普通高等教育“十二五”规划教材

C++程序设计

CHENGXU SHEJI

主　编　王海涌　李　鹏

电子科技大学出版社

C++程序设计

主　编　王海涌　李　鹏
副主编　罗胜荣　张　程

电子科技大学出版社

图书在版编目（CIP）数据

C＋＋程序设计 / 王海涌，李鹏主编. —成都：电子科技大学出版社，2013.1

ISBN 978-7-5647-1357-7

Ⅰ. ①C… Ⅱ. ①王… ②李… Ⅲ. ①C语言－程序设计－高等职业教育－教材 Ⅳ. ①TP312

中国版本图书馆CIP数据核字（2012）第302226号

C＋＋程序设计

主　编　王海涌　李　鹏

出　　版：电子科技大学出版社
策划编辑：曾　艺
责任编辑：曾　艺
主　　页：www.uestcp.com.cn
电子邮件：uestcp@uestcp.com.cn
发　　行：新华书店经销
印　　刷：北京佳顺印务有限公司
成品尺寸：185 mm×260 mm　**印张** 17.25　**字数** 405千字
版　　次：2013年1月第1版
印　　次：2013年1月第1次印刷
书　　号：ISBN 978-7-5647-1357-7
定　　价：38.00元

前　言

随着计算机科学技术的发展和计算应用技术的普及，越来越多的人需要操作和使用计算机。一般来说，操作和使用计算机分为两个层次。一是基于商业软件，进行简单的鼠标点击和键盘输入等基本操作。例如，使用 Microsoft Excel 进行业务数据处理，包括排序、分类汇总、绘制趋势线、分布图和饼图等。二是使用计算机语言，如 C++、C 和 Java 等进行定制开发。例如，使用 C++语言编制一个聊天应用程序，使用 Java 语言编制一个计算器应用程序等。现实生活中，大部分人对计算机的使用都属于第 1 个层次。而科研工作人员和技术开发人员等往往面临很多个性化的应用，这些应用不能简单地使用商业软件进行处理，必须自己开发应用软件来满足个性化需求，这属于第 2 个层次。作为高等学校的大学生，绝大多数人都越来越多地面临第 2 个层次的使用需求。因此，掌握一门计算机语言成为高等学校对大学生培养的基本要求。

众多的计算机语言中，如 C++、C、Java、C#和 VB 等，C++是一门最为复杂的计算机编程语言。造成这种复杂性的原因是，C++是一门兼有面向过程方法和面向对象方法的混合编程语言。一方面，C++这种混合编程语言的特点使得我们起步开始学习这门编程语言相比较学习其他编程语言要困难很多；另一方面，C++作为一门混合编程语言，涉及了面向过程方法的很多基本概念，同时也涉及了面向对象方法的很多基本概念，可以说，C++编程语言中涵盖了几乎所有编程语言的基本概念，使得我们在学习其他编程语言时，尤其是当下非常流行且得到大规模应用的 C#和 Java 语言时，能够进行快速的知识迁移。从这个角度来说，C++作为学习编程语言的入门是非常合适的。

C++是近年来国内外广泛使用的现代计算机语言，它既支持基于过程的程序设计，也支持面向对象的程序设计。国内许多高校陆续开设了 C++程序设计课程。但是，由于 C++涉及概念很多，语法比较复杂，内容十分广泛，使不少人感到学习难度较大，难以入门。

本书编者深入调查了大学的程序设计课程的现状和发展趋势，参阅了国内外数十种有关 C++的教材，认真分析了读者在学习中的困难和认识规律，设计了这本读者易于学习的教材。

本书降低入门起点，不需要 C 语言的基础，从零起点介绍程序设计和 C++，以通俗易懂的语言对 C++的许多难懂的概念作了透彻而通俗的说明，大大降低了初学者学习的困难，是一本初学者学习 C++的好教材。

学习程序设计是一个充满新奇、不断探索的过程，在这个过程中学习者希望感到主动探究的乐趣，而不是被枯燥的语法规定所左右。因此，程序设计课程和教材的设计就需要以学习者为中心，通过实例引导学习者在实践中学会编写程序。

程序设计的目的，是对问题和解决问题的方法进行模拟，用计算机语言描述，进而利用计算机进行问题求解。C++语言是一种面向对象的高级语言，而面向对象的观点正是人类看待自然的观点。不过人类以自然语言思维，而计算机程序设计语言的语法要比自然

语言简单很多，词汇量也少很多，因此当读者以程序设计语言来编写程序时，就会感到困难。因此读者需要学习计算思维，也就是以计算机可以理解的方式思维，以程序设计语言能够表达的方式描述问题和解决问题的方法。

本书内容全面，例题丰富，概念清晰，循序渐进，易于学习，即使没有教师讲授，读者也能看懂本书的大部分内容。本书是学习C++的入门教材，可供各类专业学生使用，也可作为计算机培训班的教材以及读者自学参考。

在本书的编写过程中，编者参阅了许多国内外优秀教材和教学参考资料，恕不一一列出，谨在此一并致谢！

尽管本书的编写凝聚了编者们多年来积累的丰富教改研究和教学实践经验，但由于时间较紧且学识所限，疏漏和不妥之处在所难免，期望读者谅解并指正，以便在今后的重印或再版中改进和完善。

编　者

目　　录

项目一

C++语言基础知识

项目简介

C++语言是一种同时支持面向过程程序设计方法和面向对象程序设计方法的优秀编程语言,已经在计算机相关的工程领域得到了广泛应用。本项目从C++语言的初步知识入手,进而介绍了C++上机实践的步骤。

任务一　C语言与C++语言

计算机的本质是“程序的机器”,计算机的一切操作都是由程序驱动的。程序和指令的思想是计算机系统中最基本的概念。只有懂得程序设计,才能进一步懂得计算机,真正了解计算机是怎样工作的。

在计算机诞生后的初期,人们使用机器语言或汇编语言(由于它们贴近计算机,被称为低级语言)编写程序,难学、难记、难用、难修改,使计算机只限于少数专业人员使用。

1954年出现了世界上第一种计算机高级语言,它是用于科学计算的FORTRAN语言。高级语言的出现是计算机发展过程中的一个划时代的事件,使计算机的广泛推广成为可能。随着计算机的推广应用,先后出现了多种计算机高级语言,如BASIC,ALGOL,Pascal,COBOL,ADA,C等。其中使用最广泛、影响最大的当推BASIC语言和C语言。

BASIC语言是1964年在FORTRAN语言的基础上简化而成的,它是为初学者设计的小型高级语言。它的语法相对简单,采取交互方式,功能也比较丰富,容易学习和掌握,因此受到广大初学者的欢迎。尤其在20世纪70年代微型计算机出现后,BASIC语言成为与微型计算机天然匹配的计算机语言,为计算机在大范围内的推广应用作出了重要的贡献。BASIC语言被称为“大众语言”。

C语言是1972年由美国贝尔实验室的D. M. Ritchie研制成功的。它不是为初学者设计的,而是为计算机专业人员设计的。最初它是作为写UNIX操作系统的一种工具,在贝尔实验室内部使用。后来C语言不断改进,人们发现它功能丰富、表达能力强、使用灵活方便、应用面广、目标程序效率高、可移植性好,既具有高级语言的优点,又具有低级语言的许多特点,特别适合于写系统软件,因此引起了人们的广泛重视。在短短的十几年中,C语言从实验室走向社会,从美国走向世界,到20世纪80年代,它已风靡全世界。被安装在几乎所有的巨型机、大型机、中小型机以及微型机上。大多数系统软件和许多应用软件都是用C语言编写的。无论在外国还是在中国,C语言都成为了计算机开发人员的一项基本功,C语

言的设计者在当初做梦也没有想到 C 语言日后会如此辉煌,会如此深刻地影响了几代计算机工作者。

但是随着软件规模的增大,用 C 语言编写程序就显得有些吃力了。C 语言是结构化和模块化的语言,它是基于过程的。在处理较小规模的程序时,程序员用 C 语言还比较得心应手。但是当问题比较复杂、程序的规模比较大时,结构化程序设计方法就显出它的不足。C 程序的设计者必须细致地设计程序中的每一个细节,准确地考虑到程序运行时每一时刻发生的事情,例如各个变量的值是如何变化的,什么时候应该进行哪些输入,在屏幕上应该输出什么等。这对程序员的要求是比较高的,如果面对的是一个复杂问题,程序员往往感到力不从心。当初提出结构化程序设计方法的目的是解决软件设计危机,但是这个目标并未完全实现。

为了解决软件设计危机,在 20 世纪 80 年代提出了面向对象的程序设计(Object Oriented Programming,OOP),需要设计出能支持面向对象的程序设计方法的新的语言。Smalltalk 就是一种面向对象的语言。在实践中,人们发现由于 C 语言是如此的深入人心,使用如此广泛,面对程序设计方法的革命,最好的办法不是另外发明一种新的语言去代替它,而是在它原有的基础上加以发展。在这种形势下,C + +应运而生。C + +是由 AT&T Bell(贝尔)实验室的 Bjarne stroustrup 博士及其同事于 1980 年开始在 C 语言的基础上进行开发并取得成功的,1985 年开始在 AT&T 以外流行开来。C + +保留了 C 语言原有的所有优点,增加了面向对象的机制。由于 C + +对 C 的改进主要体现在增加了适用于面向对象程序设计的“类(class)”,因此最初它被 Bjarne Stroustrup 称为“带类的 C”。后来为了强调它是 C 的增强版,用了 C 语言中的自加运算符“ + +”,改称为 C + +。

AT&T 发布的第一个 C + +编译系统实际上是一个预编译器(前端编译器),它把C + +代码转换成 C 代码,然后用 C 编译系统编译它,生成目标代码。第一个真正的 C + +编译系统是 1988 年诞生的。C + +2.0 版本于 1989 年出现,它有了重大的改进,包括了类的多继承。1991 年的 C + +3.0 版本增加了模板,4.0 版本则增加了异常处理、命名空间、运行时类型识别(RTTI)等功能。1989 年 ANSI 和 ISO 联合着手制定 C + +标准,并于 1994 年公布了非正式的草案,以征求意见,该草案是以 4.0 版本为基础制订的。经过几次修改后,1998 年正式通过并发布了 C + +的国际性标准。

C + +是由 C 发展而来的,与 C 兼容。用 C 语言写的程序基本上可以不加修改地用于 C + +。从 C + +的名字可以看出它是 C 的超集。C + +既可用于基于过程的结构化程序设计,又可用于面向对象的程序设计,是一个功能强大的混合型的程序设计语言。

C + +对 C 的“增强”,表现在两个方面:

(1)在原来基于过程的机制基础上,对 C 语言的功能做了不少扩充。

(2)增加了面向对象的机制。

面向对象程序设计,是针对开发较大规模的程序而提出来的,目的是提高软件开发的效率。只有编写过大型程序的人才会真正体会到 C 的不足和 C + +的优点。

不要把面向对象和基于过程对立起来,面向对象和基于过程不是矛盾的,而是各有用途、互为补充的。在面向对象程序设计中仍然要用到结构化程序设计的知识,例如在类中定义一个函数就需要用结构化程序设计方法来实现。任何程序设计都需要编写操作代码,具

体操作的过程就是基于过程的。对于简单的问题,直接用基于过程方法就可以轻而易举地解决。

学习 C + +,既要会利用 C + +进行基于过程的结构化程序设计,也要会利用 C + +进行面向对象的程序设计。本书的第 1 篇介绍 C + +基于过程的程序设计,第 2 篇和第 3 篇介绍 C + +基于对象的程序设计和面向对象的程序设计。

应当指出:目前所使用的 C + +编译系统中,有一些是早期推出的,并未全部实现 ANSI C + +标准所建议的功能。

任务二　最简单的 C + +程序

为了使读者能了解什么是 C + +程序,下面先介绍几个简单的程序。

【例 1-1】 输出一行字符:"This is a C + + program."。

程序如下:

```
#include <iostream>                    //包含头文件 iostream
using namespace std;                   //使用 C + +的命名空间 std
int main()
{
  cout <<"Thlis is a C + +program.";
  return 0;
}
```

在运行时会在屏幕上输出以下一行信息:

```
This is a C + + program.
```

先看程序中的第 3 行,其中用 main 代表"主函数"的名字。每一个 C + +程序都必须有一个 main 函数。main 前面的 int 的作用是声明函数的类型为整型(标准 C + +规定 main 函数必须声明为 int 型[①],即此主函数带回一个整型的函数值)。程序第 6 行的作用是向操作系统返回一个零值。如果程序不能正常执行,则会自动向操作系统返回一个非零值,一般为 -1。

函数体是由大括号{　}括起来的。本例中主函数内只有一个以 cout 开头的语句。cout 是由 c 和 out 两个单词组成,顾名思义,它是 C + +用于输出的语句。cout 实际上是 C + +系统定义的对象名,称为输出流对象。在没有学习对象和输出流对象以前,为了便于理解和使用,我们把用"cout"和" <<"实现输出的语句简称为 cout 语句。" <<"是"插入运算符",与 cout 配合使用,在本例中它的作用是将运算符" <<"右侧双撇号内的字符串"This is a C + + program."插入到输出的队列 cout 中(输出的队列也称作"输出流"),C + +系统将输

① 标准 C + +要求 main 函数必须声明为 int 型。有的操作系统(如 UNIX,Linux)要求执行一个程序后必须向操作系统返回一个数值。因此,C + +的处理是这样的:如果程序正常执行,则向操作系统返回数值 0,否则返回数值 -1。在目前使用的一些 C + +编译系统并未完全执行 C + +这一规定,如果主函数首行写成 void main()也能通过,在本书中的所有例题都按 C + +规定,写成 int main(),希望读者也养成这个习惯,以免在编译时通不过。只要记住:在 main 前面加 int,同时在 main 函数的最后加一个语句"return 0;"即可。

出流 cout 的内容输出到系统指定的设备(一般为显示器)中。注意 C + +所有语句最后都应当有一个分号。

再看程序的第 1 行“#include <iostream>”,这不是 C + +的语句,而是 C + +的一个预处理指令,它以“#”开头以与 C + +语句相区别,行的末尾没有分号。“#include <iostream>”是一个“包含指令”,它的作用是将文件“iostream”的内容包含到该命令所在的程序文件中,代替该指令。文件 iostream 的作用是向程序提供输入或输出时所需要的一些信息。iostream 是 i-o-stream 3 个词的组合,从它的形式就可以知道它代表“输入输出流”的意思,由于这类文件都放在程序单元的开头,所以称为“头文件”(header file)。在程序进行编译时,先对所有的预处理命令进行处理,将头文件的具体内容代替#include 指令,然后再对该程序单元进行整体编译。

程序的第 2 行“using namespace std;”的意思是“使用命名空间 std”。C + +标准库中的类和函数是在命名空间 std 中声明的,因此程序中如果需要用到 C + +标准库(此时需要用#include 指令),就需要用“using namespace std;”作声明,表示要用到命名空间 std 中的内容①。

初学 C + +时,对本程序中的第 1,2 行可以不必深究,只须知道:如果程序有输入或输出时,必须使用“#include <iostream>”指令以提供必要的信息,同时要用“using namespace std;”使程序能够使用这些信息,否则程序编译时将出错。读者以后将会看到,本书中的程序几乎在程序的开头都包含此两行。请读者先接受这个现实,在写 C + +程序时如法炮制,在程序的开头包含此两行,在以后的学习中将会逐步加深理解。

【例 1-2】 求 a 和 b 两个数之和。

可以编写以下程序:

```
//求两数之和                          (本行是注释行)
#include <iostream>                   //预处理指令
using namespace std;                  //使用命名空间 std
int main()                            //主函数首部
{                                     //函数体开始
    int a,b> >,sum;                   //定义变量
    cin> >a> >b;                      //输入语句
    sum=a+b;                          //赋值语句
    cout<<"a+b="<<sum<<endl;//输出语句
    return 0;                         //如程序正常结束,向操作系统返回一个零值
}                                     //函数结束
```

本程序的作用是求两个整数 a 和 b 之和 sum。第 1 行“//求两数之和”是一个注释行,

① 在早期的一些 C + +程序中,使用的是从 C 语言继承下来的函数库,在程序中用#include 指令把带后缀.h 的头文件包含进来,即可在本程序中使用这些函数。在 C + +新标准中,使用不带后缀.h 的头文件,标准库中的类和函数都在“命名空间 std"中声明。因此,如果程序中包含了新形式的头文件(无后缀的头文件,如 iostream),必须使用“using namespace std;”。

C + +规定在一行中如果出现“//”,则从它开始到本行末尾之间的全部内容都作为注释。注释只是给人看的,而不是让计算机操作的。注释是源程序的一部分,在输出源程序清单时全部注释按原样输出,以便看程序者更好地理解程序。但是在对程序编译时将忽略注释部分,这部分内容不转换成目标代码,因此对运行不起作用。注释可以加在程序中任何行的右侧。为便于读者理解,在本书中用汉字表示注释,当然也可以用英语或汉语拼音作注释。

在一个可供实际应用的程序中,为了提高程序的可读性,常常在程序中加了许多注释行,在有的程序中,注释行可能占程序篇幅的三分之一。在本书中为了节省篇幅,不写太多的独立的注释行,而只在语句的右侧用“//”作简短的注释。

第6行是声明部分,定义变量a,b和sum为整型(int)变量。第7行是输入语句,cin是c和in两单词的组合,与cout类似,cin是C + +系统定义的输入流对象。“ > >”是“提取运算符”,与cin配合使用,其作用是从输入设备中(如键盘)提取数据送到输入流cin中。我们把用cin和“ > >”实现输入的语句简称为cin语句。在执行程序中的cin语句时,从键盘输入的第1个数据赋给整型变量a,输入的第2个数据赋给整型变量b。第8行将a + b的值赋给整型变量sum。第9行先输出字符串“a + b = ”,然后输出变量sum的值,cout语句中的endl是C + +输出时的控制符,作用是换行(endl是end line的缩写,表示本行结束)。因此在输出变量sum的值之后换行。

如果在运行时从键盘输入以下信息(为区别输入和输出的信息,在本书中,输入的信息都带下划线):

123　456↙

则输出为

a + b = 579

读者可能已注意到本程序的第6 ~ 9行的末尾都有一个分号,因为它们是C + +语句。

【例1-3】 给两个数x和y,求两数中的大者。

在本例中包含两个函数。

```
#include <iostream>                    //预处理指令
using namespace std;
int max(int x,int y)                   //定义 max 函数,函数值为整型,形式参数 x,y
                                         为整型
{                                      //max 函数体开始
 int z;                                //变量声明,定义本函数中用到的变量 z 为整型
 if(x > y)z = x;                       //if 语句,如果 x > y,则将 x 的值赋给 z
   else z = y;                         //否则,将 y 的值赋给 z
 return(z);                            //将 z 的值返回,通过 max 带回调用处
}                                      //max 函数结束
int main()                             //主函数
{                                      //主函数体开始
 int a,b,m;                            //变量声明
 cin > >a > >b;                        //输入变量 a 和 b 的值
```

```
    m = max(a,b);                    //调用 max 函数,将得到的值赋给 m
    cout << "max = " << m << '\n';   //输出大数 m 的值
    return 0;                        //如程序正常结束,向操作系统返回一个零值
}                                    //主函数结束
```

本程序包括两个函数:主函数 main 和被调用的函数 max。程序中第 3 ~ 9 行是 max 函数,它的作用是将 x 和 y 中较大者的值赋给变量 z。return 语句将 z 的值返回给主调函数 main。返回值是通过函数名 max 带回到 main 函数的调用处。主函数中 cin 语句的作用是输入 a 和 b 的值。main 函数中第 5 行为调用 max 函数,在调用时将实际参数 a 和 b 的值分别传送给 max 函数中的形式参数 x 和 y。经过执行 max 函数得到一个返回值(即 max 函数中变量 z 的值),把这个值赋给变量 m。然后通过 cout 语句输出 m 的值。

运行结果如下:

```
18  25↙                          (输入 18 和 25 给 a 和 b)
max = 25                         (输出 m 的值)
```

注意输入的两个数据间用一个或多个空格间隔,不能以逗号或其他符号间隔。如,输入

```
18,25↙
```

或

```
18;25↙
```

是错误的,它不能正确输入第 2 个变量的值,使第 2 个变量有不可预见的值。

在上面的程序中,max 函数出现在 main 函数之前,因此在 main 函数中调用 max 函数时,编译系统能识别 max 是已定义的函数名。如果把两个函数的位置对换一下,即先写 main 函数,后写 max 函数,这时在编译 main 函数遇到 max 时,编译系统无法知道:max 代表什么含义,因而无法编译,按出错处理。

为了解决这个问题,在主函数中需要对被调用函数作出声明。上面的程序可以改写如下:

```
#include <iostream>
using namespace std;
int main()
{ int max(int x,int y);          //对 max 函数作声明
  int a,b,c;
  cin >> a >> b:
  c = max(a,b);                  //调用 max 函数
  cout << "max = " << c << endl:
  return 0;
}
int max(int x,int y)             //定义 max 函数
{ int z;
  if(x > y)z = x;
  else z = y;
```

```
    return(z);
}
```

程序第4行就是对max函数作声明,它的作用是通知C++编译系统:max是一个函数,函数值是整型,函数有两个参数,都是整型。这样,在编译到程序第7行时,编译系统会知道max是已声明的函数,系统就会根据函数声明时给定的信息对函数调用的合法性进行检查,如果二者不匹配(例如参数的个数或参数的类型与声明时所指定的不符),编译就会出错。

细心的读者会发现程序第4行的函数声明与程序第11行max函数的第1行(称为函数首部)基本相同。很容易写出函数声明,只要在被调用函数的首部的末尾加一个分号,就成为对该函数的函数声明语句。函数声明的位置应当在函数调用之前(不能把程序第4行的内容放在第7行"c=max(a,b);"之后)。

本例用到了函数调用、实际参数和形式参数等概念,这里只做了很简单的解释。读者如对此不大理解,可以暂不予深究。在此介绍此例子,目的是使读者对C++函数的形式和使用有一个初步的了解。

下面举一个包含类(class)和对象(object)的C++程序,目的是使读者初步了解C++是怎样体现面向对象程序设计方法的。由于还未系统介绍面向对象程序设计的概念,读者可能对程序理解不深,现在只须有一个初步印象即可。

【例1-4】 包含类的C++程序。

```
#include <iostream>                    //预处理指令
using namespace std;
class Student                          //声明一个类,类名为Student
{ private:                             //以下为类中的私有部分
      int num;                         //私有变量num
      int score;                       //私有变量score
  public:                              //以下为类中公用部分
      void setdata()                   //定义公用函数setdata
      { cin>>num;                      //输入num的值
        cin>>score;                    //输入score的值
      }
      void display()                   //定义公用函数display
  { cout<<"num="<<num<<endl;           //输出num的值
    cout<<"score="<<score<<endl;       //输出score的值
  };
};                                     //类的声明结束
Student stud1,stud2;                   //定义stud1和stud2为Student类的
                                         变量,称为对象
int main()                             //主函数首部
{ stud1.setdata();                     //调用对象stud1的setdata函数
```

```
    stud2.setdata();                    //调用对象 stud2 的 setdata 函数
    stud1.display();                    //调用对象 stud1 的 display 函数
    stud2.display();                    //调用对象 stud1 的 display 函数
    return 0;
}
```

这是一个包含类的最简单的 C + +程序。程序第 3 ~ 16 行声明一个类 Student。在一个类中包含两种成员:数据(如变量 num,score)和函数(如 setdata 函数和 display 函数),分别称为数据成员和成员函数。在 C + +中把一组数据和有权调用这些数据的函数封装在一起,组成一种称为"类(class)"的数据结构。在上面的程序中,数据成员 num,score 和成员函数 setdata,display 组成了一个名为 Student 的"类"类型。成员函数是用来对数据成员进行操作的。也就是说,一个类是由一批数据以及对其操作的函数组成的。

类可以体现数据的封装性和信息隐蔽。在上面的程序中,在声明 Student 类时,把类中的数据和函数分为两大类:private(私有的)和 public(公用的)。把全部数据(num,score)指定为私有的,把全部函数(setdata,display)指定为公用的。当然也可以把一部分数据和函数指定为私有,把另一部分数据和函数指定为公用,这完全根据需要而定。在大多数情况下,都把所有数据指定为私有,以实现信息隐蔽。

凡是被指定为公用的数据或函数,既可以被本类中的成员函数调用,也可以被类外的语句所调用。被指定为私有的成员(函数或数据)只能被本类中的成员函数所调用,而不能被类以外的语句调用(除了以后介绍的"友元类"成员以外)。这样做的目的是对某些数据进行保护,只有被指定的本类中的成员函数才能调用它们,拒绝其他无关的部分调用它们,以防止误调用。这样才能真正实现封装的目的(把有关的数据与操作组成一个单位,与外界相对隔离),信息隐蔽是 C + +的一大特点。

可以看到:在类 Student 中,有两个公用的成员函数 setdata 和 display。setdata 函数的作用是给本类中的私有数据 num 和 score 赋以确定的值,这是通过 cin 语句实现的,在程序运行时从键盘输入 num 和 score 的值。display 函数的作用是输出已被赋值的变量 num 和 score 的值。由于这两个函数与私有数据 num 和 score 是属于同一个类 Student 的,因此函数可以直接引用 num 和 score。

程序中第 17 行"Student stud1,stud2"是一个定义语句,它的作用是将 stud1 和 stud2 定义为 Student 类型的变量,这种定义方法和定义整型变量"int a,b;"的方法是一样的。区别只在于 int 是系统已预先定义好的标准数据类型,而 Student 是用户自己声明(指定)的类型。Student 类与 int,float 等一样都是 C + +的合法类型。具有"类"类型特征的变量称为"对象"(object)。stud1 和 stud2 是 Student 类型的对象。和其他变量一样,对象是占实际存储空间的,而类型并不占实际存储空间,它只是给出一种"模型",供用户定义实际的对象。在用 Student 定义了 stud1 和 stud2 以后,这两个对象具有同样的结构和特性。

程序中第 18 ~ 24 行是主函数。在主函数中有 4 个语句,用来调用对象的成员函数。现在有两个对象 stud1 和 stud2,因此在类外调用成员函数时不能只写函数名(如"setdata();"),而必须说明要调用哪一个对象的函数,准备给哪一个对象中的变量赋值。因此要用对象的名字加以限定,如表 1-1 所示。

表 1-1　引用对象的成员

对象名	num(学号)	score(成绩)	setdata 函数	display 函数
stud1	stud1. num(如 1001)	stud1. score(如 98.5)	stud1. setdata( )	stud1. display( )
stud2	stud2. num(如 1002)	stud2. score(如 76.5)	stud2. setdata( )	stud2. display( )

其中,“.”是一个“成员运算符”,把对象和成员连接起来。stud1. setdata( )表示调用对象 stud1 的 setdata 成员函数,在执行此函数中的 cin 语句时,从键盘输入的值(假设为 1001 和 98.5)送给 stud1 对象的 num 和 score,作为学生 1(stud1)的学号和成绩。stud2. setdata( )表示调用对象 stud2 中的 setdata 成员函数,在执行此函数中的 cin 语句时,从键盘输入的值(假设为 1002 和 76.5)送给 stud2 对象的 num 和 score,作为学生 2(stud2)的学号和成绩。

程序中主函数中第 1 个语句用来输入学生 1 的学号和成绩。第 2 个语句用来输入学生 2 的学号和成绩。第 3 个语句用来输出学生 1 的学号和成绩。第 4 个语句用来输出学生 2 的学号和成绩。

程序运行情况如下:

```
1001  98.5↙                    (输入学生 1 的学号和成绩)
1002  76.5↙                    (输入学生 2 的学号和成绩)
num = 1001                     (输出学生 1 的学号)
score = 98.5                   (输出学生 1 的成绩)
num = 1002                     (输出学生 2 的学号)
score = 76.5                   (输出学生 2 的成绩)
```

通过这个例子,读者可以初步了解包含类的 C + +程序的形式和含义。

说明:以上几个程序是按照 ANSI C + +规定的语法编写的。由于 C + +是从 C 语言发展而来的,为了与 C 兼容,C + +保留了 C 语言中的一些规定。其中之一是头文件的形式,在 C 语言中头文件用. h 作为后缀,如 stdio. h,math. h,string. h 等。在 C + +发展初期,为了和 C 语言兼容,许多 C + +编译系统保留头文件以. h 为后缀的用法,如 iostream. h。但后来 ANSI C + +建议头文件不带后缀. h。近年推出的 C + +编译系统新版本则采用了 C + +的新方法,提供了一批不带后缀的头文件,如用 iostream,string,cmath 等作为头文件名。但为了使原来编写的 C + +程序能够运行,仍允许使用原有的带后缀. h 的头文件,即二者同时并存,由用户选用。例 1-1 也可以写成下面的形式:

```
#include <iostream.h>                    //头文件带后缀.h
int main()
{ cout <<"This is a C + +program.";
  return0;
}
```

由于 C 语言无“命名空间”,因此用带后缀. h 的头文件时不必用“using namespace std;”作声明。

目前有些介绍 C + +的书中的程序仍采用 C 的形式(如 main 函数用 void 类型,头文件带后缀. h),此外还有些以前的 C 程序会在 C + +环境下运行,当读者在看到这些程序时,

也应当能看懂这些程序,并能将它们改写为标准 C + +的形式。应当提倡在编写新的程序时按照标准 C + +的规定进行。本书中的全部程序都是按标准 C + +的规定编写的。

有了以上的基础,在以后的项目中将由简到繁、由易到难、循序渐进地介绍 C + +编程。

任务三　C + +程序的构成和书写形式

从上面几个例子中,已经初步了解了 C + +程序的结构和书写格式,现在再归纳如下:

(1)一个 C + +程序可以由一个程序单位或多个程序单位构成。每一个程序单位作为一个文件。在程序编译时,编译系统分别对各个文件进行编译,因此,一个文件是一个编译单元。任务二中介绍的 4 个例子是比较简单的程序,都是只由一个程序单位(即一个文件)构成的。

(2)在一个程序单位中,可以包括以下 3 个部分:

①预处理指令。任务二的 4 个程序中都包括#include 指令。

②全局声明部分(在函数外的声明部分)。在这部分中包括对用户自己定义的数据类型的声明和程序中所用到的变量的定义。例 1-4 就包括了对类 Student 的声明和对变量 stud1,stud2 的定义。也可以包括对函数的声明,如例 1-3 中第 2 个程序中主函数内对 max 函数的声明。

③函数。函数是实现操作的部分,因此函数是程序中必须有的和最基本的组成部分。每一个程序必须包括一个或多个函数,其中必须有一个(而且只能有一个)主函数(main 函数)。

但是并不要求每一个程序文件都必须全部具有以上 3 个部分,可以缺少某些部分(包括函数)。也就是说,有的程序文件可以不包括函数,而只包括预处理命令和(或)声明部分。完全根据需要而定。这一点在以后的学习中会逐步体会到的。

(3)一个函数由两部分组成。

①函数首部,即函数的第一行。包括函数名、函数类型、函数属性、函数参数(形参)名、参数类型。

例如,例 1-3 中的 max 函数的首部为

```
int      max       ( int          x          ,  int          y)
 ↓        ↓          ↓           ↓             ↓           ↓
```

函数类型　函数名　函数参数类型　函数参数名　函数参数类型　函数参数名

一个函数名后面必须跟一对圆括号,函数参数可以缺省,如 int main( )。

②函数体,即函数首部下面的大括号内的部分。如果在一个函数中有多个大括号,则最外层的一对{ }为函数体的范围。

函数体一般包括:

- 局部声明部分(在函数内的声明部分)。包括对本函数中所用到的类型、函数的声明和变量的定义。如例 1.3 中 main 函数中的“int a,b,m;”以及对所调用的函数声明“im max(int x,int y);”。

对数据的声明既可以放在函数之外(其作用范围是全局的),也可以放在函数内(其作