



Microsoft C 自学读本

Microsoft C

自学读本

程文斌 金相凤等 编

适用的语言范围:

- Microsoft C
- Microsoft C/C++7.0
- Visual C++各种版本
- Quick C

北京航空航天大学

TP312



北京航空航天大学出版社

Microsoft C 自学读本

程文斌 金相风等 编

适用的语言范围

- Microsoft C
- Microsoft C/C++ 7.0
- Visual C++ 各种版本
- Quick C



北京航空航天大学出版社

图书在版编目(CIP)数据

JS182/20

Microsoft C 自学读本/程文斌等编著. —北京: 北京航空航天大学出版社, 1995. 9

ISBN 7-81012-599-0

I. M… I. 程… II. C 语言-基本知识 IV. TP312C

中国版本图书馆 CIP 数据核字(95)第 10778 号

内 容 简 介

本书是针对初学 Microsoft C 进行 C 程序设计的人编写的。首先介绍 Microsoft 语言的基础知识,包括 C 程序结构、函数、流程控制、数据类型、操作符和程序设计易犯的错误;接着讨论输入和输出、动态存储分配、图形、字体和内部汇编;最后给出了 C 语言的参考和库函数参考。叙述由浅入深,言简意赅,示例丰富,是自学 Microsoft C 语言的一本好书。

● 书 名:Microsoft C 自学读本

Microsoft C ZIXUE DUBEN

● 编 著 者:程文斌 金相凤等 编

● 责任编辑:张躬行 许传安

● 出 版 者:北京航空航天大学出版社

● 地 址:北京市学院路 37 号(邮编 100083,发行科电话 2015720)

● 印 刷 者:朝阳科普印刷厂印刷

● 发 行 者:新华书店科技发行所

● 经 售:全国各地书店

● 开 本:787×1092 1/16

● 印 张:18.5

● 字 数:469 千字

● 印 数:5000 册

● 版 次:1995 年 8 月第 1 版

● 印 次:1995 年 8 月第 1 次印刷

● 书 号:ISBN 7-81012-599-0/TP·180

● 定 价:23.00 元



前 言

C 语言是 1972 年由贝尔实验室的 Dennis Ritchie 发明的。它是在另外一门语言,即 B 语言的基础上发展来的,后才由 Ken Thompson 提出。而 B 又是另外一门语言——BCPL 的演化,BCPL 由 Martin Richards 提出,用于为 DEC PDP-7 开发第一版 UNIX 系统。Dennis Ritchie 提出 C 语言的起因是为 DEC PDP-11 计算机开发 UNIX 系统。初始的 UNIX PDP-11 蓝本用汇编语言编写,以后使用 C 语言编写。此后的 UNIX 版本均用 C 编写。不久,C 就取代汇编语言成为 UNIX 环境程序员的首选编程语言。随着 C 越来越流行,大量的应用程序也采用 C 来编写。

很快,C 语言为很多不同的计算机和不同的操作系统所采纳,变化形式随之多了起来。美国国家标准学会(ANSI)——负责定义编程标准的计算机工业专业机构,成立了一个委员会以实现 C 语言的标准化。当一门编程语言标准化之后,以该语言编写的任何程序就能在不同的计算机和操作系统上使用,这样的程序具有可移植性。这个委员会给出了 ANSI 标准 C,它现在已受到绝大多数的 C 语言编译器厂商的支持。

Microsoft C 编译器就是 ANSI 标准 C。有很多厂商开发为 IBM PC 机所用的 C 编译器。Microsoft 公司从 1981 年起开始开发 C 编译器。当时 IBM PC 机才刚进入市场。许多商业应用程序,尤其是那些为 Windows 和 OS/2 所编写的,都用 Microsoft C 编译器开发。

C 语言具有如下主要优点:

- C 是一门使用灵活、功能强大的语言。它的命令、操作符和函数库能用于编写绝大多数类型的计算机程序。
- C 被职业程序员用来为大多数计算机系统(包括 IBM PC 及其兼容机)开发软件。
- C 还可用于开发操作系统、编译器、实时系统和商业应用程序。Microsoft Windows, OS/2, dBase, Paradox, Microsoft Word 和 FoxBase Pro 都是用 C 编写的,绝大多数的 C 编译器也是用 C 编写的。
- 为某一类型计算机编写的一个 C 程序,略经修改就可在另一类型的机器上使用。这就是 C 的可移植性。可移植性这一点很重要,因为大多数的现代计算机都有 C 编译器。一旦用户已学会 C,那么,在为一种新型机器编写程序时就不必去另学一门语言,同一个程序能在不同机器上运行,就没有必要重新编写程序。

C 的特色在于其运行速度。在早期的计算中,执行效率的问题是通过用汇编语言重新编写程序的全部或部分代码解决的。当时的各种程序只能在专门的某类机器上运行,如果一家公司换了机器类型,就得花财力和精力去为新机器重新编写程序。而现在,如果一个用 BASIC 或 dBase 解释编写的程序运行速度缓慢,那么用 C 语言改写这个程序就能加快执行速度。

由于用 C 编写程序变得非常普遍,专业程序员已开发了许多 C 的库,用以支持应用程序的开发。另外,还有很多商用 C 语言库具有支持数据库操作、图形、文本编辑、通讯等功能。

像大多数的现代计算机语言一样,C 语言是过程语言或者说是面向动作的语言。也就是

说在开发的过程中,程序员要把注意力放在程序对数据所执行的动作上。程序员决定程序需要哪种算法。

本书内容如下:

- 第一部分“学习 C”:包括了诸如函数及数据类型这样的基本问题。读者对本部分各章应按顺序从头到尾阅读。
- 第二部分“使用 C”:包括了诸如输入/输出和图形学这样的实际编程问题。本部分是按每个问题而组织的,不必按照任何特定顺序来进行阅读。
- 附录 A“C 语言指南”:总结了 C 语言的 Microsoft C 实现。你可在读完第一部分并对 C 有些了解之后将此附录作为一快速参考。
- 附录 B“C 库函数指南”:总结了 Microsoft C 运行库中最常使用的函数。该附录主要是当你还没有使用 Microsoft C 时,作为浏览而用的。当在 Quick 环境中,可使用 Microsoft C Advisor(联机帮助)来得到有关 C 语言性质及运行库函数的信息。

本书所描述的 C 语言内容适用于 Microsoft C 6.0、Microsoft C/C++ 7.0、Visual C++ 1.0、Visual C++ 1.5 和 Visual C++ 2.X。自学使用上述软件的读者,可阅读本书。

目 录

第一部分 学习 C

第一章 C 程序解析	3
1.1 一个典型的 C 程序	3
1.2 注释	3
1.3 语句	4
1.4 关键字和名字	4
1.5 预处理指令	4
1.6 函数	4
1.7 调用函数	5
1.8 声明及初始化变量	5
1.9 外部变量与局部变量	6
1.10 函数原型	6
1.11 关于 printf	6
第二章 函数	8
2.1 函数和结构化程序设计	8
2.2 main 函数	8
2.3 函数的位置与可见性	9
2.4 函数定义和原型	10
2.5 调用一个函数	10
2.6 传递变元	11
2.7 变元与参数	13
2.8 给参数赋值	13
2.9 值传递	14
2.10 从函数中返回值	15
2.11 使用返回值	17
2.12 声明函数的返回类型	17
2.13 函数原型	18
2.13.1 原型化无参数的函数	18
2.13.2 原型化有不定参数的函数	19
2.14 旧式的函数声明与定义	19
第三章 流程控制	21
3.1 循环: while, do 和 for	21
3.1.1 while 语句	21
3.1.2 do 语句	22

3.1.3	for 循环	24
3.2	判断语句:if,else,switch,break,continue 和 goto	26
3.2.1	if 语句	26
3.2.2	else 子句	27
3.2.3	switch 语句	28
3.2.4	break 语句	31
3.2.5	continue 语句	32
3.2.6	goto 语句	33
第四章	数据类型	35
4.1	基本数据类型	35
4.1.1	说明基本类型	35
4.1.2	说明变量	37
4.1.3	说明常量	37
4.2	聚集数据类型	39
4.2.1	数组	39
4.2.2	结构	44
4.2.3	联合	51
第五章	高级数据类型	53
5.1	可见性	53
5.1.1	局部变量	53
5.1.2	外部变量	55
5.1.3	多个源文件中的可见性	56
5.1.4	函数的可见性	57
5.2	生命期	57
5.3	转换数据类型	59
5.3.1	数据类型的级别	59
5.3.2	升级与降级	60
5.3.3	自动类型转换	60
5.3.4	强制类型转换	62
5.4	寄存器变量	63
5.5	用 typedef 对现存类型重命名	63
5.6	枚举类型	64
第六章	操作符	66
6.1	C 操作符介绍	66
6.1.1	算术操作符	66
6.1.2	关系操作符	66
6.1.3	赋值操作符	67
6.2	C 语言独特的操作符	68
6.2.1	加 1 和减 1 符	68

6.2.2	位操作符	69
6.2.3	逻辑操作符	70
6.2.4	地址操作符	71
6.2.5	条件操作符	72
6.2.6	sizeof 操作符	72
6.2.7	逗号操作符	72
6.2.8	基数操作符	73
6.3	操作符优先级	73
第七章	预处理指令	75
7.1	#include 指令	75
7.2	#define 和 #undef 指令	76
7.2.1	简单文本替换	77
7.2.2	与函数相似的宏	77
7.2.3	#undef 指令	78
7.3	条件指令	78
7.4	编译指令	80
第八章	指针	81
8.1	使用指针	81
8.2	指向简单变量的指针	81
8.2.1	声明指针变量	82
8.2.2	指针存储方式	83
8.2.3	初始化指针变量	83
8.2.4	使用指针变量	84
8.2.5	指针基础知识小结	84
8.3	数组指针	85
8.3.1	数组及指针运算	86
8.3.2	指针比较	87
8.3.3	重温 PARRAY.C	87
8.4	指针和串	88
8.5	传递指针至函数	90
8.6	指针数组	92
8.7	本章回顾	96
第九章	高级指针	97
9.1	指向指针的指针	97
9.2	数组与指针的等价性	98
9.3	获取命令行参数	101
9.4	空指针	102
9.5	指向结构的指针	102
9.6	指向函数的指针	105

9.7 有关指针的几点说明	107
第十章 程序设计中的错误	108
10.1 操作问题	108
10.1.1 赋值操作符和相等操作符的混淆	108
10.1.2 混淆操作符优先级	109
10.1.3 混淆结构元素操作符	109
10.2 数组问题	110
10.2.1 数组下标错	110
10.2.2 在处理数组时忽略了数组下标	110
10.2.3 超出数组边界	111
10.3 字符串问题	111
10.3.1 混淆字符常数与字符串	111
10.3.2 忘记了字符串结尾的空字符	112
10.3.3 忘记为字符串分配内存	112
10.4 指针问题	113
10.4.1 用错误的地址操作符来初始化指针	113
10.4.2 用错误的类型声明了一个指针	114
10.4.3 使用悬挂指针	114
10.5 库函数问题	115
10.5.1 没有检查库函数的返回值	115
10.5.2 与库函数重名	116
10.5.3 忘记包含声明库函数的包含文件	116
10.5.4 调用 scanf 时忽略了地址操作符	117
10.6 宏问题	117
10.6.1 忽略了宏参数的括号	118
10.6.2 在宏参数中使用递增和递减操作符	118
10.7 混合性问题	120
10.7.1 不匹配的 if 和 else 语句	120
10.7.2 放错了分号的位置	121
10.7.3 在 DOS 路径中忽略了双反斜杠	122
10.7.4 忽略了开关的语句中的 break 语句	122
10.7.5 混淆有符号和无符号值	123

第二部分 使用 C

第十一章 输入和输出	127
11.1 输入输出流	127
11.2 屏幕和键盘输入/输出	127
11.2.1 操作和打印字符串	127
11.2.2 打印数值	131

11.2.3	使用 scanf 进行键盘输入	134
11.3	标准磁盘输入/输出	136
11.3.1	创建文件和向文本文件写	136
11.3.2	打开文件以便二进制方式读	139
11.3.3	二进制和文本文件	140
11.3.4	数值型变量的文本格式	141
11.3.5	使用二进制格式	145
11.4	低级输入和输出	148
第十二章	动态存储分配	152
12.1	为什么要分配	152
12.2	存储分配基础	152
12.2.1	分配存储块的准备工作	154
12.2.2	指定分配块的大小	155
12.2.3	图示说明	155
12.2.4	对 malloc 返回地址赋值	156
12.2.5	检测 malloc 函数返回值	156
12.2.6	访问—已分配存块	157
12.2.7	为不同的数据类型分配存储块	158
12.2.8	采用 free 函数释放存储块	159
12.3	特定的存储分配函数	159
12.3.1	calloc 函数	159
12.3.2	realloc 函数	160
12.4	避免麻烦	160
第十三章	图形	162
13.1	图形方式	162
13.1.1	检测当前视频模式	162
13.1.2	设置视频方式	163
13.1.3	编制图形程序	164
13.1.4	使用彩色图形方式	169
13.1.5	使用彩色视频文本方式	177
13.2	文本坐标	178
13.3	图形坐标	179
13.3.1	物理屏幕	179
13.3.2	视口坐标	181
13.3.3	窗口中的实际坐标	181
第十四章	报告用图形	188
14.1	术语	188
14.2	报告用图形程序结构	190
14.3	五个图形程序例子	191

14.4	色板	200
14.4.1	颜色库	201
14.4.2	格式库	201
14.4.3	模式库	202
14.4.4	字符库	203
14.5	通用报告用图形	204
14.5.1	图形环境	204
14.5.2	titletype	204
14.5.3	axistype	205
14.5.4	窗口类型	207
14.5.5	legendtype	208
14.5.6	chartenv	209
14.6	报告用图形函数概述	210
第十五章	字体	212
15.1	Microsoft C 字体	212
15.2	使用 Microsoft C 字体库	213
15.2.1	注册字体	213
15.2.2	设置当前字体	214
15.2.3	显示文本	215
15.3	例题程序	215
15.4	一些提示	218
第十六章	内部汇编	219
16.1	内部汇编的优点	219
16.2	_asm 关键字	219
16.3	在 _asm 块中使用汇编语言	220
16.3.1	指令集	220
16.3.2	表达式	220
16.3.3	Data 伪指令及操作符	220
16.3.4	EVEN 和 ALIGN 伪指令	220
16.3.5	宏	221
16.3.6	段访问	221
16.3.7	类型和变量大小	221
16.3.8	注释	221
16.3.9	用 CodeView 调试器来调试	221
16.4	在 _asm 块中使用 C	222
16.4.1	使用操作符	222
16.4.2	使用 C 符号	222
16.4.3	访问 C 数据	223
16.4.4	编写函数	223

16.5	使用和保存寄存器	225
16.6	跳转到标号	225
16.7	调用 C 函数	226
16.8	定义 <code>_asm</code> 块为 C 宏	227
16.9	优化	228

第三部分 附录

附录 A	C 语言指南	231
A.1	一般语法	231
A.1.1	用户定义名	231
A.1.2	关键字	231
A.1.3	函数	232
A.2	流程控制	232
A.2.1	break 语句	232
A.2.2	continue 语句	233
A.2.3	do 语句	233
A.2.4	for 语句	233
A.2.5	goto 语句	234
A.2.6	if 语句	234
A.2.7	return 语句	235
A.2.8	switch 语句	235
A.2.9	while 语句	235
A.3	数据类型	236
A.3.1	基本数据类型	236
A.3.2	聚合数据类型	238
A.3.3	高级数据类型	239
A.4	操作符	240
A.5	预处理指令	242
A.6	指针	243
附录 B	C 库函数指南	244
B.1	C 运行库概况	244
B.2	缓冲区处理例程	245
B.3	字符分类与转换例程	246
B.4	数据转换例程	247
B.5	错误信息的例程	248
B.6	图形 1: 低级图形例程	249
B.6.1	配置方式与环境	250
B.6.2	设置坐标	250
B.6.3	设置色板	252

B. 6.4	设置属性	253
B. 6.5	输出图像	254
B. 6.6	输出文本	257
B. 6.7	传递图像	258
B. 7	图形 2: 报告用图形例程	259
B. 8	图形 3: 字体显示例程	261
B. 9	输入、输出例程	262
B. 9.1	流式例程	263
B. 9.2	低级例程	268
B. 9.3	控制台与端口 I/O 例程	269
B. 10	数学例程	271
B. 11	内存分配例程	274
B. 12	进程控制例程	275
B. 13	搜索与排序例程	276
B. 14	字符串处理例程	277
B. 15	时间例程	280

第一部分

学习 C

第一章	C 程序解析	3
第二章	函数	8
第三章	流程控制	21
第四章	数据类型	35
第五章	高级数据类型	53
第六章	操作符	66
第七章	预处理指令	75
第八章	指针	81
第九章	高级指针	97
第十章	程序设计中的错误	108



第一章 C 程序解析

作为一个有经验的程序员,可能会想立即投身于 C,但在此之前,应该了解所有 C 程序的基本模型。本章剖析了一个 C 程序,但没有涉及规则的正式定义也不讨论此例之外的有关内容。

这里的讨论围绕一个不大的名为 VOLUME.C 的典型 C 程序。为习惯 C 程序的形式以及组成它们的基本成分,在阅读时你应常常参考 VOLUME.C。

1.1 一个典型的 C 程序

VOLUME.C 是一个计算球体体积并打印下列结果于屏幕上的简单程序:

Volume: 113.040001

像本书中所有的典型程序一样,VOLUME.C 也在 Microsoft C 的联机帮助中。图 1.1 说明了 VOLUME.C 程序。

编译指令	注释——	/* VOLUME.C; Calculate sphere's volume. */
	装 stdio.h 文件——	#include <stdio.h>
	定义符号常量 PI ——	#define PI 3.14
	函数原型——	float sphere(int rad);
		main()
主函数定义	局部变量声明—— [{ float volume; int radius = 3;
	函数调用—— [volume = sphere(radius); printf("Volume: %f\n", volume);
		}
		float sphere(int rad)
sphere 函数定义	局部变量声明——	{ float result;
	返回值——	result = rad * rad * rad; result = 4 * PI * result; result = result /3; return result; }

图 1.1 VOLUME.C 程序

1.2 注释

VOLUME.C 的第一行是一注释:

```
/* VOLUME.C; Calculate sphere's volume. */
```


在 C 中,注释以 /* 开始,以 */ 结束。由于 C 只有很少的关键字,注释在使程序可读性好这一方面就起了重要作用。

注释不可以嵌套(将一个注释放在另一个注释里面)。下面这行含有语法错误:

```
/* Error! /* You can't */ nest comments in C. */
```

1.3 语句

C 语句总是以分号结束,例如 VOLUME.C 中的语句:

```
result = 4 * PI * result;
```

还可以用花括号将一组语句括起来以形式一个“语句块”。语句块含相关语句,诸如一函数体内的语句。

C 语言忽略源程序中的空格,所以你几乎可以以任意风格来安排程序。但是通常,典型的 C 程序中是每行一个语句,花括号垂直对齐,其中的语句要缩进。

1.4 关键字和名字

C 是一种对大小写敏感的语言(即区别大写字母与小写字母)。所有 C 的关键字都用小写给出;联机帮助中含有 C 关键字的一个完整集合。

你可以以大小写的任意组合来声明名字,但是许多程序员更喜欢用小写来表示变量和函数名,用大写来声明符号常量(一个符号常量是一表示常数的描述名。在 VOLUME.C 中,PI 就是符号常量)。

1.5 预处理指令

一个程序中并非每行语句都是可执行的:程序可以包含“预处理指令”——用于 Microsoft C 编译器的命令。这样的指令以 # 开始,结尾处不加分号。

VOLUME.C 的第二、三行就是预处理指令。#include 指令告诉 Microsoft C 在编译 VOLUME.C 时插进文件 stdio.h:

```
#include <stdio.h>
```

STDIO.H 是 Microsoft C 提供的许多包含文件中的一个。包含文件含有为 C 库函数所需的声明及定义(“库函数”是和 Microsoft C 一起提供而非由用户编写的)。在程序 VOLUME.C 中,printf 库函数需要来自 stdio.h 包含文件中的信息。

#define 指令定义一名为 PI 的符号常量:

```
#define PI 3.14
```

这样,每当 PI 在程序后面的出现时,Microsoft C 就用 3.14 代替。符号常量可以是字母、数字或其它字符的任意组合。

1.6 函数

函数是 C 的基本组块:每个 C 程序都至少有一个函数,每个可执行的 C 语句都出现在