

零起点 学通C语言

— (多媒体范例教学) —



1DVD 光盘

从零基础到精通C语言，一步到位

■

404个实用范例，穿插海量经验和技巧，重点难点全透析

■

10小时高清语音教学视频抢先发布，百万读者翘首以待，再掀C语言学习风暴

范磊 编著



科学出版社

零起点 学通C语言

(多媒体范例教学)

范磊 编著

科学出版社

北京

内 容 简 介

本书吸取了十多本 C 语言图书及教材的优点,摒弃了语言拖沓、层次结构混乱等缺陷,从零开始、由浅入深、层层递进、细致而又详尽地讲解 C 语言这门大型编程语言。

本书知识系统全面,拥有字典般的容量,可随用随查,涵盖指针、函数、字符串、编译处理、算法、数据结构、底层开发、跨平台开发、文件操作、宏等主流 C 语言开发技术。为了使读者能够活学活用,本书针对重要的概念精心设计了 404 个实用范例,囊括大量经验和技巧,即使已从事 C 语言工作多年的朋友,也能从中摄取新的养料。

本书适合从未学习过任何编程语言的新手,以及学习 C 语言多年,仍不能融会贯通的读者,对于正在使用 C 语言进行开发的程序员也有很好的参考价值。

图书在版编目(CIP)数据

零起点学通 C 语言:多媒体范例教学/范磊编著. —北京:科学出版社, 2013. 5

ISBN 978-7-03-037120-1

I. ①零… II. ①范… III. ①C 语言—程序设计
IV. ①TP312

中国版本图书馆 CIP 数据核字(2013)第 049261 号

责任编辑:周晓娟 高莹 / 责任校对:杨慧芳
责任印刷:华程 / 封面设计:杨英

科学出版社 出版

北京东黄城根北街 16 号

邮政编码:100717

<http://www.sciencep.com>

三河市李旗庄少明印装厂印刷

中国科技出版传媒股份有限公司新世纪书局发行 各地新华书店经销

*

2013 年 5 月第 一 版

开本:16 开

2013 年 5 月第一次印刷

印张:41.5

字数:1009 000

定价:78.00 元(含 1DVD 价格)

(如有印装质量问题,我社负责调换)

前言

2010 年 12 月中旬，赵编辑突然联系到我，希望我写一本适合初学者的 C 语言图书。

当时没有任何迟疑就答应了，但随之而来的则是压力，因为毕竟 C 语言的图书太多了。能否在这些书中脱颖而出，尚且是个未知数。

为此我翻阅了大量 C 语言图书，希望能找到它们的不足。值得庆幸的是，几乎没费多少劲，就找到了，因为这些书似乎都不是为初学者而写的。

传统 C 语言图书的不足

1. 章节安排不合理

无论国内还是国外的书，都试图在第 1 章就将 C 语言的全貌展现给初学者，有的甚至在第 1 章就讲了算法，这样做只会吓跑初学者。不过假如他们的目的就是吓跑初学者，那么，很显然，他们做得很成功。

2. 总想把一句话讲得天衣无缝

很多国外的作者是大师级的人物，他认为说话一定要完全标准，要天衣无缝，要严谨，然而，这样往往会把话说得很复杂、很啰唆，使人不知所云。这样的书明显不适合初学者，不过在读者精通 C 语言后，将其作为字典进行查询也是很不错的。

3. 理论多于实践

大部分图书喜欢不厌其烦地讲理论，很少用实践去证明理论的真实性，得出的结论也很难令人信服。

4. 过多纠缠细枝末节

有些图书热衷于讨论事物的细枝末节，而对于事物的本质却一带而过。如果立志成为“书呆子”，那么阅读这些书将有很大的帮助。

5. 喜欢跳跃

有些书的毛病是喜欢将一个特性讲完后，再去讲另一个特性，由于每个特性都与其他特性相互联系，因此要看这样的书，需要来回跳跃，初学者非常痛苦，没有体会到学习和读书的乐趣。这样的书其实会一点点抹杀初学者的兴趣，直到他们彻底厌恶并远离 C 语言为止。

6. 大量公式

有些书最擅长用 C 语言来解决许多数学问题，这是很不好的习惯，毕竟计算机不是专门为了解决数学问题而设计的，过多的公式只会让初学者望而生畏，从而达到厌倦学习 C 语言的目的。

7. 枯燥乏味

有些书的最大毛病就是枯燥乏味，很难引起读者的兴趣。这样的书，即使作者名气很大，也大多会被束之高阁，等同于没有。

本书特色

C++就像大海，要全面了解 C++，必须从大海中跳出来，从高处往下俯瞰，但是 C 语言则不同，C 语言就像小溪流，不需要高处俯瞰就可以纵览全貌，因此本书不会再从大局讲解 C 语言，而是从局部着手讲解 C 语言的每个特性，不会放过每个细节。

本书的目标是易学、实用、通俗、全面、标准、精炼、准确。在语言描述上力求准确、精炼、通俗、易懂；在实例安排上力求全面、标准、实用；在章节编排上力求循序渐进；在语法阐述时尽量避免术语和公式；相对于同类书来说，本书具备以下特点。

1. 章节安排合理

本书不会在读者还不能写出一行代码时，就开始讲算法，而是根据读者的理解能力合理安排学习进度。

为了达到这个目的，我先后找了十多名未曾学过任何编程语言的人，让他们来阅读此书，每当遇到有读不下去的段落，就将该部分后移，直到能像阅读小说一样顺畅为止，这个过程付出了很多心力，也付出了很多时间，所幸的是，目标达成了。

2. 通俗易懂

为了使大多数读者都能看懂，本书尽量采用故事的形式进行说明，如条件选择语句那章的开头，则通过古希腊一个流传甚广的小故事来说明 if 语句的特性。

3. 尽量避免公式

实际编程工作中涉及的数学知识并不多，况且本书讲的是 C 语言而不是数学，因此没有必要罗列大量的数学公式。

4. 主题分明

并不试图在讲述一个语法时，将围绕该语法的所有知识都列举出来，这样做虽然很省事，但是往往会使初学者头晕眼花，产生一个 C 语言很难学的错觉。

本书在描述一个语法时就只讲它，对于它的附属，则在用到时才进行讲解。这样不会偏离主题，使读者思路清晰，因此会感觉 C 语言很好学。另外，知识在用到时方才进行说明，读者的印象会更深刻。

5. 循序渐进地进行补充

避免空洞的讲述语法，而是让语法特性伴随着一个个鲜明的实例不断进行渗透，这个过程就像海绵吸水一样，不是硬性要求去背诵语法规则，而是告诉读者为什么这么设置语法规则，并且用大量实例来演示这样制定语法规则的好处。

6. 剥茧抽丝的讲解方式

对于千头万绪、盘根错节的问题，并不是一头扎到乱麻中，而是善于找到问题的源头，通过源头将问题一点点解开。实践证明，经常采用这种方式讲解，会使读者学会独立解决问题的方法。

7. 实用性强

本书运用大量实例来演示 C 语言的语法，容易令人信服，不像其他书那样理论多于实践。

另外，本书也不罗列规则，而是将规则渗透到一个个实例中，使读者灵活运用规则。自古以来，看着兵法书打仗的将军都不是好将军，只有灵活运用兵法的将军才是好将军。

8. 全面细致

本书内容庞大，就像一本大型的字典，几乎所有的疑问都可以在本书中查到。

9. 准确精炼

语言描述力求准确、精炼、通俗、易懂，尽量用朴实生动的语言阐述最复杂的问题，而且力求准确。文字描述尽量简练，能用一句话概括的，绝不用两句话。

10. 严扣标准

基于大多数编译器采用的是 C89 标准，因此本书严格按照 C89 标准进行阐述，只有在用到 C99 标准的新特性时才会讲一下。所有实例均已在多个平台下测试通过，对于非标准的代码，则针对每个平台给出了相应的实现。

本书的附录列出了标准库中的所有常用函数，包括 C99 标准。其中也包括一些非标准的函数，这些函数通常是频繁使用的，为了与 C 标准库函数相区别，这些函数的名称下面都注有“非标准”3 个字。

如何迈出坚定的第一步

马克思说过，万事开头难，每门科学都是如此。

柏拉图说过，良好的开端，是成功的一半。

由此可见，如何迈出第一步，是很重要的一件事。一个好的开始，等于成功了一半。

那么如何迈出第一步呢？不同的人有不同的方式，大多数人选择了制定计划。

从 2009 年 7 月起到现在，大约有 200 名读者来信向我讨教如何制定学习计划，有的甚至列了一大堆计划表，要我帮他选一个。

每当我看到这些密密麻麻的计划表时，都不由自主地感觉有些眩晕，随之想到的是，这个计划，你能执行多久？

一个计划写得再好，也不代表你有着良好的开端，只有有效的施行，才是最关键的。

那么如何才能算是有效的施行呢？那就是：不带任何功利性思想的行动！

纯粹的行动，不能带有任何功利性的思想，我先前也不信，通过几件事才认识到，只要有功利性的思想，受到打击或挫折时，就会格外沮丧，就会放弃努力，假如没有功利性的思

想，那么做这个纯粹是兴趣，也就不会感到受挫，更不会觉得沮丧，如玩游戏，纯粹是兴趣，没有任何的功利思想，所以从来不会因为被某个玩家打败而感到失败沮丧，从而放弃玩游戏。

从这个观点来看，定目标也是错误的，你玩游戏的时候从来不会给自己定目标，一天要玩几个小时吧，而总是克制不住自己，偷偷地玩。学习也不要制定任何目标，假如你制定了目标却没有完成，那么你就会产生受挫感，而受挫感又会使你对学习这件事情失去兴趣，从而产生厌恶感。我们的目标是对学习产生兴趣，这样显然是背道而驰了，所以最好的方式是不要定目标，而是慢慢养成一种习惯。一种习惯的养成只需要一周时间，为了促进一种良好习惯的养成，完全可以每天只学习 5 分钟，5 分钟是非常容易做到的事情，这样一周内你都会洋溢在一种成功的喜悦之中，从而越来越喜欢做这件事情，自发地延长学习的时间，从 5 分钟到 10 分钟，再到半小时、1 小时、8 小时，慢慢地你就会发现自己一天都沉浸在学习中，并且快乐得不能自己。

光盘内容

为了达到教学的最佳效果，我综合了 200 多名来信读者的意见，整理出 10 小时 252 段视频教程，并提供了 404 个实例源程序，与图书内容对应，以便读者最大限度地提高学习效率。

致谢

本书虽是笔者长期教学和 C 语言开发经验的总结，但也凝聚了众多文献的精华，包括：

- 《C 语言参考手册（第五版）》，Samuel P. Harbison III & Guy L. Steele Jr.
- 《C 语言核心技术》，Peter Prinz & Tony Crawford
- 《C 语言大全（第 4 版）》，Herbert Schildt
- 《你必须知道的 495 个 C 语言问题》，Steve Summit
- 《C Primer Plus（第五版）》，Stephen Prata
- 《21 天学通 C 语言》，Bradley L. Jones & Peter Aitken
- 《C 标准库》，P. J. Plauger

- 《C 语言程序设计：现代方法》，K.N.King
- 《C 程序设计语言（第 2 版）》，Brian W.Kernighan & Dennis M.Ritchie
- 《C 程序设计（第二版）》，谭浩强
- 《Linux 程序设计》，Neil Matthew & Richard Stones
- 《程序设计语言：实践之路》，Michael L.Scott
- 《计算机组成原理》，唐朔飞
- 《深入理解计算机系统》，Randal E. Bryant & David O'Hallaron
- 《操作系统》，刘腾红
- 《操作系统原理》，谢青松
- 《Linux 下的 C 库函数源代码》
- 《标准 C 语言的官方文档》
- 《C99 标准的官方文档 (ISO/IEC 9899:1999)》
- 《C 标准库源码解剖》

同时也得到了一些软件行业朋友的大力支持，在此深表感谢。在本书的编写过程中，虽力求完美，但难免存在一些不足之处，敬请广大读者批评指正。联系方式：najiushifeng@126.com。

编著者
2013 年 3 月

目 录

第 1 章 基础知识	1
1.1 计算机语言概述	1
1.1.1 机器语言——面向机器的语言	1
1.1.2 汇编语言——面向机器的语言	1
1.1.3 高级语言——面向过程、面向对象的语言	2
1.2 C 语言的起源	2
1.3 为什么要学习 C 语言	4
1.4 集成开发环境	5
第 2 章 一个最简短的 C 程序	6
2.1 程序示例	6
2.2 注释	7
2.3 语句	9
2.4 块	9
2.5 表达式	9
2.6 表达式语句	10
2.7 总结	11
2.3 语句	9

4.6	局部变量	22
4.7	全局变量	23
4.8	全局变量的缺点	25
4.9	作用域	26
4.10	输出函数	27
4.11	输入函数	29
4.12	字符输出函数	31
4.13	字符输入函数	31
4.14	函数的声明与定义	33
4.15	总结	34
第 5 章 数据类型		35
5.1	C 语言为什么要有数据类型	35
5.2	变量的通用原则	35
5.2.1	变量的声明	35
5.2.2	变量的命名	36
5.2.3	变量的定义	37
5.2.4	变量的赋值与初始化	38
5.2.5	初始化的用途	38
5.2.6	变量在内存中的排列	39
5.3	整型变量	39
5.3.1	整型变量的取值范围	40
5.3.2	超出取值范围的后果	40
5.3.3	long 与 int 的区别	41
5.3.4	输出短整型变量的值	42
5.3.5	输出整型变量的值	42
5.3.6	输出长整型变量的值	43
5.3.7	输出双长整型变量的值	44
5.3.8	输出八进制数和十六进制数	45
5.3.9	为类型设置别名	46
5.4	浮点型变量	47
5.4.1	float 型变量的取值范围	48
5.4.2	double 型变量的取值范围	49
5.4.3	long double 型变量的取值范围	50
5.4.4	浮点型变量的精度	51
5.4.5	超出精度的误差	51
5.4.6	输出浮点型变量的值	53
5.4.7	输入数值到浮点型变量	54
5.5	字符型变量	54
5.5.1	字符型变量与 ASCII 码	55
5.5.2	字符型变量的赋值	56

5.5.3	字符型变量只能保存单个字符	57
5.5.4	输出所有的可见字符	57
5.5.5	数字与数字字符	58
5.5.6	显式转换与隐式转换	58
5.5.7	类型提升	59
5.5.8	类型下降	60
5.5.9	特殊字符	60
5.6	_Bool 型变量	62
5.7	常量	63
5.7.1	宏	63
5.7.2	const 常量	65
5.7.3	const 常量与宏的区别	65
5.7.4	枚举型常量	65
5.8	总结	66
第 6 章 条件选择语句		67
6.1	关系运算符	67
6.2	if 语句	68
6.2.1	else 语句	70
6.2.2	else if 语句	70
6.2.3	if 语句的嵌套	72
6.2.4	if 与 else 的配对	73
6.2.5	复杂的嵌套	75
6.3	逻辑运算符	76
6.3.1	逻辑与	77
6.3.2	逻辑或	77
6.3.3	逻辑非	78
6.3.4	括号改变优先级	79
6.3.5	真与假	80
6.4	三目运算符	81
6.4.1	三目运算符的优先级	82
6.4.2	灵活运用三目运算符	82
6.4.3	三目运算符的操作数类型	83
6.4.4	三目运算符最常见的用法	83
6.5	switch 语句	84
6.5.1	switch 语句的用法	84
6.5.2	去掉 break 的 switch 语句	85
6.6	总结	86
第 7 章 循环语句		87
7.1	循环语句的前身——goto 语句	87
7.2	慎用 goto 语句	88

7.3	while 语句	88
7.3.1	带逻辑运算符的 while 语句	89
7.3.2	限定 while 循环的次数	90
7.3.3	continue 语句	91
7.3.4	break 语句	92
7.3.5	exit 函数	92
7.3.6	猜数字	93
7.3.7	永不休止的 while 循环	95
7.4	do...while 循环	96
7.5	for 循环	97
7.5.1	for 循环的执行过程	98
7.5.2	for 循环的注意细则	99
7.5.3	域	99
7.5.4	灵活的 for 循环	100
7.5.5	条件为空的 for 循环	100
7.5.6	导航菜单	102
7.5.7	执行为空的 for 循环	103
7.5.8	嵌套的 for 循环	104
7.5.9	九九乘法表	105
7.6	总结	106

第 8 章 综合实例——博弈机 107

8.1	模块 1——游戏规则	108
8.2	模块 2——确定选择方式与支出积分	108
8.3	模块 3——掷出骰子	109
8.4	模块 4——判断点数的大小	110
8.5	模块 5——计算输赢的积分	111
8.6	模块 6——检查积分够不够	113
8.7	程序的入口——主函数	114
8.8	完整程序	116
8.9	总结	123

第 9 章 初识指针 124

9.1	什么是地址	124
9.2	什么是指针	124
9.3	通过指针进行读取	125
9.4	通过指针进行修改	126
9.5	空指针	127
9.6	变更指针保存的地址	128
9.7	指针自身的地址	128

9.8 指针运算	129
9.8.1 指针的加减运算	129
9.8.2 指针的赋值运算	130
9.8.3 指针的相减运算	131
9.8.4 指针的比较运算	131
9.9 指针的类型	132
9.10 const 与指针	133
9.10.1 常量指针	133
9.10.2 指向常量的指针	134
9.10.3 指向常量的常指针	134
9.11 总结	135
第 10 章 堆和栈	136
10.1 内存中的数据	136
10.2 什么是堆和栈	138
10.2.1 栈	138
10.2.2 堆	138
10.2.3 堆和栈的应用	138
10.3 堆与指针	139
10.3.1 利用指针访问堆	139
10.3.2 释放指针指向的堆空间	140
10.3.3 关于 free 函数的注意事项	141
10.3.4 内存泄露	142
10.3.5 迷途指针	143
10.4 总结	144
第 11 章 数组	145
11.1 数组的用法	145
11.1.1 数组的定义	145
11.1.2 数组的元素	145
11.1.3 数组的下标越界	146
11.1.4 倒序存放数据	147
11.1.5 将数组的长度定义为常量	148
11.1.6 新手容易犯的错误	149
11.1.7 数组的初始化	149
11.2 数组的实例	150
11.2.1 求平均成绩	150
11.2.2 斐波那契数列	151
11.2.3 查找最大数	152
11.2.4 查找某数	154
11.3 排序算法	155

11.3.1	冒泡排序法	155
11.3.2	选择排序法	157
11.3.3	插入排序法	160
11.4	数组在内存中的分布	163
11.5	数组名	165
11.6	数组名的地址	165
11.7	数组与指针	166
11.7.1	利用指针输出数组元素	166
11.7.2	查找最大数	167
11.8	数组与枚举常量	168
11.9	什么是多维数组	169
11.10	多维数组的初始化	170
11.11	打印杨辉三角形 (1)	171
11.12	打印杨辉三角形 (2)	172
11.13	总结	174

第 12 章 字符串

12.1	字符数组	175
12.1.1	定义字符数组	175
12.1.2	初始化字符数组	175
12.1.3	输出字符数组中的所有字符	176
12.2	字符串	177
12.2.1	字符串的初始化	177
12.2.2	字符串的输入	178
12.2.3	字符串输入的常见问题	179
12.3	常用字符串函数	180
12.3.1	strlen 函数——得到字符串长度	180
12.3.2	strlwr 函数——将大写转为小写	180
12.3.3	strupr 函数——将小写转为大写	181
12.3.4	puts 函数——字符串输出	181
12.3.5	strcpy 函数——字符串复制	182
12.3.6	memcpy 函数——内存复制	182
12.3.7	memccpy 函数——内存复制	183
12.3.8	strcat 函数——字符串连接	184
12.3.9	strncat 函数——连接 n 个字符	184
12.3.10	strchr 函数——查找字符	185
12.3.11	strset 函数——设置字符串	186
12.3.12	strcmp 函数——比较字符串	186
12.3.13	strrev 函数——反转字符串	187
12.3.14	strstr 函数——查找子串	188
12.3.15	常见的字符串函数列表	188

12.4	字符串函数的实现思路	189
12.4.1	strlen 函数的实现思路	189
12.4.2	strcat 函数的实现思路	189
12.4.3	strcpy 函数的实现思路	190
12.4.4	strcmp 函数的实现思路	191
12.4.5	strrev 函数的实现思路	192
12.5	字符串函数的应用实例	192
12.5.1	密码验证	193
12.5.2	插入字符串	194
12.5.3	姓名排序	195
12.6	指针与字符串	197
12.6.1	字符指针	197
12.6.2	字符串常量	198
12.6.3	删除字符串中的字符	198
12.6.4	字符指针数组	199
12.6.5	指向指针的指针	200
12.6.6	英语词典	201
12.7	总结	203
第 13 章 结构体和共用体		204
13.1	什么是结构体	204
13.2	什么是结构变量	204
13.3	结构变量的多种定义方式	205
13.4	结构变量的简单应用	206
13.5	结构变量的初始化	207
13.6	结构变量的赋值	208
13.7	结构体数组	209
13.8	英汉词典	210
13.9	结构体的嵌套	217
13.9.1	在职员工档案登记系统	218
13.9.2	退休员工档案登记系统	220
13.10	结构体的大小	222
13.11	结构体与指针	224
13.11.1	指向结构体变量的指针	224
13.11.2	指向结构体数组的指针	225
13.11.3	指向结构体数组的指针运算	226
13.12	结构体与 typedef	227
13.13	共用体	228
13.13.1	共用体的大小	229
13.13.2	共用体变量不能被初始化	229
13.13.3	共用体类型可以在结构体类型中定义	230

13.13.4 共用体的作用	231
13.14 总结	232
第 14 章 深入函数	233
14.1 函数的运行机理	233
14.2 函数的特殊调用方式	235
14.3 函数的求参顺序	235
14.4 函数的嵌套调用	236
14.5 函数的递归调用	237
14.5.1 直接调用自身	237
14.5.2 间接调用自身	237
14.5.3 终止递归调用	238
14.5.4 递归调用的用途	239
14.6 指针作为函数参数	240
14.6.1 按值传递	240
14.6.2 按址传递	241
14.6.3 实现系统库函数 <code>strlwr</code>	242
14.7 数组作为函数参数	243
14.8 数组的长度作为函数参数	244
14.9 数组只有一个原本	245
14.10 数组形参退化为指针	246
14.11 指针数组作为函数参数	247
14.12 指向指针的指针作为函数参数	249
14.13 结构体变量作为函数的参数	250
14.14 结构体数组作为函数的参数	251
14.15 内部函数	253
14.16 外部函数	253
14.17 可变参数	255
14.18 外部变量	255
14.19 静态变量	256
14.19.1 静态局部变量	257
14.19.2 静态局部变量的应用	258
14.19.3 静态全局变量	259
14.19.4 全局变量	260
14.20 自动变量	260
14.21 寄存器变量	261
14.22 总结	261
第 15 章 预处理指令	262
15.1 宏定义指令	262