

宝典丛书

200万

C语言

(第2版)

宝典

对结构化程序设计的三种基本结构以及 C 语言中复杂的数据类型给出相匹配的实例分析, 有效提高初学者程序分析能力

讲述 C 语言在底层操作中的应用, 给出简单易懂的对应实例

提供三大案例, 从需求分析开始, 真实体现利用 C 语言解决具体问题的整个过程

刘丕顺 主编

迟 剑 刘艳飞 等编著



电子工业出版社

Publishing House of Electronics Industry

<http://www.phei.com.cn>

宝 典 丛 书

C 语言宝典

(第2版)

刘丕顺 主编

迟 剑 刘艳飞 等编著

電 子 工 業 出 版 社

Publishing House of Electronics Industry

北京 · BEIJING

内 容 简 介

C 语言是一种计算机程序设计语言,既有高级语言的特点,又具有汇编语言的特点。C 语言可以作为系统设计语言来编写工作系统应用程序,也可以作为应用程序设计语言编写不依赖计算机硬件的应用程序,因此它的应用范围十分广泛。对操作系统和系统使用程序以及需要对硬件进行操作的场合,用 C 语言明显优于其他解释型高级语言,有一些大型应用软件也是用 C 语言编写的。

本书共分为 5 部分,详细介绍了 C 语言的基础知识、C 语言的程序分析方法、调试方法、常见问题的编程方法以及初学者易犯的错误。本书循序渐进地讲述了 C 语言基础知识、三种基本控制结构、数组、指针、结构体共用体、文件、底层操作等主要内容,并穿插讲述了一些编译预处理知识和程序设计方法及调试方法等。本书的最后部分,介绍了利用 C 语言编写功能比较完备的程序的设计过程以及源代码,使读者对 C 语言程序设计有一个比较完整的设计思路,同时又可以加强对 C 语言常用的基础知识的理解和程序调试的掌握,还能掌握通用的软件开发过程。

本书既适合没有编程语言经验的初学者,也适合相关专业在校生。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有,侵权必究。

图书在版编目(CIP)数据

C 语言宝典 / 刘丕顺主编;迟剑等编著. —2 版. —北京:电子工业出版社,2011.3

(宝典丛书)

ISBN 978-7-121-12738-0

I. ①C… II. ①刘… ②迟… III. ①C 语言—程序设计 IV. ①TP312

中国版本图书馆 CIP 数据核字(2011)第 001159 号

责任编辑:张月萍

印 刷:北京东光印刷厂

装 订:三河市皇庄路通装订厂

出版发行:电子工业出版社

北京市海淀区万寿路173信箱

邮编:100036

开 本:787×1092 1/16

印张:41 字数:1155千字

印 次:2011年3月第1次印刷

印 数:3000册 定价:79.00元

凡所购买电子工业出版社图书有缺损问题,请向购买书店调换。若书店售缺,请与本社发行部联系,联系及邮购电话:(010)88254888。

质量投诉请发邮件至zltts@phei.com.cn,盗版侵权举报请发邮件至dbqq@phei.com.cn。

服务热线:(010)88258888。

前 言

C 语言是一种计算机程序设计语言,既有高级语言的特点,又具有汇编语言的特点。C 语言可以作为系统设计语言来编写工作系统应用程序,也可以作为应用程序设计语言编写不依赖计算机硬件的应用程序,因此它的应用范围十分广泛。对操作系统和系统实用程序,以及需要对硬件进行操作的场合,用 C 语言明显优于其他解释型高级语言,有一些大型应用软件也是用 C 语言编写的。

但是很多初学者对于学习 C 语言都感到不能很快上手,不能做到自己编写 C 语言的程序。实际上,这些感觉都是由于没有做大量的练习所致。学习 C 语言这种结构化很强的程序设计语言,必须经过大量的练习,多读程序、多写程序,才能有比较好的效果。

笔者长期以来从事计算机课程的基础教学工作,因此比较清楚做程序设计的初学者在学习 C 语言的过程中常见的一些问题和学习过程中的一些困惑。本书的目的是为了让 C 语言的初学者对于 C 语言乃至结构化程序设计有一个系统、全面的认识,同时能为今后继续学习其他编程语言打下良好的基础。本书可以使初学者快速地掌握 C 语言的基础知识,能使读者对 C 语言有一个全面、直观、系统的认识,同时能掌握结构化程序设计的精髓,最终可以完成独立、专业的 C 语言程序设计。读者还能利用书中的大量实例,打下学习其他高级语言的基础。另外,本书在最后的实例篇涉及到了一些软件开发方面的知识,使得今后读者无论使用何种编程语言独立开发、设计程序的时候都能够使用本书中的软件开发通用过程。

本书的特点

本书帮助读者真正掌握 C 语言的基础知识,以便遇到问题时,可以立刻使用 C 语言的思路分析问题;可以单独地分析程序,对 C 语言的程序段和完整的程序有清楚的分析方法;可以利用已学习的知识,对一些具体的问题有一定的编程思路,可以设计简单的程序;可以调试大型的 C 语言程序,对程序的不同分支,设计不同的数据进行测试;了解一些完整的实际问题的处理例子,为计算机后续课程的学习打下一个坚实的基础。

本书对 C 语言的基础知识的介绍比较详细,对于基础知识点配有其在程序中的应用介绍,有效地把枯燥的基础知识转化成程序中的用法,可以加强读者对 C 语言基础知识的记忆。比较详细地介绍了 C 语言选择结构、循环结构常见的程序分析方法,尤其是针对循环结构,讲述了单层循环、带有跳转的单层循环,以及二层循环的程序分析方法,介绍得比较清晰、易懂。对于一些常见问题给出了套路式的解决问题的方法,为初学者学写程序提供了一个练习的途径。对于数组、函数等知识点以详细分类的方法介绍说明,便于读者对不熟悉的知识点进行查找和学习。

本书的特点主要体现在以下几个方面。

- ◆ 本书的编排采用循序渐进的方式,适合初学者逐步掌握 C 语言的基础、技巧和精髓。
- ◆ 本书编排了大量的实例,即使比较少的代码和错误的代码示例也有配套的源代码,为读者进行调试提供了方便。
- ◆ 养成良好的程序调试习惯是语言学习者必须具备的,因此本书加入了调试程序这一章,读

者可以在调试程序中随时翻阅本章,不受基础知识的限制。

- ◆ 在比较重要的知识点或者数据类型的章节后面,都通过一个具体问题的解决来应用这个知识点。书中的大部分实例都给出了编程思路和步骤,同时在源代码后面对程序的运行状态和问题的转化以及对该代码的进一步诠释,可以帮助读者利用这些问题解决方案来解决另外的问题。随书光盘中提供了书中实例的源代码,方便读者使用。
- ◆ 在每一章或者每一个比较重要的知识点后面,都有相关的常见错误解析和改正方法,有助于初学者了解和学习。另外,对于每一章中初学者经常会提出的一些问题,均以问答的形式予以解答。
- ◆ 本书最后的实例渗透了部分软件工程学的思想,旨在给读者介绍通用的软件开发过程,使读者在今后使用其他开发语言工具的时候也能找到正确简便的开发步骤。

本书的内容安排

本书共分为 5 部分,共 23 章,由浅入深,从理论到实践,尤其适合初级读者逐步学习和完善自己的知识结构。

第 1 部分“基础篇”(第 1 章~第 6 章)介绍 C 语言的基本语法和 C 语言主要控制语句的基础知识。本部分包括:C 语言概述、数据类型与常量变量、表达式与运算符、顺序程序设计、选择程序设计、循环程序设计等。力求浅显易懂,从第一次接触 C 语言入手,逐步深入到 C 语言的组成元素和基本的控制结构,让读者一开始就有系统考虑程序设计的思想。

第 2 部分“进阶篇”(第 7 章~第 11 章)介绍 C 语言中比较复杂的知识。本部分包括:数组、函数、作用域程序设计、指针、字符串等几章。本部分的内容是 C 语言的精髓所在,其中包括了一些比较复杂的内容,如指针等。本部分使用大量的实例代码和图表解释,尽可能做到让读者理解程序运行的过程。

第 3 部分“高级篇”(第 12 章~第 15 章)介绍 C 语言中的高级应用以及程序调试的方法。本部分包括:编译预处理、程序调试、结构体与共用体、自定义类型与链表等。本部分内容属于比较高级的应用,有一部分知识已经涉及到了简单数据结构的范畴。本部分中仍然使用较多的实例代码和图表解释,做到让读者见图知义,能够理解 C 语言程序在运行时的内存使用情况。

第 4 部分“底层开发篇”(第 16 章~第 20 章)介绍 C 语言中的底层开发技术。本部分包括位运算、文件操作、屏幕和键盘的控制等。本部分内容属于底层的应用,仍然使用较多的实例代码和图表解释,让读者能够使用 C 语言程序对文件、屏幕和键盘进行操作。在本部分的最后,还介绍了 C++ 语言对 C 语言进行的扩展,让读者能够在未来学习类似 C++ 等面向对象的设计语言时有所了解,为以后的学习做知识储备。

第 5 部分“应用实例篇”(第 21 章~第 23 章)利用几个实例介绍 C 语言解决实际问题时的应用。本部分包括 3 个实例程序,每个实例程序均遵循“问题提出”→“问题分析”→“模块划分”→“模块内部设计”→“编程实现”→“测试”6 个环节,力求让读者能够从实例中学会如何从一个具体问题出发直到设计出可以解决实际问题的软件的过程。

本书所涉及的代码,请访问博文视点公司网站(www.broadview.com.cn)下载。

本书由刘丕顺主编,迟剑、刘艳飞编写,其他参与编写的人员有张金霞、曾光、张双、朱照华、黄永湛、刘新、孟祥嵩、李勇、张贺军、关涛、王岩、李晓白、魏星等。在此一并表示感谢!

编 者

目 录

第 1 部分 基础篇	1
第 1 章 认识 C 语言	2
1.1 计算机语言及程序的发展	2
1.1.1 计算机语言的历史	2
1.1.2 C 语言的特点	3
1.2 从 Hello 程序认识 C 程序	4
1.3 程序的实例解析与升华	4
1.3.1 简单例题的引申	5
1.3.2 C 语言程序的组成	6
1.3.3 常见的错误解析与解决办法	7
1.4 C 语言基本组成	9
1.4.1 常见的语法规则	9
1.4.2 代码编写规范	10
1.5 C 语言程序的调试过程	11
1.5.1 Turbo C 2.0 集成开发环境介绍及使用	11
1.5.2 Turbo C 2.0 运行程序的步骤	14
1.5.3 程序编译中常见问题	15
1.6 本章的问与答	16
1.6.1 编译和运行有什么不同	16
1.6.2 如何评价一个程序的好坏	16
1.7 小结	17
第 2 章 数据类型、常量和变量	18
2.1 数据及数据类型	18
2.1.1 C 语言常见的数据类型介绍	18
2.1.2 常量	19
2.1.3 变量	20
2.2 常量及常量的应用	21
2.2.1 整型常量	21
2.2.2 实型常量	23
2.2.3 字符常量	24
2.2.4 字符串常量	26
2.2.5 常量使用时的错误分析	26
2.3 变量及变量的应用	28
2.3.1 整型变量	28
2.3.2 实型变量	30
2.3.3 字符型变量	32
2.3.4 变量的使用规范	33
2.3.5 变量常见错误分析方法	35
2.4 本章的问与答	36

2.4.1	编写程序的风格很重要吗	36
2.4.2	数据结构与数据类型是一回事吗	36
2.4.3	main 是关键字吗	36
2.4.4	说明一个变量和定义一个变量有什么区别	36
2.4.5	用 const 说明常量有什么好处	37
2.5	小结	37
第 3 章	运算符与表达式	38
3.1	C 语言中的运算符及表达式	38
3.1.1	运算符及其学习要点	38
3.1.2	表达式及其求值顺序	40
3.2	运算中的数据类型转换	41
3.2.1	强制类型转换	41
3.2.2	隐含类型转换	42
3.3	算术运算符与算术表达式	43
3.3.1	基本算术运算符	43
3.3.2	特殊的算术运算符——自加和自减	44
3.3.3	常见错误分析与解决方法	46
3.4	赋值运算符和赋值表达式	48
3.4.1	简单的赋值运算符	48
3.4.2	复合赋值运算符	49
3.4.3	赋值表达式的运算	50
3.4.4	常见错误分析与解决方法	51
3.5	逗号运算符和逗号表达式	52
3.6	关系运算符和关系表达式	53
3.6.1	C 语言中的真值和假值	53
3.6.2	关系运算符	53
3.6.3	关系表达式	54
3.6.4	常见错误分析	55
3.7	逻辑运算符和逻辑表达式	56
3.7.1	逻辑运算符	56
3.7.2	逻辑表达式	56
3.7.3	常见错误分析	59
3.8	条件运算符和条件表达式	59
3.9	sizeof 运算符	60
3.10	取地址运算符	61
3.11	几种运算符混合运用与常见的问题分析	62
3.12	本章的问与答	63
3.12.1	什么是左值 (lvalue), 什么是右值 (rvalue)	63
3.12.2	运算符的优先级总能保证“自左至右”或“自右至左”的顺序吗	63
3.12.3	表达式和语句有区别吗	64
3.12.4	什么时候应该使用类型强制转换	64
3.12.5	对不同类型的变量进行算术运算会有问题吗	64
3.13	小结	64
第 4 章	顺序程序设计	66
4.1	C 程序基本结构	66
4.2	C 语句概述	67

4.2.1	表达式语句	67
4.2.2	流程控制语句	69
4.2.3	跳转语句	69
4.2.4	复合语句	69
4.2.5	空语句	70
4.3	库函数的调用与输入输出的关系	71
4.4	格式化输出	72
4.4.1	printf 函数的一般格式	72
4.4.2	printf 函数中的格式控制说明	73
4.4.3	整型数据的输出	74
4.4.4	字符型数据的输出	75
4.4.5	实型数据的输出	76
4.4.6	字符串的输出	77
4.4.7	格式化输出的使用实例分析与常见错误	78
4.5	字符输出	79
4.6	格式化输入	80
4.6.1	scanf 函数的一般格式	80
4.6.2	scanf 函数中的格式控制说明	81
4.6.3	格式化输入的使用实例分析与常见的错误	82
4.7	字符输入	85
4.7.1	getch()和 getche()函数	85
4.7.2	getchar 函数	86
4.8	变量如何得到正确的输入数据	87
4.9	顺序结构程序典型例题分析与升华	87
4.10	本章的问与答	91
4.10.1	为什么有时不应该用 scanf()来接收数据	91
4.10.2	怎样用 0 补齐一个数字	92
4.10.3	怎样判断一个字符是否是字母或数字	92
4.11	小结	93
第 5 章	选择程序设计	94
5.1	典型的选择问题的程序设计思路	94
5.2	用 if 语句设计选择程序	94
5.2.1	if 语句的形式: 单条件单分支的 if 语句	95
5.2.2	if 语句的形式: 单条件双分支的 if 语句	96
5.3	if 语句的嵌套	99
5.3.1	单条件单分支 if 语句嵌套单条件单分支的 if 语句	100
5.3.2	单条件单分支 if 语句嵌套单条件双分支的 if 语句	100
5.3.3	单条件双分支 if 语句的嵌套	101
5.3.4	嵌套时 if 与 else 的配对规则	105
5.4	if 语句的使用实例与升华	108
5.5	if 语句的常见错误与解决方法	112
5.6	用 switch 分支设计选择程序	113
5.6.1	switch 分支介绍	114
5.6.2	switch 分支处理的程序特点	117
5.6.3	switch 分支的使用实例	118
5.6.4	switch 分支的常见错误与解决方法	119
5.7	选择型问题实例的剖析与升华	121

5.8 本章常见问题的问与答	123
5.8.1 选择程序中如何加入注释	123
5.8.2 switch 语句的最后一个分支可以不要 break 语句吗	124
5.8.3 共有几种退出 switch 分支的方法	124
5.8.4 什么时候用一条 switch 语句比用多条 if 语句更好	124
5.9 小结	124
第6章 循环结构程序设计	125
6.1 典型的循环问题程序设计思路	125
6.2 while 语句处理的循环	126
6.2.1 while 语句介绍	126
6.2.2 while 语句使用要求	128
6.2.3 while 语句的使用实例	130
6.2.4 常见 while 语句典型程序分析	132
6.2.5 while 语句的常见错误与解决方法	135
6.3 do...while 循环	136
6.3.1 do...while 语句介绍	136
6.3.2 do...while 语句与 while 语句的联系与区别	138
6.3.3 do...while 语句的使用实例	139
6.3.4 常见 do...while 语句的典型程序分析	140
6.3.5 do...while 语句的常见错误与解决方法	143
6.4 for 循环	144
6.4.1 for 语句介绍	144
6.4.2 for 语句的特点	146
6.4.3 for 语句的使用实例与升华	147
6.4.4 常见 for 语句的典型程序分析	149
6.4.5 for 语句的常见错误与解决方法	153
6.5 流程控制语句	154
6.5.1 break 语句	154
6.5.2 continue 语句	157
6.5.3 break 与 continue 常见的使用错误	159
6.5.4 goto 语句及其使用	160
6.6 循环程序的分析方法	161
6.6.1 while 循环分析	162
6.6.2 do...while 循环分析	162
6.6.3 for 循环分析	163
6.6.4 带 break 与 continue 语句的循环程序分析	164
6.7 循环程序的例题	166
6.8 多重循环——循环的嵌套	168
6.8.1 循环嵌套的含义	169
6.8.2 循环嵌套实例与升华	169
6.8.3 循环嵌套的应用	171
6.8.4 循环嵌套程序的易错点与程序分析	175
6.9 循环实例分析	177
6.10 本章常见问题的问与答	179
6.10.1 如何为循环程序添加注释	179
6.10.2 循环结构应该如何学习	180
6.10.3 怎样才能知道循环是否提前结束	180

6.10.4	空循环和无穷循环有什么区别	180
6.11	小结	181
第 2 部分	进阶篇	183
第 7 章	数组	184
7.1	认识数组	184
7.2	一维数组	185
7.2.1	一维数组的定义	185
7.2.2	一维数组的引用	186
7.2.3	一维数组的初始化	188
7.3	一维数组的使用	190
7.3.1	一维数组的使用实例与升华	190
7.3.2	一维数组使用中的常见错误与解决方法	192
7.3.3	常见一维数组典型程序分析	193
7.4	二维数组	198
7.4.1	二维数组的定义	198
7.4.2	二维数组的引用	199
7.4.3	二维数组的初始化	201
7.5	二维数组的使用	203
7.5.1	二维数组的使用实例与升华	203
7.5.2	二维数组使用中的常见错误与解决方法	205
7.5.3	常见二维数组典型程序的分析与书写	206
7.6	使用数组的程序分析	210
7.7	数组实例的剖析与升华	212
7.8	本章常见问题的问与答	214
7.8.1	数组是基本数据类型吗	214
7.8.2	定义数组以后, 如果处理的元素超出了数组的范围, 如何处理	214
7.8.3	数组的下标总是从 0 开始吗	214
7.8.4	为什么要小心使用数组不要越界	214
7.8.5	数组可以是左值吗	215
7.9	小结	215
第 8 章	函数	216
8.1	程序的模块化	216
8.2	函数的定义	218
8.2.1	C 语言函数的简介	219
8.2.2	函数的定义	220
8.3	函数的调用	222
8.3.1	函数调用的一般形式	222
8.3.2	被调函数声明和函数原型	224
8.4	数据在函数间的传递	226
8.4.1	形式参数和实际参数	226
8.4.2	函数的返回值	228
8.5	数组作为函数的参数	229
8.5.1	一维数组作为函数的参数	230
8.5.2	二维数组作为函数的参数	233
8.6	函数的嵌套调用	234
8.6.1	嵌套调用的程序结构	234

8.6.2 程序实例的应用	235
8.7 函数的递归调用	236
8.7.1 递归的含义	236
8.7.2 递归程序的分析	238
8.7.3 递归程序的实例构思	240
8.8 函数使用的常见错误分析	244
8.9 典型函数程序实例	245
8.10 本章常见问题的问与答	247
8.10.1 什么叫函数	247
8.10.2 函数的说明与定义的区别	247
8.10.3 什么是函数参数以及如何传递函数参数	248
8.10.4 一个函数可以有多少个参数	248
8.10.5 如果一个函数没有返回值, 是否需要加入 return 语句	248
8.10.6 exit()和 return 有什么不同	248
8.10.7 为什么要说明函数原型	248
8.11 小结	249
第9章 作用域与程序设计	250
9.1 变量的作用域	250
9.1.1 局部变量	250
9.1.2 全局变量	252
9.2 变量的存储属性	255
9.3 局部变量的存储类别	256
9.3.1 自动变量	256
9.3.2 静态局部变量	257
9.3.3 寄存器变量	258
9.4 全局变量的存储类别	259
9.4.1 外部全局变量	259
9.4.2 静态全局变量	260
9.5 常见变量存储的错误分析与解决	261
9.6 应用不同存储属性的变量的程序例题分析	263
9.7 函数的作用域	264
9.7.1 外部函数	264
9.7.2 内部函数	265
9.7.3 函数的应用	265
9.7.4 多个源程序文件的编译和连接	266
9.8 简单的程序设计实验	267
9.9 本章常见问题的问与答	273
9.9.1 文件中增加了函数与变量的存储属性的时候, 常见的注释风格是怎样的	273
9.9.2 变量存储在内存中的什么地方	273
9.9.3 所有变量必须初始化吗	274
9.10 小结	274
第10章 指针	275
10.1 地址和指针的概念	275
10.2 变量的指针	277
10.2.1 指针变量的定义	277
10.2.2 指针变量的赋值与引用	278

10.3	指针变量的运算	281
10.3.1	&和*运算	281
10.3.2	++和--运算以及加减整型常量的运算	282
10.3.3	比较运算	283
10.3.4	减法运算	284
10.4	指针变量作为函数的参数	285
10.4.1	以普通变量作为参数完成交换函数	287
10.4.2	以指针变量作为参数实现数据的交换	287
10.4.3	交换指针变量指向的内容	287
10.5	指针变量实例中常见的错误分析与解决	288
10.6	指针与一维数组	289
10.6.1	指向一维数组元素的指针	289
10.6.2	用指针访问一维数组元素	291
10.6.3	一维数组作为函数参数	293
10.6.4	一维数组作为函数参数的实例	296
10.7	指针与二维数组	298
10.7.1	处理二维数组中所有元素的指针变量	300
10.7.2	使用指向一维数组的指针处理二维数组	301
10.7.3	二维数组作为函数参数	303
10.7.4	二维数组作为函数参数的实例	304
10.8	指针与数组常见的错误分析与解决	305
10.9	指针数组与二级指针	306
10.9.1	指针数组的概念	306
10.9.2	指针数组的应用	309
10.9.3	二级指针	311
10.9.4	指针数组作为主函数的参数	313
10.9.5	指针数组与二级指针常见错误分析与解决	314
10.10	指向函数的指针及返回指针的函数	315
10.10.1	用函数指针调用函数	315
10.10.2	用指向函数的指针作为函数的参数	316
10.10.3	返回指针的函数	319
10.11	应用例题及实验设计模板	320
10.12	本章常见问题的问与答	322
10.12.1	指针使用的常见形式有哪些	322
10.12.2	什么是 void 指针	322
10.12.3	什么时候使用 void 指针	322
10.12.4	NULL 总是被定义为 0 吗	322
10.12.5	用指针作 if 语句的条件表达式意味什么	322
10.12.6	NULL 和 NUL 有什么不同	323
10.13	小结	323
第 11 章	字符串	324
11.1	字符串与字符串的结束标志	324
11.2	字符数组与字符串	324
11.2.1	字符数组的定义	324
11.2.2	字符数组的初始化	325
11.2.3	字符数组的输入输出	327
11.2.4	字符数组处理字符串的应用实例	330

11.2.5	字符数组处理字符串常见的错误分析与解决	333
11.3	常用字符串处理函数	334
11.3.1	strcpy(字符数组,字符串): 字符串复制函数	335
11.3.2	strcat(字符数组 1,字符数组 2): 字符串连接函数	336
11.3.3	strcmp(字符串 1,字符串 2): 字符串比较函数	336
11.3.4	strlen(字符串): 求字符串长度函数	337
11.4	指针与字符串	339
11.4.1	字符串的表示形式	339
11.4.2	字符串指针作为函数参数	342
11.4.3	字符串常见错误分析	344
11.5	典型例题分析与升华	345
11.6	本章常见问题的问与答	346
11.6.1	字符是带符号的还是无符号的	346
11.6.2	字符串和数组有什么不同	347
11.6.3	字符类型与整型的根本区别是什么	347
11.6.4	可以将数字形式的字符串转换为数字吗	347
11.7	小结	348
第 3 部分	高级篇	349
第 12 章	编译预处理	350
12.1	预处理程序	350
12.2	宏定义指令	351
12.2.1	不带参数的宏定义	351
12.2.2	带参数的宏定义	356
12.2.3	常见的宏定义错误解析	360
12.3	文件包含指令	361
12.4	条件编译指令	363
12.4.1	#if~#endif 条件编译指令	364
12.4.2	#if~#else~#endif 条件编译指令	364
12.4.3	#if~#elif~#else~#endif 条件编译指令	365
12.4.4	#ifdef 格式	366
12.4.5	#ifndef 格式	366
12.4.6	条件编译的应用	367
12.5	本章常见问题的问与答	369
12.5.1	都有哪些预处理命令	369
12.5.2	下面程序的运行结果是什么	369
12.5.3	可以用#include 指令包含类型名不是“.h”的文件吗	369
12.5.4	使用宏更好,还是使用函数更好	369
12.5.5	用#define 指令说明常量有什么好处	370
12.6	小结	370
第 13 章	程序设计与调试	371
13.1	程序设计简介	371
13.2	程序调试	374
13.2.1	调试的概念	374
13.2.2	调试程序	375
13.2.3	查错	376
13.2.4	常用的避免错误的方法	380

13.3	程序的挂起	382
13.3.1	未知原因的挂起	382
13.3.2	死循环	383
13.3.3	运行时间比期望的时间长	384
13.3.4	等待正确的输入	385
13.4	小结	385
第 14 章	结构体和共用体	386
14.1	结构体的概念及结构类型定义	386
14.2	结构体变量	388
14.2.1	结构体变量的定义	388
14.2.2	结构体变量的引用	391
14.2.3	结构体变量的初始化	393
14.2.4	结构体变量作为函数的参数	395
14.3	结构体数组	397
14.3.1	结构体数组的定义及引用	398
14.3.2	结构体数组的初始化	400
14.3.3	结构体数组作为函数的参数	402
14.4	结构体指针	405
14.4.1	指向结构体变量的指针	405
14.4.2	指向结构体数组的指针	407
14.4.3	用指向结构体的指针作为函数参数	410
14.5	结构体变量常见的错误分析与解决	412
14.6	共用体	414
14.6.1	共用体的概念	415
14.6.2	共用体变量的定义和引用	416
14.6.3	共用体应用	420
14.7	枚举类型	422
14.7.1	枚举类型简介	422
14.7.2	枚举变量的定义与应用	423
14.7.3	枚举类型常见的错误分析	426
14.8	本章常见问题的问与答	427
14.8.1	结构体变量的数据传到另外的函数中有几种方法	427
14.8.2	用 <code>enum</code> 关键字说明常量有什么好处	427
14.8.3	和用 <code>#define</code> 指令说明常量相比, 用 <code>enum</code> 关键字说明常量有什么好处	428
14.9	小结	428
第 15 章	自定义类型与链表	429
15.1	<code>typedef</code> 自定义类型	429
15.2	动态内存管理	431
15.2.1	<code>malloc</code> 函数	432
15.2.2	<code>calloc</code> 函数	432
15.2.3	<code>free</code> 函数	433
15.2.4	<code>realloc</code> 函数	433
15.3	链表	435
15.3.1	链表简介	435
15.3.2	建立单链表	437
15.3.3	尾插法建立单链表	437

15.3.4	头插法建立单链表	440
15.3.5	单链表元素的遍历	441
15.4	单链表中进行的基本操作	442
15.4.1	单链表中查找元素	442
15.4.2	按照序号查找	443
15.4.3	按照指定值进行查找	443
15.4.4	单链表中元素的插入和删除	444
15.4.5	单链表中元素的合并	448
15.5	单循环链表和双向链表简介	450
15.5.1	单循环链表	450
15.5.2	双向链表	451
15.5.3	在双向链表中插入一个节点	452
15.5.4	在双向链表中删除一个节点	453
15.6	单链表设计实例与升华	454
15.7	自定义类型与链表常见的错误分析与解决	457
15.8	本章常见问题的问与答	458
15.8.1	什么是线性结构, 单链表是不是线性结构, 为什么	458
15.8.2	类型 <code>void *</code> 说明什么样的返回值	458
15.8.3	本章中函数的参数类型已经是指针类型, 为什么还通过返回值把链表的 头指针的值返回, 而不是通过参数直接改变呢	459
15.8.4	<code>NULL</code> 总是被定义为 0 吗	459
15.8.5	用 <code>malloc()</code> 函数更好, 还是用 <code>calloc()</code> 函数更好	459
15.9	小结	459
第 4 部分	底层开发篇	461
第 16 章	位与字节	462
16.1	数值数据的表示和编码	462
16.1.1	数据进制	462
16.1.2	数据存储	464
16.2	位运算	465
16.2.1	位运算简介	465
16.2.2	按位与运算符 “&”	466
16.2.3	按位或运算符 “ ”	468
16.2.4	按位异或运算符 “^”	469
16.2.5	按位取反运算符 “~”	471
16.2.6	按位左移运算符 “<<”	471
16.2.7	按位右移运算符 “>>”	472
16.2.8	位运算赋值运算符	473
16.2.9	位运算符的使用情况和实例分析	474
16.3	位运算的应用	475
16.4	位段及应用	478
16.4.1	位段的定义	478
16.4.2	位段的应用	480
16.5	位运算的常见错误分析与解决	484
16.6	本章的问与答	485
16.6.1	对位 (bit) 和字节 (byte) 的区分到底是什么	485
16.6.2	用什么方法存储标志 (flag) 效率最高	485

16.6.3	移位和乘以 2 这两种方式中哪一种更好	486
16.6.4	什么是高位字节和低位字节	486
16.6.5	16 位和 32 位的数是怎样存储的	486
16.7	小结	487
第 17 章	文件	488
17.1	文件系统概述	488
17.2	文件指针	491
17.3	文件打开与关闭	492
17.3.1	打开文件	493
17.3.2	关闭文件	495
17.4	文件读写函数	497
17.4.1	读写字符	497
17.4.2	使用 fgets()和 fputs()处理字符串	501
17.4.3	读写磁盘文件	504
17.4.4	成组读写	507
17.5	文件的定位	509
17.5.1	定位到开头	509
17.5.2	随机读写	510
17.5.3	返回位置	512
17.6	错误检测	513
17.6.1	ferror()函数	513
17.6.2	clearerr()函数	513
17.7	文件使用常见错误分析	514
17.8	文件例题分析与升华	516
17.9	本章常见问题的问与答	519
17.9.1	什么是 C 语言中的文件	519
17.9.2	在对文件进行操作的时候, 如果要求文件的位置回到文件的开头, 应当 用到哪个函数	519
17.9.3	设 fp 为文件指针, 如果打开一个已经存在的非空文件 aa.txt 进行修改, 应该用什么语句实现	519
17.9.4	文件的分类是什么, 到底什么是流式文件, 是不是文件分类的一种	519
17.9.5	读文件和写文件是什么	520
17.9.6	如何选择读写函数	520
17.9.7	缓冲文件系统是什么	520
17.10	小结	520
第 18 章	屏幕和键盘的控制	521
18.1	文本窗口	521
18.1.1	文本方式	521
18.1.2	文本窗口的使用	522
18.1.3	窗口内文本的输入输出	524
18.1.4	键盘的使用	527
18.1.5	具体应用实例——下拉式菜单	529
18.2	C 语言绘图概述	532
18.2.1	图形显示	532
18.2.2	图形系统初始化	533
18.2.3	图形屏幕的颜色设置	536

18.2.4	清除屏幕	537
18.3	C 语言中绘图函数的使用	539
18.3.1	绘图函数	539
18.3.2	线型函数	545
18.3.3	填充函数	546
18.3.4	自定义窗口	548
18.4	图形文本	549
18.4.1	文本输出函数	550
18.4.2	设置图形文本属性	551
18.5	绘图常见错误说明	553
18.6	绘图典型例题分析	555
18.7	本章的问与答	556
18.7.1	什么是像素	556
18.7.2	什么是分辨率	556
18.7.3	怎样在屏幕上定位光标	557
18.7.4	什么时候使用 far 指针	557
18.8	小结	557
第 19 章	系统调用	559
19.1	DOS、BIOS 与中断简介	559
19.2	DOS 下的设备控制	560
19.2.1	软中断调用函数	560
19.2.2	中断设置函数	563
19.2.3	端口输入输出函数	565
19.2.4	内存操作函数	566
19.3	系统驻留程序简介	567
19.4	系统函数调用常见错误解析	569
19.5	利用中断实现多任务的实例分析	570
19.6	本章常见问题的问与答	571
19.6.1	DOS 是什么	571
19.6.2	如何获得具体信息的内存位置	572
19.6.3	什么是缓冲区	572
19.6.4	什么是环境	572
19.7	小结	572
第 20 章	从 C 到 C++	573
20.1	C++对 C 的扩充	573
20.1.1	C++中输入/输出方式	573
20.1.2	新增运算符和关键字简介	575
20.1.3	引用	577
20.1.4	函数重载	579
20.1.5	函数内联	580
20.2	面向对象语言的概念	581
20.2.1	面向对象的概念	581
20.2.2	类的定义	582
20.2.3	对象的创建	584
20.2.4	对象的继承	586
20.2.5	多态性	588