

21 世纪高等院校计算机系列课程教材

C语言程序设计

主编 高 巍 张立忠

(第2版)

21 世纪高等院校计算机系列课程教材

C 语言程序设计

(第 2 版)

主编 高 巍 张立忠

编者 姜 楠 张丽秋 梅 旻

王淮中 张 颜

 北京理工大学出版社

BEIJING INSTITUTE OF TECHNOLOGY PRESS

内 容 简 介

C 语言是一种高效使用的程序设计语言,目前仍在国内外广泛使用,是计算机应用人员应该掌握的一种程序设计工具。

本书深入浅出、循序渐进地介绍了使用标准 C 语言编程的必备知识,内容新颖、体系合理、通俗易懂、概念清晰,适用于教学。

如无特殊说明,本书程序均在 Visual C++6.0 下调试通过。本书可作为高等院校各专业学生学习 C 语言的教材及相关领域的培训教材,也可供各类报考 C 语言考试的读者参考。

本书同时配有《C 语言程序设计习题与上机指导》,以方便教师教学及学生自学。

版权专有 侵权必究

图书在版编目(CIP)数据

C 语言程序设计/高巍,张立忠主编. —2 版. —北京:北京理工大学出版社,2010.1

ISBN 978-7-5640-0315-9

I. C… II. ①高…②张… 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(2009)第 086234 号

出版发行/北京理工大学出版社

社 址/北京市海淀区中关村南大街 5 号

邮 编/100081

电 话/(010) 68914775(办公室) 68944990(批销中心) 68911084(读者服务部)

网 址/http://www.bitpress.com.cn

经 销/全国各地新华书店

印 刷/北京国马印刷厂

开 本/787 毫米×1092 毫米 1/16

印 张/16.25

字 数/370 千字

版 次/2010 年 1 月第 2 版 2010 年 1 月第 5 次印刷

印 数/18501~22500 册

定 价/32.00 元

责任校对/陈玉梅

责任印制/边心超

图书出现印装质量问题,本社负责调换

前 言

C 语言是一门在国内外都获得广泛应用的计算机语言。自从 20 世纪 70 年代 C 语言诞生以来, 凭借其自身的优点, 表现出了强大的生命力, 至今在许多学科和领域中仍然占据着绝对的统治地位。可以预见到, 在竞争激烈的 21 世纪, C 语言仍然是一门有前途的计算机语言。在大学计算机基础教学中, C 语言是核心课程, 我们只有学好它, 才能在今后的学习和工作中得心应手。

C 语言功能丰富, 使用灵活, 目标程序效率高, 可移植性好, 在高级语言中具有突出的优点。同时, 又具有汇编语言的一些特点, 因此在使用时更灵活方便。许多单片机在提供汇编语言编程工具的同时, 提供了标准 C 语言编译器, 支持使用标准 C 语言为单片机编程, 极大地提高了工作效率。另一方面, C 语言是 C++ 的基础, 有志于学习面向对象编程的读者, 可以选择使用 C++ 编程, 这样, 就必须学习 C 语言了。

程序设计是一门实践性很强的课程, 在掌握概念的基础上, 必须自己动手编程, 并且上机调试运行, 这样才能真正掌握程序设计的灵魂。如果仅仅将书读几遍, 忽视实践环节, 就希望能掌握程序设计的精髓, 那么日后不可避免地会在实践中碰壁。希望读者能够重视实践, 注重理论联系实际, 锻炼动手能力, 多思考、多练习。要将学习的目标放在能独立解决问题上, 而不是其他方面, 例如考试。当然, 这需要付出努力才行。

本书适合做各类高校计算机专业和非计算机专业的程序设计基础课教材。使用本书可以达到以下培养目标:

- (1) 程序设计入门, 领略什么是过程程序设计;
- (2) 掌握程序设计方法;
- (3) 把握 C 语言程序设计规律, 熟练使用 C 语言编程。

全书共分 12 章。前 4 章介绍 C 语言程序设计的基础知识, 这是必备的基础。以后各章依次介绍了数组、函数、指针预编译、结构体和共用体、位运算及文件的知识, 最后一章介绍了 C 语言的一些高级应用。

本书由高巍、张立忠主编, 姜楠、张丽秋、梅旻、王淮中、张颜等参编。由于编者水平有限, 加之时间仓促, 书中难免有不妥之处, 我们真诚希望得到广大读者的批评指正。

编者

目 录

第1章 C语言概述	1
1.1 C语言的发展历史	1
1.2 为什么要使用C语言	1
1.3 C语言程序范例及其结构特点	2
1.4 C语言程序的开发过程	4
本章小结	6
第2章 数据类型	7
2.1 数据类型	7
2.2 常量	9
2.3 变量	13
2.4 标准输入/输出函数	15
本章小结	21
第3章 运算符和表达式	22
3.1 算术运算符和算术表达式	22
3.2 赋值运算符和赋值表达式	26
3.3 关系运算符和关系表达式	29
3.4 逻辑运算符和逻辑表达式	30
3.5 条件运算符和条件表达式	33
3.6 逗号运算符和逗号表达式	34
3.7 长度运算符	35
3.8 数据类型转换	36
本章小结	38
第4章 语句	39
4.1 基本语句	39
4.2 流程控制语句	40
本章小结	61
第5章 数组	62
5.1 一维数组	62
5.2 二维数组和多维数组	71
5.3 字符数组和字符串	78
本章小结	89
第6章 函数	90
6.1 函数的概念	90

6.2 函数的定义和声明	91
6.3 函数调用	95
6.4 函数传递机制	98
6.5 递归	101
6.6 变量的作用域及存储类型	105
6.7 库函数	112
本章小结	116
第 7 章 指针	117
7.1 指针的概念	117
7.2 指针变量的定义和引用	118
7.3 指向数组的指针	121
7.4 字符串的指针	128
7.5 函数与指针	131
7.6 指针数组	139
7.7 指向指针的指针	141
本章小结	144
第 8 章 预编译	145
8.1 宏定义	145
8.2 文件包含	150
8.3 条件编译	153
本章小结	154
第 9 章 结构体与共用体	155
9.1 结构体类型变量定义和引用	155
9.2 结构体数组	160
9.3 结构体指针	162
9.4 链表	167
9.5 共用体	173
本章小结	177
第 10 章 位运算	178
10.1 概述	178
10.2 位运算符和位运算	178
10.3 位运算应用举例	185
10.4 位段	189
本章小结	192
第 11 章 文件	193
11.1 文件的概述	193
11.2 文件类型指针	195
11.3 文件的输入和输出	196
11.4 文件的定位操作	210

11.5 文件的错误检测	213
本章小结	214
第 12 章 C 语言综合应用	215
12.1 排序及查找算法	215
12.2 经典回溯问题	221
12.3 递归问题及实现	223
12.4 动态规划问题	227
12.5 趣味数学	230
12.6 数值计算	235
本章小结	242
附录 I ASCII 字符编码一览表	243
附录 II C 语言中的关键字	244
附录 III 库函数	245
附录 IV 运算符、优先级和结合性	249

第1章 C语言概述

1.1 C语言的发展历史

电子计算机自从 1946 年问世以来,其应用已渗透到社会的各个领域,并且硬件和软件都取得了突飞猛进的发展。作为计算机软件的基础,程序设计语言也得到不断的充实和完善。一些功能全面、使用便利的计算机程序语言相继问世。20 世纪 70 年代初期,在种类繁多的程序语言家族中又增添了一名新成员——C 语言。

C 语言是一种编译型程序设计语言,它的前身是马丁·理查德 (Martin Richards) 在 20 世纪 60 年代开发的 BCPL 语言。BCPL 语言是计算机软件人员在开发系统软件时,作为记述语言使用的一种程序语言。1970 年,美国贝尔实验室的肯·苏姆普逊 (Ken Thompson) 在软件开发工作中,继承和发展了 BCPL 语言的特点,进而提出了“B”语言。当时最新型的小型计算机,美国 DEC 公司的 PDP-7 型机中的 UNIX 操作系统就是使用 B 语言记述和开发的。此后,在美国贝尔实验室为更新型的小型机 PDP-11/20 进行 UNIX 操作系统的开发工作中,戴尼斯·M·利奇 (Dennis M Ritchie) 和布朗·W·卡尼汉 (Brian W Kernighan) 对 B 语言作了进一步的充实和完善,于 1972 年推出了一种新型的程序语言——C 语言。

C 语言功能强大而灵活,因此很快被传播到贝尔实验室之外,世界各地的程序员都使用它来编写各种程序。然而,在 C 语言出现不久,不同的组织便开始使用自己的 C 语言版本。由于没有统一的标准,使得不同版本的 C 语言之间出现了一些不一致的地方。为了改变这种状况,美国国家标准研究所 (ANSI) 为 C 语言制定了一套 ANSI 标准,成为现行的 C 语言标准。目前流行的 C 语言编译器绝大多数都遵守这一标准。

1.2 为什么要使用 C 语言

在当前的计算机编程领域中,有高级语言可供选择,如 C、Perl、Java 和 C++。这些都是非常卓越的计算机语言,能够完成大部分编程任务。但是,基于以下几个原因,很多人认为 C 语言是其中最佳的。

① C 语言功能强大,并且语言简洁、紧凑,使用方便、灵活。C 语言仅有 32 个关键字 (见附录 II),9 种控制语句,程序的书写形式也很自由。C 语言可用于操作系统、字处理器、图形、电子表格等项目,甚至可用于编写其他语言的编译器。具体的实例可参考本书第 12 章的内容。

② C 语言具有结构化的控制语句,用函数作为程序模块以实现程序的模块化。

③ 数据类型丰富。C 语言除具有基本数据类型整型 (int)、实型 (float 和 double)、字符型 (char) 外,还有各种构造类型。利用这些数据类型可以实现复杂的数据结构,如堆栈、队列、链表等。

④ C 语言表达能力强、语言简练，可以直接访问内存物理地址和硬件寄存器，可以表达直接由硬件实现的针对二进制位（bit）的运算。同时，C 语言在语言成分的表达方法上比较简洁，I/O 操作通过 C 库函数实现，因而程序也简洁，编译生成的程序体积小。

⑤ C 语言是可移植的。这意味着为一种计算机系统（如 IBM PC）编写的 C 语言，只需作少量的修改，甚至无需修改，就可以在其他系统中编译并运行。在不同的操作系统下的情形也类似。例如，在使用 Windows 操作系统的计算机上编写的 C 程序，可以不必修改或作少量修改就可成功移植到使用 Linux 操作系统的计算机上。C 语言的 ANSI 标准（有关编译器的一组规则）进一步加强了可移植性。

⑥ C 语言生成的目标代码质量高，程序执行效率高。代码质量是指 C 程序经编译后生成的目标程序在运行速度上和存储空间上开销的大小。一般而言，运行速度越高，占用的存储空间越少，则代码质量越高。一般高级语言相对于汇编语言而言，其代码质量要低得多，但 C 语言在代码质量上几乎可以与汇编语言相媲美。

由于 C 语言的上述众多特点，使它成为一个实用的通用程序设计语言，既可用于编写系统软件，又可编写应用软件，特别适用于编写各种与硬件环境相关的系统软件，是一种强有力的系统程序设计语言。

1.3 C 语言程序范例及其结构特点

任何计算机程序语言，都具有特定的语法规则和一定的表现形式。程序的书写格式和程序的构成规则是程序语言表现形式的一个重要方面。按照规定的格式和构成规则书写程序，不仅可以使程序设计人员和使用程序的人容易理解，更重要的是，当把程序输入给计算机时，计算机能够充分“认识”，从而能够正确执行它。

1.3.1 C 语言程序范例

一个完整的 C 语言程序可由若干个函数构成，其中必须有且只能有一个以 `main` 命名的主函数。下面介绍几个简单的 C 程序例子，从中可以看出 C 程序的组成结构。

例 1.1 编写一个程序，显示出以下一行文字：`I love China.`

```
#include "stdio.h"

void main()
{
    printf("I love China.\n");
}
```

运行这个程序时，在屏幕上显示一行英文：

`I love China.`

这是一个仅由 `main` 函数构成的 C 语言程序。`main` 是函数名，函数名后面一对圆括号内是写函数参数的，本程序的 `main` 函数没有参数，所以圆括号内什么也不写，但圆括号不能省略。`void` 表示此函数没有返回值。`main()` 后面被大括号 `{ }` 括起来的部分称为函数体。一般情况下，函数体由“说明部分”和“执行部分”组成。本例中只有执行部分而无说明部分。执行部分由若干语句组成。在 `printf` 函数的圆括号内，“`\n`”是换行符，即在输出“`I love China.`”

后回车换行。`#include "stdio.h"`是将标准输入/输出头文件包含到程序源文件中,以便确保能够正确执行输入/输出操作。

例 1.2 计算 2 个正整数的和与积。

```
#include "stdio.h"
void main()
{
    int a,b;                /* 定义 a、b 为整型变量 */
    int sum;                /* 定义 sum 为整型变量,存储 a 与 b 的和 */
    int prod;               /* 定义 prod 为整型变量,存储 a 与 b 的积 */
    printf("Please input 2 numbers:"); /* 输出提示字符串信息 Please input 2 numbers: */
    scanf("%d,%d",&a,&b);    /* 输入变量 a、b 的值 */
    sum=a+b;                /* 计算变量 sum 的值 */
    prod=a*b;               /* 计算变量 prod 的值 */
    printf("a+b=%d\n",sum);  /* 输出 sum 的值 */
    printf("a*b=%d\n",prod); /* 输出 prod 的值 */
}
```

程序的运行结果如下:

Please input 2 number: 3, 4✓

a+b=7

a*b=12

本程序也是一个仅由 main 函数构成的 C 语言程序。其中:

① `int a,b; int sum; int prod;` 这是变量定义语句,定义 4 个整型变量,变量名分别为 a、b、sum、prod。其中 sum 和 prod 分别用来存放 a、b 的和与积。

② `printf("Please input 2 numbers:");` 这是一条输出语句,通过调用 printf 库函数在显示屏上输出指定的内容,此例输出 "Please input 2 numbers:" 字符串。

③ `scanf("%d,%d",&a,&b);` 这是一条输入语句,通过调用 scanf 库函数从键盘上输入 a、b 的值,&a 和 &b 中的 "&" 含义是 "取地址",此 scanf 函数的作用是将 2 个数值分别输入到变量 a 和 b 的地址所标识的单元中,也就是输入给变量 a 和 b。关于 scanf 函数的介绍详见第 2 章内容。

④ 在 /* 和 */ 之间的内容是程序的注释。程序的注释对程序的运行没有任何影响。但是对阅读程序的用户会有帮助。

例 1.3 求 2 个整数中的大数。

```
#include "stdio.h"
int max(int x, int y)
{
    return(x>y ? x : y);
}
void main()
{ int num1,num2;
```

```
scanf("%d, %d", &num1, &num2);  
printf("max=%d\n", max(num1, num2));  
}
```

运行这个程序时, 输入 3,9 ✓ (输入 3 和 9 给 num1, num2)

在屏幕上显示:

max=9

本程序是由 main 函数和一个被调用函数 max 构成的。max() 函数的作用是返回 x 和 y 中较大的值, 即通过 return 语句将 x 和 y 中较大的数返回给主调函数 main。返回值是通过函数名 max 带回到 main 函数的调用处。main 函数中第 4 行为调用 max 函数, 在调用时将实际参数 num1、num2 的值分别传送给 max 函数中的形式参数 x、y。调用 max 函数后, main 函数得到一个返回值, 并用 printf 函数输出这个值。printf 函数中双引号内 “max=%d\n” 在输出时, 其中 “%d” 将由 max 的返回值代替, 而 “max=” 原样输出。

通过对上述几个例子的分析, 我们可以归纳出 C 语言程序的结构特点。

1.3.2 C 语言程序结构特点

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, C 语言中的其他函数, 相当于其他高级语言中的子程序。

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, 而不论其在程序中的位置。当主函数执行完毕时, 亦即程序执行完毕。

③ 分号 “;” 是 C 语句的一部分。

④ C 程序书写格式自由, 一行内可写多条语句, 且语句中的空格和回车符均可忽略不计。

⑤ 程序的注释部分应括在 “/*” 和 “*/” 之间, “/*” 和 “*/” 必须成对出现。“/” 和 “*” 之间不允许留有空格, 注释可以用中文, 也可以用英文。注释可以出现在程序中的任何位置上。注释部分对程序的运行没有影响。在注释中可以说明变量的含义、程序段的功能, 以便帮助人们阅读程序。因此一个好的程序应该有详细的注释。

1.4 C 语言程序的开发过程

C 语言采用编译方式将源程序转换为二进制的目标代码。一个编写完成的 C 程序在成功运行之前, 一般经过编辑源代码、编译、连接、运行 4 个步骤。

(1) 编辑源代码

使用一个文本编辑器 (如 Turbo C 2.0 系统自带的编辑器) 编辑 C 源代码, 创建一个包含源代码的磁盘文件, 文件的扩展名为 “.c”。源代码是一系列的语句或命令, 用于指示计算机要执行的任务。

(2) 编译

使用一个 C 语言编译系统 (如 Turbo C 2.0 系统或者 Microsoft VC++ 6.0 系统) 对 C 源程序进行语法检查和翻译, 生成同名的 “.obj” 目标文件。

因为计算机只能识别二进制形式的机器语言, 所以要将源代码转换为机器语言后才能在计算机上运行。这种转换工作是由编译器来完成的。编译器将源代码文件作为输入, 如果编

译器没有发现源代码文件的任何错误，就会生成一个与源代码文件同名，但扩展名为“.o”或“.obj”的目标文件，目标文件中包含了与源代码语句对应的机器语言指令。编译器创建的机器语言指令被称为目标代码，包含目标代码的磁盘文件被称为目标文件。如果发现源代码中的错误，编译器将生成报告提交给程序员，以便程序员在源代码文件中进行修改。

(3) 连接

将目标文件和库函数等连接在一起并生成一个“.exe”的可执行文件。

ANSI C 语言定义中包含一个函数库，其中包含预定义的函数的目标代码（已编译过的函数的代码）。预定义的函数包含编写好的 C 代码，由编译器软件包以可直接使用的方式提供。如前面的程序范例中使用的 `printf()` 和 `scanf()` 便是库函数中常用的 2 个。

如果程序中使用了库函数，则必须将编译源代码时生成的目标文件和库函数中的目标代码组合起来，并生成最终的可执行程序。这一过程称为连接，是由连接程序完成的。

使用连接程序进行连接时，如没有检查到语法或其他错误，则生成一个与源代码主文件名相同，但扩展名为“.exe”的可执行程序。图 1.1 说明了编译器将 C 程序源代码转换为目标代码，然后被连接程序转换为可执行文件的过程。

(4) 运行可执行文件

将程序进行编译和连接，创建出可执行文件后，就可以运行了。如果运行程序时得到的结果与期望的不同，则需要回到第 1 步，找出导致问题的原因，并在源代码中进行更正。修改源代码后，需要重新编译和连接程序，创建更正后的可执行文件，如果还有错误，则要不断循环下去，直到程序的执行情况同期望的完全相符。

图 1.2 说明了上述程序的开发步骤。

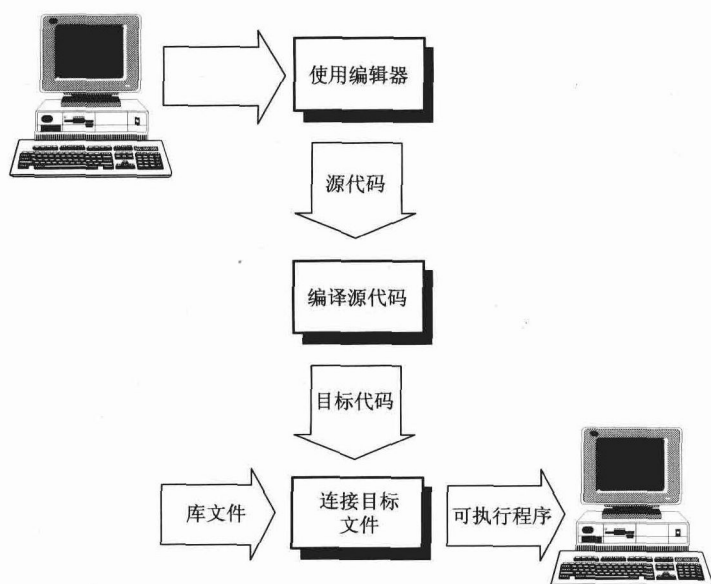


图 1.1 源程序到可执行文件的转换过程

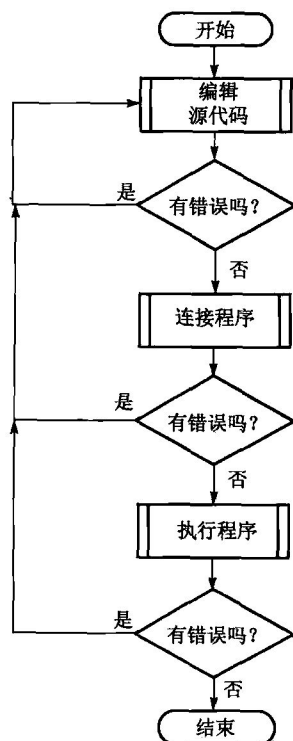


图 1.2 C 语言程序的开发步骤

本章小结

本章简要介绍了 C 语言的发展简史,回答了为什么要学习 C 语言这一初学者一直在思考的问题;同时介绍了 C 语言程序范例及其结构特点。要求熟练掌握 C 程序的基本结构,重点理解以下内容:

- ① 所有的 C 程序都由一个或多个函数组成。
- ② 在所有函数中,至少包含一个名为 `main()` 的主函数。
- ③ C 程序总是从主函数 `main()` 开始执行, `main()` 函数可以放在程序的任何位置。
- ④ 熟练掌握 C 语言程序的开发过程。C 程序的开发包括 4 个步骤:
 - 编辑源程序。使用一个文本编辑器(如 Turbo C 2.0 系统自带的编辑器)编辑 C 源程序。
 - 编译。使用一个 C 语言编译系统(如 Turbo C 2.0 系统)对 C 源程序进行语法检查和翻译,生成同名的“.obj”目标文件。
 - 连接。将目标文件和库函数等连接在一起并形成一个扩展名为“.exe”的可执行文件。
 - 运行。运行得到可执行文件。

在开发 C 语言程序时,有多种集成的开发环境可供选择。例如 Turbo C 2.0 和 Microsoft Visual C++6.0 等。需要注意的是,各种开发环境稍微有些差别。本书所提供的程序在 Microsoft Visual C++6.0 中均调试通过。

第2章 数据类型

计算机能处理的信息有很多种，例如文字、声音、图形、图像等。这些信息都是以一定的形式存储的。一般来说，数据在存储器中的存储情况由数据类型决定。每一种编程语言都有其支持的数据类型。在本章中，我们将要介绍 C 语言所支持的数据类型。在以后的学习中，我们将使用这些数据类型进行编程。

2.1 数据类型

C 语言可以使用的数据类型如图 2.1 所示。

由图 2.1 可见，C 语言的数据类型非常丰富，并且可根据实际需求，利用这些基本数据类型构造更为复杂的数据结构。需要注意的是，在 C 语言中，不论程序中用到哪种数据，都要明确地指明其数据类型，并遵循先说明后使用的原则。

在编程时，程序中所使用的每个数据都要声明为上述某一种类型，而且需要先正确定义，然后才可以使用。

1. 基本类型

C 语言提供了 4 种基本数据类型，分别是：整型、字符型、浮点型和枚举型。表 2.1 给出了整型、短整型和长整型数据在存储时所占二进制的位数和所能表示的数的范围。

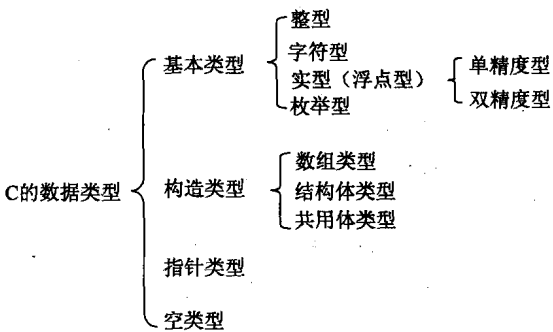


图 2.1 C 语言的数据类型

表 2.1 C 语言中的整数类型

类 型	位长度	值 域
unsigned int	32	0~4294967295 (0~(2 ³² -1))
[signed] int	32	-2147483648~2147483647 (-2 ³¹ ~(2 ³¹ -1))
unsigned short [int]	16	0~65535 (0~(2 ¹⁶ -1))
[signed] short [int]	16	-32768~32767 (-2 ¹⁵ ~(2 ¹⁵ -1))
unsigned long [int]	32	0~4294967295 (0~(2 ³¹ -1))
[signed] long [int]	32	-2147483648~2147483647 (-2 ³¹ ~(2 ³¹ -1))

注：[]里的内容可以省略。

(1) 整型数据

整型是指没有小数部分的数据类型。除了基本整型 int 外，为了处理不同取值范围的整数，

C 语言提供了扩展的整数类型, 它们的表示方法是在 `int` 之前加上关键字 `short`、`long` 或 `unsigned`。

有符号 (signed) 和无符号 (unsigned) 的数据的区别很小, 仅仅是它们对数据最高位的定义不同。对于二进制数, 如果最高位定义为符号位, 则这个数就是有符号数。此时, 最高位为 0, 表示这是一个正数; 如果最高位为 1, 表示这是一个负数。如果一个二进制数的最高位没有定义成符号位, 而是仅仅表示一个有效的数值位, 那么这个数就是无符号的数。无符号数都是正数。在位数相同的情况下, 有符号数和无符号数所能表示的数的范围是不同的。

例如:

$(+125)_{10}$ 的二进制为: 00000000 01111101

$(-125)_{10}$ 的二进制为: 10000000 01111101

在这里, 假设 +125 和 -125 都是短整型数据, 因此每个数据占 16 个二进制位。

但是, 大多数计算机都使用的是二进制补码, 因此还不能简单地使用上述代码。通常, 采用这样的约定: 正数的二进制补码就是该数的二进制形式, 而负数的二进制补码是将其对应的正数的各位求反 (符号位除外), 再加 1。所以:

$(+125)_{10}$ 的二进制补码为: 00000000 01111101

$(-125)_{10}$ 的二进制补码为: 11111111 10000011

C 语言并未规定各类整型数据的长度, 只要求 `short` 型不长于 `int` 型, `long` 型不短于 `int` 型。在 Turbo C 编译系统中, `int` 和 `unsigned` 型数据的长度为 16 位。本书中讨论的整型类型, 以 Visual C++ 编译系统为准, `int` 和 `unsigned` 型数据长度为 32 位, 如表 2.1 所示。

(2) 字符型数据

说明符用 `char` 表示, 如 'a', '0', '#' 都是字符类型数据。每个字符型数据在内存中占用一个字节, 即 8 个二进制位, 存储的是字符所对应的 ASCII 码。字符的 ASCII 码表请参见附录 I。由于 ASCII 码具有数值特征, 所以有时候可以用整数表示字符。

例如, 假设 `ch` 是字符型变量, 字符 'A' 的 ASCII 码值是 65, 则 `ch = 'A'` 和 `ch = 65` 是等价的。

需要注意的是, 字符型数据和整型数据所占的存储空间是不一样的, 字符型数据占一个字节。

C 语言中字符常量是用单引号包围起来的单个字符, 由双引号包围起来的是字符串。

除了单引号标志的单个字符外, 还有一类特殊字符: 转义字符。转义字符是由 "\" 引导的字符。转义字符一般表示一些屏幕动作或者不能在键盘上输入的字符数据。这一类字符外观上虽然不是单个字符, 但也属于字符型数据。关于字符型数据的用法, 将在 2.2 节中详细介绍。

(3) 枚举型数据

枚举型数据是一个整型常数的集合, 这个集合所包含的值通常是某一变量的所有合法取值。

枚举在日常生活中的运用十分常见, 例如一个星期有 7 天, 则星期的取值只有 7 个, 不应超出此范围。声明枚举类型的数据要用关键字 `enum`。

以星期为例:

```
enum week {Mon,Tue,Wed,Tur,Fri,Sat,Sun};
```

```
enum week SA;
```

此处声明了一个枚举类型 `enum week`，枚举列表中列出了每个星期所有合法的取值，然后又定义了一个 `enum week` 类型的变量 `SA`，那么变量 `SA` 所有的取值都在所定义的范围内。

需要注意的是，枚举是一个整型常数的集合，因此枚举列表中的元素只可以是整型数据，不允许是其他类型的数据。除非进行了初始化，否则第一个枚举元素的序号值为 0，第二个的序号值为 1，依次类推。则在上例中，`Mon` 的序号值为 0，`Tue` 的序号值为 1 等。

(4) 实型数据

C 语言所支持的实型包括单精度实型、双精度实型和长双精度实型，通常也称实型为浮点类型。这 3 种不同类型的实数可以用如下类型修饰符来修饰。

`float` 单精度实型

`double` 双精度实型

`long double` 长双精度实型

对实型数据而言，它的有效数字和数值范围在不同的系统上略有差异。表 2.2 列出了 Visual C++6.0 编译系统对常用的实型数据的取值情况。

表 2.2 实数类型

类 型	位长度	有 效 数 字
<code>float</code>	32	约有 7~8 位
<code>double</code>	64	约有 15~16 位

2. 构造类型

前面已经介绍过 C 语言的基本数据类型，如整型、字符型、实型等，但是仅有这些数据类型还不能满足实际应用中的需求。C 语言允许用户根据实际需要定义自己的数据结构，即构造数据类型。这些构造数据类型为用户提供了更简洁、更方便、更自然的数据描述方法。

C 语言的构造类型包括数组类型、结构体类型和共用体类型。构造类型的使用方法将在第 5 章和第 9 章中详细介绍。

3. 指针类型

指针是一种特殊的数据类型，指的是内存的地址值。在 C 语言中将内存的地址值称为指针。对于指针的意义、语法及使用，将在第 7 章中详细介绍。

4. 空类型

C 语言中空类型用 `void` 表示，它通常有 2 个用途：第一个用途是声明一个函数不需要返回任何值；第二个用途是用作指针的类型，描述一个可以指向任何数据类型的指针。关于这 2 个用途将在后续章节中介绍。

2.2 常量

常量是指在程序运行期间值不会改变的量。常量可以分为不同的类型，如整型常量、实型常量、字符串常量等，常量在程序中比较容易识别。

2.2.1 整型常量

整型常量包括基本整型常量和字符常量。

1. 基本整型常量的表示

基本整型常量通常用下述 3 种形式表示:

(1) 十进制表示: 由 0~9 共 10 个数组成, 如 263, -65, 0。

(2) 八进制表示: 由 0~7 共 8 个数组成。在 C 语言中, 为了将八进制数与十进制数相区别, 八进制数要以数字 0 开头。如 0236, 表示八进制数 $(236)_8$, 转换成相应的十进制数为:

$$(236)_8 = 2 \times 8^2 + 3 \times 8^1 + 6 \times 8^0 = (158)_{10}$$

又如:

$$(-037)_8 = -(3 \times 8^1 + 7 \times 8^0) = (-31)_{10}$$

(3) 十六进制表示: 由 0~9、A~F 共 16 个数组成, 以 0x 开头, 如 0x236, 表示十六进制数 $(236)_{16}$, 转换成相应的十进制数为:

$$(236)_{16} = 2 \times 16^2 + 3 \times 16^1 + 6 \times 16^0 = (566)_{10}$$

又如:

$$(3A5.F)_{16} = 3 \times 16^2 + 10 \times 16^1 + 5 \times 16^0 + 15 \times 16^{-1} = (933.9375)_{10}$$

2. 字符常量

在 C 语言中, 字符常量是用单引号括起来的单个字符, 如 'a', 'B', '*', '?' 等都是字符常量。由上述可知, 存储字符常量时存的是字符对应的 ASCII 码, 由于大写字母和小写字母的 ASCII 码是不同的, 因此字符常量大小写也是不同的, 例如 'B' 与 'b' 不同, '5' 与数值 5 也不同。附录 I 的 ASCII 码表列出了标准字符对应的 ASCII 码值。

在 ASCII 码表中, 有一些 ASCII 码对应的字符不能显示在屏幕上, 或者不能够从键盘上输入, 例如回车符的 ASCII 码是 13, 响铃符的 ASCII 码是 7。这样的字符在 C 语言中要用转义字符的形式来表示。转义字符的语法是在单引号中, 用 "\" 开头, 后跟一个替代字符或者一个数字。转义字符的含义是将 "\" 后面的字符转变成如表 2.3 所示的意义。

例如:

'\104' 代表 ASCII 码值等于八进制数 $(104)_8$ 的字符, 而 $(104)_8 = (68)_{10}$, 查 ASCII 表可知它是大写字母 'D'。

'\n' 代表换行符。

表 2.3 列出了常见的转义字符。

表 2.3 转义字符及其含义

转义字符	代 表 意 义
\n	换行, 将光标从当前位置移至下一行开头
\r	回车, 将光标从当前位置移至本行开头
\t	水平制表, 将光标水平方向跳至下一个 tab 位
\v	垂直制表, 将光标垂直方向跳至下一个 tab 位