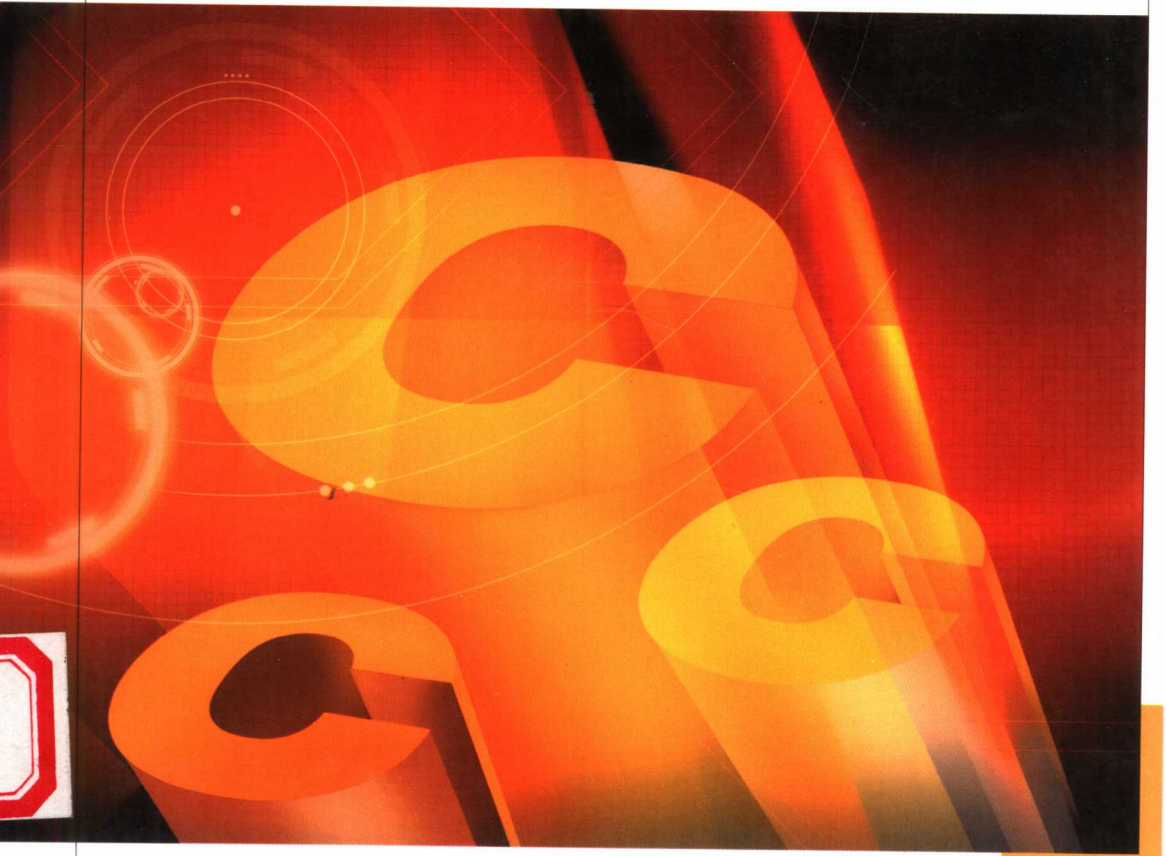
应用型本科人才培养创新教材出版工程

C语言程序设计

张宝森　陈　彦　主编
周海燕　主审



科学出版社
www.sciencep.com

●应用型本科人才培养创新教材出版工程

C语言程序设计

张宝森　陈　彦　主　编

周海燕　主　审

科学出版社

北　京

内 容 简 介

全书对C程序设计的内容结构做了新的安排，将指针与动态存储分配合为一章并安排在函数一章之前，宏定义和存储类别放在文件之后，以便提前学习文件内容。为了加深对指针的理解，本书提出了“无名变量”、“无名数组”的概念，以期读者对这些无名的对象只能使用指针进行存储。另外为了简化 switch 语句流程图，大胆使用了一个扇形图形符号。

本书以解决实际问题为中心，安排了大量实例，每个实例都采取“问题”、“问题分析”、“程序”三个步骤书写。在“问题分析”里对解决问题的思路与算法做了比较详尽讨论，并配有流程图，程序中有详细的注释。之所以采取先分析后给出程序的次序，是为了模仿自然的设计过程，以期开拓思维，提高读者的程序设计能力。

书中全部例题均在 TURBO C2.0 环境编译运行过。

本书内容丰富，通俗易懂，结构合理，操作性强，适合应用型本科、高职高专院校的学生使用，也可供广大计算机工程技术人员参考使用。

本书是为应用型本科、高职高专院校计算机、电子信息类专业一年级本科生而写，也适于专科学生和自学的读者。

图书在版编目(CIP)数据

C语言程序设计/张宝森，陈彦主编. —北京：科学出版社，2004.9
应用型本科人才培养创新教材出版工程
ISBN 7-03-014399-X
Ⅰ.C… Ⅱ.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(2004)第 095979 号

责任编辑：许 远 / 责任校对：包志虹
责任印制：安春生 / 封面设计：王凌波

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
印刷
科学出版社发行 各地新华书店经销
*
2004 年 9 月第 一 版 开本：B5(720×1000)
2004 年 9 月第一次印刷 印张：21 1/2
印数：1—3 500 字数：408 000

定价：32.00 元（含光盘）

（如有印装质量问题，我社负责调换（环伟））

前　言

C语言程序以其丰富的数据类型、高效快速的代码，易于操纵硬件设备的特点，被用作编写操作系统、硬件驱动程序和各种应用程序的强大工具。其结构化的流程简单合理、逻辑清晰，蕴含丰富的程序概念，仍然是当今程序设计的基础。即使现代的开发工具发展到了面向对象阶段，依然离不开结构化的编程方法，依旧可以见到C程序设计中诸多的概念和方法。这也是许多大学为什么将C语言作为程序设计入门语言的原因所在。

如果将C语言作为第一门程序设计语言，就要面临两方面的问题：语言的语法概念和程序设计的方法。笔者认为，语言的语法与概念好比是游戏的规则，程序设计好比游戏的攻防方案，已有的程序设计方法好比曾经出现过的游戏攻防案例，而程序的设计人员好比新游戏的玩家和导演。每个“玩家”和“导演”都可以充分利用上述“素材”，创造出优美杰出的软件作品。

在学习程序设计的语言与方法之前，作者要提醒大家注意的有三点：

第一，计算机比任何人都笨，它等着你去教它。只有你按照游戏规则教会了它，它才能很好地完成你的任务。计算机做的任何事情都是人教的。计算机没有智慧，它的智慧要等着你去赋予它。

第二，计算机与人相比，处理问题的方式略有差异。举例来说，桌子上有3个苹果，如果有人问“桌子上有多少苹果时”，人只需要瞄上一眼就可以了，但计算机却要一个一个地数，这是差异所在。但是如果给人蒙上眼睛，那么人也只能一个一个地数了。深入探究这一过程也会发现，即使不蒙上眼睛，当苹果的数量很大时，人也只能一个一个地数了。只不过在一般情况下，“瞄上一眼”的处理速度非常快，人们已经忽略了“数”的过程了。计算机的处理方式与人蒙上眼睛时的处理方式的确有极大的相似之处，因此解决实际问题时要挖掘思维的本质，不要被表面现象所蒙蔽。在程序设计中，我们常用这样的方式去思考问题会有很大好处的。

第三，任何程序都是对数据进行处理的，而数据在内存中的逻辑结构和物理结构又决定了处理方法。因此学习C语言程序设计，必须紧紧抓住数据的存储结构，必要时，要画出数据的存储结构图，才能进一步确定处理方法。笔者极力要求读者能够经常画图，养成用图形进行分析的良好习惯，这将使你分析问题、解决问题的能力得到极大提高。

本书可按64/32学时授课，选取不同例题，课时也可以调整为48/32。

第1章到第5章为基本语法部分。主要讲解数据的类型、运算符及其运算规律、各种语句（顺序、分支、循环）。在这几章里，初学者往往对各种数据类型感到迷茫，弄不清为什么要有这么多种类型，搞不清运算符要运算的数据的类型不同时，其运算效果是不一样的。另外对关系运算、逻辑运算以及它们的混合运算不能深刻理解，写不出能够正确反映自己观点的逻辑表达式（有时候自己的逻辑意图都是错误的），这是一大难点。

传统流程图难于描述 switch（）语句的逻辑流程，本书采用一个扇形的图形符号，借用扇形的结构来阐述 switch（）语句的功能，笔者认为这样或许可以起到直观的效果。

程序是由一条条的语句组成的。运行时，每一时刻都在运行着某条语句，程序当前被运行着的语句可以称为“运行点”，“运行点”的移动就构成了程序实际的运行过程，并最终出现程序中所安排的结果，初学者往往没有这样的概念，非要反复强调才能理解。建立程序“运行点”（执行点）的概念对循环语句的理解会有很大帮助。

第6章讲解指针与动态存储分配。由于指针在C语言里占有很重要的地位，大量数据是借助指针管理的，因此让初学者及早接触指针是件好事。本书通过使用动态存储分配，提出使用“无名变量”的概念来阐述指针与所指存储空间的关系，从而剥去指针的神秘面纱；通过建立“一维无名数组”，来阐述指针与数组的关系，建立对指针的深刻理解，为学习后面的知识打下基础。

第7章讲解函数。初学者对这一章感到困惑主要有如下几个地方，形式参数及参数传递、函数的返回值及类型、函数原型问题，通过指针形参使用其他主调函数的变量问题。

第8章讲解数组。数组是C语言程序设计的重要概念，其中一维数组显得尤为重要，它不仅涉及到数据的组织，而且还涉及到数据的存储。它既可以存储成批的数据，又可以存储字符串。不仅如此，它还是二维数组乃至高维数组的基础，即它们可以看作是由一维数组扩展而来。对一维数组，有大量初学者必须要掌握的程序设计算法，这些都是程序设计的基础。

第9章讲解结构体类型、位段类型、共用体类型和枚举类型，结构体类型是本章的基础。如果说结构体是将整字节组合起来，那么位段类型就是将整字节拆开来使用。共用体类型可以看作是在一个包装箱里只能放一个物品，即使物品很小，浪费空间也在所不惜。枚举类型原则上属于整数类型，但由于知识要点过于短小，没有再单独另分一章。

第10章讲解文件。文件一章也是内容繁多的一章，并且与操作系统紧密相连。初学者往往对文本文件与二进制文件的差别感到困惑。对文本文件，数据以一定的格式和次序写入文件，而在读取数据时，读取的格式与次序必须与写入时

的格式和次序一一对应。对二进制文件，初学者常常感到惧怕，不敢碰及。如果对二进制文件格式建立了良好概念，并且对二进制文件读写过程有所理解，能够和内存中的变量、数组联系起来，就会应用自如。

本书将编译预处理命令和变量的存储类别、生存期与可见性分别放在第 11 章和第 12 章。之所以将它们放在后面是基于突出主干的考虑，而这两章内容实际主要用于较大程序的研发。

目前关于 C 语言程序设计的书籍已有很多，那么作者为什么还要再加上一本呢？作者觉得在众多的书籍中，缺少一种入门级书籍，尤其是缺少转变学生的思维模式的书籍。本书力图做到这一点。基于此目的，本书全部例题均采用问题、问题分析、程序三个步骤书写，部分问题的解决方案配有算法或流程图，大部分程序做了详细的注释。

学完本书，读者仅仅是掌握了一门编程语言，懂得了程序设计中的一些概念，了解了程序设计中经常使用的一些基本方法，知道了计算机是如何解决实际问题的，所以本书最初命名为《C 语言程序设计》，意喻只有初级水平。但是作者想要告诉读者的是，程序设计最重要的是思维方式的转变，是思维的开拓。如果在读完本书后，读者在设计程序时的思维方式有所开拓，那么笔者就感到欣慰了。但由于作者水平有限，书中难免存有纰漏与不足，敬请同行赐教指正，笔者将不胜感激。

本书周海燕教授审阅了全稿，进行了全部修改或给出修改意见，王本教授给予了指导意见，并在高林教授、尤克教授、袁玫教授的关心下得以出版，在此一并感谢。

联系方法：北京市朝阳区北四环东路 97 号　北京联合大学电子信息技术实训基地教学部

邮政编码：100101，电话：010-64900229

E-mail：ldtbaosen@buu.com.cn，ldzhygz@163.com

张宝森

2003 年 11 月 30 日

目　录

第 1 章

第1章 概　　述

1.1 C语言的特点及简介

1.1.1 特点

1973年，贝尔实验室的工作人员创建了C语言，并用它重新编写了UNIX操作系统。1983年，美国国家标准协会（ANSI）制定了新的C语言标准——ANSI C，从此用C语言开发应用程序便普及发展起来。C语言的诞生与发展是由于它的特殊目的促成的，人们希望它既是高级语言，同时又兼有汇编语言直接操纵硬件的特点，从而更加利于编写操作系统一类的软件。

确实，与其他开发工具相比，C语言有极为独特的特点，特别容易与汇编语言接口，运行速度快，充分利用系统资源，更适合编写设备驱动程序，操作系统，应用系统，尤其在通信、自动化，信息处理领域得到广泛的应用。

1.1.2 简单的C程序介绍

本节要说明C语言程序的样子，我们来看具体的例子。

例1.1 给定一个长度值5作为棱长，求立方体的体积，再将这个长度值作为半径，求圆的面积。

问题分析 对于这个问题，我们首先想到的是立方体体积公式 $V = a^3$ 和圆的面积公式 $S = \pi r^2$。其次给定的长度值应当保存起来，以备重复使用。最后我们想到的是输出。

以上这个次序有点乱，在计算机程序设计中，最讲究的是次序的先后，给定的长度值应当最先保存起来，才能作为棱长，计算完立方体体积之后再作为半径计算圆的面积。这里我们先给出程序，再作解释。为了便于说明，我们给程序的每一行都加了行号（实际的程序是不应当有这些行号的）。

程序 a

```
1: /* 这是一个演示例子，先计算立方体体积，再计算圆的面积 */
2:  #include "stdio.h" /* 要使用的库函数的头文件 */
3:   main ( )          /* 程序的开始 */
4:    {
```

```
5:    int a = 5 , r; /* 定义整型变量，a作为棱长，r作为半径 */
6:    int V;         /* 定义整型变量V，作为立方体体积 */
7:    float S;       /* 定义单精度实型变量S，作为圆面积 */
8:    V = a * a * a;/* 按体积公式计算，结果保存在变量V中 */
9:    r = a;    /* 原来作为棱长的数据，现在当作半径大小赋给变量r */
10:   S = 3.141592 * r * r; /*按圆面积公式计算，结果保存在变量S中*/
11:   printf("The volume of the cube is %d\n",V);/*输出立方体积的信息 */
12:   printf("The area of the circle is %f\n",S);/*输出圆面积的信息*/
13:   }
```

这里，我们仅用一个被称为主函数 main（）的程序完成问题提出的任务。我们先不要管程序的第 1、2 行的作用。C 语言规定，一个程序无论大小，只能有而且必须有一个主函数 main，它是程序的起始点。任何 C 语言程序都是从 main（）函数开始运行的。另外，C 语言中，程序是以函数为单位的，主函数也是一个函数。一个函数的组成通常有两部分——函数头和函数体。在例 1.1 的程序 a 中，第 3 行 main（）就是函数头，4～13 行为函数体。其中第 4 行与第 13 行的一对花括号，指明了函数体的界限。函数体又分为两部分：变量说明（定义）部分（第 5、6、7 行）和执行语句部分（第 8～12 行）。说明部分用来定义计算过程中所需要的各个变量，并为这些变量设置初始值；执行语句部分是编程人员的思想体现，是程序员教导计算机如何一步一步地完成任务的操作序列，是计算过程。这两部分不能混杂交叉。程序是从第 8 行开始顺序向下逐条执行的。

程序的第 1 行以及每一行的后半部分是由一对“/* */”扩起来的文字，称为注释。注释可以占多行，也可以放在每行后面。它的作用仅仅是供程序员阅读理解程序，而计算机并不执行它。并非每一行语句都需要有注释。

第 11、12 条语句是调用库函数 printf（）。它的作用是输出信息和数据，它是设计 C 语言的工程师事先编好的函数。在 C 语言里有大量的库函数供程序员使用以便减轻编程的负担。要正确的使用库函数 printf（），应当在程序的前面包含它的头文件“stdio.h”，正像程序的第 2 条一样。在头文件“stdio.h”里，有对库函数 printf（）的说明。在 C 语言里任何函数、变量都必须先说明后使用。任何未经说明的函数、变量，都不能使用。

关于库函数 printf（）以后还有专门的说明，在这里，只想指出本程序中第 11、12 条语句的差别。每个 printf（）函数在输出由一对双引号括起来的文字时，前者最后有一个符号%d，而后者是%f。这些都是所谓的“格式控制符”。%d的意义是：在它所在的位置“放上”整型数据；而%f 的意义是：在它所在的位置“放上”实型数据。这些数据要在“输出项列表”（即引号、逗号后面的输出项）里一对一的匹配。第 11 条的 printf（）的输出项列表里只有变量 V，而第 12 条的 printf（）的输出项列表里只有变量 S，它们分别填入各自对应的

格式控制符，一行信息便输出完毕。

这个程序运行完毕后输出的信息是：

```
The volume of the cube is 125
The area of the circle is 78.539803
```

程序 a 并不能完全反映 C 语言程序的风貌，C 语言程序的特点是，一个程序是由众多的具有单一功能的函数组合起来的，函数是 C 语言程序的基本单位。就像多米诺骨牌，每一张都是简单的长方体，当把它们按不同的线路，不同的位置放置时，就会组成五彩缤纷的图案，波澜壮阔的景色。而 C 语言的一个函数就像一张多米诺骨牌，众多的函数相互组织起来，就可以构成功能强大、完成复杂任务的程序。

对例 1.1 的问题，我们更改上面的程序，采用多个函数的方法设计程序。我们经过分析可以得知，这个问题中，包含着计算与输出两种任务；而计算中又分为立方体的计算和圆面积的计算。在对这些任务分解之后，每项工作就很单一，很简单。我们可以分别用函数来完成。

为了从主体上了解 C 语言程序，而不马上陷入细节，我们也是先给出程序，详细内容以后讲解。

程序 b

```
1: #include "stdio.h"
2: int getcubevaluem (int a);
3: float getcirclearea (int r);
4: void outmessage (int V, float S);
5: main ()
6: {
7:    int n = 5;
8:    int V;
9:    float S;
10:   V = getcubevaluem(n);      /* 数据 n 传给求体积函数,V 接受计算结果 */
11:   S = getcirclearea(n);      /* 数据 n 传给求面积函数, S 接受计算结果 */
12:   outmessage(V , S);         /* 调用函数输出立方体积和圆面积的信息 */
13:  }
14:
15: int getcubevaluem(int a)    /* 形参 a 接受数据,作为棱长 */
16: {
17:   int cube;
18:     cube = a*a*a;           /* 按公式计算 */
19:   return cube;              /* 返回计算结果 */
20: }
```

```
21:
22: float getcirclearea(int r) /*形参 r 接受数据,作为半径 */
23: {
24:   float S;
25:   S = 3.141592 * r * r; /*按公式计算 */
26:   return S;             /*返回计算结果 */
27: }
28:
29: void outmessage(int V,float S)
30: {          /* 形参 V,S 接受主函数传给的数据:体积和面积 */
31:   printf("The volume of the cube is %f\n",V);/*输出立方体积的信息*/
32:   printf("The area of the circle is %f\n",S); /*输出圆面积的信息*/
33: }
```

借助注释，阅读完这个程序我们就会发现，这个程序的主函数仅仅做些传送与接收数据的工作，计算、输出工作都分摊给相应的函数完成。各个函数也非常简单，功能单一。整个程序结构简洁清晰，易于检查纠错。的确这个程序具备了C语言程序的主要风貌。

在这个程序中，第2、3、4行称为“函数原型”，分别与后面的三个函数相对应，其作用是将这三个函数的作用范围提前到主函数之前，达到未曾谋面而先闻其名的效果。详细内容将在第7章讲解。

主函数用变量n保存了数值5，在第9行通过语句 V = getcubevaluem (n)；将n保存的数据传给函数getcubevaluem，并停下来等待它运行之后的返回值。而从函数getcubevaluem角度认识，它是通过形式参数r接受主函数传来的数据5后，把它当作立方体的棱长，利用公式计算出立方体的体积，最后将其返回。到此，函数getcubevaluem运行结束，它将计算出的值返回给主函数，主函数在刚才停下来的位置接受了返回值，并且将返回值通过赋值运算赋给变量V，之后接着向下运行。以后的过程与此类似，读者可以自行模仿分析。这个分析里我们又见到了许多新名词，“形式参数”，数据在函数间的传递，返回值，这些都留在第7章讲解。

观察例1.1的程序a和例1.1的程序b这两个程序，我们会发现它们都有如下类似的语句：

```
int n = 5;
int V;
float S;
```

在这些语句中，变量前面为什么要有int与float？而且程序使用了常数5和3.141592，这两个数在C语言里有没有区别呢？这些问题都要涉及C语言的数

据类型问题，我们留待第2章中学习。

另外需要指出，这两个程序都有一个严重的缺憾，那就是只能计算数据5作为棱长的立方体体积和作为半径的圆面积，如果想采用其他数值，就必须改动“int a = 5，r;”或“int n = 5;”这两条语句。改动后还要编译连接，最后生成的程序也只能计算改动后的数值作为棱长和半径的立方体体积和圆的面积。你想不想制作一个可以计算任意数值作为棱长的立方体体积和作为半径的圆的面积？我们放在后面章节讲。你也可以自己先阅读一下后面章节，看看自己能否解决。

1.2 用C语言解决实际问题的步骤

用C语言编制程序解决实际问题，应当事先有一个计划，按计划一步一步的完成，否则极易出错，这也是初学者最容易出现的毛病。总的说来，要遵循如下步骤：

(1) 分析、确定问题的性质（数学的还是非数学的）以及所需数据和类型。

(2) 研究人处理问题的方法，找到适于计算机程序处理的方法。

(3) 将复杂任务分解为多个简单任务。

(4) 定义每个算法所需的数据及类型。

(5) 写出每个任务的算法。

(6) 编制、测试函数。

(7) 合成，联调。

在以后各章节里，我们尽量按上述步骤进行程序的设计。由于我们是初学，遇到的问题都极其简单，有些步骤就难免省略了。

1.3 学习C语言要注意的问题

1.3.1 要注意C语言概念和细节

由于C语言主要用来开发操作系统、设备驱动等程序，因此，它所涉及概念非常多，有大量的细节需要注意，初学者容易产生厌倦情绪，并且一不留神就会出错，这又会造成初学者对C语言的恐惧。

譬如除法运算，分为整数运算和实数运算，但是运算符却只有一个：“/”。具体做何种运算要取决于其两边的运算对象，当两边运算对象均为整型量时，所作运算为整数运算，结果为整数；当两边运算对象有一个是实型量时，所作运算为实型运算，结果为double（双精度实数型）。对于这个规则，如果运算对象都是常数，也许大家都会注意，如果是表达式，就容易犯错误。

要避免犯错误，就必须细心，在书写程序的每一个符号时，都要想一想它所涉及的基本概念与规则。这样做编程初期速度会很慢，但一段时间后就会很快，并且基本功会很扎实。

1.3.2 要注意变量或数据的存储关系

在C语言程序中，变量之间往往存在某种逻辑关系，这种逻辑关系表现了数据的组织关系。这些关系我们看不见，摸不着。此时只有凭着概念，画出它们的存储关系图，才能一目了然，理清它们的关系。因此，“画图”是分析问题、设计程序有力的辅助工具，初学者应当养成经常画图的良好习惯，经常通过图形去理解程序，设计程序，这样才能少犯错误，少走弯路。

习　题

1.1 在C语言中，程序从哪里开始运行?

1.2 什么是库函数?

1.3 什么是“格式控制符”? %d与%f的区别是什么?

1.4 #include “stdio.h”起到什么作用?

1.5 C语言程序的基本单位是什么?

1.6 函数原型有什么特点?

第 2 章

C 的基本数据类型与表达式

“工欲善其事，必先利其器”。在开始程序设计这一“游戏”之前，我们必须先学习游戏的规则。只有学懂了规则，掌握了规则，才能够在程序设计这个游戏之中，游刃有余，发挥自己的聪明才智。

2.1 数 据 类 型

2.1.1 基本数据类型

在 C 语言中，数据并不是散乱的，它们都具有类型这个特性，没有类型就无所谓数据。基本数据类型如表 2.1 所示。其中最常用的五种类型为整型（int）、长整型（long int）、字符型（char）、单精度实数型（float）和双精度实数型（double）。长整型又可以简写为 long。

表 2.1 C 语言的基本数据类型

类 别		名 称	类型名称	占用的字节数	取值范围
整型数据	有符号	整型	int	2	−32768 ～ +32767
		短整型	short int	2	−32768 ～ +32767
		长整型	long int	4	−2147483648 ～ +2147483647
		字符型	char	1	−128 ～ +127
	无符号	无符号整型	unsigned int	2	0 ～ 65535
		无符号短整型	unsigned short	2	0～65535
		无符号长整型	unsigned long	4	0 ～ 4294967295
		无符号字符型	unsigned char	1	0 ～ 255
实 数（浮点数）		单精度实数型	float	4	绝对值 3.4×10^{-38}～$3.4\times10^{+38}$ 6、7 位精度
		双精度实数型	double	8	绝对值 1.7×10^{-308}～$1.7\times10^{+308}$ 15、16 位精度
		长双精度型	long double	10	绝对值 3.4×10^{-4932}～$1.1\times10^{+4932}$
指针类型			类型 *	通常为 2 字节	

C 语言中为什么需要这么多的数据类型呢？答案很简单，为了适用不同方面问题的需要，尽量节约内存，提高运算速度。大家注意到表中有一列“占用存储单元的字节数”反映的就是这一特性。就整型数据而言，所占用的字节数越多，