条件运算符和条件表达式 .....           | (49) |
| 技能训练 .....                      | (50) |
| 本章小结 .....                      | (52) |
| 习题 .....                        | (53) |
| 项目实训 .....                      | (57) |
| 第5章 循环结构程序设计 .....              | (58) |
| 5.1 循环结构程序设计简介 .....            | (58) |
| 5.2 while 循环 .....              | (60) |
| 5.3 do while 循环 .....           | (61) |
| 5.4 for 循环 .....                | (63) |
| 5.5 空循环 .....                   | (65) |
| 5.6 几种循环的比较 .....               | (66) |
| 5.7 嵌套循环 .....                  | (66) |
| 5.8 循环的跳离 .....                 | (68) |
| 5.8.1 break 语句 .....            | (69) |

|                            |              |
|----------------------------|--------------|
| 5.8.2 continue 语句 .....    | (69)         |
| 技能训练 .....                 | (70)         |
| 本章小结 .....                 | (72)         |
| 习题 .....                   | (73)         |
| 项目实训 .....                 | (77)         |
| <b>第6章 数组应用</b> .....      | <b>(79)</b>  |
| 6.1 数组简介 .....             | (79)         |
| 6.2 一维数组 .....             | (81)         |
| 6.2.1 一维数组的定义 .....        | (81)         |
| 6.2.2 一维数组的引用 .....        | (82)         |
| 6.2.3 一维数组的初始化 .....       | (84)         |
| 6.2.4 一维数组程序设计举例 .....     | (85)         |
| 6.3 二维数组 .....             | (87)         |
| 6.3.1 二维数组的定义 .....        | (87)         |
| 6.3.2 二维数组的引用 .....        | (88)         |
| 6.3.3 二维数组的初始化 .....       | (90)         |
| 6.3.4 二维数组程序设计举例 .....     | (91)         |
| 6.4 多维数组 .....             | (93)         |
| 6.5 字符数组 .....             | (94)         |
| 6.5.1 字符串与字符数组的区别 .....    | (94)         |
| 6.5.2 字符数组的定义、引用及初始化 ..... | (95)         |
| 6.5.3 字符数组的输入与输出 .....     | (96)         |
| 6.5.4 字符串输入/输出函数 .....     | (97)         |
| 6.5.5 字符串处理函数 .....        | (98)         |
| 技能训练 .....                 | (100)        |
| 本章小结 .....                 | (102)        |
| 习题 .....                   | (102)        |
| 项目实训 .....                 | (106)        |
| <b>第7章 函数应用</b> .....      | <b>(108)</b> |
| 7.1 C语言函数概述 .....          | (108)        |
| 7.1.1 库函数和用户自定义函数 .....    | (108)        |
| 7.1.2 有返回值函数和无返回值函数 .....  | (109)        |
| 7.1.3 无参函数和有参函数 .....      | (109)        |
| 7.1.4 字符类型 .....           | (109)        |
| 7.2 函数的定义 .....            | (110)        |
| 7.2.1 无参函数的定义 .....        | (110)        |
| 7.2.2 有参函数的定义 .....        | (111)        |
| 7.3 函数的参数及返回值 .....        | (112)        |
| 7.3.1 形式参数和实际参数 .....      | (112)        |



|       |                     |       |
|-------|---------------------|-------|
| 7.3.2 | 函数的返回值 .....        | (114) |
| 7.4   | 函数的一般调用 .....       | (115) |
| 7.4.1 | 函数调用的一般形式 .....     | (115) |
| 7.4.2 | 函数调用的方式 .....       | (115) |
| 7.4.3 | 被调用函数的声明和函数原型 ..... | (116) |
| 7.5   | 函数的嵌套调用和递归调用 .....  | (117) |
| 7.5.1 | 嵌套调用 .....          | (117) |
| 7.5.2 | 递归调用 .....          | (118) |
| 7.6   | 数组作为函数参数 .....      | (121) |
| 7.6.1 | 数组元素作为函数实际参数 .....  | (121) |
| 7.6.2 | 数组名作为函数参数 .....     | (122) |
| 7.7   | 变量的作用域 .....        | (127) |
| 7.7.1 | 局部变量 .....          | (127) |
| 7.7.2 | 全局变量 .....          | (129) |
| 7.8   | 变量的存储属性 .....       | (131) |
| 7.8.1 | 变量的存储类别 .....       | (131) |
| 7.8.2 | auto 变量 .....       | (132) |
| 7.8.3 | static 变量 .....     | (132) |
| 7.8.4 | register 变量 .....   | (134) |
| 7.9   | 内部函数和外部函数 .....     | (135) |
|       | 技能训练 .....          | (136) |
|       | 本章小结 .....          | (138) |
|       | 习题 .....            | (138) |
|       | 项目实训 .....          | (141) |
| 第8章   | 指针应用 .....          | (142) |
| 8.1   | 指针概述 .....          | (142) |
| 8.1.1 | 指针的概念 .....         | (142) |
| 8.1.2 | 为什么要用指针 .....       | (143) |
| 8.2   | 指针变量 .....          | (143) |
| 8.2.1 | 指针变量的定义 .....       | (143) |
| 8.2.2 | 指针变量的初始化 .....      | (144) |
| 8.2.3 | 指针变量的引用 .....       | (145) |
| 8.3   | 指针运算 .....          | (147) |
| 8.3.1 | 指针变量的赋值运算 .....     | (147) |
| 8.3.2 | 指针变量与整数的加减运算 .....  | (148) |
| 8.3.3 | 指针变量之间的减法运算 .....   | (149) |
| 8.3.4 | 指针变量的关系运算 .....     | (149) |
| 8.4   | 指针与数组 .....         | (149) |
| 8.4.1 | 指向一维数组元素的指针 .....   | (150) |

|                               |              |
|-------------------------------|--------------|
| 8.4.2 指向二维数组元素的指针 .....       | (151)        |
| 8.5 双重指针 .....                | (153)        |
| 技能训练 .....                    | (155)        |
| 本章小结 .....                    | (156)        |
| 习题 .....                      | (156)        |
| 项目实训 .....                    | (160)        |
| <b>第9章 结构体、共同体和枚举类型 .....</b> | <b>(162)</b> |
| 9.1 结构体类型概述 .....             | (162)        |
| 9.1.1 结构体介绍 .....             | (162)        |
| 9.1.2 结构体类型定义 .....           | (162)        |
| 9.2 结构体类型变量 .....             | (163)        |
| 9.2.1 结构体类型变量的一般定义 .....      | (163)        |
| 9.2.2 结构体类型变量的嵌套定义 .....      | (164)        |
| 9.2.3 结构体的引用 .....            | (165)        |
| 9.3 结构体数组 .....               | (167)        |
| 9.4 结构体指针 .....               | (170)        |
| 9.4.1 指向结构体变量的指针 .....        | (170)        |
| 9.4.2 指向结构体数组的指针 .....        | (171)        |
| 9.5 共同体类型 .....               | (172)        |
| 9.6 枚举类型 .....                | (175)        |
| 技能训练 .....                    | (179)        |
| 本章小结 .....                    | (181)        |
| 习题 .....                      | (181)        |
| 项目实训 .....                    | (185)        |
| <b>第10章 文件 .....</b>          | <b>(186)</b> |
| 10.1 文件概述 .....               | (186)        |
| 10.2 文件的打开与关闭 .....           | (188)        |
| 10.2.1 文件打开函数 .....           | (188)        |
| 10.2.2 文件关闭函数 .....           | (190)        |
| 10.3 字符读写函数 .....             | (191)        |
| 10.3.1 读字符函数 fgetc() .....    | (191)        |
| 10.3.2 写字符函数 fputc() .....    | (192)        |
| 10.4 字符串读写函数 .....            | (194)        |
| 10.5 数据块读写函数 .....            | (196)        |
| 10.6 格式化读写函数 .....            | (199)        |
| 10.7 文件的随机读写 .....            | (201)        |
| 10.8 文件的出错检测和处理 .....         | (203)        |
| 技能训练 .....                    | (205)        |
| 本章小结 .....                    | (206)        |

习题 ..... (206)

项目实训 ..... (209)

第 11 章 综合应用实例 ..... (210)

11.1 程序开发步骤 ..... (210)

11.2 菜单的设计 ..... (212)

11.3 系统功能的实现 ..... (217)

本章小结 ..... (222)

习题 ..... (223)

附录 A Turbo C 操作环境介绍 ..... (224)

附录 B 运算符和结合性 ..... (229)

附录 C C 库函数 ..... (230)

附录 D 字符与 ASCII 代码对照表 ..... (235)

附录 E 常见错误分析 ..... (237)

参考文献 ..... (242)

(531) 1.1.1 变量的存储类别 ..... (531)

(531) 1.1.2 变量的存储类别 ..... (531)

(531) 1.1.3 变量的存储类别 ..... (531)

(531) 1.1.4 变量的存储类别 ..... (531)

(531) 1.1.5 变量的存储类别 ..... (531)

(531) 1.1.6 变量的存储类别 ..... (531)

(531) 1.1.7 变量的存储类别 ..... (531)

(531) 1.1.8 变量的存储类别 ..... (531)

(531) 1.1.9 变量的存储类别 ..... (531)

(531) 1.1.10 变量的存储类别 ..... (531)

(531) 1.1.11 变量的存储类别 ..... (531)

(531) 1.1.12 变量的存储类别 ..... (531)

(531) 1.1.13 变量的存储类别 ..... (531)

(531) 1.1.14 变量的存储类别 ..... (531)

(531) 1.1.15 变量的存储类别 ..... (531)

(531) 1.1.16 变量的存储类别 ..... (531)

(531) 1.1.17 变量的存储类别 ..... (531)

(531) 1.1.18 变量的存储类别 ..... (531)

(531) 1.1.19 变量的存储类别 ..... (531)

(531) 1.1.20 变量的存储类别 ..... (531)

(531) 1.1.21 变量的存储类别 ..... (531)

(531) 1.1.22 变量的存储类别 ..... (531)

(531) 1.1.23 变量的存储类别 ..... (531)

(531) 1.1.24 变量的存储类别 ..... (531)

(531) 1.1.25 变量的存储类别 ..... (531)

(531) 1.1.26 变量的存储类别 ..... (531)

(531) 1.1.27 变量的存储类别 ..... (531)

(531) 1.1.28 变量的存储类别 ..... (531)

(531) 1.1.29 变量的存储类别 ..... (531)

(531) 1.1.30 变量的存储类别 ..... (531)

(531) 1.1.31 变量的存储类别 ..... (531)

(531) 1.1.32 变量的存储类别 ..... (531)

(531) 1.1.33 变量的存储类别 ..... (531)

(531) 1.1.34 变量的存储类别 ..... (531)

(531) 1.1.35 变量的存储类别 ..... (531)

(531) 1.1.36 变量的存储类别 ..... (531)

(531) 1.1.37 变量的存储类别 ..... (531)

(531) 1.1.38 变量的存储类别 ..... (531)

(531) 1.1.39 变量的存储类别 ..... (531)

(531) 1.1.40 变量的存储类别 ..... (531)

(531) 1.1.41 变量的存储类别 ..... (531)

(531) 1.1.42 变量的存储类别 ..... (531)

(531) 1.1.43 变量的存储类别 ..... (531)

(531) 1.1.44 变量的存储类别 ..... (531)

(531) 1.1.45 变量的存储类别 ..... (531)

(531) 1.1.46 变量的存储类别 ..... (531)

(531) 1.1.47 变量的存储类别 ..... (531)

(531) 1.1.48 变量的存储类别 ..... (531)

(531) 1.1.49 变量的存储类别 ..... (531)

(531) 1.1.50 变量的存储类别 ..... (531)

(531) 1.1.51 变量的存储类别 ..... (531)

(531) 1.1.52 变量的存储类别 ..... (531)

(531) 1.1.53 变量的存储类别 ..... (531)

(531) 1.1.54 变量的存储类别 ..... (531)

(531) 1.1.55 变量的存储类别 ..... (531)

(531) 1.1.56 变量的存储类别 ..... (531)

(531) 1.1.57 变量的存储类别 ..... (531)

(531) 1.1.58 变量的存储类别 ..... (531)

(531) 1.1.59 变量的存储类别 ..... (531)

(531) 1.1.60 变量的存储类别 ..... (531)

(531) 1.1.61 变量的存储类别 ..... (531)

(531) 1.1.62 变量的存储类别 ..... (531)

(531) 1.1.63 变量的存储类别 ..... (531)

(531) 1.1.64 变量的存储类别 ..... (531)

(531) 1.1.65 变量的存储类别 ..... (531)

(531) 1.1.66 变量的存储类别 ..... (531)

(531) 1.1.67 变量的存储类别 ..... (531)

(531) 1.1.68 变量的存储类别 ..... (531)

(531) 1.1.69 变量的存储类别 ..... (531)

(531) 1.1.70 变量的存储类别 ..... (531)

(531) 1.1.71 变量的存储类别 ..... (531)

(531) 1.1.72 变量的存储类别 ..... (531)

(531) 1.1.73 变量的存储类别 ..... (531)

(531) 1.1.74 变量的存储类别 ..... (531)

(531) 1.1.75 变量的存储类别 ..... (531)

(531) 1.1.76 变量的存储类别 ..... (531)

(531) 1.1.77 变量的存储类别 ..... (531)

(531) 1.1.78 变量的存储类别 ..... (531)

(531) 1.1.79 变量的存储类别 ..... (531)

(531) 1.1.80 变量的存储类别 ..... (531)

(531) 1.1.81 变量的存储类别 ..... (531)

(531) 1.1.82 变量的存储类别 ..... (531)

(531) 1.1.83 变量的存储类别 ..... (531)

(531) 1.1.84 变量的存储类别 ..... (531)

(531) 1.1.85 变量的存储类别 ..... (531)

(531) 1.1.86 变量的存储类别 ..... (531)

(531) 1.1.87 变量的存储类别 ..... (531)

(531) 1.1.88 变量的存储类别 ..... (531)

(531) 1.1.89 变量的存储类别 ..... (531)

(531) 1.1.90 变量的存储类别 ..... (531)

(531) 1.1.91 变量的存储类别 ..... (531)

(531) 1.1.92 变量的存储类别 ..... (531)

(531) 1.1.93 变量的存储类别 ..... (531)

(531) 1.1.94 变量的存储类别 ..... (531)

(531) 1.1.95 变量的存储类别 ..... (531)

(531) 1.1.96 变量的存储类别 ..... (531)

(531) 1.1.97 变量的存储类别 ..... (531)

(531) 1.1.98 变量的存储类别 ..... (531)

(531) 1.1.99 变量的存储类别 ..... (531)

(531) 1.1.100 变量的存储类别 ..... (531)

# 第 1 章

## C 程序设计概述



### 学习目标

1. 了解 C 语言的由来和特点。
2. 正确掌握 C 语言程序的结构和书写格式。
3. 正确掌握集成开发环境的基本操作。

C 语言是由贝尔实验室的 Dennis M Ritchie 于 1972 年推出的，是一种通用的程序设计语言。它具有表达能力强、编译目标文件质量高、语言简单灵活、容易移植、容易实现等特点，是学习和掌握更高层语言的开发工具，是 C++/C#、Visual C++ 和 Java 语言程序设计的基础。本章主要介绍 C 语言概述、C 程序的结构、简单 C 程序的编写、C 程序的调试运行等内容。

### 1.1 C 语言概述

#### 1.1.1 C 语言的由来

C 语言诞生于 1972 年，由美国电话电报公司贝尔实验室的 Dennis M Ritchie 设计，并首先在一台 UNIX 操作系统的 DEC PDP-11 计算机上实现的。C 语言的诞生至今已有 40 多年。一种较好的高级语言的出现，往往要经历一个长期的演变过程。C 语言是一种出现比较晚的高级语言，它吸取了早期高级语言的长处，克服了其中的某些不足，形成了自己的风格和特点。C 语言的演变过程如下：ALGOL（1960 年）→CPL（1963 年）→BCPL（1967 年）→B 语言（1970 年）→C 语言（1972 年）。

#### 1.1.2 C 语言的特点

任何一种计算机语言的发展都有其目的。在 C 语言诞生之前，已经有很多程序设计语言产生。例如，BASIC 语言，其主要目的是要让计算机的初学者可以很容易编写程序，其语法近似英文，而且浅显易懂。此外，应科学计算与商业用途的需要，FORTRAN 与 COBOL 语言也应运而生；其他高级语言如 Pascal 等也有其特定的用途。但是这些语言常因发展背景与语言本身的限制而无法兼顾实用与性能。C 语言的诞生恰恰可以弥补这些缺憾。C 语言是一种简洁明了、功能强大、可移植性好的结构化程序设计语言。C 语言具有如下特点：

(1) C 语言是一种结构化语言，它层次清晰，便于按模块化方式组织程序，易于调试和维护。

(2) C 语言的表现能力和处理能力极强，它具有丰富的运算符和数据类型，便于实现各类复杂的数据结构。

(3) C 语言实现了对硬件编程操作，C 语言集高级语言和低级语言的功能于一体，广泛应用于系统软件和应用软件的开发研制中。

(4) C 语言具有可移植性强的特点，因此广泛地被移植到了各种类型的计算机上，从而形成了多种版本的 C 语言。

(5) C 语言允许直接访问物理地址。C 语言中含有位运算和指针运算，能直接对内存地址进行访问操作，可以实现汇编语言的大部分功能，即直接对硬件进行操作。

(6) 生成的目标代码质量高，程序执行效率高。它比一般的高级语言生成的目标代码质量高约 20%。

(7) 语法检查不太严格，程序书写自由度大。比如，数组下标不进行超界检查；整型、字符型、逻辑型可以通用等；可以多个语句占一行，也可以一个语句占一行。

总之，程序员使用 C 语言编写程序会感到限制少、灵活性大、功能性强，可以编写出任何类型的程序。

## 1.2 C 程序的结构

### 1.2.1 C 程序的结构分析

通过下面两个简单的 C 程序介绍，了解 C 程序的结构。

**【例 1-1】** 一个简单的 C 程序。

```
main()                                /*主函数*/
{
    printf("This is a C program.\n");    /*屏幕上输出一行文本信息并换行*/
}
```

程序的运行结果：

This is a C program.

**说明**

(1) 这是一个完整的 C 程序，每个 C 程序都必须有且只有一个 `main()` 函数（主函数）。

(2) 一个函数的函数体由大括号 `{}` 括起来。

(3) C 语言中可以使用注释，注释的格式：`/*注释内容*/`，只是对程序起说明作用，程序执行时注释语句不执行。

(4) `printf()` 是 C 语言的输出函数。

(5) `\n` 是转义字符，表示换行。

(6) 一条语句结束必须使用分号“`;`”。

**【例 1-2】** 求两个数的和。

```
#include "stdio.h"
```

```
main() /*主函数*/
{
    int a,b; /*定义两个整型变量*/
    a=2;
    b=a+1;
    printf("%d",b); /*屏幕上输出b的值*/
}
```

程序的运行结果:

3

说明

(1) printf()是系统函数, 包含在 stdio.h 文件中。由于 printf()函数经常使用, 因此文件包含可以省略。

(2) 使用变量时要先定义再使用。

**【例 1-3】**由 main()函数调用其他函数。

```
#include "stdio.h"
output()
{
    printf("This is a C program.\n");
}
main()
{
    printf("*****\n");
    output();
}
```

程序的运行结果:

\*\*\*\*\*

This is a C program.

说明

(1) 一个程序可以由多个函数组成。

(2) C 程序总是从 main()函数开始执行。

### 1.2.2 C 程序的格式要求

在书写 C 语言程序时每个人的习惯不尽相同, 按照规范书写 C 语言程序不仅方便自己及他人阅读, 而且也会使程序的可维护性大大增强, 所以有必要掌握 C 语言的基本书写格式。格式要求如下。

(1) 程序中包含多条语句, 一般情况下每条语句占用一行。如果多条语句写在同一行中, 只要用分号隔开即可。

(2) 每条语句结束时加分号 “;”, 作为语句结束符。

(3) 同一层次中的语句缩进同样多的字符数, 可用空格键调整各行的起始位置。

(4) 程序通常使用英文小写字母书写, 大写字母通常作为常量或其他用途。

(5) 在程序中可以使用注释信息, 增强程序的可读性。

### 1.2.3 C 程序的结构特点

由上述的程序可以总结出 C 程序结构的主要特点。

(1) 一个完整的 C 程序有且只有一个 `main()` 函数。

(2) 一个 C 程序可以由多个函数组成, 但程序总是从主函数开始执行, 不管 `main()` 函数在整个程序的哪个位置。

(3) 程序中用到的变量要先定义后使用。

(4) C 程序中用 “{}” 表示程序的结构层次范围, 注意 “{}” 必须配对使用。

(5) C 程序书写格式自由, 即一行中可以有多个语句, 一个语句也可以占用多行。

(6) 由 # 开头的行称为宏定义或文件包含, 结束无分号, 是 C 语言中的编译预处理命令, 不是真正的执行语句。

## 1.3 C 程序的调试与运行

### 1.3.1 C 程序的实现过程

C 语言采用的编译方式是将源程序转换成二进制目标代码。编写一个 C 程序到完成运行得到结果一般都需要经过以下几个步骤。

#### 1. 编辑

在开发环境下, 将 C 语言源程序通过键盘输入到计算机中, 将源程序保存到磁盘文件中, 其文件扩展名为 “.c”。

#### 2. 编译

编译是将已编辑好的源程序翻译成二进制的目标代码。在编译时, 还要对源程序进行语法检查, 如果程序有错误, 则显示出错信息, 需要对其进行修改, 然后重新编译直至没有错误, 才算编译成功, 并生成扩展名为 “.obj” 的同名文件。

#### 3. 连接

编译后生成的目标文件不能直接执行, 需要经过连接之后才能生成可执行的代码。连接后得到可执行的文件, 其扩展名为 “.exe”。连接的目的是将目标文件各库函数和其他目标程序连接成可执行程序。

#### 4. 执行

一个经过编译和连接的可执行的目标文件, 只有在操作系统的支持和管理下才能执行它。

### 1.3.2 Turbo C 的操作过程

#### 1. 进入 TC 的编辑环境

TC 界面的上方有一行“主菜单”, 其中包括八个菜单项: File、Edit、Run、Compile、Project、Option、Debug、Break/watch, 分别代表文件、编辑、运行、编译、项目、选项、调试、中断或观察等功能。



Turbo C 主界面中常用的快捷键如下:

(1) **【F10】**: 切换主菜单和编辑方式。

(2) **【F2】**: 保存。

(3) **【F3】**: 调用一个已经存在的源文件。

每个菜单项可用键盘中的 **【←】** 键和 **【→】** 键进行选择, 选中某一菜单并按 **【Enter】** 键后出现其下拉菜单。

利用键盘中的 **【↓】** 键选择下拉菜单, 并按 **【Enter】** 键实现。如利用 **【↓】** 键选择 File 的下拉菜单 New 命令, 并按 **【Enter】** 键, 则新建一个 C 源程序。

#### 2. 编辑源程序

在 Edit 状态下根据需要输入或修改源程序。

#### 3. 编译/连接程序

选中 Compile 菜单, 在此下拉菜单中, 若选择 Compile to OBJ 命令可得到扩展名为 “.obj” 的目标程序; 若选择 Link EXE file 命令可得到扩展名为 “.exe” 的可执行文件。用户也可将编译和连接合为一个步骤, 选择 Make EXE file 命令或按 **【F9】** 键, 即可一次完成编译和连接操作。

如果在编译和连接时出现错误, 需要反复修改, 重复进行编译和连接, 直到不再显示错误为止。

#### 4. 运行程序

选择 Run 菜单, 在此下拉菜单中选择 Run 命令, 并按 **【Enter】** 键后运行一个已编译和连接的 C 程序。

#### 5. 退出 TC

选择 File 菜单, 在此下拉菜单中选择 Quit 命令, 并按 **【Enter】** 键可退出 TC 环境。

Turbo C 中常用的快捷键如下。

(1) **【F10】**: 切换主菜单和编辑方式。

(2) **【F2】**: 保存。

(3) **【F3】**: 调用一个已经存在的源文件。

(4) **【F6】**: 窗口切换。

(5) **【Ctrl+F9】** 组合键: 运行。

(6) **【Alt+F5】** 组合键: 显示程序的运行结果, 可按任意键返回编辑界面。

(7) **【Ctrl+Break】** 组合键: 中断程序执行。

(8) **【Alt+X】** 组合键: 退出 Turbo C。

## 本章小结

本章主要介绍了 C 语言的由来、特点, 并详细介绍 C 语言程序的格式要求和结构特点及集成开发环境的使用, 编辑、调试和运行一个 C 语言程序的具体步骤。通过本章的学习, 读者能够对 C 语言程序有初步的认识, 可以上机调试简单的 C 语言程序, 初步掌握 C 语言程序的构成和书写格式, 体会到学习 C 语言程序并不是一件很难的事情。



## 习 题

## 一、选择题

1. ( ) 是构成 C 语言程序的基本单位。  
A. 函数                      B. 过程                      C. 子程序                      D. 子例程
2. 一个 C 程序的执行是从 ( )。  
A. 本程序的 main() 函数开始, 到 main() 函数结束  
B. 本程序文件的第一个函数开始, 到本程序文件的最后一个函数结束  
C. 本程序文件的第一个函数开始, 到本程序的 main() 函数结束  
D. 本程序文件从头到尾执行
3. C 语言规定在源程序中, 主函数的位置 ( )。  
A. 必须在开始                      B. 必须在系统调用的库函数的后面  
C. 可以任意                      D. 必须在最后

## 二、简答题

1. C 语言主要有哪些特点?
2. C 语言程序的基本单位是什么?
3. C 语言程序的结构特点是什么?

## 项目实训

## 【实训目的】

了解在 C 语言开发环境上如何编辑、编译、连接和运行一个 C 语言程序, 掌握 C 语言程序运行、调试过程。通过编写简单的 C 语言程序, 初步了解 C 语言程序的书写格式及结构特点。

## 【实训内容】

1. 请编写一个程序, 要求在屏幕上显示两行文字:

I am a student.

I love China.

2. 已知长方形的长、宽分别是 10cm、5cm, 编写程序求长方形面积。