高等学校教材

C/C++ 程序设计教程

谭浩强 张基温 编著

高等教育出版社

高 等 学 校 教 材

C/C++程序设计教程

谭浩强　张基温　编著

高 等 教 育 出 版 社

内容提要

本书以C语言为背景(兼顾C++),介绍高级语言程序设计的方法。全书共分7章。前6章全面、系统地介绍了C语言的基本概念、基本语法,并把重点放在提高学生程序设计和解题能力上。本书不要求学生具有程序设计的基础,当学完第1章(C语言程序设计初步)后,便可以初步了解C语言程序设计的基本方法。以后每学一章,程序设计能力就会有一个新的提高。最后一章,介绍了有关C++的基本概念和基本语法,为有意从C语言编程迈向C++编程的读者奠定一定的基础。

本书蕴含了作者丰富的教学和教材编写经验;例题、习题丰富;结构新颖、紧凑;讲解通俗、易懂。可作为高等院校有关专业C语言程序设计课程的教材,也可供各类培训班学员或应用开发人员学习参考。

图书在版编目(CIP)数据

C/C++程序设计教程/谭浩强,张基温编著.—北京:高等教育出版社,2001
ISBN 7-04-008911-4

Ⅰ.C... Ⅱ.①谭...②张... Ⅲ.C语言-程序设计-教材 Ⅳ.TP312

中国版本图书馆CIP数据核字(2000)第70705号

C/C++程序设计教程
谭浩强 张基温 编著

出版发行	高等教育出版社		
社　　址	北京市东城区沙滩后街55号	**邮政编码**	100009
电　　话	010-64054588	**传　　真**	010-64014048
网　　址	http://www.hep.edu.cn		
经　　销	新华书店北京发行所		
印　　刷	济南新华印刷厂		
开　　本	787×1092　1/16	**版　　次**	2001年1月第1版
印　　张	16.75	**印　　次**	2001年1月第1次印刷
字　　数	400 000	**定　　价**	21.20元

本书如有缺页、倒页、脱页等质量问题,请到所购图书销售部门联系调换。

前　言

20 世纪 80 年代以来，国内学习 C 语言的人愈来愈多，高等学校理工科专业大都开设了 C 语言程序设计的课程。根据高等学校计算机基础教育发展的需要，我们在 1992 年编写了《C 语言程序设计教程》，由高等教育出版社出版。1998 年修订后出版了《C 语言程序设计教程（第二版）》。该书受到广大读者的欢迎，被许多大学选为教材。

随着计算机技术的发展，面向对象程序设计方法逐步得到推广，C++语言则是当前被广泛使用的、支持面向对象的程序设计语言之一。许多学习 C 语言的人希望了解 C++语言。实践表明，有 C 语言基础的人进一步学习 C++是不太困难的。

根据当前教学的需要，我们编写了这本《C/C++程序设计教程》。本书有以下几个特点：

（1）本书既介绍 C 语言程序设计，又介绍 C++语言。前 6 章介绍 C 语言程序设计，第 7 章介绍 C++的基本知识。读者可以根据自己的情况决定如何使用本书。例如，可以全部学习本书的内容，也可以只学习 C 语言程序设计部分，而将 C++部分作为自学或留作以后需要时学习。

（2）在 C 语言中，指针是一个很重要的概念，使用很灵活，也较难掌握。本书在第 1 章就开始介绍指针的概念，然后在后续各章结合有关的内容加以应用（如指针与变量、指针与数组、指针与函数），以不断深化指针的应用。这是一个尝试，是否能取得预期的效果，还要在教学实践中加以验证。

（3）本书着眼于努力提高读者的编程能力。学习程序设计，应该把重点放在算法上，语言只是一种工具。只要掌握了解题的算法，就可以用任何一种计算机语言编程序去解决问题。在本书的例题和习题中，有一些相对难一些，希望读者下功夫学习和掌握算法，并能举一反三。

（4）本书只介绍 C++的基础。C++语言比较复杂，概念较多，比较难学。事实上，只有编制过大程序的人，才能真正体会 C++的语言的优越性。考虑到大学一般专业的学生学习 C++，目的不是马上从事大型软件开发工作，而且本课程学时有限，因此不宜对学习 C++提过高的要求。本书只介绍 C++的初步知识，使读者对面向对象的程序设计方法和 C++有一个初步的了解，为读者今后进一步学习和使用 C++打下一定的基础。

（5）为帮助读者学习本课程，我们另外编写了一本《C/C++程序设计题解与实验教程》，作为本书的配套辅助教材也由高等教育出版社出版。该书内容包括：①《C/C++程序设计教程》一书各章习题的参考解答；②学习本课程的实验；③C/C++的实践环境；④C++的出错信息。相信会对读者有一定的帮助。

除了谭浩强和张基温教授外，参加本书部分工作的还有董惠丽、张秋菊等老师。由于作者水平有限，本书会有不足之处，敬请专家和读者指正。

作　者

2000 年 8 月

目　录

第 1 章　C 语言程序设计初步

1.1　程序与程序开发

1.1.1　程序、程序设计方法与程序设计语言

众所周知，程序就是要计算机完成某项工作的代名词。要计算机完成某项工作就要运行一个相应的程序，如某一动画程序、某一计算程序、某一管理程序等。其实，程序就是对计算机工作规则的描述，例如让它演出动画与进行计算或管理，规则是不相同的。为完成一项工作的规则的过程设计就称为程序设计。

从根本上说，程序设计是人的智力克服客观问题的复杂性的过程。作为个体的自然人的智力是有限的，而客观世界的复杂性是无限的。著名计算机科学家 Dijkstra 说过：“我只有一个小小的脑袋，但是我要靠它去工作。”这就要求在程序设计时，首先要解决一个如何正确地应用自己的思维的问题。这就是程序设计方法。高级程序设计语言的进步的主要动力也正是来自人们对程序设计方法的不断探索。

从大的方面来说，程序设计方法主要分为两大类：面向对象的程序设计方法和面向过程的程序设计方法。

1. 面向对象的程序设计方法和程序设计语言

面向对象的程序设计把世界看作是由对象组成的，对象是运动的，向对象发送消息，将会激活对象的行为，或者说对象行为靠消息触发而激活。正是由于对象的复杂性和运动的多样性，才组成了客观问题的多样性和复杂性。每一个对象都具有两种属性：静态属性和动态属性。例如某一家商店，店名、店主、地址、营业证号、资本、利润、经营范围、雇员姓名、库存等是它的静态属性；进、销、存、交税、雇佣等都是它的动态属性，称为方法；顾客购买某一种货物，将会激活该商店的卖行为，而通过卖，将会使库存和利润产生变化。

按面向对象的方法进行程序设计的关键，首先是确定对象；研究问题领域内所含的对象；将对象分类；并按层次关系对类进行组织和定义，如将车按图 1.1 所示的层次结构进行定义和管理。

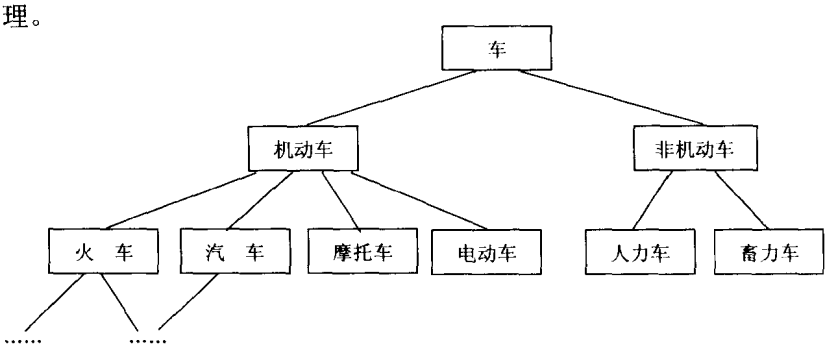


图 1.1　对象的分类

在层次结构中，每一个下层的类（如火车）都是在具有上层（父层——如机动车）属性的基础上，再增加一些属性而生成的。这一特征，称为类的继承。

类的定义，把对象看作是某一个类的特例，或者说由类生成对象；然后研究对象和类之间的关系以及运动规律。

在第7章中将更详细地介绍对象、类等概念。

把现实世界中的对象及其运动用程序设计描述，就称为面向对象的程序。或者说，面向对象的程序，要用面向对象的程序设计语言描述。最早引入类和继承机制的程序设计语言是Simula 67。公认的第一个商业性的面向对象的程序设计语言是Smalltalk 80。面向对象的程序设计语言在发展过程中形成了两大分支：纯面向对象的程序设计语言，如Smalltalk、Eiffel；混合（多范）型面向对象的程序设计语言，如C++、Objective C、LOOPS、Flavor和CLOS等。

2. 面向过程的程序设计方法和C程序设计语言

面向过程的程序设计是基于下面的认识：

- 程序的作用是实现某些功能。
- 每个功能由计算机的一个操作过程实现。
- 这些按一定的顺序安排操作序列的方法，也是一种常用的解题方法，其解题过程也称为程序。例如，下面是某一个会议的程序过程：
- 宣布会议开始；
- 奏国歌；
- 校长讲话；
- 宣布获奖者名单；
- 颁奖；
- 获奖代表讲话；
- 宣布会议结束。

再如，下面是做某一个菜的程序过程：

- 洗菜；
- 备菜；
- 把锅放在炉子上，点火；
- 锅热后加油；
- 油热后，放入葱、姜、蒜和调料略煸炒后，加入菜煸炒；
- 加入些水；
- 菜八成熟时，加入酱油、味精；
- 出锅，并装盘。

简单地说，程序主要用于描述完成某项功能所涉及的对象和动作规则。如上述的国歌、校长、名单、代表、话、奖以及菜、锅、油、水、酱油、味精等都是对象，而奏、作、颁、讲、宣布、放入、出锅等都是动作，这些动作的先后顺序以及它们能作用的对象，要遵守一定的规则。如“颁”的作用对象是“奖”而不是“话”；不能先颁奖，后宣布获奖名单。再如，做菜时，若锅热后，先放水、再放菜，就变成了煮菜，而不是炒菜了。

面向过程的程序要用面向过程的程序设计语言描述。C程序设计语言就是一种面向过程

的程序设计语言。下面是几个简单的 C 语言程序的例子：

例 1.1

```
#include <stdio.h>
main()
{
    printf("This is a C program.");
}
```

说明：

（1）C 语言程序的基本形式为

```
main（）
 {
     <语句序列>
 }
```

“main()”是 C 语言程序的标志，称为 C 程序主函数。所有的 C 语言程序都必须有一个主函数。

（2）C 语言用语句对对象和操作进行描述，语句要写在“main()”后面的一对花括弧中。语句用分号结尾。

（3）本例中只有一条语句，它使用一个函数 printf()将双引号中的一串字符原样输出。printf()的具体操作是预先定义好的。这样的预先定义的函数还有许多，被系统分门别类地存放在称为函数库的地方，所以也将它们称为库函数。使用库函数，将使程序十分简洁、清晰。stdio.h 称为 printf()的头文件，它定义了要使用 printf()等输入输出函数所必需的信息，程序中要使用函数 printf()时，应当用预处理命令#include 将这头文件包含到程序中。

例 1.2

```
#include <stdio.h>
main()
{
    int a,b,c;
    a=5;
    b=3;
    c=a*b;
    printf("c=%d",c);
}
```

说明：

（1）“int a,b,c;”称为对变量的声明，它定义了 a、b、c 3 个变量。通常，变量具有如下特点：

• 占有一片可用于存放数据的内存空间，空间的大小决定于它要存放什么类型的数据。本例中，用关键字“int”定义了 a、b、c 是可存放整型数（integer）的变量。

• 变量的值即它所存放的数据值不是固定不变的，通过赋值操作（如 a=5;b=3;c=a*b;）可以用一个新的数据替换原来的数据。

·每个变量都要有一个名字（如本例中分别为 a、b、c）。给变量命名要按一定的规则进行。一般说来，变量名应以字母打头，后面是数字和字母组成的串。

（2）在 C 语言中，赋值操作用符号“=”表示。注意，它不是等号。在 C 语言中，等号为“==”。赋值号表示一种操作，作用是把其后面的数据送到其前面的变量中；等号则表示一种关系——其前后两个数据是否相等。

（3）字符“*”在 C 语言中表示乘运算。在 C 语言中，四则运算的运算符分别为+、－、*和/。

（4）库函数 printf()称为格式化输出函数，圆括号中的双引号中的字母表示格式。如本例中“%d”表示输出的数据是一个整型数，其前的“c=”将以原样输出。本例的输出结果为：c=15。

例 1.3

```
#include <stdio.h>
main()
{
    int a=5,b=3,c;
    c=a;
    a=b;
    b=c;
    printf("a=%d,b=%d",a,b);
}
```

说明：

（1）在声明语句“int a=5,b=3,c；”中，“a=5,b=3”称为对变量 a 和 b 的初始化，即定义变量 a 的初始值为 5，变量 b 的初始值为 3。

（2）“c=a；a=b；b=c；”3 个语句的操作结果是交换了变量 a 和 b 的值，变量 c 称为中间变量。具体操作为：先将变量 a 的值赋给变量 c（即 a、c 的值均变为 5），再将变量 b 的值赋给变量 a（即 b、a 的值均变为 3），最后将变量 c 的值赋给变量 b（即 b、c 的值均变为 5）。这一交换过程相当于有 a、b 两盘磁带，各录有不同的内容，要交换 a、b 两盘磁带的内容时，必须借助于第三盘磁带 c。

（3）语句“printf("a=%d,b=%d",a,b);”的输出结果为：

```
a=3,b=5
```

格式参数中的两个“%d”，分别指示要输出的两个数据 a 和 b 的格式，其余字符将依原样输出。

例 1.4

```
#include <stdio.h>
main()
{
    int a=5,b=3,max;
    if(a>b)
        max=a;
```

```
    else
        max=b;
    printf("max=%d",max);
}
```

说明：这个程序完成下面的功能：取 a、b 之中的较大者，赋值给变量 max，最后输出 max 的值。具体的操作为：如果 a>b，则将 a 的值赋给 max，否则将 b 的值赋给 max，接着输出 max 的值。

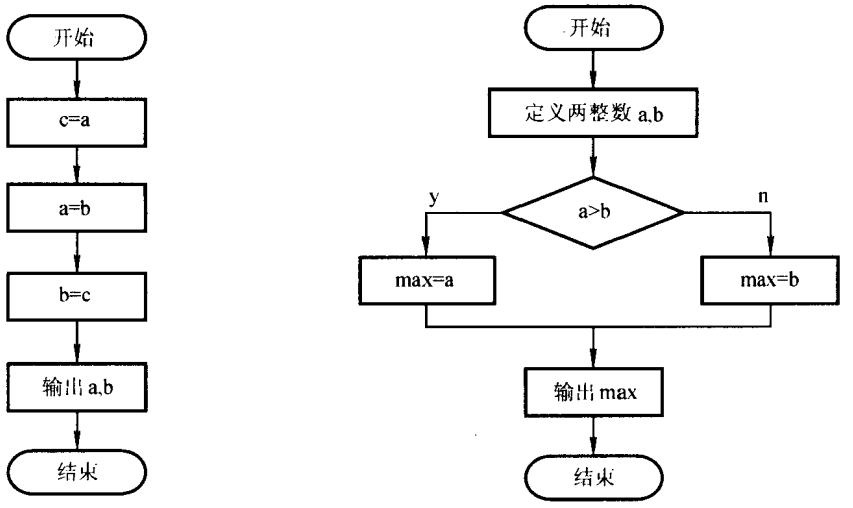
前面给出了几个小的 C 程序。一般说来，对不同的问题，有不同的解题思路；对同一个问题，也可能有不同的解题思路；对同一个问题的同一个解题思路，可能有不同的程序描述。

程序设计是一项艰巨的脑力劳动。为了使得问题更容易求解，在程序设计的初期要把精力集中在问题的求解思路的考虑上，暂不考虑程序描述的细节。解题思路的考虑分为两个方面：如何组织求解对象、制定求解过程的操作步骤和规则。这两个方面称为数据结构和算法。数据结构描述了问题涉及对象间的联系和组织结构；算法描述了求解问题的步骤或规则。著名的计算机科学家 Niklaus Wirth 有一个简洁的描述：

算法+数据结构=程序

也就是说，程序设计的关键是构造程序的数据结构并描述问题所需要的、施加在这些数据结构上的算法。

算法可以用自然语言描述，也可以用专用的算法描述语言描述，还可以用程序设计语言描述，用程序设计语言描述的算法就是程序。常用的算法专用描述语言是程序流程图。图 1.2(a)、(b)分别为例 1.3、例 1.4 的程序流程图。



（a） （b）

图 1.2 程序流程图示例

图中，矩形框表示操作，菱形框表示选择，带箭头的线表示流程。

3. 程序的一般概念

程序的概念因程序设计方法而异。下面，给出其一般的概念：程序是对完成某项功能所

涉及的对象和动作规则的描述。就像同样一个意思，用汉语、英语和日语描述出来的形式各不相同一样，要计算机实现同样一个功能，用不同的计算机程序设计语言描述出来，形式也各不相同。

对一个具体的问题来说，是采用面向对象的程序设计方法还是采用面向过程的程序设计方法，主要取决于以下几点：

（1）问题的规模以及其本身的性质。

（2）程序设计的环境：具体的程序设计语言支持不同的程序设计方法。

1.1.2　程序开发过程

程序开发的一般过程为：

1. 分析

一般说来，一个具体的问题要涉及许许多多的方面，这是问题的复杂性所在。为了便于求解，往往要忽略一些次要方面。这种通过忽略次要方面，而找出解题规律称为建立模型。分析问题就是要搞清问题；问题的分析能力来自对与问题相关领域的了解和所具备的知识。

2. 设计

设计分为两步：总体设计和详细设计。

（1）总体设计

当问题较大时，常常要把问题分为一些子问题；当子问题还大时，还应进一步分割。这样，一个较大的问题就被分成一些容易求解的子问题，每一个子问题将作为程序的一个模块。这是总体设计的一个任务。但是，仅仅分解还是不够的，在总体设计时，还要考虑如何组织程序模块。图 1.3 为一个教学管理程序的结构图。

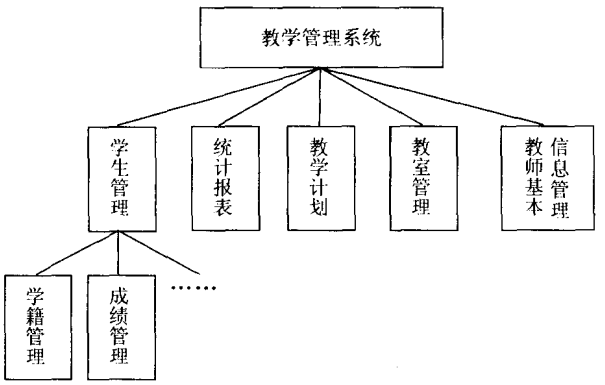


图 1.3　程序结构示例

（2）详细设计

详细设计就是设计每个模块的数据结构和算法。

3. 程序编码以及编辑、编译和链接

程序编码就是用具体的程序设计语言表现各模块的算法和数据结构。这一阶段的关键是运用某一种具体的计算机语言，来描述前面设计的数据结构及其算法。为此就要掌握一种计算机程序设计语言。

程序编码可以先写在纸上，也可以直接在计算机上用编辑软件编写，形成一个程序文件。在一定的环境下进行程序的书写以及修改的过程称为程序的编辑。

写出一个高级语言程序后，并不是就可以立即拿来执行。要让计算机执行，还要做下列工作：

（1）编译

用户用程序设计语言编写的程序称为源程序，这样的程序，计算机是不能理解的。要计算机理解，必须要将源程序文件按该语言的词法和语法规则翻译成用二进制表示的程序文件——目标程序文件。当然，同时还要对源程序的词法、语法和逻辑结构进行检查，这一过程称为编译。编译工作是由称做编译器（compiler）的软件完成的。不同的计算机程序设计语言有不同的编译器。

（2）链接

现在程序设计环境为用户提供了丰富的可利用资源，几乎所有的程序都使用到系统提供的资源模块。而且目标程序就是一些目标程序模块。对于大的程序来说，它们可能是被存放在几个文件中。因而，目标程序文件还不是完整的程序，不能被执行。为此，要将目标程序文件中的目标程序模块以及程序所需的系统中固有的目标程序模块（如执行输入输出操作的模块）链接，生成一个完整的程序文件才是可执行文件。完成链接过程的软件称为链接器（linker）。显然，不同的程序设计语言提供有不同的链接器。

图 1.4 为 C 语言程序的编辑、编译和链接过程示意图。

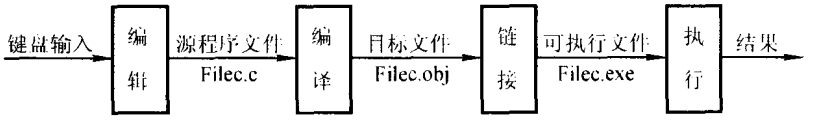


图 1.4 C 语言程序的编辑、编译和链接过程示意图

目前，程序的编辑、编译、链接以及运行，都可以在一些集成环境下进行。后面介绍的 Turbo C 就是一种集成环境。

4. 程序测试

每一个人编写出一个程序后，在正式交付使用前，总要试通一下。“试通”就是试运行程序，也就是对程序进行测试。今天，尽管程序测试作为程序设计的一个重要组成环节已再无人怀疑，但是在下面的两个问题上，并不是所有的人都能有一个正确的认识。

（1）程序测试的目的

基于不同的立场，存在着两种截然相反的观点：一种人认为，测试的目的是为了证明程序是正确的；另一种人认为，测试的目的是为了发现程序中的错误。实际上前一种观点将指导一种自欺欺人的行为，它对于提高程序的质量毫无价值。正确的观点是后者，Glenford J.Myers 把它归结为如下 3 句话：

- 测试是程序的执行过程，目的在于发现错误；
- 一个好的测试实例在于能发现至今未发现的错误；
- 一个成功的测试是发现了至今未发现的错误的测试。

（2）测试方法和技术

测试是以程序通过编译，没有语法和链接上的错误为前提的。在此基础上，通过让程序

试运行一组数据，来检测程序的逻辑以及程序中的各语句有无错误。这一组测试数据应是以假设“任何程序都是有错误的”为前提精心设计出来的，而不是随心所欲地乱凑而成的。它不仅应含有被测程序的输入数据，而且还应包括程序执行它们后预期的结果；每次测试都要把实际的结果与预期的结果相比较，以观察程序是否出错。

著名计算机科学家 Dijkstra 曾断言：“程序测试只能证明错误的存在，而不能证明错误不存在。”可以证明，除很小的程序外，无论使用任何方法，要想做到彻底的测试，即发现程序中的所有错误，是不现实的。现实是如何提高测试的效率，尽可能多地检测出程序中的错误。这就要求注重测试方法和技术。不同的测试方法所对应的测试用例的设计方法不同。

5. 编写程序文档

经过了问题分析、设计、程序编码、测试后，程序开发的工作基本上结束了。但是，这时还不能交付使用。因为还存在两个问题没有解决：一是用户拿到程序后会不会使用它；二是用户在使用过程中，对程序是否满意。

实际上，随着程序规模的增大和日益复杂化，程序的使用和运行也越来越复杂，用户要运行程序，还需要知道许多信息，如

- 程序的功能；
- 需要输入的数据类型、格式和取值范围；
- 需要使用的文件数量、名称、内容以及存放位置等；
- 程序运行需要的软、硬件环境；
- 程序的装入、启动方法以及交互方式等。

为此，程序开发者需要向用户提供这些资料——称为程序使用说明书或用户文档。需要说明的是，在许多软件中，这些内容已经部分或全部地以“readme”或“help”的文件形式提供。

另一方面，一个程序开发好后并不是一劳永逸了，由于下列原因，程序还要进行必要的修改：

（1）由于在开发过程中，开发者同用户之间的交流不可能非常充分、开发者对用户的要求没有充分理解等原因，程序同用户要求之间还会存在一些差距；

（2）软件界有一句这样的谚语：“没有不存在错误的软件”。由于开发者的疏忽以及测试的不完全性限制，程序一定会存在这样或那样的错误；

（3）在程序的使用过程中，由于条件、环境以及应用对象的改变。

在使用过程中进行的这些修改，称为程序的维护。为了便于程序的维护，也需要提供一个专门的文档——程序技术说明书或称程序技术文档，其内容包括：

- 程序各模块的描述；
- 程序使用硬件的有关信息；
- 主要算法的解释和描述；
- 各变量的名称、作用；
- 程序代码清单。

目前，程序文档已经成为软件开发产品的必要部分。文档在程序使用和维护中的重要性也改变了软件的概念，软件由早期的“计算机程序的总称”演化为“计算机的程序连同计算机化的文档的总称”。

1.2 数值数据与算术运算

计算机进行处理时，不仅要对数据施行运算，还要对数据进行存储。为了便于处理和存储，高级语言要把数据规格化，分成不同的类型。不同类型的数据存储和运算方式不同。大致说来，C 语言具有如下一些基本数据类型：

· 数值数据：有 int 类型、float 类型和 double 类型等。

· 字符数据：有 char 类型。

本节介绍数值数据，下一节介绍字符数据。

数值数据是能够实行算术运算的数据。由于不同大小和精度的数值数据，需要相应的存储空间，为了减少处理上的复杂性，C 语言将它们分为整型和实型两大类。简单地说，整型就是不带有小数点的数，如

-32768 -32767 0 5 32767 32766

等，都称为整型数。带有小数点的数，如

2. 3.14159 0.5 0.

等，都称为实型数。

1.2.1 整数类型

1. int、short int 和 long int 类型

为了方便应用，C 语言将整型数据分为 short int、int 和 long int 3 种类型，并要求 short int 类型的存储长度（所占用的内存字节数——通常每字节为 8 个二进制位）不大于 int 类型，int 类型的长度不大于 long int 类型。多数系统中，采用两种规格：有的将 short int 类型和 int 类型用一种规格存放（如在基于 16 位字长的微型计算机中，一般用两个字节长的定点格式来存放 short int 类型和 int 类型数据，用 4 个字节长的定点格式来存放 long int 类型数据）。使用不同的长度存放整型数，所能表示的数据范围不同。采用两个字节（16 位）存储，可表示 -32 768（即 -2^{15}）~32 767（即 $2^{15}-1$）之间的整数。采用 4 个字节（32 位）存储，可表示 -2 147 483 648（即 -2^{31}）~2 147 483 647（即 $2^{31}-1$）之间的整数。

本书介绍的是基于 16 位微型计算机系统的 C 语言。

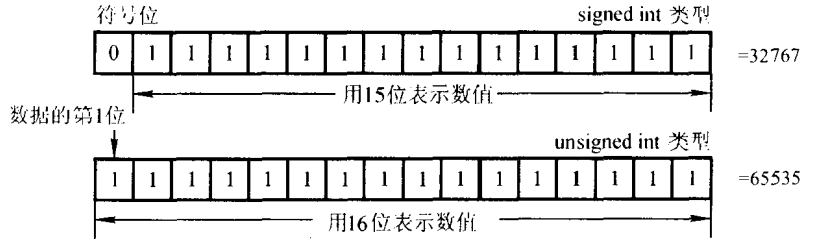
2. 带符号的整型数和不带符号的整型数

C 语言还把整型数分为带符号和不带符号两大类，即除了前面提到的外，还有

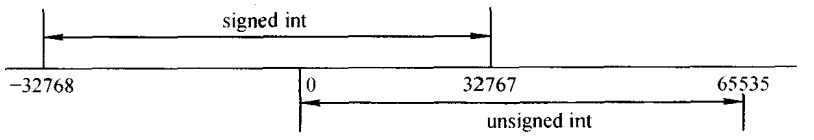
unsigned int

unsigned long

带符号和不带符号整数的区别，仅在于在机器内部带符号数要用 1 位二进制码表示正负号，而不带符号数用全部字长表示数值。因此，在同样的字长下，它们的表数空间相同，但不带符号数的最小值为 0，最大值比同样字长的带符号数扩大一倍。图 1.5 为 unsigned int 与 signed int 类型的表数范围的比较。



（a）最大数的存储形式比较



（b）表数范围比较

图 1.5　unsigned int 类型与 signed int 类型比较

3. 整数的书写形式

在 C 语言程序中，整型数允许使用十进制（用 0,1,2,3,4,5,6,7,8,9 十个码）、八进制（用 0 打头，并使用 0,1,2,3,4,5,6,7 八个码）和十六进制（用 0x 打头，并使用 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F 或 0,1,2,3,4,5,6,7,8,9,a,b,c,d,e,f 十六个码）书写。其中，八进制和十六进制码只能书写非负整数。下面是几个合法的整型数：

3225——（十进制正整数）

0111——（八进制数，等于十进制数 73）

0177777——（八进制数，等于十进制数 65537）

0XFFFF——（十六进制数，等于十进制数 65537）

-32768——（十进制负数）

0xA8——（十六进制数，等于十进制数 168）

下面是不合法的整型数：

09876——（非八进制数——含数字 8 和 9，又非十进制数——以 0 开头）

321fa——（非十六进制数，又非十进制数——含数字 f 和 a，但非 0x 打头）

0x10ak——（含非法字母 k）

由于整型数有 int 和 long int 之分，对于超出 int 类型范围的数，系统可以自动判断其为 long int 类型，但对于未超出 int 类型的数，想要表明其属于 long int 类型时，可以加后缀“L”或“l”（为避免小写的“l”与数字“1”混淆，建议写大写的“L”）。

1.2.2　实型类型

1. float 类型和 double 类型

当数值超出 long int 或要表示带小数点的数时，应使用实型数。实型数分为 float 类型和 double 类型两类。在基于 16 位的微型计算机系统的 C 语言中，通常 float 类型占 4 个字节空间，表数范围为±（3.4×10^{-38}~3.4×10^{38}），精度为大约 7 位十进制有效数字；而 double 类

型占 8 个字节空间，表数范围为±（1.7×10^{-308}~1.7×10^{308}），精度为大约 16 位十进制有效数字。

为了进一步提高表数精度和表数范围，必要时还可以使用长双精度（long double）类型。long double 类型在机器内用 10 个字节表示，大约有 19 位有效数字，其数值的表示范围约为±（1.2×10^{-4932}~1.2×10^{4932}）。

2. **实型数的小数形式和指数（科学记数法）形式**

下面是实数的小数形式书写的例子：

3.5　.008　3.　0.567

下面是指数（科学记数法）的小数形式书写的例子：

2E5(即 2×10^{5})　3.56e-3(即 3.56×10^{-3})　2.E-5(即 $2.\times10^{-5}$)

3. **区分 float 类型数和 long double 类型数**

通常，C 语言把实常数作为 double 类型来处理。当不做任何说明时，把带小数点的数都认为是 double 类型。当一个常数是 float 类型或 long double 类型时，需加后缀（f 或 F，l 或 L 以声明）。

1.2.3 数据类型长度的测试

前面曾多次强调在不同的系统——机器的字长和 C 语言的版本中，数值数据的存放长度和表数范围是不同的。表 1.1 为基于 16 位计算机系统和 ANSI C 的几种主要类型数据的存储空间和值域。

表 1.1 基于 16 位计算机系统的 ANSI C 数值数据的存储空间和值域

类　型	存储空间	值　域
int/short int	16 位	-32 768（即-2^{15}）~32 767（即$2^{15}-1$）
unsigned int	16 位	0~65 535（即$2^{16}-1$）
long int/long	32 位	-2 147 483 648（即-2^{31}）~2 147 483 647（即$2^{31}-1$）
unsigned long	32 位	0~4 294 967 295（即$2^{32}-1$）
float	32 位	±（3.4×10^{-38}~3.4×10^{38}）
double	64 位	±（1.7×10^{-308}~1.7×10^{308}）
long double	80 位	±（1.2×10^{-4932}~1.2×10^{4932}）

但是，由于已有多种 C 语言版本，并且计算机系统也在不断升级，现在 64 位的计算机逐步会成为主流，因此在程序设计之前除应查阅手册，了解所使用系统中各类型的数值数据的存储长度和表数范围外，还可以使用运算符函数 sizeof()来计算出各数据类型的存储长度的字节数。sizeof()函数的格式为：

sizeof(**类型符或变量名**)

例如：

```
sizeof(short int );
sizeof(int);
sizeof(unsigned int);
```