



陈锐 等编著

C语言 从入门到精通

 电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

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

C 语言从入门到精通

陈 锐 等编著

電 子 工 業 出 版 社

Publishing House of Electronics Industry

北京 · BEIJING

内 容 简 介

本书从最基本的概念入手,由浅入深,综合典型的实例,引导初学者由浅入深地掌握 C 语言。本书共 21 章。其中,第 1~2 章是起步篇,包括程序设计基础知识和开发工具。第 3~12 章是基础篇,包括基本数据类型、运算符与表达式、语句、C 语句与数据的输入输出、结构化程序设计、数组、函数、指针、结构体和联合体、位运算与预处理。第 13~20 章是提高篇,包括链表、文件、图形界面设计、键盘与鼠标操作、网络编程、常用算法、队列和栈、排序算法。

本书内容全面,不仅涵盖了 C 语言的基本语法与简单的程序设计知识,还包括高级的程序设计技术与常见算法。本书每个知识点都给出了程序实例和完整源码,语言通俗,不仅适合 C 语言的初学者学习,还适合有一定基础,希望进一步提高的程序开发人员阅读。

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 语言从入门到精通 / 陈锐等编著. —北京:电子工业出版社,2010.10

ISBN 978-7-121-11869-2

I. ①C… II. ①陈… III. ①C 语言-程序设计 IV. ①TP312

中国版本图书馆 CIP 数据核字(2010)第 184403 号

责任编辑:李红玉

文字编辑:谭丽莎

印 刷:北京天竺颖华印刷厂

装 订:三河市鑫金马印装有限公司

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

北京市海淀区翠微东里甲 2 号 邮编:100036

开 本:787×1092 1/16 印张:30.5 字数:780 千字

印 次:2010 年 10 月第 1 次印刷

定 价:57.00 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 zllts@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语言是目前国内外使用最为广泛的程序设计语言之一。它具有功能丰富、表达能力强、使用方便灵活、执行效率高、可移植性好等优点，几乎可用于所有领域。C语言既具有高级语言的特点，也具有汇编语言的功能，还具有很强的系统处理能力，可以直接对硬件和外部接口进行控制。C语言被广泛应用于系统软件和应用软件的开发。

使用C语言进行程序设计和软件开发，可以熟悉并理解计算机内部的工作原理，对于深入学习计算机技术是大有裨益的。C语言是计算机科学与技术专业的基础课程，是以后学习数据结构与算法的基础，也为以后选择 Visual C++或 Java 软件开发奠定了基础。因此，只有熟练地掌握了C语言，以后才能更加深入地掌握计算机技术。

1. 本书的特点

1) 内容全面，讲解详细

为了使读者系统、全面地掌握C语言知识，本书不仅讲解了C语言的基本程序设计知识，包括数据类型、结构化程序设计、数组、指针、结构体等，还介绍了链表、文件、图形界面设计、键盘与鼠标操作、网络编程、常用算法、队列和栈等高级程序设计技术。这是市场上同类图书不具备的。

在讲解方法上，本书本着通俗、易学的指导思想，尽量用通俗的语言，针对C语言中每个知识点都进行了详细的讲解，以便读者能快速理解并掌握每个知识点。

2) 层次清晰，结构合理

本书将C语言知识分为篇、章、节和小节，将每一个知识点进行细化，这样便于读者理解和掌握。经过这样的详细划分之后，内容重点突出、结构层次感强，使读者容易抓住重点。通过将概念与例子结合，还可使读者更加容易理解与消化。

3) 结合图表，通俗易懂

C语言概念较多，知识点零碎，为便于读者学习，在概念出现之后都给出了相应的例子和表格进行说明，以使读者领会其含义；对于复杂的程序，均结合程序流程图进行讲解，以方便读者理解程序的执行过程；在语言的叙述上，普遍采用了短句子、易于理解的语言，而避免了使用复杂句子和晦涩难懂的语言。

4) 实例典型，深入剖析

在讲解每一个大的知识点时，本书结合具体实例进行了剖析。本书选取的都是典型且涵盖知识点丰富的实例。在每一章的最后或比较大的知识点后面，还给出了一个完整且实用的程序。本书在给出程序的同时，还对程序通过图进行具体讲解，深入分析，并在程序的最后给出了运行结果。读者在学习的过程中，可结合例子和运行结果来验证程序的正确性。

5) 实用性强，延伸知识

本书讲解的内容不仅全面，还具有很强的实用性。除了介绍C语言中的语法知识外，本书还讲解了常用算法、数据结构、键盘和鼠标操作、网络编程等技术，这些内容都是软件开发常

用的技术。通过这些知识点的介绍，可使读者进一步理解计算机相关技术的实现原理，具有很强的引导性，也为读者今后的继续深入学习奠定了基础。

2. 适合的读者

- (1) 没有任何基础，准备学习 C 语言的初学者。
- (2) 有一定 C 语言基础，想进一步掌握 C 语言的中、高级读者。
- (3) 大中专院校学生。
- (4) 软件开发人员。
- (5) 计算机相关的科研工作者。

由于时间仓促，书中难免存在错误，敬请广大读者批评指正。

目 录

第 1 篇 起步篇

第 1 章 C 语言基础	1	1.3.3 浮点数	13
1.1 为什么要选择 C 语言	1	1.4 小结	15
1.1.1 选择 C 语言的好处	1	第 2 章 C 语言常用开发环境的介绍	16
1.1.2 C 语言的特点	2	2.1 Turbo C 3.0 开发环境的介绍	16
1.1.3 如何学好 C 语言	2	2.1.1 Turbo C 3.0 的开发环境	16
1.2 程序设计语言基础——进制转换	3	2.1.2 使用 Turbo C 3.0 运行 C 语言程序	18
1.2.1 二进制与位权	4	2.1.3 C 程序的开发步骤	19
1.2.2 二进制数与十进制数的相互转换	5	2.2 Visual C++ 6.0 开发环境的介绍	20
1.2.3 十六进制数与十进制数的相互转换	6	2.2.1 使用 Visual C++ 6.0 新建项目	20
1.2.4 十进制数与八进制的转换	8	2.2.2 在 Visual C++ 6.0 的项目中新建程序文件	21
1.2.5 各种计算机进制数的转换	9	2.2.3 Visual C++ 6.0 的开发环境	23
1.3 计算机中数的表示	10	2.2.4 使用 Visual C++ 6.0 运行 C 语言程序	24
1.3.1 计算机中的正数与负数表示	10	2.3 小结	26
1.3.2 原码、补码	11		

第 2 篇 基础篇

第 3 章 基本数据类型	27	整型变量的值	34
3.1 数据类型的定义及分类	27	3.4 实型变量	35
3.2 变量与常量	29	3.4.1 实型变量的定义与赋值	35
3.2.1 变量	29	3.4.2 实型变量的表示范围	36
3.2.2 常量	30	3.4.3 一个简单的 C 程序——输出实型变量的值	36
3.3 整型变量	31	3.5 字符型变量	36
3.3.1 整型变量的定义	32	3.5.1 字符型变量的定义与赋值	37
3.3.2 整型变量占用的字节数与表示范围	32	3.5.2 字符型数据的存储形式	37
3.3.3 整型变量的存储形式	33	3.5.3 字符数据与字符串数据的区别	39
3.3.4 整型变量的赋值	34	3.6 小结	39
3.3.5 一个简单的 C 程序——输出			

第 4 章 运算符与表达式	40	输入函数	65
4.1 运算符与表达式的基础	40	5.2.3 getch 函数——另一个字符数据 输入函数	66
4.1.1 运算符的分类	40	5.3 格式化数据的输入与输出	67
4.1.2 运算符的优先级与结合性	41	5.3.1 printf 函数——格式化数据的 输出	67
4.1.3 多种类型数据的混合运算—— 自动类型转换	41	5.3.2 scanf 函数——格式化数据的 输入	74
4.2 算术运算符与算术表达式	43	5.4 小结	77
4.2.1 +, -, *, /, %——双目 运算符	43	第 6 章 结构化程序的设计	79
4.2.2 算术表达式	43	6.1 顺序结构程序的设计	79
4.2.3 强制类型转换	44	6.1.1 顺序结构	79
4.2.4 ++与—、+与———单目 运算符	45	6.1.2 顺序结构程序设计应用举例	79
4.3 赋值运算符与赋值表达式	47	6.2 选择结构程序的设计	81
4.3.1 直接赋值运算符与直接赋值 表达式	47	6.2.1 if 选择结构	82
4.3.2 赋值表达式中的类型转换	48	6.2.2 switch 选择结构	90
4.3.3 复合算术赋值运算符与 表达式	53	6.2.3 条件运算符与条件表达式	95
4.4 关系运算符与关系表达式	54	6.2.4 选择结构程序设计应用举例	96
4.4.1 关系运算符	54	6.3 循环结构程序设计	100
4.4.2 关系表达式	55	6.3.1 while 循环语句	101
4.5 逻辑运算符与逻辑表达式	56	6.3.2 do-while 循环语句	102
4.5.1 逻辑运算符	56	6.3.3 for 循环语句	104
4.5.2 逻辑表达式	56	6.3.4 break 语句	109
4.6 逗号运算符与逗号表达式	58	6.3.5 continue 语句	110
4.6.1 逗号运算符	58	6.3.6 goto 语句	112
4.6.2 逗号表达式	59	6.3.7 循环结构的嵌套	113
4.6.3 逗号运算符应用举例	59	6.3.8 循环结构程序设计应用举例	116
4.7 小结	60	6.4 小结	120
第 5 章 C 语句与数据的输入/输出	61	第 7 章 数组	121
5.1 语句	61	7.1 一维数组	121
5.1.1 语句的分类	61	7.1.1 定义一维数组	121
5.1.2 赋值语句	63	7.1.2 引用一维数组	122
5.1.3 变量赋初值	64	7.1.3 初始化一维数组	123
5.2 字符数据的输入与输出	64	7.1.4 一维数组应用举例	123
5.2.1 putchar 函数——字符数据 输出函数	65	7.2 二维数组	127
5.2.2 getchar 函数——字符数据		7.2.1 定义二维数组	127

7.3 字符数组与字符串	135	8.9.1 内部函数	187
7.3.1 定义字符数组与初始化	135	8.9.2 外部函数	187
7.3.2 引用字符数组	136	8.10 小结	189
7.3.3 字符数组与字符串	136	第9章 指针	190
7.3.4 字符数组的输入与输出	137	9.1 地址和指针	190
7.3.5 字符串的相关函数	138	9.1.1 什么是地址	190
7.3.6 字符串应用举例	143	9.1.2 什么是指针——间接存取	191
7.4 小结	147	9.2 指针与变量	192
第8章 函数	149	9.2.1 定义指针变量	192
8.1 函数初识与函数的分类	149	9.2.2 引用指针变量	192
8.1.1 函数初识	149	9.2.3 指针变量作为函数的参数	195
8.1.2 函数的分类	150	9.3 指针与数组	198
8.2 函数的定义	150	9.3.1 指向数组元素的指针与指向 数组的指针	198
8.2.1 无参函数的定义	150	9.3.2 指向多维数组的指针变量	201
8.2.2 有参函数的定义	151	9.3.3 数组名(指针)作为函数 参数	205
8.2.3 有参函数传统的定义方式	152	9.3.4 指针数组	214
8.3 函数的参数与函数返回值	152	9.3.5 二级指针	215
8.3.1 实际参数与形式参数	152	9.4 指针与字符串	217
8.3.2 函数返回值	154	9.4.1 字符串指针	217
8.4 函数的调用	156	9.4.2 字符串指针作为函数参数	218
8.4.1 函数的一般调用	156	9.4.3 字符指针数组	220
8.4.2 函数原型	159	9.4.4 指针数组作为 main 的参数	222
8.4.3 函数的嵌套调用	162	9.5 指针与函数	223
8.5 函数的递归调用	165	9.5.1 函数指针——指向函数的 指针	224
8.5.1 递归调用的定义	165	9.5.2 函数指针作为函数的参数	225
8.5.2 递归调用应用举例	166	9.5.3 指针函数——返回指针值的 函数	228
8.6 数组作为函数的参数	169	9.5.4 void 指针	230
8.6.1 数组元素作为函数参数	169	9.6 指针与 const	230
8.6.2 数组名作为函数参数	171	9.6.1 常量指针——指向常量的 指针	230
8.6.3 多维数组名作为函数参数	174	9.6.2 指针常量——指针不可以 改变	231
8.7 变量的作用域	176	9.6.3 常量指针常量——指向常量的 指针常量	232
8.7.1 局部变量	176	9.7 小结	233
8.7.2 全局变量	177		
8.8 变量的存储类别	181		
8.8.1 自动变量	181		
8.8.2 静态变量	182		
8.8.3 寄存器变量	183		
8.8.4 外部变量	184		
8.9 内部函数与外部函数	187		

第 10 章 结构体与联合体	234	10.6.3 枚举类型应用举例	262
10.1 结构体	234	10.7 小结	265
10.1.1 定义结构体类型	234	第 11 章 位运算	266
10.1.2 定义结构体变量	235	11.1 位运算符与位运算	266
10.1.3 引用结构体变量	237	11.1.1 位与运算	266
10.1.4 初始化结构体变量	238	11.1.2 位或运算	268
10.1.5 结构体应用举例	240	11.1.3 异或运算	268
10.2 结构体数组	241	11.1.4 取反运算	270
10.2.1 定义结构体数组	241	11.1.5 左移运算	270
10.2.2 初始化结构体数组	242	11.1.6 右移运算	273
10.2.3 结构体数组应用举例	243	11.1.7 位复合赋值运算符	275
10.3 指针与结构体	246	11.2 位段	275
10.3.1 指向结构体变量的指针	246	11.2.1 定义位段	275
10.3.2 指向结构体数组的指针	248	11.2.2 引用位段成员	276
10.3.3 结构体变量作为函数的 参数	249	11.3 小结	277
10.3.4 指向结构体变量的指针作为 函数的参数	251	第 12 章 预处理命令	278
10.4 用 typedef 定义类型	252	12.1 宏定义 #define	278
10.4.1 使用 typedef 定义类型	252	12.1.1 不带参数的宏定义	278
10.4.2 typedef 应用举例	254	12.1.2 带参数的宏定义	280
10.5 联合体	255	12.1.3 条件编译命令中的运算符 # 和 ##	282
10.5.1 定义联合体类型及变量	255	12.2 文件包含命令 #include	283
10.5.2 引用联合体	256	12.2.1 文件包含命令的两种方式	283
10.5.3 使用联合体应注意的问题	257	12.2.2 文件包含命令应用举例	284
10.5.4 联合体的应用举例	258	12.3 条件编译命令	285
10.6 枚举类型	261	12.3.1 条件编译命令——#ifdef	285
10.6.1 定义枚举类型及变量	261	12.3.2 条件编译命令——#ifndef	287
10.6.2 使用枚举类型的一些说明	261	12.3.3 条件编译命令——#if	287
		12.4 小结	288
第 3 篇 提高篇			
第 13 章 链表	291	13.2.4 链表的插入操作	298
13.1 什么是链表	291	13.2.5 链表的删除操作	299
13.1.1 链表的基本概念	291	13.3 链表应用举例	303
13.1.2 动态内存分配	292	13.3.1 直接插入排序——使用链表 实现	303
13.2 链表的操作	293	13.3.2 一元多项式的相加	309
13.2.1 创建链表	293	13.4 小结	312
13.2.2 输出链表	296	第 14 章 文件	314
13.2.3 链表的查找操作	297		

14.1 文件与文件类型指针	314	15.5 基本绘图函数	349
14.1.1 文件的分类	314	15.5.1 画点类函数	349
14.1.2 文件类型指针	315	15.5.2 画线类函数	350
14.2 打开文件与关闭文件	315	15.5.3 圆弧类函数	351
14.2.1 打开文件	315	15.5.4 多边形函数	352
14.2.2 关闭文件	316	15.5.5 填充函数	354
14.3 文件的读写	317	15.6 图形模式下的文本输出	356
14.3.1 fgetc 函数与 fputc 函数	317	15.6.1 文本输出函数	356
14.3.2 fread 函数与 fwrite 函数	319	15.6.2 文本属性函数	357
14.3.3 fscanf 函数与 fprintf 函数—— 格式化读写函数	323	15.6.3 汉字的输出	359
14.3.4 fgets 函数与 fputs 函数—— 字符串读写函数	325	15.7 小结	362
14.4 文件的定位	325	第 16 章 键盘与鼠标操作	363
14.4.1 rewind 函数——重置文件 指针	326	16.1 键盘操作	363
14.4.2 fseek 函数——定位文件 指针	327	16.1.1 键盘编码	363
14.4.3 ftell 函数——得到文件指针 位置	329	16.1.2 键盘操作函数	364
14.5 出错检测	329	16.2 鼠标操作	366
14.5.1 ferror 函数	330	16.2.1 鼠标的工作原理	366
14.5.2 clearerr 函数	330	16.2.2 鼠标综合应用举例	371
14.6 小结	330	16.3 小结	374
第 15 章 图形界面设计	331	第 17 章 网络编程基础	375
15.1 相关概念	331	17.1 网络基础知识	375
15.1.1 图形显示与适配器	331	17.1.1 什么是计算机网络	375
15.1.2 显示器的工作原理	332	17.1.2 网络协议	375
15.2 文本屏幕操作	333	17.1.3 协议分层	376
15.2.1 屏幕操作函数	333	17.1.4 网络参考模型	377
15.2.2 字符属性函数	336	17.1.5 端口	379
15.2.3 文本操作函数	338	17.2 Winsocket 基础	379
15.2.4 屏幕状态函数	341	17.2.1 套接字 (socket)	379
15.3 图形系统的初始化与关闭	343	17.2.2 基于 TCP 的 socket 编程	380
15.3.1 图形系统的初始化	343	17.2.3 基于 UDP 的 socket 编程	381
15.3.2 图形系统的关闭	346	17.3 Winsocket 的相关函数	381
15.4 图形屏幕管理及属性	346	17.3.1 WSASStartup 函数——启动 套接字库	381
15.4.1 图形屏幕管理	346	17.3.2 socket 函数——建立套 接字	382
15.4.2 设置图形属性	347	17.3.3 bind 函数——绑定本地 IP 地址和端口	383
		17.3.4 listen 函数——侦听客户端 请求	384

17.3.5	accept 函数——等待客户端的请求	384	18.5.2	数制转换	409
17.3.6	send 函数——发送数据	384	18.5.3	组合问题	411
17.3.7	recv 函数——接收数据	385	18.6	分治算法	412
17.3.8	connect 函数——建立连接	385	18.6.1	算法思想	412
17.3.9	recvfrom 函数——接收数据	385	18.6.2	求 n 个数的最大值和最小值	413
17.3.10	sendto 函数——发送数据	385	18.6.3	赛程安排问题	415
17.4	基于 TCP 的简单网络程序	386	18.7	贪心算法	418
17.4.1	服务器端应用程序的实现	386	18.7.1	算法思想	418
17.4.2	客户端应用程序的实现	389	18.7.2	加油站问题	419
17.5	基于 UDP 的简单网络聊天程序	391	18.7.3	找零钱问题	420
17.5.1	服务器端应用程序的实现	391	18.8	小结	422
17.5.2	客户端应用程序的实现	393	第 19 章	简单数据结构——队列和栈	424
17.6	小结	395	19.1	队列	424
第 18 章	常用算法设计	396	19.1.1	队列的定义	424
18.1	算法基础	396	19.1.2	队列的表示与实现	425
18.1.1	什么是算法及算法的描述语言	396	19.1.3	顺序循环队列	426
18.1.2	算法的特性	397	19.1.4	顺序循环队列的实现	427
18.1.3	算法设计的目标	398	19.1.5	链式队列的表示与实现	429
18.1.4	算法的时间复杂度和空间复杂度	398	19.1.6	队列的应用——商品货架模拟	432
18.2	迭代算法	399	19.2	栈	436
18.2.1	算法思想	399	19.2.1	栈的定义	436
18.2.2	求一个数的平方根	399	19.2.2	顺序栈的存储结构与实现	436
18.2.3	谷角猜想	400	19.2.3	链式栈的存储结构与实现	439
18.3	递推算法	401	19.2.4	栈的应用举例——算术表达式求值	441
18.3.1	认识递推	402	19.3	小结	445
18.3.2	斐波那契数列	402	第 20 章	常用技术——排序	446
18.3.3	分西瓜	404	20.1	排序的基础知识	446
18.3.4	该存多少钱	404	20.1.1	排序的相关概念	446
18.4	穷举算法	405	20.1.2	排序算法的分类	447
18.4.1	算法思想	405	20.2	插入类排序	447
18.4.2	完全数	406	20.2.1	直接插入排序	447
18.4.3	背包问题	407	20.2.2	折半插入排序	449
18.5	递归算法	409	20.2.3	希尔排序	450
18.5.1	算法思想	409	20.3	选择类排序	451
			20.3.1	简单选择排序	451
			20.3.2	堆排序	453

20.4 交换类排序.....	460	20.6 分配类排序.....	468
20.4.1 冒泡排序.....	460	20.6.1 基数排序的算法思想.....	468
20.4.2 快速排序.....	462	20.6.2 基数排序算法的实现.....	470
20.5 归并类排序.....	465	20.7 各种排序方法的比较.....	473
20.5.1 二路归并排序的算法思想.....	465	20.8 小结.....	474
20.5.2 二路归并排序算法的实现.....	465		

第1篇 起步篇

第1章 C语言基础

C语言是世界上非常流行、有着广泛前途的计算机高级语言。C语言从诞生之日起就深受人们的喜爱。正是因为C语言的广泛普及，才使得后来的编程语言都遵循或延续了C语言的习惯。因此，C语言自然成为几乎所有程序员的起步语言。本章主要介绍为什么要选择C语言、各种进制的转换及计算机中数的表示。

1.1 为什么要选择C语言

目前世界上流行的计算机编程语言有许多，如C，C++，Java，Pascal，Basic，Fortran。为什么要选择C语言呢？本节主要介绍选择C语言的好处、C语言的特点及如何学好C语言。

1.1.1 选择C语言的好处

C，C++，Java，Basic是目前非常流行的编程语言，几乎每一个程序员都有过学习C语言的经历。C语言对于初学者是一个非常不错的选择，因为选择C语言有如下好处。

(1) C语言是C++、Java、C#的基础。C++、Java、C#是目前最为流行的语言，这些语言都是建立在C语言基础之上的，当你学了C语言之后，再学习C++、Java或C#会变得非常容易。

(2) C语言涵盖了计算机编程语言中的几乎所有知识，对于今后学习其他编程语言和理解计算机系统的工作方式有很大的帮助。

(3) 使用C语言编写的程序运行效率高。C语言是一门高级语言，使用它编写的程序仅仅比汇编语言编写的程序的运行效率低10%~20%。

(4) C语言可以直接访问内存，并可以直接对硬件进行操作。同时，C语言要比汇编语言更加容易学习和使用。

(5) 过去许多系统都是使用C语言实现的，如Windows和Linux操作系统的大部分代码是用C语言编写的，它们都可以得到重新利用。

(6) 有了C语言基础，便可为今后继续学习数据结构和算法提供保障，因为目前绝大部分的数据结构和算法教材都是以C语言描述的。

(7) 使用C语言既可以编写系统软件，也可以编写应用软件。现在的游戏软件、嵌入式开发和手机开发都是选择C语言作为开发语言的。

(8) 使用C语言编写的程序有很好的移植性。在一种计算机系统（如IBM PC机）上编

写的 C 语言程序，可以直接在其他系统（如 DEC VAX 系统）上运行。

C 语言拥有如此多的优点，你也一定坚定了学习 C 语言的决心。接下来，就让我们来了解一下 C 语言的特点吧！

1.1.2 C 语言的特点

C 语言自从 1973 年诞生于贝尔实验室以来，经历了将近 40 年的历史。它现在仍然经久不衰、拥有广大的用户，这是与 C 语言自身的特点分不开的。

1. C 语言程序结构紧凑、简洁、规整，表达式简练、灵活——容易理解与学习

C 语言程序结构紧凑、简洁、规整，容易阅读与理解；其表达式简练，去除了一些不必要的成分，书写简单，使用起来比较灵活。因此，C 语言更加容易掌握。

2. C 语言的数据类型丰富——可以描述各种复杂的数据结构

C 语言的数据类型包括整型、实型、字符型、数组类型、指针类型、结构体类型、联合体类型等。其中，结构体类型和联合体类型是用户自定义的类型，即用户根据具体需要自己定义的类型。C 语言所提供的数据类型可以描述各种复杂的数据结构。

3. C 语言的运算符丰富——实现复杂的运算比较方便

C 语言包含了 34 种运算符，丰富的运算符与丰富的数据类型结合，构成了多样的表达式。灵活的运算符可以很容易实现比较复杂的运算。

4. C 语言是一种结构化的语言——拥有 3 种控制语句、函数作为程序的模块单元

C 语言是一种结构化的程序设计语言，适用于大型的模块化程序设计。它拥有 3 种控制语句，其函数是 C 语言程序的模块单元，每个函数各自独立。C 语言的源程序可以分为多个源文件，先对其分别进行编译，然后链接在一起便可构成可执行程序，这为大型的软件开发提供了方便。

5. C 语言可以对位进行操作，可以实现汇编语言的大部分功能——既是高级语言，也是低级语言

C 语言可以直接访问内存单元，直接对位一级进行操作，也就是说它可以实现汇编语言的功能。C 语言本身又是一门高级语言，但是它具备了汇编语言的许多功能，因此 C 语言既是高级语言，也是一门低级语言。正是用于 C 语言的这种双重特性，所以我们常常称 C 语言为中级语言。

6. C 语言的指针是它区别于其他语言的显著特性

C 语言有一种非常特殊的类型——指针。指针的存在，使得它可以直接访问内存。指针使得原本灵活多样的 C 语言变得更加灵活，使得编写出的程序的运行效率更加高效。

1.1.3 如何学好 C 语言

在选择了对 C 语言和对 C 语言的特点了解之后，想要学好 C 语言，需要把握好以下几点。

1. 确立离散性思维方式，摒弃连续性思维方式

在学习计算机语言时，一定要确立离散性的思维方式，这是决定你是否能够学好 C 语言的一个非常重要的因素，这是因为计算机中的数据存取形式是二进制形式，是一种离散的数据表

示方式。在处理类似连续性函数、积分等问题时，需要将问题转化为离散的方式进行处理。在学习 C 语言时，你会深刻地体会到这一点。

2. 熟练掌握二进制与十进制、十六进制、八进制之间的相互转换

在计算机中，所有的数据都是以二进制形式存储的，而我们熟悉的是十进制，再加上二进制数据表示起来又太长，因此为了方便表示，需要将二进制转换为十进制、十六进制、八进制，这样看起来就比较直观。

3. 理解字符与 ASCII 码之间的关系

通过键盘输入的数据是字符数据，而计算机是以二进制形式存储的，这需要将字符转换为对应的二进制形式并存放起来。美国的标准协会 ANSI 专门规定了字符与 ASCII 之间的对应关系。

4. 掌握运算符及运算符的优先级

C 语言提供了 34 种运算符，每种运算符都有优先级与结合性。如果有多个运算符出现在同一个表达式中，需要选择优先级别高的运算符进行计算。如果运算符相同，则需要根据运算符的结合性进行运算。

5. 掌握 3 种程序控制结构

C 语言是一种结构化的程序设计语言，它具有 3 种控制结构：顺序结构、选择结构和循环结构。使用这 3 种结构可以解决所有的问题。

6. 掌握一些常用的算法

在学习 C 语言的过程中，常常需要对一些数据进行排序及查找给定的数据，这就是排序算法和查找算法。排序算法和查找算法在程序设计过程中是非常常用的算法，排序算法可以分为冒泡排序、插入排序、选择排序等，查找算法可以分为顺序查找、折半查找等。掌握一些常用的算法对今后学习数据结构和算法是大有裨益的。

7. 熟练使用指针

指针是 C 语言区别于其他语言的一个重要标志。指针是 C 语言的灵魂，熟练使用指针可以使编写程序更加灵活，编写出来的程序运行效率也会更加高效。指针是一把双刃剑，使用得好则可以提高运行效率，使用得不当，则很容易造成难以意料的错误。因此，需要大家在学习的过程中熟练掌握指针。

8. 熟练掌握一个开发工具

要想学好一门语言，必须熟练掌握一个开发工具。只有多上机练习，才能知道程序是否正确。C 语言的开具有许多，目前比较流行的有 Turbo C 2.0, Turbo C 3.0, Visual C++ 6.0, Win-TC, LCC-Win32 等。我们建议初学者可以学习 Turbo C 2.0 或 Turbo C 3.0，有了基础之后可以选择 Visual C++ 6.0 (Visual C++ 6.0 是一个非常专业的开发工具)。

1.2 程序设计语言基础——进制转换

在计算机内部，信息的存储与处理都采用了二进制形式。二进制不容易阅读且需要用很长

的数码来表示一个数，为了表示方便与符合人们的阅读习惯，常常需要将二进制转换为十进制、十六进制、八进制。本节主要讲解二进制数与十进制数、十进制数与十六进制数、十进制数与八进制数，以及各种进制数的相互转换。

1.2.1 二进制与位权

二进制形式主要是由计算机内部结构决定的。计算机是由电子器件构成的，而计算机中的数据是用电子器件的物理状态表示的，其中高电平表示 1，低电平表示 0。

1. 二进制

二进制数中只有两个数字符号：0 和 1。例如，101、1110101、1101110 都是二进制。在计算机中，为了将二进制与十进制、十六进制、八进制区别开来，会在一个数的末尾加一个下标。例如，上面的二进制常常写成如下形式：

$$(101)_2、(1110101)_2、(1101110)_2$$

二进制的特点是逢二进一，即满二就向高位进一。例如，在二进制数中，有 $0+0=0$ ， $0+1=1$ ， $1+0=1$ ， $1+1=10$ 。需要注意，由于 $1+1$ 满 2 需要向高位进 1，所以是 10，而不是 2。对于 $10+11=101$ ，其运算过程如图 1.1 所示。

2. 位权

二进制数与十进制数一样，每一个数字符号在不同的位上表示的意义不同。例如，十进制数的 1 在个位上表示 1，在十位上表示 10，在百位上表示 100，在千位上表示 1000。这种同一个数字符号在不同位上的数值表示就是位权或权。一个 n 位的十进制整数从低位到高位对应的位权分别是 10^0 ， 10^1 ， 10^2 ， \dots ， 10^{n-1} 。对于带小数点的数，小数点后的位权分别是 10^{-1} 、 10^{-2} 、 $10^{-3}\dots$ 。例如，十进制数 2813.65 对应的位权如图 1.2 所示。

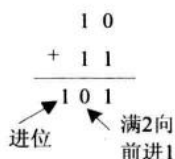


图 1.1 10 和 11 的运算过程示意图

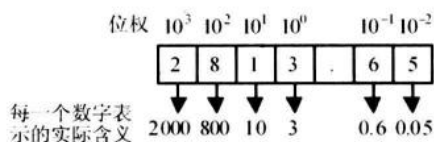


图 1.2 2813.65 对应的位权示意图

同理，一个 n 位的二进制整数的位权从低位到高位依次是 2^0 、 2^1 、 2^2 、 \dots 、 2^{n-1} 。对于带小数点的二进制数，小数点后的各位数对应的位权依次是 2^{-1} ， 2^{-2} ， 2^{-3} ， \dots ，小数点前面的各位数对应的位权是 2^0 、 2^1 、 2^2 、 \dots 。例如，二进制数 $(1011101.101)_2$ 的位权如图 1.3 所示。

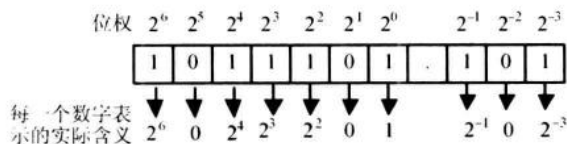


图 1.3 $(1011101.101)_2$ 对应的位权示意图

1.2.2 二进制数与十进制数的相互转换

二进制数与十进制数的相互转换分为二进制数转换为十进制数、十进制数转换为二进制数。

1. 二进制数转换为十进制数

二进制数转换为十进制数的方法比较简单，只需要将二进制数每一位上的数码与对应的位权相乘并求和，即可得到对应的十进制数。例如，二进制数 $(101110)_2$ 和 $(1011.101)_2$ 转换为十进制数的过程如下：

$$(101110)_2 = 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = (22)_{10}$$

$$(1011.101)_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} = (11.625)_{10}$$

这里，我们使用下标 10 表示该数是十进制数。从上面可以看出，要将一个二进制数转换为对应的十进制数，只需要将整数部分与小数部分分别转换即可。

2. 十进制数转换二进制数

十进制数转换为二进制数分为 3 种情况：十进制整数转换为对应的二进制整数、十进制小数转换为对应的二进制小数、一般的十进制数转换为二进制数。

1) 十进制整数转换为二进制整数

十进制整数转换为对应的二进制整数有两种方法：除 2 取余法和降幂法。

(1) 十进制整数转换为二进制整数——除 2 取余法。

下面先介绍除 2 取余法。所谓除 2 取余法，就是将一个十进制整数除以 2，得到一个商和余数，并记下这个余数。然后再将商作为被除数除以 2，得到一个商和余数，并记下这个余数。继续不断重复以上过程，直到商为 0 为止。每次得到的余数（0 和 1）分别是对应二进制整数的低位到高位上的数字。例如，十进制整数 86 转换为对应二进制整数的过程如图 1.4 所示。

(2) 十进制整数转换为二进制整数——降幂法。

所谓降幂法，就是将十进制整数不断地减去与该整数最接近的二进制整数的位权，如果够减，则对应的二进制位上的数字应为 1。否则，对应的二进制位上应为 0。得到的差值作为新的被减数重新进行下一次计算，直到被减数为 0 为止。这样就得到了对应的二进制整数。例如，十进制整数 93 转换为对应的二进制整数的过程如图 1.5 所示。

	86	余数
2	43	0 即 $a_0=0$
2	21	1 即 $a_1=1$
2	10	1 即 $a_2=1$
2	5	0 即 $a_3=0$
2	2	1 即 $a_4=1$
2	1	0 即 $a_5=0$
	0	1 即 $a_6=1$ 商为 0，结束

$$(86)_{10} = (a_6 a_5 a_4 a_3 a_2 a_1 a_0)_2 = (1011010)_2$$

图 1.4 使用除 2 取余法将十进制整数转换为二进制整数的过程示意图

	93	操作	二进制位
	$93 - 2^6 = 93 - 64 = 29$	够减, $a_6=1$	1
	$29 < 2^5 = 32$	不够减, $a_5=0$	0
	$29 - 2^4 = 29 - 16 = 13$	够减, $a_4=1$	1
	$13 - 2^3 = 13 - 8 = 5$	够减, $a_3=1$	1
	$5 - 2^2 = 5 - 4 = 1$	够减, $a_2=1$	1
	$1 < 2^1 = 2$	不够减, $a_1=0$	0
	$1 - 2^0 = 1 - 1 = 0$	够减, $a_0=1$	1

$$(93)_{10} = (a_6 a_5 a_4 a_3 a_2 a_1 a_0)_2 = (1011101)_2$$

图 1.5 使用降幂法将十进制整数转换为二进制整数的过程示意图

首先将 93 减去与之最为接近的二进制的位权，即 $2^6 (=64)$ ，得到一个差值 29，因为够减，