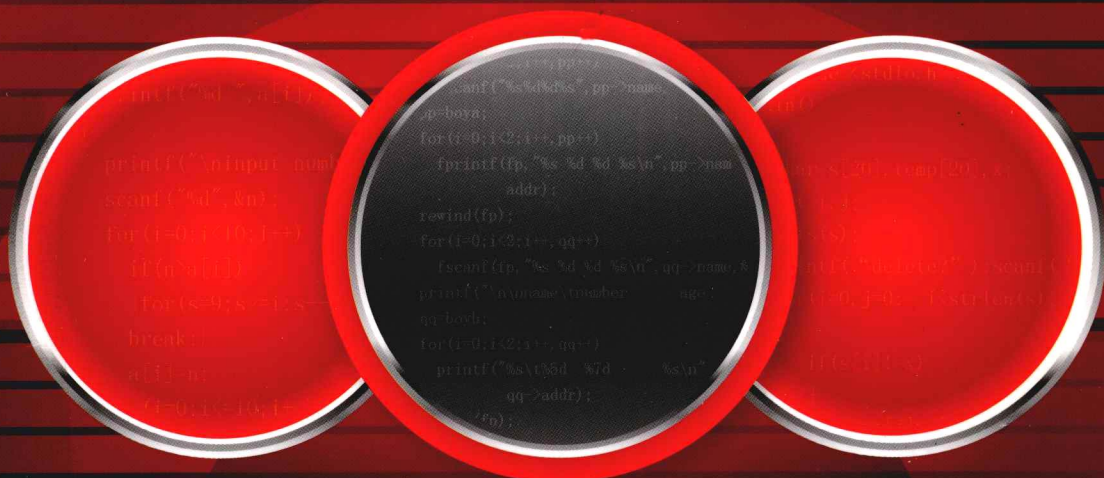


普通高等教育“十二五”规划教材

C语言程序设计教程

张思卿 谭晓玲 主编



化学工业出版社

普通高等教育“十二五”规划教材

C 语言程序设计教程

张思卿 谭晓玲 主 编

赵 娟 宋 斌 副主编



化学工业出版社

· 北 京 ·

本书基于 Visual C++ 6.0 平台,通过大量的实例详细讲解了 C 语言的语法规则及结构化程序设计的思想和方法,旨在培养学生程序设计能力并能解决某些实际问题,着重加强程序设计的思维方式和算法设计、分析能力的培养。本书对 C 语言的精华部分做了较为细致的介绍,还针对目前高等院校和社会上举办的各种计算机的等级考试,精心组织教学内容。

本书共 13 章,其主要内容包括 C 语言概述,基本数据类型和运算,顺序结构程序设计,运算符和表达式,循环结构程序设计,数组,函数,编译预处理,指针,结构体与共用体,位运算,文件,C 程序中的常见错误等。

本书适用于计算机类专业的本科生、高职高专、专升本的学生,也可以作为大学各专业公共教材和全国计算机等级考试的 C 语言参考书,同时也很适合广大计算机程序设计者学习参考。

图书在版编目(CIP)数据

C 语言程序设计教程 / 张思卿,谭晓玲主编. —北京:化学工业出版社,2012.6
普通高等教育“十二五”规划教材
ISBN 978-7-122-14187-3

I. C… II. ①张… ②谭… 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(2012)第 087591 号

责任编辑:王听讲

装帧设计:关 飞

责任校对:宋 夏

出版发行:化学工业出版社(北京市东城区青年湖南街 13 号 邮政编码 100011)

印 装:化学工业出版社印刷厂

787mm×1092mm 1/16 印张 14³/₄ 字数 364 千字 2012 年 8 月北京第 1 版第 1 次印刷

购书咨询:010-64518888(传真:010-64519686) 售后服务:010-64518899

网 址: <http://www.cip.com.cn>

凡购买本书,如有缺损质量问题,本社销售中心负责调换。

定 价: 30.00 元

版权所有 违者必究

前 言

本书根据高等院校计算机基础教学改革的需要,结合编者多年讲授 C 语言程序设计课程的教学经验编写而成。全书基于 Visual C++ 6.0 平台,通过大量的实例详细讲解了 C 语言的语法规则及结构化程序设计的思想和方法,旨在培养学生程序设计能力并能解决某些实际问题,着重加强程序设计思维方式和算法设计、分析能力的培养。

本书共 13 章,其主要内容包括 C 语言概述,基本数据类型和运算,顺序结构程序设计,运算符和表达式,循环结构程序设计,数组,函数,编译预处理,指针,结构体与共用体,位运算,文件,C 程序中的常见错误等。本书具有如下特点。

(1) 内容全面,编排合理。本书几乎涵盖了 C 语言的全部知识点,包括位运算等,能够满足各种专业学生的学习需求。书中内容的组织较为合理,包括前后内容的衔接、实例与理论的结合、理论与应用的结合等。

(2) 表述通俗易懂。本书力求概念准确、表述清楚、语言简洁,根据需要设计了相应的图、表,在程序中的关键位置添加了注释,便于读者的学习。

(3) 实践性强。学习程序设计语言需要大量的实践。本书配备了丰富的实例和习题,覆盖面广,综合性强,难度适宜。这些资源能够帮助读者逐步掌握 C 语言,并培养和提升读者的程序设计能力。

(4) 基于 Visual C++ 6.0 平台。现有的 C 语言类教材大都以 Turbo C 为平台,给教师教学、学生学习都造成了一定的困扰。本书在编写时以 Visual C++ 6.0 为平台,这与全国计算机等级考试的要求也是一致的。

与本书同时配套出版的《C 语言程序设计实训指导与习题解答》,提供了上机实验和每章习题与解答的内容,同时为配合全国计算机等级考试二级 C 语言笔试,还提供了近年来的部分真题与解析。我们还将为使用本书的教师免费提供电子教案和教学资源,需要者可以到化学工业出版社的教学资源网站 <http://www.cipedu.com.cn> 免费下载使用。

本书由郑州科技学院张思卿、重庆三峡学院谭晓玲担任主编,天津青年职业技术学院赵娟、南京理工大学宋斌担任副主编。具体编写人员分工为:张思卿编写第 4、5、7 章,谭晓玲编写第 3、6 章,宋斌编写第 9 章,赵娟编写第 11 章,郑州科技学院周丽编写第 8、10 章,郑州科技学院郑睿编写第 1、2 章,郑州科技学院李志伟编写第 13 章、附录,甘肃建筑职业技术学院张润花编写第 12 章,本书最后由张思卿统稿。

在本书的编写过程中,参考了其他大量的同类教材和网络上的相关资源,在此向其作者表示衷心的感谢。由于作者水平有限,加上编写时间仓促,书中难免会有不妥之处,殷切地希望广大读者提出宝贵意见。

编 者
2012 年 4 月

目 录

第1章 C语言概述	1
1.1 C语言的发展和特点	1
1.1.1 C语言的发展	1
1.1.2 C语言的特点	1
1.1.3 面向对象的程序设计语言	2
1.2 C语言程序的结构与书写规则	2
1.2.1 C语言程序的总体结构	2
1.2.2 函数的一般结构	3
1.2.3 源程序书写格式	5
1.3 C语言的语句和关键字	5
1.3.1 C语言的语句	5
1.3.2 关键字	6
本章小结	6
第2章 基本数据类型和运算	7
2.1 标识符	7
2.2 基本数据类型	8
2.3 变量和常量	9
2.4 运算符和表达式	13
2.5 不同类型数据的混合运算	18
本章小结	20
第3章 顺序结构程序设计	21
3.1 基本的数据输入输出	22
3.1.1 数据输入	22
3.1.2 数据输出	25
3.2 C程序的基本结构及书写规则	28
3.3 顺序结构程序设计	29
本章小结	30
第4章 运算符和表达式	31
4.1 关系运算符和关系表达式	31
4.1.1 关系运算符	31
4.1.2 关系表达式	31
4.2 逻辑运算符和逻辑表达式	32
4.2.1 逻辑运算符	32
4.2.2 逻辑表达式	32
4.3 语句和复合语句	33
4.4 分支结构	33
4.4.1 if语句	33
4.4.2 switch语句	36
4.4.3 条件运算符	38

4.5 程序设计举例	38
本章小结	41
第5章 循环结构程序设计	42
5.1 为什么要有循环结构	42
5.1.1 如何重复输出多个 hello world	42
5.1.2 如何求连续的 n 个自然数的和	43
5.2 “当”型循环	44
5.3 “直到”型循环	46
5.4 for 循环	47
5.5 多重循环结构的嵌套	49
5.6 break 语句与 continue 语句	52
5.6.1 break 语句	52
5.6.2 continue 语句	53
5.7 循环结构程序应用举例	53
本章小结	55
第6章 数组	56
6.1 一维数组	56
6.1.1 一维数组的定义方式	56
6.1.2 一维数组元素的引用	57
6.1.3 一维数组的初始化	58
6.1.4 一维数组程序举例	59
6.2 二维数组	60
6.2.1 二维数组的定义	60
6.2.2 二维数组元素的引用	60
6.2.3 二维数组的初始化	61
6.2.4 二维数组程序举例	62
6.3 字符数组与字符串	64
6.3.1 字符数组的定义	64
6.3.2 字符数组的初始化	64
6.3.3 引用字符数组	65
6.3.4 字符串和字符串结束标志	66
6.3.5 字符串的输入和输出	66
6.3.6 字符串常用函数	68
本章小结	78
第7章 函数	80
7.1 函数的概述	80
7.2 函数的定义	82
7.2.1 无参函数的定义	82
7.2.2 有参函数的定义	83
7.2.3 函数设计举例	84
7.3 函数的调用	85

7.3.1 函数的调用.....	85
7.3.2 函数原型说明.....	87
7.3.3 函数的定义、调用以及函数声明的比较.....	88
7.3.4 函数间数据传递.....	88
7.4 函数的嵌套与递归调用.....	90
7.4.1 函数的嵌套调用.....	90
7.4.2 函数的递归调用.....	91
7.5 数组作为函数的参数.....	95
7.5.1 数组元素作为函数的参数.....	95
7.5.2 数组名作为函数的参数.....	96
7.6 局部变量与全局变量.....	98
7.6.1 局部变量.....	98
7.6.2 全局变量.....	99
7.7 变量的存储类别.....	100
7.7.1 自动变量 (auto).....	101
7.7.2 静态变量 (static).....	101
7.7.3 外部变量 (extern).....	102
7.7.4 寄存器变量 (register).....	103
7.8 内部函数和外部函数.....	103
本章小结.....	103
第8章 编译预处理	105
8.1 宏定义.....	105
8.1.1 无参数的宏定义.....	106
8.1.2 带参数的宏定义.....	108
8.2 文件包含处理.....	114
8.3 条件编译.....	116
本章小结.....	120
第9章 指针	121
9.1 指针的概念.....	121
9.1.1 地址.....	121
9.1.2 指针.....	121
9.1.3 指针变量.....	122
9.2 指针变量的定义与运算.....	122
9.2.1 指针变量的定义.....	122
9.2.2 指针变量的运算.....	122
9.3 指针变量作函数参数.....	126
9.4 数组与指针.....	128
9.4.1 指向数组元素的指针.....	128
9.4.2 通过指针引用数组元素.....	128
9.4.3 指向二维数组的指针.....	131
9.4.4 数组名作函数参数.....	132

9.5 字符串与指针	133
9.5.1 字符串的表示形式	133
9.5.2 指向二维字符数组的指针	134
9.5.3 字符串指针作函数参数	135
9.6 返回指针值的函数	136
9.7 指针数组和指向指针的指针	137
9.7.1 指针数组	137
9.7.2 指向指针的指针	138
本章小结	139
第 10 章 结构体与共用体	142
10.1 概述	142
10.2 结构体类型与结构体类型变量的定义	143
10.2.1 结构体类型定义的一般形式	143
10.2.2 结构体变量的定义	144
10.2.3 结构体变量成员的引用	147
10.2.4 结构体变量的赋值	148
10.2.5 结构体变量的初始化	149
10.2.6 结构体数组的定义	150
10.2.7 指向结构体类型数据的指针	153
10.2.8 指向结构体变量的指针	153
10.2.9 指向结构体数组的指针	155
10.2.10 用结构体指针变量作函数参数	156
10.3 链表	157
10.3.1 动态分配存储空间	157
10.3.2 链表的概念	159
10.4 共用体	161
10.4.1 共用体类型、共用体类型变量的定义	161
10.4.2 共用体变量的引用	162
10.5 枚举类型	164
10.5.1 枚举类型的定义和枚举变量的说明	165
10.5.2 枚举类型变量的赋值和使用	165
10.6 自定义类型符 typedef	167
本章小结	168
第 11 章 位运算	169
11.1 什么是位运算	169
11.2 常见位运算	169
11.3 位域	171
11.3.1 位域的定义和位域变量的说明	171
11.3.2 位域的使用	172
本章小结	173

第 12 章 文件	175
12.1 文件的概念	175
12.2 文件指针	176
12.3 文件的打开与关闭	177
12.3.1 文件的打开 (fopen 函数)	177
12.3.2 文件关闭函数 (fclose 函数)	178
12.4 文件的读写	179
12.4.1 字符读写函数 fgetc (getc) 和 fputc (putc)	179
12.4.2 字符串读写函数 fgets 和 fputs	182
12.4.3 数据块读写函数 fread 和 fwrite	184
12.4.4 格式化读写函数 fscanf 和 fprintf	185
12.5 文件定位函数	187
12.5.1 fseek 函数	187
12.5.2 ftell 函数	188
12.5.3 rewind 函数	188
12.6 文件检测函数	189
12.7 C 库文件	190
本章小结	191
第 13 章 C 程序中常见错误分析	192
13.1 编译错误	193
13.2 连接错误分析	194
13.3 诊断错误及其处理	199
13.4 C 语言调试过程中常见的错误信息	202
13.4.1 致命错误	202
13.4.2 一般错误	202
13.4.3 警告	211
本章小结	214
附录	215
附录 1 C 语言中的运算符及优先级	215
附录 2 C 语言常用库函数	216
附录 3 常用字符与 ASCII 代码对照表	222
附录 4 VC++ 环境下的 C 程序调试	223
参考文献	228

第 1 章 C 语言概述

【教学提示】

本章对 C 语言的基本知识做了简单的介绍，后续章节还将对这些内容进行详细的讲解，通过本章的学习，使学生对 C 程序有个初步的了解，以利于后面章节的学习。

【技能目标】

- (1) 了解什么是计算机程序。
- (2) 了解 C 语言的发展、特点。
- (3) 掌握 C 语言程序的结构、规则。
- (4) 掌握 C 语言的关键字。

1.1 C 语言的发展和特点

1.1.1 C 语言的发展

C 语言是贝尔实验室于 20 世纪 70 年代初研制出来的，后来又被多次改进，并出现了多种版本。20 世纪 80 年代初，美国国家标准化协会（ANSI），根据 C 语言问世以来的各种版本对 C 语言的发展和扩充，制定了 ANSI C 标准（1989 年再次做了修订）。

早期的 C 语言主要是用于 UNIX 系统。由于 C 语言的强大功能和各方面的优点逐渐为人们认识，到了 20 世纪 80 年代，C 开始进入其他操作系统，并很快在各类大、中、小和微型计算机上得到了广泛的使用，成为当代最优秀的程序设计语言之一。

目前，流行的 C 语言有以下几种：

- Microsoft C 或称 MS C；
- Borland Turbo C 或称 Turbo C；
- AT&T C。

这些 C 语言的版本不仅实现了 ANSI C 标准，而且在此基础上各自做了一些扩充，使之更加方便、完美。

本书以 ANSI C 新标准来介绍 C 语言。

1.1.2 C 语言的特点

(1) C 语言简洁、紧凑，使用方便、灵活。ANSI C 一共只有 32 个关键字，9 种控制语句，程序书写自由，主要用小写字母表示，压缩了一切不必要的成分。

Turbo C 扩充了 11 个关键字：

asm _cs _ds _es _ss cdecl far
hugeinterrupt near pascal

注意：在 C 语言中，关键字都是小写的。

(2) 运算符丰富。共有 34 种。C 语言把括号、赋值、逗号等都作为运算符处理。从而使 C 语言的运算类型极为丰富，可以实现其他高级语言难以实现的运算。

(3) 数据结构类型丰富。

(4) 具有结构化的控制语句。

(5) 语法限制不太严格，程序设计自由度大。

(6) C 语言允许直接访问物理地址，能进行位 (bit) 操作，能实现汇编语言的大部分功能，可以直接对硬件进行操作。因此有人把它称为中级语言。

(7) 生成目标代码质量高，程序执行效率高。

(8) 与汇编语言相比，用 C 语言编写的程序可移植性好。

但是，C 语言对程序员要求也高，程序员用 C 语言编写程序会感到限制少、灵活性大，功能强，但在学习上较其他高级语言要困难一些。

1.1.3 面向对象的程序设计语言

在 C 语言的基础上，1983 年又由贝尔实验室的 Bjarne Stroustrup 推出了 C++。C++ 进一步扩充和完善了 C 语言，使其成为一种面向对象的程序设计语言。C++ 目前流行的最新版本是 Borland C++4.5, Symantec C++6.1 和 Microsoft VisualC++ 2.0。C++ 提出了一些更为深入的概念，它所支持的这些面向对象的概念容易将问题空间直接地映射到程序空间，为程序员提供了一种与传统结构程序设计不同的思维方式和编程方法。因而也增加了整个语言的复杂性，掌握起来有一定难度。

C 语言是 C++ 的基础，C++ 和 C 语言在很多方面是兼容的。因此，掌握了 C 语言，再进一步学习 C++ 就能以一种熟悉的语法来学习面向对象的语言，从而达到事半功倍的目的。

1.2 C 语言程序的结构与书写规则

1.2.1 C 语言程序的总体结构

一个完整的 C 语言程序，是由一个 main() 函数（又称主函数）和若干个其他函数结合而成的，或仅由一个 main() 函数构成。

【例 1.1】 仅由 main() 函数构成的 C 语言程序。

/*功能：仅由 main() 函数构成的 C 语言程序示例*/

```
main()
{
    printf("This is a C program.\n");
}
```

程序运行结果：

This is a C program.

【例 1.2】 由 main() 函数和 1 个其他函数 max() 构成的 C 语言程序。

/*功能：由 main() 函数和 1 个其他函数 max() 构成的 C 语言程序示例*/

```
int max(int x, int y)
{
```

```

        return( x>y ? x : y );
    }
main()
{
    int num1,num2;
    printf("Input the first integer number: ");
    scanf("%d", &num1);
    printf("Input the second integer number: ");
    scanf("%d", &num2);
    printf("max=%d\n", max(num1, num2));
}

```

程序运行结果:

```

Input the first integer number:6↵
Input the second integer number:9↵
max=9

```

【例 1.3】 改写【例 1.2】，交换 main()函数和 max()函数的前后位置。

源程序略。

程序运行结果:

```

Input the first integer number:6↵
Input the second integer number:9↵
max=9

```

通过以上几个例子，可以看到：

(1) 函数是 C 语言程序的基本单位。main()函数的作用，相当于其他高级语言中的主程序；其他函数的作用，相当于子程序。

(2) C 语言程序总是从 main()函数开始执行。一个 C 语言程序，总是从 main()函数开始执行，而不管其在程序中的位置。当主函数执行完毕，亦即程序执行完毕。

习惯上，将主函数 main()放在最前头。

1.2.2 函数的一般结构

任何函数（包括主函数 main()）都是由函数说明和函数体两部分组成。其一般结构如下：

[函数类型] 函数名 (函数参数表) 函数说明部分

```

{   说明语句部分;
  执行语句部分;
}   } 函数体部分

```

1) 使用的语法符号约定

[...]——方括号表示可选（即可以指定，也可以缺省）。

.....——省略号表示前面的项可以重复。

| ——多（含 2）中选 1。

2) 函数说明

由函数类型（可缺省）、函数名和函数参数表 3 部分组成，其中函数参数表的格式为：

数据类型 形参[, 数据类型 形参 2.....]

例如, [例 1.2]中的函数 `max()`, 其函数说明各部分如图 1.1 所示。



图 1.1

注意: 在旧标准中, 函数可以缺省参数表。

3) 函数体

在函数说明部分的下面、大括号（必须配对使用）内的部分。

函数体一般由说明语句和可执行语句两部分构成：

(1) 说明语句部分。说明语句部分由变量定义、自定义类型定义、自定义函数说明、外部变量说明等组成。

(2) 可执行语句。一般由若干条可执行语句构成。如图 1.2 所示是[案例 1.2]的 `main()` 函数体的示意图。

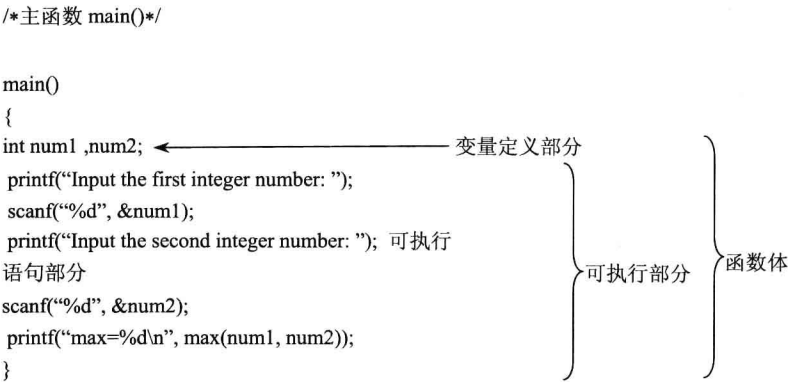


图 1.2

4) 说明

① 函数体中的变量定义语句，必须在所有可执行语句之前。下面程序中变量定义语句“`int max;`”的位置是非法的：

```
main ()
{ int x,y; /*变量定义语句：定义 2 个整型变量 x、y*/
  x=3; /*可执行的赋值语句：将 3 赋值给变量 x*/
  y=6; /*可执行的赋值语句：将 6 赋值给变量 y*/
  int max; /*变量定义语句：出现在可执行的赋值语句"x=3;"和" y=6;"之后，
  非法! */
  max=x>y?x:y;
  printf("max=%d\n",max);
}
```

解决办法很简单，请读者自己思考。

② 如果不需要，也可以默认变量定义语句。

1.2.3 源程序书写格式

(1) 所有语句都必须以分号“;”结束，函数的最后一个语句也不例外。

(2) 程序行的书写格式自由，既允许 1 行内写几条语句，也允许 1 条语句分写在几行上。

例如，[例 1.2]的主函数 `main()`，也可改写成如下所示的格式：

```
main()
{
    int num1,num2;
    printf("Input the first integer number: ");
    scanf("%d", &num1);
    printf("Input the second integer number: ");
    scanf("%d", &num2);
    printf("max=%d\n", max(num1, num2));
}
```

如果某条语句很长，一般需要将其分写在几行上。

(3) 允许使用注释。C 语言的注释格式为： `/* ..... */`。

例如，在【例 1.1】和【例 1.2】中，以及本节其他部分给出的源程序中，凡是用“/*”和“*/”括起来的文字，都是注释。

① “/*”和“*/”必须成对使用，且“/”和“*”、以及“*”和“/”之间不能有空格，否则会出错。

技巧：为避免遗漏必须配对使用的符号，例如注释符号、函数体的起止标识符（花括号）、圆括号等等，在输入时，可连续输入这些起止标识符，然后再在其中进行插入来完成内容的编辑。在起止标识符嵌套时，尤其相距较远时，这样做更有必要。

② 注释的位置，可以单占 1 行，也可以跟在语句的后面。

③ 如果一行写不下，可另起一行继续写。

④ 注释中允许使用汉字。但在非中文操作系统下，看到的会是一串乱码，但这不影响程序的运行。

1.3 C 语言的语句和关键字

1.3.1 C 语言的语句

与其他高级语言一样，C 语言也是利用函数体中的可执行语句，向计算机系统发出操作命令。按照语句功能或构成的不同，可将 C 语言的语句分为 5 类。

1) 控制语句

控制语句完成一定的控制功能。C 语言只有 9 条控制语句，又可细分为 3 种：

(1) 选择结构控制语句。

`if() ~ else ~, switch() ~`

(2) 循环结构控制语句。

`do ~ while(), for() ~, while() ~, break, continue`

(3) 其他控制语句。

goto, return

2) 函数调用语句

函数调用语句由一次函数调用加一个分号（语句结束标志）构成。

例如, `printf("This is a C function statement.");`

3) 表达式语句

表达式语句由表达式后加一个分号构成。最典型的表达式语句是, 在赋值表达式后加一个分号构成的赋值语句。

例如, “`num=5`” 是一个赋值表达式, 而 “`num=5;`” 却是一个赋值语句。

4) 空语句

空语句仅由一个分号构成。显然, 空语句什么操作也不执行。

例如, 下面就是一个空语句:

;

5) 复合语句

复合语句是由大括号括起来的一组（也可以是 1 条）语句构成。例如:

```
main()
```

```
{ .....
```

```
{.....} /*复合语句。注意: 右括号后不需要分号。*/
```

```
.....
```

```
}
```

复合语句的性质如下:

(1) 在语法上和单一语句相同, 即单一语句可以出现的地方, 也可以使用复合语句。

(2) 复合语句可以嵌套, 即复合语句中也可出现复合语句。

1.3.2 关键字

C 语言的关键字共有 32 个, 根据关键字的作用, 可分其为数据类型关键字、控制语句关键字、存储类型关键字和其他关键字 4 类。

(1) 数据类型关键字 (12 个): `char, double, enum, float, int, long, short, signed, struct, union, unsigned, void`。

(2) 控制语句关键字 (12 个): `break, case, continue, default, do, else, for, goto, if, return, switch, while`。

(3) 存储类型关键字 (4 个): `auto, extern, register, static`。

(4) 其他关键字 (4 个): `const, sizeof, typedef, volatile`。

本章小结

本章主要介绍 C 语言的特点, 通过简单的 C 语言程序, 了解并掌握 C 语言程序的特点、C 语言程序的结构和书写规则。

学习 C 语言时有一点应当注意: 应该把精力放在最基本、最常用的内容上。开始不要钻牛角尖, 有一些细节, 随着对 C 语言的了解逐步深入和实践经验的逐步丰富, 会自然地掌握, 而有些细节则要通过长期的实践才能真正熟练掌握。

第 2 章 基本数据类型和运算

【教学提示】

本章介绍 C 语言提供的基本数据类型及数据运算规则，这些知识是程序设计的基础。通过本章的学习，学生将对 C 语言的数据处理能力有更进一步的了解。

【技能目标】

- (1) 掌握 C 语言运算符。
- (2) 掌握 C 语言数据类型。
- (3) 掌握 C 语言运算规则和使用场景。

2.1 标识符

所谓标识符，实际上就是一个字符序列。在 C 语言中，标识符用来标记常量、变量、数据类型、函数及程序的名字。在 C 语言中构成标识符必须符合下列语法规则：

以字母或下划线 “_” 符号中任一字符打头。

在第一个字符之后，可以是任意的字母 (A~Z,a~z)、下划线(_)或数字(0~9)组成的字符序列，这个序列可以是空串。

C 语言中的标识符分为以下 3 类。

1) 关键字

关键字是用来说明 C 语言中某一固定含义的字。例如 float 是关键字，用以说明变量是浮点类型。新的 ANSI C 标准已经将关键字扩充到 32 个见表 2.1。

表 2.1 ANSI C 标准关键字

auto	break	case	char	const	continue	default
do	double	else	enum	extern	float	for
goto	if	int	long	register	return	short
signed	static	sizeof	struct	switch	typedef	union
unsigned	void	volatile	while			

这些关键字为 C 语言专用符号，不得赋予其他含义，C 语言中的习惯是用小写字母书写关键字，所有这些关键字也都是由小写字母构成的。新标准已将原来的 entry 关键字取消。

2) 特定字（预定义标识符）

特定字是具有特定含义的标识符，主要有如下 7 个。

#define #include #undef #ifdef #ifndef #endif #line

特定字主要用在 C 语言的预处理程序中，这些标识符虽然不是关键字，但由于这些字赋予特定含义，所以人们习惯把特定字看做是关键字。因此在程序中不能把这些特定字当做一般标识符使用。

3) 用户标识符

用户标识符通常是用户根据程序中的需要自己按照标识符规则定义的标识符。这些标识符可根据所标识的内容用英文或汉语拼音来表示, 或根据习惯符号来表示。下面的字符串为合法的标识符:

```
dw    name    f_1    ff_2    x    y    abc    area    _age
```

而下面的字符串都是不合法的标识符:

```
2a  -xy  name#  no*    a/bb
```

注意: 如果用户标识符与关键字相同, 在对程序进行编译时系统将给出出错信息; 如果用户标识符与预定义标识符相同, 系统并不报错, 只是该预定义标识符将失去原定义的含义, 代之以用户确认的含义, 这样有可能引发一些运行时错误。

4) 命名规则

C 语言中的大小写字母具有不同的含义, 例如, **name** 和 **NAME** 就代表不同的标识符。原来的 C 语言中虽然规定标识符的长度不限, 但只有前 8 个字符有效, 所以对下面两个变量是无法区别的:

```
aabbccddRadio
```

```
aabbccddTV
```

现在流行的为 32 位操作系统配备的 C 编译器已经能识别长文件名, 不再受 8 位长度的限制。另外, 在选取时不仅要保证正确性, 还要考虑容易区分, 不易混淆。例如, 数字 1 和字母 l 在一起时就不易辨认。

用户在定义标识符时, 应注意以下几个问题。

① C 语言中有 32 个关键字, 每个关键字在 C 程序中都代表着某一固定含义。用户在定义标识符时不应采用与它们同名的标识符。

② 系统已经定义了一些预定义标识符, 包括预处理命令和 C 语言提供的库函数的名字。为了增强程序的可读性, 建议用户不要将其定义为标识符使用。

③ C 语言是“大小写敏感”的语言, 即将大写字母和小写字母作为不同的字符处理, 如系统认为 **sum**、**SUM**、**Sum**、**sUM** 是不同的标识符。C 语言中的关键字和预定义标识符全部以小写字母表示。

④ 不同的 C 语言版本对标识符的长度有不同的规定, 许多系统通常取前 8 个字符, 因此, 在定义标识符时, 最好不要超过 8 个字符。

⑤ 见名知义。标识符虽然可由程序员随意定义, 但标识符是用于标识某个变量或函数等的符号, 因此, 命名应尽量有相应的意义, 以便阅读理解。

由于教材中的程序一般很短, 变量的特定意义不明显, 所以有时就使用诸如 **x**、**y**、**z**、**a**、**b**、**c** 等简单的变量名。编制实用程序时不要这样做, 应该从现在开始就养成良好的命名习惯。

2.2 基本数据类型

C 语言规定, 在 C 程序中使用的每一个数据都属于唯一的一种数据类型, 没有无类型的数据, 一个数据也不可能同时属于多种数据类型。C 语言的数据类型如图 2.1 所示。