第1章 C++的初步知识

1.1 从C到C++

C语言是结构化和模块化的语言,它是面向过程的。在处理较小规模的程序时,程序员用C语言较为得心应手。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)思想,这就需要设计出能支持面向对象的程序设计方法的新语言。Smalltalk就是当时问世的一种面向对象的语言。而在实践中,人们发现由于C语言是如此深入人心,使用如此广泛,以至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20世纪80年代初在C语言的基础上开发成功的。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,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,它作了重大的改进,包括了类的多继承。1991年的C++ 3.0版本增加了模板,C++ 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。ANSI C++标准草案是以C++ 4.0版本为基础制定的,1997年ANSI C++标准正式通过并发布。但是目前使用的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。

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++既可用于面向过程的结构化程序设计,也可用于面向对象的程序设计,是一种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) 在原来面向过程的机制基础上,对C语言的功能作了不少扩充。

(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能真正体会到C的不足和C++的优点。

不要把面向对象和面向过程对立起来,面向对象和面向过程不是矛盾的,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面向过程的。对于简单的问题,直接用面向过程方法就可以轻而易举地解决。

读者在学习C++之后,既能进行面向过程的结构化程序设计,也能进行面向对象的程序设计。本书着重介绍C++面向对象程序设计的基本知识。

1.2 最简单的C++程序

为使读者了解什么是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>                          //用cout输出时需要用此头文件
using namespace std;                         //使用命名空间std
int main()
{ cout <<" This is a C++ program.\n";        //用C++的方法输出一行
  return 0;
}
```

程序运行时输出:

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

请读者分析一下,本程序和以前见过的C程序有什么不同?

(1) 在C++程序中,一般在主函数main前面加一个类型声明符int,表示main函数的返回值为整型(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5行的作用是向操作系统返回0。如果程序不能正常执行,则会自动

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++是这样处理的: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,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,以免在严格遵循C++标准的编译系统中通不过。只要记住:在main前面加int,同时在main函数的最后加一条语句“return 0;”即可。

向操作系统返回一个非零值，一般为-1。

（2）在C++程序中，可以使用C语言中的“/*……*/”形式的注释行，还可以使用以“//”开头的注释。从例1.1可以看到：以“//”开头的注释可以不单独占一行，它可以出现在一行中的语句之后。编译系统将“//”以后到本行末尾的所有字符都作为注释。应注意：它是单行注释，不能跨行。C++的程序设计人员多愿意用这种注释方式，它比较灵活方便。

（3）在C++程序中，一般用cout进行输出。cout是由c和out两个单词组成的，见名知意，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**。对象和输出流对象的概念将在后面介绍。为了便于理解，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.\n”插入到输出的队列cout中（输出的队列也称作“输出流”），C++系统将输出流cout的内容输出到系统指定的设备（一般为显示器）中。除了可以用cout进行输出外，在C++中还可以用printf函数进行输出。

（4）使用cout需要用到头文件iostream。程序的第1行“#include <iostream>”是一个预处理命令，学过C语言的读者对此应该是很清楚的。文件iostream的内容是提供输入或输出时所需要的一些信息。iostream是i-o-stream三个词的组合，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”（head file）。

请注意：在C语言中所有的头文件都带后缀.h（如stdio.h），而按C++标准要求，由系统提供的头文件不带后缀.h，用户自己编制的头文件可以有后缀.h。在C++程序中也可以使用C语言编译系统提供的带后缀.h的头文件，如“#include <math.h>”。

（5）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使用C++标准库中的有关内容（此时需要用#include命令行），就需要用“using namespace std;”语句作声明，表示要用到命名空间std中的内容。命名空间的概念暂可不必深究，只需知道：如果程序有输入或输出时，必须使用“#include <iostream>”命令以提供必要的信息，同时要用“using namespace std;”语句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：本书中几乎所有程序的开头都包含此两行。请读者先接受这个现实，在写C++程序时也如法炮制，在程序的开头包含此两行。在第8章中将对命名空间作详细介绍。

例1.2　求a和b两个数之和。

可以写出以下程序：

```
// 求两数之和                          （本行是注释行）
#include <iostream>                    //预处理命令
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
}
```

本程序的作用是求两个整数 a 与 b 之和 sum。第 1 行“//求两数之和”是一个注释行,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。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。为了节省篇幅,本书中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 两个单词的组合,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。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,与“\n”作用相同)。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

```
123  456↙
```

则输出为

```
a + b = 579
```

例 1.3 输入两个数 x 和 y,求两数中的大者。

在本例中包含两个函数。

```
#include <iostream>
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);
}
```

本程序包括两个函数：主函数 main 和被调用的函数 max。max 函数的作用是将 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 函数第 5 行中调用 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 c。然后通过 cout 语句输出 c 的值。

程序运行情况如下：

```
18  25↙        （输入 18 和 25 给 a 和 b）
max=25         （输出 c 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，它不能正确输入第二个变量的值，将使第二个变量有不可预见的值。

程序第 4 行是对 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。学过 C 的读者对此例是很容易理解的。

下面举一个包含类（class）和对象（object）的简单程序，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，现在只需有一个初步印象即可。在第 2 章中将会详细介绍。

例 1.4　包含类的C++ 程序。

```
#include <iostream>
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行到第16行声明一个被称为“类”的类型 Student。class 是声明“类”类型时必须使用的关键字,如同声明结构体类型时使用关键字 struct 一样。在 C 语言的结构体中只能包含数据成员,而在C++的类中可以包含两种成员,即**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。如在上面的程序中,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**)。在上面的C++程序中把全部数据(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。大多数情况下会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**的数据或函数只能被本类中的成员函数所调用,而不能被类以外调用(以后介绍的“友元类”成员除外)。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:在例 1.4 中声明的类 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 赋予确定的值,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 到 24 行是主函数。在主函数中有 4 条语句,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,见表 1.1。

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”,把对象和成员联接起来。stud1. num 表示对象 stud1 中的 num,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。

程序的主函数中第 1 条语句用来输入学生 1 的学号和成绩。第 2 条语句用来输入学生 2 的学号和成绩。第 3 条语句用来输出学生 1 的学号和成绩。第 4 条语句用来输出学生 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。本章例 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.";
```

```
    return 0;
}
```

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。

此外,C语言不要求main函数返回整数,main函数不必指定为int型,一般用void型(无返回值),这样,main函数中最后一条语句“return 0;”也无必要了。因此,如果将例1.1写成下面的形式:

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

在一些C++编译系统中也能识别和通过,但在一些新版本的C++编译系统中不能通过编译,因为新版本的C++编译系统严格执行C++标准。

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,读者看到这些程序时,也应当能看懂,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。

1.3 C++对C的扩充

C++既可用于面向过程的程序设计,也可用于面向对象的程序设计。在面向过程程序设计的领域,C++继承了C语言提供的绝大部分功能和语法规定,并在此基础上作了不少扩充,主要有以下几个方面。

1.3.1 C++的输入输出

C++为了方便用户,除了可以利用printf和scanf函数进行输出和输入外,还增加了标准输入输出流cout和cin。cout是由c和out两个单词组成的,代表C++的输出流,cin是由c和in两个单词组成的,代表C++的输入流。它们是在头文件iostream中定义的。键盘和显示器是计算机的标准输入输出设备,所以在键盘和显示器上的输入输出称为标准输入输出,标准流是不需要打开和关闭文件即可直接操作的流式文件。

C++预定义的标准流如表1.2所示。

表1.2 C++预定义的标准流

流名	含义	隐含设备	流名	含义	隐含设备
cin	标准输入	键盘	cerr	标准出错输出	屏幕
cout	标准输出	屏幕	clog	cerr的缓冲形式	屏幕

1. 用 cout 进行输出

cout 必须和输出运算符“ <<”一起使用。“ <<”在这里不作为位运算的左移运算符,而是起**插入**的作用,例如:“cout <<"Hello! \n";”的作用是将字符串“Hello! \n” 插入到输出流 cout 中,也就是输出在标准输出设备上。

也可以不用“\n”控制换行,在头文件 iostream 中定义了控制符 endl 代表回车换行操作,作用与“\n”相同。endl 的含义是 end of line,表示结束一行。

可以在一个输出语句中使用多个“ <<”运算符将多个输出项插入到输出流 cout 中,“ <<”运算符的结合方向为自左向右,因此各输出项按自左向右顺序插入到输出流中。例如:

```
for (i=1; i<=3;i++)
cout <<"count="<<i <<endl;
```

输出结果为

```
count=1
count=2
count= 3
```

注意:每输出一项要用一个“ <<” 符号。不能写成“cout <<a,b,c,"A"; ”形式。

用 cout 和“ <<”可以输出任何类型的数据,如:

```
float a=3.45;
int b=5;
char c='A';
cout <<"a="<<a <<","<<"b="<<b <<","<<"c="<<c <<endl;
```

输出结果为

```
a=3.45, b=5, c=A
```

可以看到在输出时并未指定数据的类型(如浮点型、整型等),系统会自动按数据的类型进行输出。这比用 printf 函数方便,在 printf 函数中要指定输出格式符(如%d,%f,%c 等)。

如果要指定输出所占的列数,可以用控制符 setw 设置(注意:若使用 setw,必须包含头文件 iomanip.h),例如 setw(5) 的作用是为其后面一个输出项预留 5 列,如输出项的长度不足 5 列,则数据向右对齐,若超过 5 列,则按实际长度输出。如将上面的输出语句改为

```
cout <<"a="<<setw(6) <<a <<endl <<"b="<<setw(6) <<b <<endl <<"c="<<setw(6)
 <<c <<endl;
```

输出结果为

```
a=␣␣3.45
b=␣␣␣␣5
c=␣␣␣␣A
```

在C++中将数据送到输出流称为“**插入**”(inserting)或“**放到**”(putting)。“<<”常称为“插入运算符”。

2. 用 cin 进行输入

输入流是指从输入设备向内存流动的数据流。标准输入流 cin 是从键盘向内存流动的数据流。用“>>”运算符从输入设备键盘取得数据送到输入流 cin 中,然后送到内存。在C++中,这种输入操作称为“**提取**”(extracting)或“**得到**”(getting)。“>>”常称为“提取运算符”。

cin 要与“>>”配合使用。例如:

```
int a;                          //定义整型变量 a
float b;                        //定义浮点型变量 b
cin >>a >>b;                    //输入一个整数和一个实数,注意不要写成 cin >>a,b
```

可以从键盘输入

```
20 32.45↙                       (数据间以空格分隔)
```

a 和 b 分别获得值 20 和 32.45。用 cin 和“>>”输入数据同样不需要在本语句中指定数据类型。

例 1.5 cin 与 cout 一起使用。

```
#include <iostream>
using namespace std;
int main( )
{cout <<"please enter your name and age:" <<endl;
 char name[10];
 int age;
 cin >>name;
 cin >>age;
 cout <<"your name is " <<name <<endl;
 cout <<"your age is " << age <<endl;
 return 0;
}
```

运行情况如下:

```
please enter your name and age:
Wang-li↙
19↙
your name is Wang-li
your age is 19
```

细心的读者可能已发现,程序中对变量的定义放在执行语句之后。C 语言是不允许这样的,它要求声明部分必须在执行语句之前。而C++允许将变量的声明放在程序的任何位置(但必须在使用该变量之前)。这是C++对 C 限制的放宽。

C++为流输入输出提供了格式控制,如:dec(用十进制形式),hex(用十六进制形