赵国瑞　汪大菊　陆明　主编
毕玉玲　主审

计算机软件技术基础

——C++、数据结构、软件工程

（第2版）

天津大学出版社
TIANJIN UNIVERSITY PRESS

计算机软件技术基础

—C++、数据结构、软件工程

（第2版）

赵国瑞　汪大菊　陆明　主编

毕玉玲　主审

天津大学出版社

内 容 提 要

本书由两部分组成:第1部分介绍C++程序设计语言,在介绍了使用C++进行面向过程的结构化程序设计的基础上,详细阐明了使用C++进行面向对象的程序设计方法;第2部分介绍数据结构和软件工程。每章内容由浅入深,并含有大量的程序例题供读者自学。

本书主要是为非计算机专业的本科生学习计算机软件技术基础课程而编写的,也可作为高职和自考学生学习C++语言基础的教科书,还可供各类计算机软件人员和软件开发人员参考。

图书在版编目(CIP)数据

计算机软件技术基础:C++、数据结构、软件工程/赵国瑞主编.—天津:天津大学出版社,2002.9
ISBN 7-5618-1634-0

Ⅰ.计… Ⅱ.赵… Ⅲ.①C语言-程序设计-高等学校-教材②数据结构-高等学校-教材③软件工程-高等学校-教材 Ⅳ.TP31

中国版本图书馆CIP数据核字(2002)第052775号

出版发行 天津大学出版社
出 版 人 杨风和
地 址 天津市卫津路92号天津大学内(邮编:300072)
网 址 www.tdcbs.com
电 话 营销部:022-27403647 邮购部:022-27402742
印 刷 保定市印刷厂
经 销 全国各地新华书店
开 本 185mm×260mm
印 张 24
字 数 600千
版 次 2002年9月第1版
印 次 2002年9月第1次
印 数 1—5 000
定 价 32.00元

第 2 版序言

自出版以来，本书已被许多院校选为非计算机专业的计算机基础教学的教科书使用，受到了广大教师和同学的好评。本书还荣获 2000 年度天津大学优秀教材二等奖。但是，编者也感到本书在许多方面尚不能满足同学们的学习要求和软件开发人员对本书的期望，同时，也需要反映 C++ 语言近年来的重大发展。因此，值此发行第 2 版之际，我们以 ISO/ANSI 1998 C++ 程序设计语言标准为依据，花了近一年的时间，对原书的内容和结构进行了重大改动。主要改动如下：

①全书分为两部分，第一部分讲解 C++ 的结构化程序设计和面向对象的程序设计，第二部分介绍数据结构和软件工程；

②完全重写了 C++ 面向对象程序设计部分，内容更加充实，叙述更加深入，例题更加丰富，篇幅也从原书的一章扩展成三章。

③删去了原书 C 语言外部文件的使用方式，改为 C++ I/O 流库方法；

④更加强调面向对象的程序设计，不仅给出了 C++ 对 C 语言的主要扩展，而且充实了C++ 面向对象特征的描述，如模板、虚基类、友元类、静态成员、虚函数、抽象类、I/O 流库、异常处理等都给出了详细说明；

⑤在本书各部分都补充了大量例题，使概念和算法更加清晰，也更便于自学，同时统一了全书的风格——每章末均有小结及例题；

⑥软件工程部分在章末给出了一个较复杂的 C++ 程序例子，该程序具有多个类与继承、子对象使用、菜单驱动、查找、排序、对象数据的输入、对外部文件的存取等多种功能（限于篇幅，例中未给出需求分析和设计过程等内容），学习本例可以使读者尤其是初学者对建立一个实用程序有更深刻的体会；

⑦同时编写了与本书配套使用的《计算机软件技术基础实验指导与习题解答——C++、数据结构、软件工程》。

参加本书第 2 版修编的有汪大菊、陆明、赵国瑞、孙桂茹、陆琪、张玉莉，并请毕玉玲担任主审。最后由赵国瑞定稿。苗君秋录入了本书的大部分例题。

如果讲授本书全部内容，需要讲课 48 学时，实验 24 学时，课外上机 40 学时（每学时 45 分钟）。讲授时最好使用电子教案或课件进行多媒体教学。建议讲课学时数分配如下：第 1 章 3 学时，第 2 章 3 学时，第 3 章 6 学时，第 4 章 4 学时，第 5 章 2 学时，第 6 章 4 学时，第 7 章 4 学时，第 8 章 2 学时，第 9 章 6 学时，第 10 章 6 学时，第 11

章4学时，第12章4学时。课程学时数较少的专业，可以略讲或简介部分内容，如虚基类、抽象类、模板、异常处理、流插入和提取运算符重载、哈夫曼树、最短路径、关键路径、堆排序等。如果不采用电子教案或课件，则讲课学时数需增加三分之一。

在本书编写过程中，得到了许多教师和学生的支持和帮助，有许多好的意见被采纳，在此一并致谢。

由于作者水平所限，错误和不当之处在所难免，敬请广大读者指正。编者的电子邮箱地址为：

guoruizhao@163.com

编者

2002年5月

目 录

第 1 部分 C++ 程序设计

第 1 部分　C++ 程序设计

第 1 章　C++ 程序设计基础

1.1　C++ 语言概述

1.1.1　C++ 与程序设计

语言是人类交流思想的工具。在人和计算机打交道的时候，要让计算机按人们预先安排的步骤进行工作，就要解决人和计算机进行交流的问题。人和计算机进行交流的语言，称为计算机语言。

最早的计算机语言称为机器语言。机器语言是一条条的二进制形式的指令，每一条二进制指令表示一个功能，例如取数、加运算等。由计算机专业人员用这些指令编写的程序难读、难写、难修改。为简化机器语言，人们用符号代替机器语言中二进制形式的指令，这种便于记忆的符号语言称为汇编语言。用汇编语言写出的程序称为汇编源程序。汇编源程序上机运行时必须通过一个“汇编程序”将汇编源程序翻译成二进制的指令，机器才能执行。汇编语言的出现，为程序设计人员提供了很大方便。但用汇编语言编写程序同样是一件繁琐的工作，它需要程序设计人员了解计算机的许多硬件细节，因而影响了计算机的推广、应用。高级语言的出现为广大非计算机专业人员应用计算机提供了极大的方便。目前常用的高级语言有若干种，例如 BASIC、FORTRAN、PASCAL、C 及 C++ 、JAVA 等。用高级语言编写的程序称为源程序。计算机不能直接执行源程序，必须经过“编译程序”或“解释程序”将源程序翻译成机器指令，机器才能执行。不同的高级语言有不同的编译程序或解释程序。

因为计算机语言是程序设计使用的语言，所以又称为程序设计语言。

程序设计就是将解决某个问题的过程用程序设计语言描述出来，计算机按这个描述去逐步实现。不同的高级语言进行程序设计的方法不同。因此，从程序设计方法的角度来看，高级语言中的 FORTRAN、PASCAL、C 等都是面向过程的程序设计语言；而 C++ 、JAVA 等在面向过程语言的基础上增加了面向对象的语言内容，常称为面向对象的程序设计语言。面向对象的程序设计方法被认为是最有希望、最有前途的方法。它是为适应计算机发展，特别是操作系统等软件资源的发展而产生的。面向对象的程序设计方法是对面向过程的程序设计方法的一次革命。这两者的根本区别是：在面向过程的程序设计中，程序设计人员把重点放在解决某个问题的过程上；而在面向对象的程序设计中，设计人员把着眼点放在解决“什么”问题上，而不是问题的解决过程上，也就是只需关心一个对象能做什么，而不必关心对象的内部构成，从而使

程序设计人员以更开阔的视野来观察问题、解决问题，使计算机的求解过程更接近人的思维活动，从而可以更充分发挥计算机系统的潜在能力。

C++是在C语言基础上为支持面向对象的程序设计研制的程序设计语言。C语言的前身是1967年英国剑桥大学Matin Richards推出的BCPL语言。1970年美国贝尔实验室的Ken Thompson以BCPL为基础，设计出简单而又很接近硬件的B语言。1973年贝尔实验室的D.M. Ritchie在B语言的基础上保留了它简练、更接近硬件等特点，设计出了C语言。特别是用C语言成功地改写了UNIX操作系统，从而使C语言迅速得到推广，先后被移植到大、中、小型机及微机上。C语言是一种优秀的程序设计语言。它既有高级语言的优点，也有低级语言的特点，多年来受到普遍欢迎。C++是C语言的扩充，它保留了C语言高效灵活、易于理解、可移植性好等优点，又克服了C语言类型检查不严格，以及程序规模较大时很难查找和排除错误等缺点，同时增加了面向对象的特征，并扩充了许多新功能，使得C++成为目前最流行的程序设计语言。

由于C++与C兼容，从而使许多C程序代码和用C语言编写的大量的库函数不经修改就可以在C++中使用。正是由于C++是一个C的扩充而不是一个纯正的面向对象的语言，所以C++既支持面向过程的结构化程序设计，又支持面向对象的程序设计。从1980年由美国贝尔实验室的Bjarne Stroutstrup提出C++以后，C++几经修改完善，并于1998年颁布了ISO/MSI C++标准。目前，C++语言仍在不断发展之中。

1.1.2 C++语言和面向对象的程序设计

C++与C的最大区别是支持面向对象的程序设计方法。面向对象的程序设计方法简称OOP(object-oriented programming)，是程序设计中的新概念，是结构化程序设计以及模块化、数据抽象等方法的发展。因此，作为程序设计者，不仅要学习结构化程序设计方法，而且要学习面向对象的程序设计方法。

在面向对象的程序设计中首先要理解“对象”一词的含义。客观世界中的任何事物都可称为对象。这些事物小到一个螺丝钉、一本书，大到一个工厂、一个学校。对象是具有某种特性和某种功能的东西。对于使用者来说，选择一个对象主要不是考虑对象内部是怎样构成的，而是考虑对象能做什么，也就是说它的功能是什么。就如同使用电视机的用户那样，只要知道怎样操纵各按钮就行了，具体电视机内部的构造用户不必关心。面向对象是一种崭新的方法论，用这种方法观察世界，将客观世界看成由许多不同种类的对象组成，每种对象有特定的特征和操作，不同对象的相互作用构成了完整的客观世界。

把具有共同特征(属性)和操作(方法)的对象抽象出来，形成了类。类是面向对象程序设计中最重要的概念。显然，对象是类的具体化，是某个类的一个实例。正因为引入类和对象，才使得面向对象的程序设计具有封装性、继承性和多态性。

在程序设计中，封装是指将数据和与这些数据有关的操作结合在一起，形成一个有机的整体。封装是一种信息隐藏技术，通过封装将数据和其有关的行为隐蔽起来，而将一部分行为作为对外的接口。从用户或应用人员的角度看，对外的接口就是为其提供的操作功能，即为一组服务，而服务的具体实现，即对象的内部却被隐藏着。在C++中是利用类的形式来实现封装的。

继承是面向对象程序设计中最重要的特征。继承所表达的是一种类之间的相互关系。它

使某个类除了具有特有的属性和方法外，还可以继承其他类的属性和方法。在实际工作中，一个特定问题的认识和处理，往往前人已经进行过较为深入的研究和探讨，他们的研究成果后人可以利用，并且在此基础上有所发展和创造，这正是社会发展的过程。C++ 提供的类的继承机制，允许程序设计人员在保持原有类的基础上，通过继承进行扩充，从而成为新的类。

多态性也是面向对象的程序设计中的一个重要特征。在 C++ 中，程序设计人员用向一个对象发送消息来完成一系列的动作。而多态性是指不同的对象接收到相同的消息时，会产生不同的动作。C++ 语言支持两种多态性，即编译时的静态多态性和程序运行时的动态多态性。静态多态性通过函数重载实现，动态多态性通过虚函数机制实现。

1.1.3　典型 C++ 环境

一个 C++ 系统通常由程序开发环境、语言和 C++ 标准库等部分组成。

在 C++ 程序开发环境中，一个 C++ 程序要经过编辑(edit)、预处理(preprocess)、编译(compile)、连接(link)、装入(load)和执行(excute)等六个阶段。使用像 Microsoft Visual C++ 6.0 等的 Windows 下的集成程序开发环境(IDE)，比使用在 UNIX 下的 GNU 等程序开发环境要简单，除了编辑源程序要由用户自己完成外，其他阶段只需要执行一个命令就可以由 C++ 系统自动完成。

一个在 Windows 下的典型 C++ 程序开发环境的各阶段及其关系如图 1.1 所示。

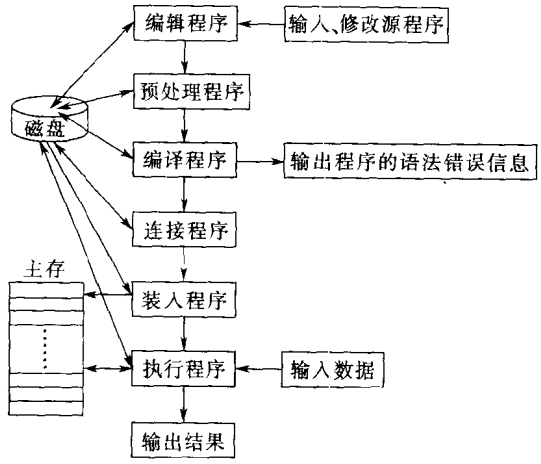

图 1.1　典型 C++ 环境

在编辑阶段，使用编辑程序输入、修改源程序，并存入磁盘中，文件名由用户指定。文件的扩展名一般为.cpp(源程序)或.h(头文件)。

预处理、编译两个阶段是用一个编译(compile)命令自动先后完成的。预处理阶段处理各种预处理命令，并生成一个临时的 C++ 源程序文件存入磁盘中，交由编译阶段进行语法检查和语义分析。如果发现语法错误，则显示尽可能多的错误信息，以便用户查找和修改错误后再次编译。如果未发现语法错误，则将生成目标程序文件并存入磁盘(扩展名一般为.obj)。

在连接阶段，连接程序把编译阶段生成的目标程序(可能有多个，一个.cpp 文件生成一个目标文件)与 C++ 的标准库的代码拼装成一个完整的可执行程序文件并存入磁盘(扩展名一

般为.exe)。

装入程序把可执行程序从磁盘放入主存,并做好执行前的一切准备。如把代码段、各种数据段的首地址存入相关寄存器中。然后,在 CPU 的控制下执行该程序,遇到从键盘读入语句时,便等待用户输入数据并完成提交(一般按下回车键)。程序执行期间也可能存取磁盘中的外部数据文件。当程序执行中发生错误时,如除数为 0 及负数取对数或开平方等,计算机将显示出错信息。如果程序正常执行结束,则按程序中输出语句的要求输出结果。

在上述各阶段中都可能出现错误。语法错误是最常见的,但这类错误可以由编译程序找出来,而且常常由于前面一个语法错误而引起后面很多语法错误。这类错误通过修改源程序很容易排除。运行期间会出现致命错误而使程序中止执行(如负数取对数等)。有些错误通过仔细分析程序流程也比较容易查找和排除。最难以查找和排除的错误是属于程序逻辑上的错误(如不慎将"=="写为"="或"<="写为"<")和程序运行时发生的非致命错误(如数组下标越界)。所以,常常需要从后面的某个阶段返回到前面的编辑等阶段。

如果使用 UNIX 等系统,上述各阶段都有相应的命令,请参看系统手册或请教他人。

1.2 C++ 程序实例

这里给出几个简单的 C++ 程序实例,以便先对 C++ 程序有感性认识。

例 1.1

```
/*   Hello   program   */
#include <iostream.h>
void main()
{    cout << "Hello, you are welcome!";          //你好,欢迎你!
}
```

当在计算机上执行这个程序后,显示如下内容:

```
Hello, you are welcome!
```

程序说明如下。

①程序中第 1 行"/* Hello program */"为注释。在 C++ 中有两种注释形式。一是由"/*"开始到"*/"为止,中间的所有内容为注释。这种注释可以占若干行,可以出现在程序的任何地方。另一种注释形式是用//开始,//后的所有内容都是注释,该注释内容直到本行结束处终止。如例 1.1 程序第 4 行的"//你好,欢迎你!"。//可出现在程序的任何行后,也可单独占一行。一般前者常用于较长内容的注释,而后者多用于较短内容的注释。程序中插入注释为阅读程序提供了方便,但编译系统并不处理它。

②程序第 2 行 #include <iostream.h> 是编译预处理行。其中 iostream.h 为一文件名。该文件是个系统库文件,文件中定义了程序中使用的"cout"和"<<"操作运算的有关信息。"cout"和"<<"是系统提供的输出操作。当在程序中使用系统提供的输入、输出、数学函数运算等操作时,必须在程序开始处嵌入相关文件,称为包含文件或头文件。C++ 系统手册会给出各种库文件所提供的功能。注意,这样的行必须独占一行。

③从第 3 行开始的结构

```
void main( )
```

```
{
   ⋮
}
```

是一个函数。其中 main 是个特殊函数名。每个 C++ 程序必须有且只能有一个main( )函数。所有 C++ 程序都从 main 函数开始执行,程序中的其他操作都是由 main( )函数调用或激活的。void 说明该函数的类型。void 表示该函数执行后没有任何返回值。main( )函数的缺省类型为整型。花括号{…}之间的内容为函数体。函数所需要的语句序列或过程都在两花括号之间。花括号可以不单独占一行。

④第 4 行的 cout 是标准输出,它表示的标准输出设备一般是屏幕。“ << ”是个输出流插入运算符。和 cout 连用时,它表示把右边的内容在屏幕上显示出来。因为显示的内容是字符串,所以需要用双引号将“Hello,you are welcome!”括起来。

⑤第 4 行字符串"Hello, you are welcome!"后有一分号“;”作为一个语句的结束标志。在 C++ 源程序中,一行可以写若干条语句,一个语句也可写在多行上,每个语句后必须用一个分号作为语句的结束。

⑥在 C++ 程序中可以从每一行的任意位置开始书写语句,不影响程序的执行。

例 1.2

```
#include <iostream.h>
void main()
{  int a,b,n;
   cout <<"Input a and b:";
   cin >> a >> b;
   n = a + b;
   cout <<"a = " << a << endl;
   cout <<"b = " << b << endl;
   cout <<"a + b = " << n << endl;
}
```

程序说明如下。

①程序第 3 行的“int a,b,n;”为变量定义,定义 a、b、n 为整型变量。在 C++ 程序中使用的变量都要在使用前定义。

②第 5 行的“cin >> a >> b;”中的 cin 代表标准输入设备,一般指键盘。“ >> ”是提取运算符。当它和 cin 联用时,接收用户从标准输入设备上输入的数据,并把接收到的数据赋给“ >> ”右边的变量。在一个 cin 语句中,“ >> ”可以连续使用多个。本例表示将从键盘上输入的第 1 个数送给变量 a,第 2 个数送给变量 b。从键盘上输入数据时,以空格、制表符或回车作为数据的分隔符。

③第 6 行是赋值运算,表示将从键盘上输入的 a 和 b 相加,结果存放在变量 n 中。

④第 7 行、第 8 行和第 9 行是标准输出,其中“a = ”、“b = ”和“a + b = ”原样输出。a、b 和 n 是变量,表示将变量中的内容在输出设置上输出。第 7、8 行和第 9 行中的 endl 为换行符,用来开始一个新行。若试图将下一个内容在新一行输出,可以使用 endl。endl 也可以用“' \n'”代

替,效果一样。例如:

```
cout << "a = " << a << ' \n';
```

例 1.2 程序执行的示例如下:

```
input a and b:123  456
a = 123
b = 456
a + b = 579
```

例 1.2 程序也可以写成如下形式:

```
#include <iostream.h>
void main()
{  int a,b;
   cout << "Input a and b:";
   cin >> a;
   cin >> b;
   int n = a + b;
   cout << "a = ";
   cout << a;
   cout << endl;
   cout << "b = ";
   cout << b;
   cout << ' \n';
   cout << "a + b = " << n << endl;
}
```

这个程序的运行结果与例 1.2 完全相同。注意这个程序中变量 n 的说明位置。C++ 允许根据需要随时说明新变量。

下面例 1.3 和例 1.4 读者可参考上述两例自行写出程序的运行结果。

例 1.3

```
#include <iostream.h>
void main(  )
{  int age;
   cout << "Enter your age:";
   cin >> age;
   cout << "your age is " << age << endl;
}
```

例 1.4

```
// This is a c++ sample
#include <iostream.h>
```

```
void main()
{   int a,b;
    a = 10;   b = 20;
    cout << "a-b = " << a-b;
}
```

例 1.5　由两个函数组成的 C++ 程序。

```
# include <iostream.h>
int add1(int a, int b)
{   int sum;
    sum = a + b;
    return sum;
}
main( )
{   int v1, v2;
    v1 = 1020;
    v2 = 2035;
    cout << "sum = " << add1(v1, v2) << endl;
    return 0;
}
```

执行例 1.5 程序时,在主函数 main( )的 cout 中调用前边定义的 add1 函数。将 add1 函数执行后返回的结果在主函数的 cout 中显示输出。一个 C++ 程序可以由若干个函数组成,程序由 main( )开始执行,一般结束也是在main( )。注意,该程序中的 main( )函数为整型,故在程序结束前返回一个整数 0。

学习程序设计的过程,总是由模仿开始,逐步深入,最后才能创造性地设计出自己的程序。这里的关键是要亲自动手编写程序,并且一定要上机调试、运行程序,从中发现问题、积累经验,使程序设计能力不断提高。

用 C++ 语言编写的程序称为 C++ 源程序,在 C++ 开发环境下被存储为扩展名 .cpp 的源程序文件,经过编译连接后产生扩展名为 .exe 的可执行文件。开发环境如何使用请参考与本书配套的实验指导书。

1.3　基本数据类型

数据是程序处理的对象。根据数据的特点,可将数据分为不同的类型。类型不同,存储方式不同,使用的场合也不同。程序中使用的数据必须属于某种数据类型。C++ 数据类型可以分为两大类,即基本数据类型和用户自定义的数据类型。这里先介绍基本数据类型。

基本数据类型是 C++ 系统已定义的类型。可以直接利用这些类型名定义数据。表 1.1. 给出 C++ 基本数据类型的名称,以及在计算机内所占的位数(二进制)和数据的取值范围。

表 1.1 C++ 基本数据类型

类型名	名 称	位数	取值范围
bool	布尔型	8	true,false
char	字符型	8	ASCII 字符集中所有字符
singned char	有符号字符型	8	-128 ~ 127
unsigned char	无符号字符型	8	0 ~ 255
short int	短整型	16	-32 768 ~ 32 767
signed short int	有符号短整型	16	-32 768 ~ 32 767
unsigned short int	无符号短整型	16	0 ~ 65 535
int	整型	16	-32 768 ~ 32 767
signed int	有符号整型	16	-32 768 ~ 32 767
unsigned int	无符号整型	16	0 ~ 65 535
long int	长整型	32	-2 147 483 648 ~ 2 147 483 647
signed long int	有符号长整型	32	-2 147 483 648 ~ 2 147 483 647
unsigned long int	无符号长整型	32	0 ~ 4 294 967 275
float	浮点型	32	±(3.4E-38 ~ 3.4E+38)
double	双浮点型	64	±(1.7E-308 ~ 1.7E+308)
long double	长双浮点型	80	±(3.4E-4932 ~ 1.1E+4932)

对表 1.1 中的基本数据类型说明如下。

①关键字 signed 和 unsigned 以及 short、long 被称为类型修饰符。使用 signed 修饰的数据类型称为有符号类型,这种类型的数据可以取正数、负数和 0。使用 unsigned 修饰的数据类型称为无符号类型,这种类型的数据仅能取正数和 0。没有修饰符的数据类型都是有符号的类型。

②用 short 修饰的数据类型的值小于或等于 int 类型的值域,但目前绝大多数编译系统都是等于 int 类型的值域。用 long 修饰的 int 数据类型的值大于或等于 int 类型的值域,具体值域的大小取决于具体的机器系统和所用的 C++ 编译系统。例如,在 32 位微机及 32 位 C++ 编译系统中,int 和 long 类型均占用 32 位。

③用 short、long、unsigned 修饰 int 时可以省略 int。

1.4 常量、变量及类型说明

1.4.1 常量

程序中可以直接使用的数据称为常量。常量分为整型、浮点型、字符型以及字符串常量和布尔常量。

(1)整型常量

C++ 可以处理的整型常量如下。

1)十进制 由0~9数字组成的正负整数,如0、15、-237。

2)八进制 以数字0开始的为八进制数。八进制数由数字0到7组成,如06,-0234,-0512。

3)十六进制 以0x或0X开始的为十六进制数。十六进制数由数字0~9和字母a~f(或大写A~F)组成。例如:0x173,0x8A,-0xB2F。

当任一整型常数后跟字母l(或L)时,表示是长整型。若任一整型常数后跟字母u(或U)时,为无符号整型数。例如:123456l,7896u。

(2)浮点型常量

浮点型又称实型,由整数部分和小数部分组成。浮点型只有十进制形式。浮点型常数有两种表示形式。

1)小数形式浮点型 如3.1416,-0.5,-123.0。

2)指数形式浮点型 如+5.2507e-10,-25.678e+5。其中+5.2507e-10表示5.2507×10^{-10},-25.678e+5表示-25.678×10^{5},字母e也可以大写。

浮点型数中的"+"号可以省略,如0.567e10;"-"不可省略,如-12e15。

浮点型数中e前可以是整数或小数,但e后的指数部分必须是整型数。

浮点型数总是按double类型存储的,只有在数的后面加上f才按floot类型存储。

(3)字符型常量

字符型常量是用单引号括起来的单个字符,例如'A'、'S'、'*'、'a'等。其说明如下。

①字符型常量中的单引号只作为定界符,不是字符型常量内容。

②字符型常量具有数值,其值就是该字符的ASCII码值。在C++中,字符型常量值可以作为整数参与运算,例如'a'+5表示将'a'字符的ASCII码值与整数5相加结果为字符'f'。又如'9'-6表示数字字符9的ASCII码值减去整数6,结果为数字字符'3'。又如'A'+32结果为'a';'f'-'d'结果为整数2。

③字符型常量可以是ASCII字符集中任意可打印字符,包括空格字符。

④字符型常量中另一种字符称为转义字符。转义字符用于表示ASCII字符集中控制代码或某些其他功能代码。转义字符由一个\开始,后跟一个字符,或一个\后跟一个八进制或十六进制数,用单引号括起来。例如:'\n'表示回车换行,'\"'表示双引号。C++转义字符序列见表1.2。

表1.2 转义字符

转义序列	功 能
\a	报警
\b	退格
\f	走纸(用于打印机)
\n	换行
\t	横向跳格(TAB键)
\v	垂直跳格
\r	回车