21 世纪高等学校计算机规划教材
21st Century University Planned Textbooks of Computer Science

C++程序设计基础教程

C++ Program Foundation

张晓如 华伟 主编
祁云嵩 王芳 於跃成 副主编

- 内容由浅入深、通俗易懂
- 教学深入浅出、循序渐进
- 案例详细丰富、实践性强

高校系列

中国工信出版集团
人民邮电出版社
POSTS & TELECOM PRESS

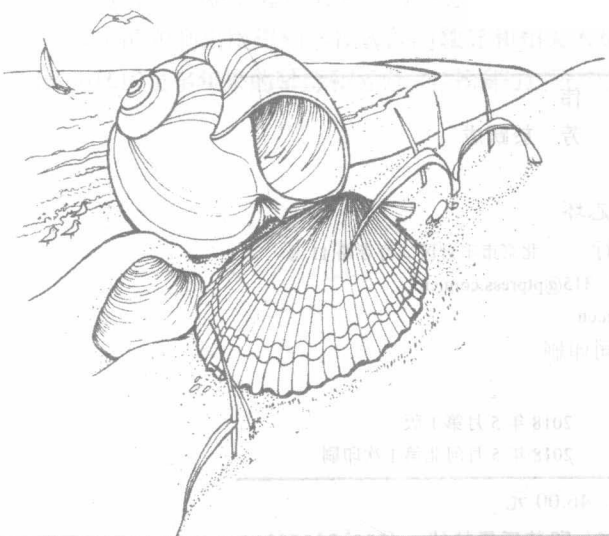
21世纪高等学校计算机规划教材
21st Century University Planned Textbooks of Computer Science

“十三五”江苏省高等学校重点教材
（编号：2017-1-011）

C++程序设计基础教程

C++ Program Foundation

张晓如 华伟 主编
祁云嵩 王芳 於跃成 副主编



高校系列

人民邮电出版社
北京

图书在版编目（CIP）数据

C++程序设计基础教程 / 张晓如，华伟主编. -- 北京 : 人民邮电出版社，2018.5
21世纪高等学校计算机规划教材
ISBN 978-7-115-47958-7

Ⅰ. ①C… Ⅱ. ①张… ②华… Ⅲ. ①C++语言－程序设计－高等学校－教材 Ⅳ. ①TP312.8

中国版本图书馆CIP数据核字(2018)第036657号

内 容 提 要

本书利用通俗易懂的语言以及大量的典型实例，循序渐进地介绍了C++程序设计的基础知识与编程方法，将C++程序设计的难点、要点分层次、分阶段地逐步展示出来。全书共分为10章，包括初识C++程序设计语言、C++语言编程基础、函数、数组、结构体与简单链表、类和对象、继承与多态性、友元函数与运算符重载、模板和异常处理、输入/输出流。

本书结构严谨，兼有普及与提高的双重功能，可作为高等院校计算机及相关专业的“C++程序设计”课程的教材，也适合作为软件开发人员及其他相关人员的参考书。

◆ 主　　编　张晓如　华　伟
副 主 编　祁云嵩　王　芳　於跃成
责任编辑　李　召
责任印制　沈　蓉　彭志环

◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路11号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
三河市潮河印业有限公司印刷

◆ 开本：787×1092　1/16
印张：13.5　　2018年5月第1版
字数：356千字　　2018年5月河北第1次印刷

定价：46.00元

读者服务热线：(010)81055256　印装质量热线：(010)81055316
反盗版热线：(010)81055315

前 言

自十多年前本书编写团队出版第一本C++教材以来，团队全体成员一直坚守在教学一线，不断研究符合新阶段的教学内容，探索适应新对象的教学模式，提升满足新要求的教学质量，通过不间断的课程建设，积累了大量较有价值的资料，其间陆续出版了几套系列教材。所幸多年来的努力使得教学成效明显，不枉这些年团队全体成员的付出。

本书将近年来团队在教学中的想法融入其中，根据学生的认知规律，以及高校程序设计教学的新特征，在章节的安排、内容的选择等方面都做了较大改进，注重学生计算思维能力和编程能力的培养，力求做到精讲、精练，内容易学易懂、方便教学。

本书共分为10章，第1章概述计算机程序设计语言；第2章介绍C++程序设计的编程基础；第3章通过函数介绍程序设计的基本思想和方法；第4章介绍C++顺序结构——数组；第5章介绍链式结构——简单链表；第6章介绍面向对象程序设计的基本思想和方法——类和对象；第7章介绍面向对象程序设计的基本特性——继承与多态性；第8章介绍友元函数与运算符重载；第9章介绍模板和异常处理；第10章介绍输入/输出流。

参加本书编写的有张晓如、华伟、祁云嵩、王芳、於跃成、王逊、段旭、范燕、石亮、严熙、潘舒、王红梅等老师，其中第1章和第6章由祁云嵩执笔，第2章、第9章和第10章由张晓如执笔，第3章由於跃成执笔，第4章和第7章由华伟执笔，第5章和第8章由王芳执笔，最后由张晓如负责统稿。限于编者水平，书中定有不足之处，恳请各位专家和读者批评指正。

本书的顺利出版得到学校各部门领导和相关人员的大力支持，在此作者深表感谢，同时感谢团队全体成员多年来的坚持和努力。本书编写过程中参考了大量已出版的教材，在此一并表示感谢。

编 者

2018年1月

目 录

第1章 初识C++程序设计语言

C++是一种面向对象的程序设计语言。本章从回顾程序设计语言的历程出发，叙述C++程序设计语言的产生、发展及特点。通过一个简单的C++程序引导读者初步认识C++程序设计。

1.1 计算机程序设计语言

计算机程序是人们为解决某个实际问题而编写的需要计算机完成的一系列操作指令的有序集合。程序设计语言是人与计算机交流的工具，是计算机可以识别的语言，具有特定的词法与语法规则。计算机程序设计语言能够使程序员准确地定义计算机需要使用的数据，并精确定义在不同情况下应当采取的行动。计算机语言从其发展历程看，可以分成机器语言、汇编语言、高级语言3个阶段，其中高级语言又可分为面向过程与面向对象的程序设计语言等。

1.1.1 机器语言与汇编语言

机器语言是直接用二进制代码指令表达的计算机语言，指令是由0和1组成的一串代码，它们有一定的位数，并分成若干段，各段的代码表示不同的含义。例如，某台计算机的字长为16位，即用16个二进制位表示一条指令或其他信息。16个0和1可组成各种排列组合，通过线路变成电信号，让计算机执行各种不同的操作。例如，将100与200相加的机器语言程序由下列两条指令实现。

```
1101 1000 0110 0100 0000 0000
0000 0101 1100 1000 0000 0000
```

虽然机器语言能被计算机直接识别和执行，但对于人类来说却十分晦涩难懂，更难以记忆与编写。在计算机发展的初期，程序员只能用机器语言编写程序，在这一阶段，计算机编程语言与人类的自然语言之间存在巨大的鸿沟，软件开发难度大、周期长，修改维护困难。

为了解决机器语言编程的困难，程序员使用类似英文缩写的助记符来表示指令，从而产生了程序设计的汇编语言（Assembly Language）。例如，使用ADD、SUB助记符分别表示加、减运算指令。将100与200相加的汇编语言实现如下。

```
MOV AX, 100
ADD AX, 200
```

使用汇编语言编写的程序，机器不能直接识别，要由一种程序将汇编语言翻译成机器语言，这种起翻译作用的程序叫汇编程序，汇编程序是系统软件中的语言处理系统软件，汇编程序将汇

编语言翻译成机器语言的过程称为汇编。汇编语言实质上仍是机器语言，同样属于低级语言。

汇编语言是面向具体机型的，它离不开特定计算机的指令系统，因此，不同型号的计算机有不同结构的汇编语言，而且，对于同一问题编制的汇编语言程序在不同类型的计算机之间是互不相通的。

虽然汇编语言比机器语言提高了一步，使编程语言与人类自然语言之间的鸿沟略有缩小，但仍然与人类的自然表达方式相差甚远，而且由于汇编语言的抽象层次太低，一个简单的任务需要大量的语句实现，程序员还需考虑大量的机器细节，因此使用汇编语言编程的难度仍然很大。

1.1.2 高级语言

为了进一步方便编程，人们开发了更加接近人类自然语言习惯的高级语言，程序使用更有意义和容易理解的语句，使程序更容易描述具体的事物与过程，编程效率大大提高。例如，仍然是将 100 与 200 相加，其高级语言可描述如下。

```
100+200
```

高级语言与计算机的硬件结构及指令系统无关，有更强的表达能力，可方便地表示数据的运算和程序的控制结构，能更好地描述各种算法，而且容易学习掌握。但用高级语言编译生成的程序代码一般比用汇编语言设计的程序代码要长，执行的速度也慢。

与汇编语言相比较，高级语言不但将许多相关的机器指令合成为单条指令，并且去掉了与具体操作有关，但与完成工作无关的细节，如使用堆栈、寄存器等，这大大简化了程序中的指令。同时，由于省略了很多细节，编程者也就不需要具备太多的专业知识。

使用高级语言编写的程序，需要相应的编译器翻译成机器语言程序才可执行。

1.1.3 面向过程与面向对象的程序设计语言

早期的计算机主要用于数值计算，其软件设计的主要工作是设计计算方法或解决问题的过程，因此早期的高级程序设计语言是一种面向过程的程序语言。随着计算机应用的日渐普及，人们需要利用计算机来解决更为复杂的问题，相应的程序软件也更加庞大，许多大型软件的开发遇到了严重的困难。20 世纪 60 年代产生的结构化程序设计方法为上述困难提供了较好的解决手段。

结构化程序设计方法的基本思想是：对要解决的整个问题采用自顶向下、分而治之、逐步求精的方法分解模块化功能。其程序结构按功能划分为若干基本模块，各模块功能尽可能地简单并且相对独立。每一个模块内部均是由顺序、选择和循环 3 种基本结构组成。在结构化计算机程序中，各模块以子程序或函数的方式设计，各模块之间的联系通过子程序或函数间的相互调用实现。

结构化程序设计方法将复杂的系统分解成易于实现和控制的子任务，显著减少了软件开发的复杂性，提高了软件的可靠性、可测试性和可维护性。结构化程序设计方法的一系列优点使得高级程序设计语言获得了更为广泛的应用，先后出现了 BASIC、ALGOL、COBOL、Pascal、Ada 和 C 语言，其中应用最为广泛、影响最大的是 C 语言。

结构化程序设计方法是面向过程的，其程序特点是描述问题的数据与解决问题的过程（数据处理的方法）相互独立，当数据结构改变时，所有相关的处理过程都要进行相应的修改。同时，由于图形界面的应用，使得软件开发过程越来越复杂，从而催生了面向对象的程序设计方法

（Object Oriented Program，OOP）。

面向对象程序设计方法的基本思想是：将描述问题的数据与解决问题的方法封装成一个不可分离的整体——对象。在面向对象的程序设计方法中，一个问题用一个对象来表示，对象内部包含了描述问题的数据以及对这些数据操作的方法。程序设计时，抽象出同类型对象的共性，形成类。类是抽象的“概念”，对象是类的实例。

正如结构化程序设计方法对计算机技术应用产生的巨大影响和促进那样，OOP方法更加强烈地影响、推动和促进计算机技术应用的更大发展。1986年在美国举行了首届“面向对象编程、系统、语言和应用（OOPSLA）”国际会议，使面向对象受到世人瞩目。现在，除了一些纯粹的面向对象的程序设计语言，如Smalltalk、Java等外，早期的一些高级程序语言也扩展了面向对象的功能。C++就是在C语言基础之上扩展出来的面向对象的程序设计语言。

1.2 C++程序设计语言

1.2.1 C++程序设计语言简介

C++语言是目前应用最为广泛的计算机程序设计语言之一。C++是由C语言扩充、改进而来的。C语言之所以要起名为C，是因为它主要参考B语言，C语言是B语言的进步，所以就起名为C语言。但是B语言并不是因为之前还有个A语言，而是B语言的设计者为了纪念其妻子，设计者妻子名字的第一个字母是B。当C语言发展到顶峰时，出现了一个版本叫C with Class，那就是C++最初的版本，在C语言中增加class关键字和类，那时有很多版本的C都希望在C语言中增加类的概念。后来C标准委员会决定为这个版本的C起个新的名字，在征集了很多名字后，最后以C语言中的++运算符来体现它是C语言的进步，故而叫C++，并成立了C++标准委员会。虽然C++是作为C语言的增强版出现的，就目前学习C++而言，它是一门独立的语言。读者可以完全不学C语言，而直接学习C++。

C++程序设计语言具有下列特点。

（1）C++完全兼容C，具有C语言的“简洁、紧凑、运算符丰富，可直接访问机器的物理地址，使用灵活方便，程序书写形式自由”等特点。大多数的C语言程序代码略做修改或不做修改就可在C++集成环境下运行。

（2）C++作为一种面向对象的程序设计语言，程序的各个模块间更具独立性，可读性更好，代码结构更加合理，设计和编制大型软件更为方便。

（3）用C++语言设计的程序可扩充性更强。

与其他高级程序设计语言一样，C++程序从开始编码到运行需要经过以下步骤。

（1）编辑源程序。可以在普通的文本编辑器（如Windows记事本）或一些专业开发软件（Dev C++、C-Free 等）提供的编辑器中对程序进行编码。由高级语言编写的程序称为源程序。C++源程序默认的扩展名为.cpp。

（2）编译源程序。使用编译程序对源程序进行编译，目的是将高级语言编写的源程序翻译成计算机硬件可以识别的二进制机器指令。源程序经编译后生成扩展名为.obj的目标程序文件。

（3）链接目标程序。用链接器将编译成功的目标程序文件与相应的系统模块链接成扩展名为.exe的可执行程序。

1.2.2 简单的C++程序框架结构

下面通过一个简单程序来了解C++程序的构成。这个程序的功能只是告知计算机显示“Hello World”。

【例1-1】 一个简单的C++程序示例。

```
/* ========================
   C++程序示例
 ======================== */
#include<iostream>                    //A，包含文件
using namespace std;                  //B，使用命名空间 std
int main() {                          //C，主函数
    cout<<"Hello World "<<endl;
    return 0;
}
```

结合上述程序示例，从以下几点粗略地介绍C++程序。

（1）程序注释。注释是程序员为程序所做的说明，是提高程序可读性的一种手段。注释并不是程序的必要部分，与其他高级语言一样，C++编译器在编译时将跳过注释语句，不对其进行处理。因此，无论源程序中有多少注释语句，均不会影响程序编译结果。

C++语言提供了两种程序注释方式：一种是界于符号“/*”和“*/”之间的内容均作为注释信息（如程序中的前三行），另一种是由符号“//”开始直至本行结束的全部内容（例如，程序中的A行、B行和C行）。

（2）文件包含。每个以符号“#”开始的行称为编译预处理指令。例1-1中的A行指令称为文件包含预处理指令。编译预处理是C++组织程序的工具。#include<iostream>的作用是在编译之前将文件iostream的内容插入程序中。iostream是系统提供的一个头文件，其中定义了C++程序输入/输出操作的有关信息，程序必须包含此文件才能进行输入/输出操作。

（3）命名空间。C++标准库中的类和函数是在std中声明的，如需要使用到其中的有关内容，就需要使用命名空间std编译。程序中的B行表示本程序使用系统提供的标准命名空间中的名称标识符。

（4）主函数。程序中的C行定义了一个函数，该函数描述程序的功能。main是函数名，其后紧跟一对圆括号。所有的C++程序有且只有一个main函数，通常称该函数为主函数。main函数是整个程序的入口，任何一个C++程序通常是从其主函数的第一条语句开始执行，执行完主函数的所有语句后，程序将自然结束。实现函数功能的语句序列必须用一对花括号括起来形成一个逻辑整体。

main前面的int表示该函数运行结束后将得到一个整数值，该整数值应该在函数执行结束前用return语句给出。在例1-1程序中，主函数最后一条语句表明，如果程序正常运行结束，将返回一个整数值0。main后的一对圆括号说明main函数运行所需的参数。例1-1中main函数后是一对空圆括号，说明本程序运行时无需提供参数。此时，也可以在圆括号中加void。

（5）信息输入/输出。在C++程序中，标准的输入/输出操作使用关键字cin或cout。

（6）程序语句。基本的C++功能语句都必须以分号结束。

（7）程序编写风格。从语法上讲，C++程序代码编写格式自由，甚至可以将多个语句写在同一行。但为了增加程序的可读性，建议在编写程序时遵守下列规则。

① 每行写一条语句，同一层次的代码左对齐。

② 配对的花括号中，上花括号“{”紧跟在上一行末尾，下花括号“}”单独另起一行，并且缩进层次同配对的上花括号“{”。花括号内的内容缩进在下一层次。

（8）源程序编译运行。编写完源程序后，将源程序文件存储为扩展名为.cpp 的文件（如果在C++编译环境提供的编辑器中编辑源程序，则编译时自动存盘）。

编辑完源程序，还需要通过编译环境进行编译和链接后才能运行程序。上述示例程序运行后输出如下。

```
Hello World
Press any key to continue. . .
```

程序执行结果在最后增加一行输出“Press any key to continue. . .”，这是系统自动添加的，目的是让用户看清屏幕输出内容，并提醒用户按任意键后程序将退出并返回到原编程环境。有些汉化的编译系统会以中文提示用户“按任意键继续. . .”。

通过上面的例子可以看出，一个简单的C++程序结构如下。

```
#include<iostream>
using namespace std;
int main(void) {
    ……
    return 0;
}
```

读者只需要将上面框架结构中的“……”替换为自己需要的功能语句，就可以改写为自己的C++程序了。

需要特别提出的是，因为C++程序代码是大小写敏感的，所以在书写程序时要注意其大小写，如函数名 main 不能写成 Main。同时，程序中语法部分不能出现中文字符（含标点符号）。

1.2.3 标准命名空间

命名空间（又称名字空间、名称空间或名域）的关键字为 namespace。

在C++中，名称（标识符）可以是符号常量、变量、宏、函数、结构体、枚举类型、类和对象等。为了避免标识符的命名相互冲突，标准C++引入了命名空间来控制标识符的作用域，不同的命名空间中可以有同名的标识符而不相冲突。

标准C++库提供的标识符都放在标准命名空间 std 中，使用命名空间 std 有以下几种方法。

（1）利用 using namespace 声明所使用的命名空间。如例 1-1 所示，程序头部使用以下语句。

```
using namespace std;
```

这是最常用的一种声明命名空间的方法，它表明此后程序中的所有系统标识符如果没有特别说明，均来自命名空间 std。

（2）用作用域运算符“::”标明标识符所属的命名空间。例如，例 1-1 程序代码可以改写成下列形式。

```
#include<iostream>
int main(void) {
     std::cout<<"Hello World "<<std::endl;
     return 0;
}
```

由于程序中没有用语句“using namespace std;”声明使用的命名空间，所以程序中使用的每一个系统标识符都必须用 std::说明。例如，程序中的 cout 应改写为 std::cout，endl 应改写为 std::endl。

（3）用 using 声明某个标识符的命名空间。例如，例 1-1 程序代码也可以改写成下列形式。

```
#include<iostream>
using std::cout;                    //A
using std::endl;                    //B
int main(void) {
     cout<<"Hello World "<<endl;
     return 0;
}
```

上述程序中 A 行和 B 行分别声明了标识符 cout 和 endl 的命名空间，表示程序中使用的标识符 cout 和 endl 均默认为来自命名空间 std。

早期的 C++标准不支持命名空间，因此程序中不需要声明使用的命名空间。C++早期的头文件都带扩展名“.h”，新版本为了与老版本兼容，也附带了这些头文件。如果用早期的头文件，例 1-1 也可以写成如下形式。

```
#include<iostream.h>
int main(void) {
     cout<<"Hello World "<<endl;
     return 0;
}
```

上述程序中使用了老版本带扩展名的头文件，因而不需要再声明命名空间。

1.3 习 题

1. C++程序中有几种注释方法？
2. 仿照例 1-1，设计一个程序，输出自己的学号、姓名、家庭住址等信息。

第2章 C++语言编程基础

学习C++语言编程，首先需要掌握其基础知识。本章将详细介绍C++语言基础，包括基本数据类型、变量和常量、运算符和表达式、流程控制语句，最后介绍简单的输入/输出方法。

2.1 C++语言数据类型

具有丰富的数据类型是C++语言的特点之一，这使得C++语言可以处理各种不同的数据。C++语言的数据类型可分为基本数据类型与构造数据类型。基本数据类型是指C++语言已经预定义、不可进一步分割的数据类型，构造数据类型是指由一种或几种数据类型组合而成的数据类型。C++语言的数据类型如图2-1所示。

图2-1 C++语言的数据类型

本章介绍基本数据类型，构造数据类型将在后面章节中分别介绍。

2.1.1 标识符

标识符是C++语言中用来表示常量、变量、函数等实体名称的有效字符序列，这里的有效字符是指键盘上除“@”“`”“$”外的所有可显示字符。C++语言标识符有两种：关键字和自定义标识符。

关键字也称保留字，是程序设计语言中约定已具有某种特定含义的标识符，不可以再作为其

他用途。表 2-1 列出了 C++语言中常用的关键字。

表 2-1 C++语言中常用的关键字

int	double	if	for
char	float	else	while
void	const	switch	do
long	short	break	return
this	struct	continue	private
inline	case	union	protected
operator	default	enum	public
virtual	auto	class	friend
static	extern	signed	delete
register	typedef	unsigned	new

自定义标识符是指在编程中，用户根据需要为变量、函数等对象起的名称。作为名称的标识符必须由字母、数字、下画线组成，且不能以数字开头。下面列出的均为合法的自定义标识符：

```
Student_003,Float,str4,_345,year,name
```

下面是不合法的自定义标识符：

```
float,003_Student,-345,$Abs,C++,A.B,π
```

命名自定义标识符时要注意如下几点。

（1）通常使用具有某种含义的标识符，尽量做到见名知义。

（2）不可以将关键字作为自定义标识符。

（3）C++语言中严格区分大小写。例如，myName 与 myname 是两个不同的标识符。

2.1.2 基本数据类型

C++语言基本数据类型包括布尔型（bool）、整型（int）、单精度实型（float）、双精度实型（double）、字符型（char）、空类型（void）等，如表 2-2 所示。在 C++语言中，不同类型的数据具有特定的精度、取值范围，在内存空间中所占字节数也有区别。

表 2-2 C++语言常用的基本数据类型

名 称		类 型	长度/字节	取 值 范 围
布尔型		bool	1	true 或 false
字符型		char	1	-128～127
整型		int	4	-2^{31}～$2^{31}-1$
实型	单精度	float	4	-3.4×10^{38}～3.4×10^{38}
	双精度	double	8	-1.7×10^{308}～1.7×10^{308}
空类型		void	0	无值

为了更准确地描述数据类型，C++语言还提供了 4 个关键字 short、long、unsigned 和 signed 用来修饰数据类型，相应的基本数据类型如表 2-3 所示。

表 2-3 C++语言中经修饰的基本数据类型

名　称	类　型	长度/字节	取值范围
无符号字符型	unsigned char	1	0～255
短整型	short int	2	-32768～32767
长整型	long int	4	-2^{31}～$2^{31}-1$
无符号短整型	unsigned short int	2	0～65535
无符号长整型	unsigned long int	4	0～$2^{32}-1$
长双精度型	long double	10	-1.7×10^{308}～1.7×10^{308}

有关数据类型的几点说明如下。

（1）数据的类型是对数据的抽象，决定数据所占内存的字节数、数据的取值范围以及其上可进行的操作。程序中使用的所有数据都必定属于某一种数据类型。

（2）布尔型又称逻辑型，有真（true）和假（false）两种状态，用整型值 1（true）和 0（false）参与运算。

（3）整数类型的数据，第一位为符号位，0 表示正数，1 表示负数，其余各位用于表示数值本身。无符号整型数据只能表示非负数，没有符号位。

（4）字符型数据按 ASCII 码存储，可以像整型数据一样参与运算。

（5）实型数据均为有符号数据，不可以用 unsigned 修饰。

（6）空类型又称无值型、空值型，意为未知类型，可以用于函数的形参、函数类型、指针等，但不能说明空类型的变量。

2.2 常　量

常量是指在程序运行中，其值不能被改变的量，包括字面常量和符号常量两种。字面常量又包括整型常量、实型常量、字符型常量、字符串常量，在程序运行时，字面常量直接参与运算，不占用内存存储空间。

2.2.1 整型常量

C++中的整型常量有十进制（D）、八进制（Q）、十六进制（H）3 种形式。默认为十进制常量，八进制常量以“0”开头，十六进制常量以“0x”或“0X”开头。例如：

```
23,+74,-18,0123,-057,0,-0x5a,0X94BC
```

还可以加后缀“L”（或“l”）和“U”（或“u”）表示长整型数或无符号型整数常量，如 0X94BCL，23l，0123U，0u 等。

2.2.2 实型常量

实型常量又称浮点小数。在 C++中实型常量只使用十进制表示，有两种表示形式：一般形式和指数形式。

用一般形式表示时，整数和小数间用小数点隔开，加后缀“F”（或“f”）表示 float 型常量，

默认为 double 型常量，也可以用“L”（或“l”）表示长双精度型实型常量。例如：

```
16.3,-13.5f,.34, 62.,-76.43L,93.88F
```

指数形式表示由尾数和指数两部分组成，中间用“E”（或“e”）隔开，其中指数部分只能是十进制整数。例如，0.573E+2 表示 0.573×10^2，-34.9E-3 表示-34.9×10^{-3}，1e4 表示 1×10^4，.3e1 表示 0.3×10^1。当以指数形式表示一个实数时，指数部分不可以省略，且应为整数，尾数的整数部分和小数部分可以省略其一，但不能都省略。例如，.263E-1 和 68.E2 都是正确的，而 E-6、5e、42e5.5 等都是错误的。

2.2.3 字符型常量

字符型常量简称字符常量，是用单引号括起来的一个字符，在内存中用 ASCII 码形式存储，占一字节，分普通字符型常量和转义字符型常量。如'a'、'8'、'?'、'$'、' '等都是普通字符型常量，其中最后一个为空格字符常量。

转义字符型常量用来表示一些不可显示，也无法通过键盘输入的特殊字符，如响铃、换行、制表符等。转义字符以“\”开头，后面跟表示特殊含义的字符序列。表 2-4 列出了常用的转义字符。

表 2-4　　常用的转义字符

字符形式	含　义
'\n'	换行
'\a'	响铃
'\t'	水平制表符（横向跳格，相当于 Tab 键）
'\v'	竖向跳格
'\0'	空字符
'\\'	反斜杠\
'\''	单引号'
'\"'	双引号"
'\ddd'	1～3 位八进制数
'\xdd'	1～2 位十六进制数

表 2-4 中最后两行是所有字符的通用表示方法，即可用八进制数或十六进制数表示，其值转换成十进制数必须在 0～255。例如，'\160'、'\x70'都表示字符'p'，这里的 160 是八进制数，x70 为十六进制数，转换成十进制数为 112，不超过 255。

2.2.4 字符串常量

字符串常量是用双引号括起来的若干字符，简称字符串。例如：

```
"We are studends. ","0234","-546.5UL","re\x70\t67\160\n0\0"
```

字符串中所含有效字符的个数称为字符串的长度，如字符串"I am a student."的长度为 15，字符串"0234"的长度为 4，字符串"-546.5UL"的长度为 8，字符串"re\x70\t67\160\n0\0"的长度为 9。

字符串常量在内存中是按顺序逐个存储串中字符的 ASCII 码，通常占用长度加 1 字节的内存