

C程序设计导引

尹宝林 编著

*P*rogramming in C
A Practical Approach for Beginners



机械工业出版社
China Machine Press

大学计算机优秀教材系列

C程序设计导引

尹宝林 编著

*P*rogramming in C
A Practical Approach for Beginners



机械工业出版社
China Machine Press

图书在版编目 (CIP) 数据

C 程序设计导引 / 尹宝林编著. —北京: 机械工业出版社, 2013. 4
(大学计算机优秀教材系列)

ISBN 978-7-111-41891-7

I. C… II. 尹… 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 (2013) 第 055396 号

版权所有·侵权必究

封底无防伪标均为盗版

本书法律顾问 北京市展达律师事务所

本书是一本讲解 C 程序设计的入门教材, 根据学生对新知识学习和认知的规律, 从 C 语言和 C 程序的基本要素以及程序设计的基本方法开始, 循序渐进地引入对程序设计专业化的要求和相关知识。全书共分 10 章 (不包括第 0 章引言), 全面地介绍了 C 语言的基本语法及 C 语言程序设计方法, 内容包括: C 程序的创建及其基本结构; 常量、变量和表达式; 条件语句和开关语句; 循环语句和 goto 语句; 函数; 数组; 指针初步; 程序设计的基本方法; 结构和联合; 输入/输出和文件。每章均配有大量的例题和习题, 附录中介绍了不同版本的 C 语言标准之间的主要区别、vi/vim 的常用命令、使用 MS VC++ 6.0 IDE 创建 C 程序的基本过程、cc/gcc 的常用命令行选项、常用的标准库函数名及其头文件、ASCII 编码、调试工具 GDB 的常用命令。

本书特别适合作为计算机和非计算机专业学生学习高级语言程序设计的教材, 也可供计算机等级考试者和其他各类学习者使用参考。

机械工业出版社 (北京市西城区百万庄大街 22 号 邮政编码 100037)

责任编辑: 余洁

蕤城市京瑞印刷有限公司印刷

2013 年 5 月第 1 版第 1 次印刷

185mm × 260mm · 16.25 印张

标准书号: ISBN 978-7-111-41891-7

定 价: 35.00 元

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

客服热线: (010) 88378991 88361066

投稿热线: (010) 88379604

购书热线: (010) 68326294 88379649 68995259

读者信箱: hzsj@hzbook.com



作者自序

本书是一本讲解 C 程序设计的入门教材，以 C 语言为工具讲解程序设计，围绕程序设计的基本方法讲解 C 语言，其主要目的是使读者通过学习能够了解 C 语言常用语句的语法和语义，掌握程序设计的基本方法和技术，较为熟练、规范地完成不涉及复杂算法和数据结构的小型 C 程序的设计、编码和调试，为进一步发展自己的专业程序设计能力打下一定的基础。

程序设计与程序设计语言是密切相关但又互不相同的两个概念。程序设计语言是程序设计的工具，主要涉及语法和语义两个层面；而程序设计所要掌握的主要内容是对问题求解过程的理解和描述方法，以及如何在此基础上运用程序设计语言将问题求解过程表达为可以由计算机执行的程序。因此，学习程序设计在内容和方法上都与单纯地学习程序设计语言有很大的不同。很多初学者往往混淆于到底是学习程序设计还是学习程序设计语言，往往发现在花了不少时间学会了一门程序设计语言之后却写不出一个像样的程序。这其中的重要原因就在于学习目标以及相关学习内容和学习方法的选择。

因为是入门级程序设计教材，所以本书不讨论复杂的算法、技巧和数据结构，也不要求读者学习过这方面的课程。书中涉及的少量数据结构，也只是作为讲解 C 语言使用的例子，而不是对数据结构本身的深入探讨。当然，复杂的程序设计离不开算法和数据结构，但那是其他课程和教材的领域了。因为是基础性教材，所以本书不过多地讨论在基本程序设计中很少遇到的语言内容和较为深入的工程性细节，而是把重点放在如何有条理地思考所面对的程序设计问题，运用已有的知识描述对问题的求解思路，运用 C 语言准确地表达自己的思想等方面。因为是讲解程序设计方法的教材，所以本书在语言标准上采用 C89，而不是更新的 C99 和 C11。C99 和 C11 与 C89 的主要区别是为满足大型复杂程序的特殊需要而增加了一些新特性，没有对 C89 进行实质性的改变，对学习基本程序设计方法没有任何影响。而且，C89 是目前得到最广泛支持的 C 语言标准。几乎所有的主流编译器，包括支持 C99 和 C11 的编译器，也都支持 C89。而 C99 和 C11 则因为标准较新，支持的编译器较少，而且这两个标准中有不少可选项，也影响了编译器之间的兼容性。采用 C89，可以在不妨碍读者今后学习 C99 和 C11 的前提下更方便地选择上机练习所使用的编译器。

“兵无常势，水无常形”，程序设计也是如此。程序设计的水平和质量在很大程度上取决于程序设计人员的想象力和创造力。因此很多计算机领域的先驱和大家把程序设计称为一种艺术。对程序设计这样一门兼具技术和艺术性质的课程的学习，自然也没有一成不变的方法；为这样一门课程编写教材更绝非易事，特别是入门级教材。

由于机械工业出版社华章公司同仁的鼓励和帮助，本书才得以早日问世。本书的主要内容在北京航空航天大学的本科生程序设计课程中讲授有年，担任过该课程教学辅导的研

究生们在学生意见的收集、例题和习题的拟定以及课程的练习、考试成绩的统计等方面贡献良多，使本书的内容能够更加贴近初学者的需要。本书在写作中也得到了不少同事的热情鼓励和多方支持，作者对此表示衷心的感谢。限于水平，书中的错漏之处在所难免，还望读者不吝指正。

尹宝林

2012 年仲夏于京北

本书的教学内容不要求学生先修任何大学课程，因此可以安排在大学一年级讲授。

本书的授课时间建议为 32 ~ 36 学时，各章的参考学时见各章内容重点说明。本书在内容安排上兼顾计算机专业和非计算机专业的使用。对于非计算机专业，可以考虑只讲到第 7.4 节，各章的教学课时也可以酌情增加。第 5.9 节、6.2 节，以及 7.5 节之后的内容主要面向计算机专业的学生，有些内容以及相关例题的难度有所增加，在讲授时可以酌选。

例题和相关的程序代码是本书的重要内容。一些重点例题可以在课堂上重点讲解，其余例题可以要求学生自行阅读，并将不完整的代码补充完整。习题，特别是编程练习题，是本书的重要组成部分，对于巩固所学的理论知识，提高实际编程能力具有不可替代的作用。建议安排与讲课时数相当的上机时间，并提供必要的检查和辅导。

各章内容重点和学时分配如下：

第 1 章 C 程序的创建及其基本结构 (2 学时)

本章简要介绍 C 程序的创建过程、基本结构和各种组成要素、C 程序在编译过程中可能出现的问题及其调试方法，以及数据的输入/输出和标准文件的概念。本章的讲授重点是 C 程序的开发过程、C 程序的基本结构，以及语法错误的定位。本章中的【例 1-4】~【例 1-6】不必讲解，也不要求学生理解，可以鼓励学生自己阅读，建立对 C 程序的感性认识。

第 2 章 常量、变量和表达式 (3 学时)

本章讨论常量、变量、算术表达式、数据类型及类型转换，以及输入/输出函数 `scanf()` 和 `printf()`，重点是数据类型、数据类型的强制转换，以及 `scanf()` 和 `printf()` 的使用方法及字段说明符。本章中的【例 2-7】时钟指针综合讲解了数据类型强制转换的使用，以及从对问题的分析到方法的选择和代码实现的过程，可以在讲解前引导学生进行适当的讨论。

第 3 章 条件语句和开关语句 (2 学时)

本章讨论关系运算符和逻辑运算符、运算符的优先级、if 语句、switch 语句，重点是使用逻辑表达式描述实际问题中给定的条件、复合语句，以及 if 语句的嵌套关系。在讲解 switch 语句时，需要强调其分支条件表达式及其各个 case 的常量表达式应为整型，以及在各个 case 中语句段落的结尾不使用 break 的条件。

第 4 章 循环语句和 goto 语句 (2 学时)

本章讨论 while、for、do while 三种循环语句和 goto 语句，重点是 while 和 for 语句，以及循环语句的嵌套和非常规控制。本章的例题较多，其中【例 4-2】最大公约数、【例 4-6】水仙花数、【例 4-7】 π 的近似值、【例 4-10】连续正整数在本章或后面的章节中再次或多次使用，对

启发学生分析和解决问题、设计恰当的程序结构等有示范作用；【例 4-3】数据读入、【例 4-8】输出提示信息，读入应答是实际程序中常用的模式，可以重点讲解。

第 5 章 函数（4 学时）

本章贯穿始终的基本思想是通过函数的定义和调用，实现对计算过程的层次化描述。本章具体内容重点包括函数的基本概念、函数的结构和定义方式、函数的调用和返回值、全局变量和局部变量。递归在理论和实际计算中占有重要的地位，但对于初学者来说可能有一定的难度；对于非计算机专业的学生，这一部分（5.9 节）的内容可以不讲。

第 6 章 数组（6 学时）

本章的内容包括一维数组、字符数组和二维数组，其中一维数组及其在程序中的使用、字符数组的相关函数，以及数组作为函数参数时的描述和使用方式是内容的重点。本章的篇幅较长，对于非计算机专业的学生，第 6.2 节可以跳过不讲。例题中【例 6-2】字符分类统计、【例 6-3】质数表、【例 6-11】数组元素的二分查找、【例 6-17】字符串倒置、【例 6-23】星期几、【例 6-24】数据中的最长行、【例 6-26】使用二维数组绘制函数图像在课时允许的情况下可以重点讲解。对于计算机专业的学生，【例 6-13】区间合并、【例 6-14】括号匹配、【例 6-16】N 位超级质数等也可以重点讲解。

第 7 章 指针初步（5 学时）

本章以 7.4 节为界，分为前后两个部分，教师可以根据教学对象选择是否讲解后一部分。本章前一部分的重点是指针的概念、指针的类型、指针的运算，以及指针与数组的关系；后一部分的重点是指向二维数组的指针和函数指针。在函数指针部分，具有函数指针参数的库函数 `qsort()` 和 `bsearch()` 是学习的难点。这两个函数所提供的功能在程序设计中经常会用到，掌握这两个函数的使用方法不仅可以帮助学生理解函数指针的使用方法，而且对于提高初学者的编程水平具有重要的作用，因此需要重点讲解。在本章的前一部分，【例 7-8】子串逆置和【例 7-9】母牛的数量是较为重要的例子。在本章的后一部分，【例 7-13】输入数据的编号可以重点讲解。

第 8 章 程序设计的基本方法（6 学时）

本章是本书的重点之一，讨论从问题分析、方案设计、编码到调试和测试的程序设计过程，以便使学生掌握正确的程序设计方法。本章的教学目的在于使学生建立起对问题的分析和对求解过程的整体规划入手进行程序设计的概念，逐步掌握自顶向下进行问题分析的方法，养成脱离编程语言和代码设计问题求解方案的习惯，以及初步掌握对程序进行测试和调试的能力。本章概念性和理论性的内容稍多，多数例子也比前面各章的例子复杂，因此需要根据学生的反馈和学生完成练习的情况，把握和调整教学进度。本章中【例 8-3】流程控制关键字的统计、【例 8-6】数字删除、【例 8-12】序列的第 N 项等对学生体会程序设计的基本方法具有启发作用，可以重点讲解。【例 8-10】花朵数较为复杂，可以根据情况酌选。

第 9 章 结构和联合（2 学时）

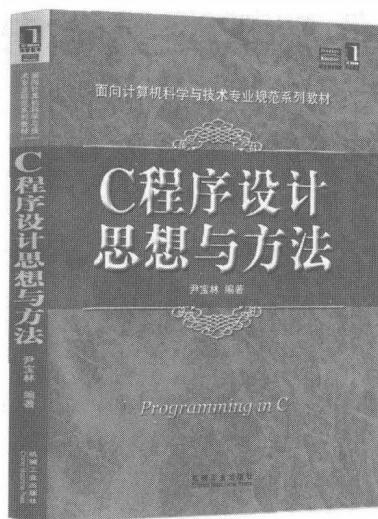
本章的重点是结构，包括结构的概念、结构变量的定义和初始化、结构成员的访问方式、结构指针等。【例 9-2】输入数据的编号、【例 9-4】同心圆、【例 9-5】单向链表、【例 9-6】链

表逆置函数是较为重要的例子。复杂类型的解读对于提高学生阅读和编写较为复杂程序的能力较为重要，在课时允许的情况下可以重点讲解。联合在初级程序设计中使用较少，可以根据课时情况酌选。

第10章 输入/输出和文件（2学时）

本章的重点是文件的概念、文件访问的操作步骤、文件读写的定位等。【例10-2】日程列表和【例10-2-1】日程列表的排序显示是本章较为重要的例子。以正文方式对文件的基本读写操作在前面的章节中使用较多，因此上面这些内容的难度不大。相比之下，二进制读写的内容不多，但对初学者可能有一定的难度，可以根据情况酌选。

推荐阅读



C程序设计思想与方法

作者：尹宝林 ISBN：978-7-111-25495-9 定价：36.00元

作者简介

尹宝林 1973年毕业于北京航空学院计算机专业，1984年获英国爱丁堡大学博士学位。现任北京航空航天大学计算机系教授、博士生导师。从事计算机专业教学和科研工作多年，曾主讲“C语言程序设计”、“高级语言程序设计”、“UNIX程序设计环境”、“计算机图形学”、“图像处理”等课程，其中“高级语言程序设计”被评为北京市精品课程。主编过《离散数学》等教材。参加全国信息学奥林匹克（NOI）活动的组织与指导工作，任全国信息学奥林匹克科学委员会副主席。

本书围绕程序设计的思想和方法组织教学内容，把C语言作为讲解程序设计的工具，把教学的重点从C语言本身转移到程序设计的思路和方法以及使用C语言编程上来，把讲授语言的概念转变为讲授语言的运用，把对概念的讲授由抽象的分析和讲解转变为通过实例和经验的学习和运用。

本书作者根据学生对新知识学习、认知的规律，从C语言和C程序的基本要素以及程序设计的基本方法开始，循序渐进地引入对程序设计专业化的要求和相关的知识；从增强学生的感性认识入手，通过多角度的对例题的分析，示范对关键知识和技术的运用，通过对关键内容在不同层次上的适当重复，深化学生对概念的理解和掌握。本书精选的例题和练习题，在功能、知识和技术上覆盖了从语言入门到实用工具多个层面，有助于学生掌握专业化的编程技能，养成专业化的编程习惯。

本书作者专门设计和实现了基于Web的C程序练习/评测系统，为学生提供具有及时监控和反馈环节的实践环境，为本书的练习题提供联网的实时评测。



目 录

作者自序
教学建议
例题索引

第0章 引言	1
第1章 C程序的创建及其基本结构	7
1.1 C程序的创建过程	7
1.2 C程序的基本组成	9
1.3 调试初步——语法错误的定位	14
1.4 数据的输入/输出和标准文件	17
习题	18
第2章 常量、变量和表达式	19
2.1 常量	19
2.1.1 数字常量	19
2.1.2 字符常量	21
2.1.3 字符串字面量	22
2.2 变量	23
2.2.1 变量名和变量类型	23
2.2.2 变量的赋值和类型转换	25
2.2.3 变量的初始化	26
2.2.4 类型限定符 const	26
2.3 算术表达式	26
2.3.1 算术运算符	27
2.3.2 增量和减量运算符	29
2.3.3 位运算	29
2.3.4 复合赋值运算符	30
2.4 强制类型转换	31
2.5 数据输出/输入函数 printf() 和 scanf()	34
2.5.1 数值的输出函数 printf	34
2.5.2 数值的输入函数 scanf	35
2.6 常量的符号表示方法	37
2.6.1 常量宏	37
2.6.2 枚举常量	38

习题	39
第3章 条件语句和开关语句	42
3.1 关系运算符和逻辑运算符	42
3.2 运算符的优先级	44
3.3 从实际问题中的条件到逻辑 表达式	45
3.4 条件语句	46
3.4.1 条件语句中的条件	46
3.4.2 复合语句	47
3.4.3 条件语句的嵌套和级联	48
3.4.4 使用条件语句时的注意 事项	51
3.4.5 条件运算符和条件表达式	53
3.5 switch 语句	53
习题	56
第4章 循环语句和 goto 语句	59
4.1 while 语句	59
4.2 for 语句	61
4.3 do while 语句	65
4.4 循环语句的选择和使用	67
4.5 逗号表达式	67
4.6 循环语句的嵌套	68
4.7 循环语句中的非常规控制	71
4.8 goto 语句	74
习题	75
第5章 函数	78
5.1 函数的基本概念	78
5.2 函数的调用	79
5.3 函数的结构	82
5.4 函数的定义	83
5.5 函数的调用关系和返回值	87
5.6 局部变量和全局变量	88
5.6.1 局部变量	88
5.6.2 全局变量	90

5.6.3 对全局变量的访问	91	第7章 指针初步	143
5.7 函数参数的传递	92	7.1 地址与指针	143
5.8 标准库函数	93	7.2 指针变量	144
5.8.1 常用的头文件	93	7.2.1 指针变量的定义和赋值	144
5.8.2 常用的数据输入/输出函数	94	7.2.2 通过指针访问数据	145
5.8.3 字符类型判断函数	96	7.2.3 作为函数参数的指针	147
5.8.4 字符串处理函数	97	7.2.4 返回指针的函数	149
5.8.5 其他常用函数	98	7.3 指针运算	150
5.9 递归初步	99	7.3.1 指针与整数的加减	150
习题	102	7.3.2 指针相减	152
第6章 数组	104	7.3.3 指针的比较	152
6.1 一维数组	104	7.3.4 指针的强制类型转换和 void *	155
6.1.1 一维数组的定义和初始化	104	7.3.5 不合法的指针运算	156
6.1.2 一维数组元素的使用	106	7.3.6 指针类型与数组类型的 差异	156
6.1.3 数组的复制	109	7.4 指针与数组	157
6.1.4 作为函数参数的一维数组	109	7.5 指向二维数组的指针	158
6.1.5 数组元素的排序和查找	112	7.6 多重指针	160
6.2 使用一维数组的常用数据结构	116	7.7 指针数组	161
6.2.1 散列表	116	7.7.1 一维指针数组	161
6.2.2 栈	118	7.7.2 命令行参数	163
6.2.3 队	120	7.8 函数指针	164
6.3 字符串和字符数组	123	7.8.1 函数指针变量的定义	164
6.3.1 字符串	123	7.8.2 具有函数指针参数的 库函数	165
6.3.2 字符数组	124	习题	168
6.4 常用的标准字符串函数	125	第8章 程序设计的基本方法	171
6.4.1 字符串输出函数	125	8.1 程序设计的基本过程	171
6.4.2 字符串输入函数	127	8.2 问题分析	172
6.4.3 字符串复制和追加函数	129	8.2.1 程序功能和输入/输出 数据	172
6.4.4 字符串比较函数	130	8.2.2 对程序性能的要求	174
6.4.5 字符串检查函数	131	8.2.3 程序中的错误处理	174
6.4.6 字符串扫描函数 sscanf()	132	8.2.4 程序的测试	174
6.5 二维数组	133	8.2.5 问题分析的结果	175
6.5.1 二维数组的定义	133	8.3 方案设计	176
6.5.2 二维数组元素的引用	134	8.3.1 解题思路	177
6.5.3 二维数组元素的遍历	135		
6.5.4 二维数组元素的排列方式	135		
6.5.5 作为参数的二维数组	137		
习题	140		

8.3.2 算法的描述	178	9.4 复杂类型的解读	216
8.3.3 数据结构和算法的选择	182	习题	217
8.4 编码:从算法到代码	187	第 10 章 输入/输出和文件	220
8.4.1 代码的结构	188	10.1 输入/输出的基本过程和文件 类型	220
8.4.2 代码的检查	194	10.2 文件的打开、创建和关闭	221
8.4.3 代码中的注释	195	10.3 文件数据的正文格式读写	223
8.5 程序的调试	195	10.4 读写操作中的定位	226
8.5.1 调试的基本方法	196	10.5 文件数据的二进制格式读写	229
8.5.2 调试工具的基本功能和 使用	197	习题	231
8.5.3 标准输入/输出的重新 定向	198	附录 A 不同版本的 C 语言标准 之间的主要区别	234
习题	200	附录 B vi/vim 的常用命令	236
第 9 章 结构和联合	202	附录 C 使用 MS VC++ 6.0 IDE 创建 C 程序的基本过程	237
9.1 结构 (struct)	202	附录 D cc/gcc 的常用命令行选项	242
9.1.1 结构类型的定义	202	附录 E 常用的标准库函数名及其 头文件	243
9.1.2 结构成员的访问	204	附录 F ASCII 编码	245
9.1.3 包含结构的结构	207	附录 G 调试工具 GDB 的常用命令	246
9.2 联合 (union)	209	参考文献和推荐书目	247
9.2.1 联合类型的定义	210		
9.2.2 联合成员的访问	210		
9.3 类型定义 (typedef) 语句	214		

例 题 索 引

【例 1-1】最简单的 C 程序	9	【例 4-5】阶乘	62
【例 1-2】产生可见效果的 C 程序	10	【例 4-2-1】最大公约数	63
【例 1-3】生成直角三角形	11	【例 4-6】水仙花数	63
【例 1-4】生成质数表	12	【例 4-7】 π 的近似值	64
【例 1-5】生成镶边的等腰三角形	12	【例 4-8】输出提示信息，读入应答	65
【例 1-6】猜数游戏	13	【例 4-2-2】最大公约数	66
【例 2-1】十进制数转换为十六进制数	20	【例 4-9】平行四边形图案	68
【例 2-2】十六进制数转换为十进制数	20	【例 4-10】连续正整数	69
【例 2-3】表达式的类型和值	27	【例 4-11】阶乘之和	70
【例 2-4】计算球体体积	28	【例 4-1-1】自然数的累加	71
【例 2-5】星期几	28	【例 4-1-2】自然数的累加	72
【例 2-5-1】星期几	28	【例 4-10-1】连续正整数	72
【例 2-6】求一元二次方程的根	28	【例 4-12】乒乓球赛	73
【例 2-7】时钟指针	32	【例 4-13】数据求和	74
【例 2-8】整数的表示方式	35	【例 4-8-1】输出提示信息，读入应答	75
【例 2-9】浮点数和字符序列的输出格式	35	【例 5-1】三角形的面积	80
【例 2-10】读入数据	36	【例 5-2】星期几	84
【例 2-11】计算实数乘积的整数部分	37	【例 5-3】最大公约数	85
【例 3-1】判断闰年	45	【例 5-4】水仙花数	86
【例 3-2】水仙花数	45	【例 5-5】 π 的近似值	86
【例 3-3】名次预测正确性的表达式	45	【例 5-6】判断整数的奇偶	88
【例 3-4】计算并输出一元二次方程的根	47	【例 5-7】局部变量和全局变量的访问	90
【例 3-5】判断季节	49	【例 5-8】全局变量的共享	91
【例 3-6】四则运算	50	【例 5-9】错误的变量交换函数	92
【例 3-7】求最大值	53	【例 5-10】printf() 的格式	95
【例 3-8】输出比较结果的文字值	53	【例 5-11】最大公约数的递归计算	99
【例 3-9】成绩信息	54	【例 5-12】组合公式的计算	99
【例 3-6-1】四则运算——使用 switch 语句	55	【例 5-13】阿克曼函数的计算	100
【例 3-9-1】成绩信息	55	【例 5-14】梵塔 (Hanoi Tower)	100
【例 4-1】自然数的累加	59	【例 6-1】摄氏 - 华氏温度对照表	106
【例 4-2】最大公约数	60	【例 6-2】字符分类统计	106
【例 4-3】数据读入	61	【例 6-3】质数表	107
【例 4-4】行数统计	61	【例 6-4】数组复制	109

【例 6-5】 三维向量的点积	110	【例 7-12】 使用 <code>qsort()</code> 对一维 <code>double</code> 数组排序 ...	165
【例 6-6】 n 维向量的点积	111	【例 7-13】 输入数据的编号	166
【例 6-7】 平均年龄	111	【例 7-14】 查质数表	168
【例 6-8】 n 维向量加法函数	112	【例 8-1】 矩阵乘法	172
【例 6-9】 选择排序	112	【例 8-2】 两条线段的交点	173
【例 6-10】 数组元素的顺序查找	113	【例 8-3】 流程控制关键字的统计	173
【例 6-11】 数组元素的二分查找	114	【例 8-1-1】 矩阵乘法	175
【例 6-12】 首个重复字母的位置	116	【例 8-4】 完全平方数	177
【例 6-13】 区间合并	117	【例 8-5】 字符串循环移位	179
【例 6-14】 括号匹配	119	【例 8-6】 数字删除	179
【例 6-15】 一个 <code>int</code> 型的队	120	【例 8-7】 整数中的数字分离	180
【例 6-16】 N 位超级质数	122	【例 8-8】 组合的生成	180
【例 6-17】 字符串倒置	124	【例 8-9】 公式求解	182
【例 6-18】 平行四边形图案	126	【例 8-10】 花朵数	184
【例 6-19】 由参数确定输出的小数位数	127	【例 8-11】 6174 问题	188
【例 6-20】 分析日期和时间信息	132	【例 8-12】 序列的第 N 项	190
【例 6-21】 第几天	134	【例 8-13】 小数化为分数	193
【例 6-22】 矩阵相加	135	【例 9-1】 结构类型的定义	202
【例 6-23】 星期几	136	【例 9-2】 输入数据的编号	205
【例 6-24】 数据中的最长行	136	【例 9-3】 矩形	207
【例 6-25】 矩阵加法函数	138	【例 9-4】 同心圆	207
【例 6-26】 使用二维数组绘制函数图像	138	【例 9-5】 单向链表	207
【例 7-1】 数据中的最长行	146	【例 9-6】 链表逆置函数	209
【例 7-2】 数据统计函数	147	【例 9-7】 联合类型的定义	210
【例 7-3】 删除换行符	149	【例 9-8】 联合成员的访问	211
【例 7-4】 字符串替换	150	【例 9-9】 后缀表达式求值	211
【例 7-5】 指针的加减运算	150	【例 9-10】 显示当前时间	216
【例 7-6】 指针相减	152	【例 10-1】 打开文件	222
【例 7-7】 多行数据的平均值	153	【例 10-2】 日程列表	224
【例 7-8】 子串逆置	154	【例 10-3】 正文数据的字符数	227
【例 7-9】 母牛的数量	157	【例 10-4】 文件按行倒置输出	227
【例 7-10】 星期几	161	【例 10-2-1】 日程列表的排序显示	228
【例 7-11】 计算命令行参数的代数和	163	【例 10-5】 二进制数据的字符数	230

引 言

计算机技术是人类在 20 世纪最重要的发明之一，对我们社会的方方面面产生了深刻而持久的影响。作为计算机技术的重要组成部分，程序设计技术在计算机技术的发展和应用过程中发挥了至关重要的作用。时至今日，程序设计技术及其相关的程序设计语言依然是计算机技术这一庞大复杂的技术体系的重要基石。在众多的程序设计语言中，C 语言由于其突出的特点，不仅在各种软件系统中得到了广泛的应用，而且在程序设计技术的学习上发挥着无可替代的作用，成为广大专业人员学习程序设计的第一语言。

计算机与程序设计

计算机系统由硬件和软件两部分组成。与其他仪器设备不同的是，计算机的硬件只提供了一个具有广泛通用性的计算平台，其具体功能和应用领域主要取决于它的软件系统。

程序是软件的主要表现形式，程序设计是软件实现的主要手段，程序设计语言是程序设计的基本工具。从描述的层面和与硬件的相关程度上看，程序设计语言可以分为低级编程语言和高级编程语言两大类。低级语言包括机器语言和汇编语言，其语言结构基本上是面向特定机器指令系统的指令序列，对计算过程的描述是在目标机操作的层面进行的。因此低级语言严格依赖于特定的硬件指令系统，可移植性差。同时，由于语言的描述层次很低，程序的可读性和可维护性差，代码较长，不适于大型软件的开发。

高级程序设计语言在与目标机无关的层面对问题的计算过程进行描述，它可以屏蔽计算过程的执行细节、突出计算过程的目标和运算方法，便于问题的分析和描述。由于高级程序设计语言在较高的层次上对计算机的执行过程进行抽象描述，一条高级程序设计语言的语句往往等价于很多条机器指令，这就大大提高了程序设计的效率。同时，高级程序设计语言需要通过编译系统转换成机器指令，或者通过语言解释系统解释执行，其本身与具体的目标机无关。只要提供相应的编译系统或解释系统，用高级程序设计语言写成的程序就可以运行在任何计算机上。高级程序设计语言的这些特点，使得它成为目前程序设计中使用的主流语言。C 语言就是这样一种目前被广泛使用的高级程序设计语言。

C 语言概述

C 语言是由美国 Bell 实验室的 Dennis M. Ritchie 于 1973 年为研制 UNIX 操作系统而专门设计的，是一种编译型的高级程序设计语言。由于它简洁精练，描述能力强，生成代码的执行效率高，适用领域宽，同时又与 UNIX 操作系统有着天然的紧密关联，因此迅速地被计算机和信

息领域的专业人员普遍接受。尽管 C 语言面世已近 40 年,但仍然是一种使用最广泛和具有重要技术影响力的程序设计语言。目前, C 语言被广泛地应用于从系统软件到应用软件的各种软件系统。据统计,近些年来 C 语言在编程领域的使用比例一直保持在 15% ~ 20% 左右,位居编程语言的前列。大量的通用软件和工具软件,特别是各类基础软件,如各种操作系统和编译器、各种脚本语言的解释器以及 Java 虚拟机等的实现,几乎毫无例外地都使用 C 语言。在 C 语言之后发展起来的很多重要语言也是建立在 C 语言语法和基本结构的基础之上的。对于 C++ 和 C# 这样的语言,仅从名字上就可以明白它们与 C 语言的关系。

C 语言是一种编译型的编程语言。所谓编译型语言,是与解释型语言相对的。使用编译型语言编写的程序,需要使用编译系统进行编译,生成以所在计算平台的机器指令码表示的可执行代码,然后再在计算平台上运行。使用编译型语言所写的程序运行效率很高,大多数主流编程语言,如 C、C++ 等都是编译型语言;而广为人知的 BASIC 语言则是解释型语言的典型代表。大量的脚本语言,如 Web 网页编程使用较多的 Perl、PHP 和 Python 等也都是解释型语言。使用解释型语言编写的程序由语言的解释系统处理。解释系统对程序进行分析,有可能生成一些中间表示形式,但是不生成以所在计算平台的机器指令码表示的可执行代码,程序的运行是以解释系统对程序或中间结果进行解释的方式完成的,因此程序的执行效率较低。

C 语言的成功得益于它的诸多特点。根据设计者自己的评价,“C 是一种通用的程序设计语言,它包含了紧凑的表达式、现代控制流和数据结构,以及丰富的运算符集合。C 不是一种‘很高级’的语言,也不‘大’,它不特定于某一个应用领域。但是 C 的限制少,通用性强,这使得它比一些被认为功能强大的语言更方便,效率更高”。对于刚刚涉足软件领域的人来说,把 C 语言作为第一门高级编程语言的重要优点是, C 语言简洁精练,所涉及的概念比较少,语法和结构也简单。这些不但便于初学者掌握 C 语言的基本要素,而且便于初学者把学习的注意力集中在掌握 C 语言的运用以及更一般的程序设计技术方面。

C 语言的早期版本定义在 Brian W. Kerninghan 和 Dennis M. Ritchie 的经典著作《C 程序设计语言》第一版的参考手册中,一般称其为 K&R C。1989 年,美国国家标准协会(ANSI)完成了一个 C 语言的标准化版本 ANSI C。这个新版本与 K&R C 基本相同,只是在一些描述方式上做了少量的改进,对语法做了少量的扩充,有些语句的语义有轻微的改变,增加了一些新的关键字,改变了函数声明和定义的语法,并且定义了一个标准的 C 函数库。但总的来说,ANSI C 保持了 K&R C 的基本风格和内容,绝大部分以 K&R C 编写的程序都可以在 ANSI C 的环境下正确地编译运行。1990 年,国际标准化组织(ISO)也采纳了这个标准(ISO/IEC 9899),称为 ISO C90。我国于 1994 年制定的国家标准 GB/T15272 等同采用该标准。因为这一标准的 ANSI 版本是在 1989 年制定的,因此也常被称为 C89。除了这两个版本之外,GCC 和 MS VC 等也是目前使用比较广的 C 语言版本。这些版本对 ANSI C 进行了少量的扩充,但是在语言描述层面没有大的改动,因此都可以正确地编译运行以 ANSI C 编写的程序。1999 年,ANSI 和 ISO 合作,提出了 ISO/IEC 9899:1999,简称 ISO C99 标准,在语言的描述层面和实际特性上都进行了一些扩展。ISO C99 提供了一些新的机制,并且在个别地方与 ANSI C 有些明显的差异。完全支持 ISO C99 标准的编译系统不多。一些号称支持 C99 标准的编译系统只是有选择地实现了 C99 对 C89 的部分扩展。有些编译系统在实现 C99 特性时也根据自己系统的特点做了部分的调整和改动。因此很多支持 C99 的编译系统之间并不兼容。实际上,C99 的这些新机制及相关的语法改动在规模较小的简单程序中不太会用到,对于初学者来说并不重要。2011 年,ISO 提出

了 ISO/IEC 9899: 2011, 简称 ISO C11 标准, 增加了一些新的特性, 以提高与 C++ 的兼容性。因为这一标准刚刚推出, 所以尚未有编译系统对其提供完整的支持。附录 A 列出了 C89 与 C99 和 C11 的主要差别。

本书的内容依从 C89 标准, 此外只采用了目前所有主流编译器都支持的“行注释”。本书不采用 C99 和 C11 的原因有二。首先, 本书所要讲解的是如何使用 C 语言进行程序设计, 而不是最新的 C 语言标准。C89 提供了 C 语言的全部基础功能, 已经超过了学习 C 语言编程的基本需要。而 C99 及 C11 所扩充的特性主要用于大型复杂程序的编写, 在学习基本编程时很难用到。本书把注意力集中在 C89 上, 有助于读者减轻学习负担, 加快掌握 C 语言程序设计的核心能力。其次, C89 是目前得到最广泛支持的 C 语言标准。即使是 C99 和 C11 的编译器, 也必然向下兼容 C89, 反之则未必。使用 C89 可以使读者在选择编译器进行练习时更容易一些。本书把内容限制在 C89, 不会影响读者所能学到的程序设计知识和能力。C89 本身完全可以胜任大规模复杂程序的设计, 而不必借助于 C99 和 C11 的新特性。当掌握了 C89 以及基本的程序设计方法后, 学习和掌握 C99 和 C11 的新特性应该是一件很容易的事。

程序运行平台和文件

程序的运行平台也称为宿主机, 由硬件和软件两部分组成。运行平台中与程序设计直接相关的硬件配置主要是计算机 CPU 的类型和内存的大小。CPU 是中央处理器的英文缩写, 是计算机进行运算的核心部件。内存是计算机在运行时保存程序代码和中间运算结果的线性存储空间, 一般以字节为单位, 各个存储单元的编号称为存储单元的地址。与程序设计直接相关的软件配置主要是操作系统和编译系统。用户通过操作系统向计算机发布各种命令, 通过编译系统将自己的程序翻译成计算机能够执行的代码。程序的运行平台与程序设计的工作平台在概念上是相互独立的, 但在多数情况下编程人员就在程序的运行平台上进行程序设计的各项工作。这时, 程序的运行与程序的设计就共享同一个工作平台了。

尽管高级程序设计语言可以屏蔽计算机硬件的大部分差异, 但仍然有一些硬件上的差异会影响到程序的设计和实现, 特别是在使用 C 语言这样兼具高级语言和低级语言特征的编程语言时。对 C 语言而言, CPU 的类型主要影响到语言中一些基本数据类型的长度, 而这会影响到数值表达和计算的范围; 内存的大小可能会影响到程序中可以使用的内存空间的大小, 以及程序运行的速度。目前大多数计算机使用的 CPU 都采用 Intel 公司的 IA32 结构 (有时也称为 x86 结构), 其所配置的内存一般都数以十亿字节 (GB) 计, 远远超过了简单程序运行时的需要, 因此硬件平台对初学程序设计的读者影响不大。

目前广泛使用的操作系统主要有 UNIX/Linux 和 MS Windows 两大类。从系统基本功能的角度看, 这两类操作系统的差异不大。从具体操作的角度看, UNIX/Linux 既提供了基于字符终端的命令行操作模式, 也提供了基于用户桌面的图形界面操作模式; 而 MS Windows 则主要提供基于用户桌面的图形界面操作模式。从程序设计的角度看, 这两类操作系统之间有一些较小的差别。例如, 在这两类操作系统上, 对于文件中的换行符和结束符的表示方法是不完全相同的; UNIX/Linux 区分文件名中的大小写字母, 而 MS Windows 对文件名中的大小写字母一视同仁。在刚刚开始学习程序设计时, 这些小小的差异可能会给初学者带来一些麻烦, 但在掌握了一定的编程经验之后, 这些问题其实是很容易解决的。使自己的程序具有一定的环境适应能