高 等 学 校 教 材

C++程序设计教程

C++ CHENGXU SHEJI JIAOCHENG

张 冰 编著
明 仲 审

人民邮电出版社
POSTS & TELECOM PRESS

高等学校教材

C++程序设计教程

张 冰 编著

明 仲 审

人民邮电出版社

图书在版编目（CIP）数据

C++程序设计教程 / 张冰编著. —北京：人民邮电出版社，2004.3

高等学校教材

ISBN 7-115-12056-0

I.C… II.张… III.C语言—程序设计—高等学校—教材 IV.TP312

中国版本图书馆CIP数据核字（2004）第008343号

内 容 提 要

C++是一种通用的程序设计语言，在商业、工程和实时系统中得到广泛的应用。本书全面、系统、详细地讲述了C++语言的基本概念、面向对象程序设计的重要特征和基本编程方法。本书通过大量的C++程序实例阐述了软件工程强调程序的可维护性、可理解性和可移植性的观点。通过本书的学习，读者能够理解和掌握面向对象程序设计的基本概念和基本方法，具备一定的运用基本数据结构和算法进行程序设计的能力。

本书的语言基础、设计和应用三部分内容相互衔接，前后呼应，便于读者循序渐进地学习。每一章都附有大量富有启发性的习题，便于读者加深理解和巩固提高。

本书可作为高等院校计算机及相关专业本科、研究生面向对象程序设计课程的教材，也可作为工程技术人员和广大计算机爱好者自学的参考书。

高等学校教材

C++程序设计教程

◆ 编　　著　张　冰

审　　明　仲

责任编辑　邹文波

◆ 人民邮电出版社出版发行　北京市崇文区夕照寺街14号

邮编　100061　电子函件　315@ptpress.com.cn

网址　http://www.ptpress.com.cn

读者热线　010-67194042

北京汉魂图文设计有限公司制作

北京隆昌伟业印刷有限公司印刷

新华书店总店北京发行所经销

◆ 开本：787×1092　1/16

印张：22　　2004年3月第1版

字数：532千字　　2004年3月北京第1次印刷

印数：1-5 000册

ISBN 7-115-12056-0/TP・3817

定价：28.00元

本书如有印装质量问题，请与本社联系　电话：(010)67129223

前　言

程序设计语言是人与计算机进行交流沟通的一种形式语言，是人们运用计算机分析问题、解决问题的一种基本工具。熟悉一门程序设计语言是每个工科学生必须掌握的基本功。C++语言作为一种高级程序设计语言是学习程序设计的首选语言。然而，笔者在多年教学过程中，发现大学本科学生在学习C++语言程序设计过程中面临概念比较抽象难理解，方法比较独特难接受，内容比较新颖难适应等问题。

本书针对这些问题，根据教育部2003年颁发的计算机基础教育白皮书——《关于进一步加强高校计算机基础教学的几点意见》，结合作者的教学和编程实践经验，力图用生动、通俗易懂的语言并结合实际应用的举例来讲解各个知识点。重要和抽象的概念将采用比拟的手法，引用学生熟悉的一些实际生活实例，以生动活泼的形式加以描述。在全书的编写过程中，注重以先进的开发工具与先进的开发方法来讲述C++语言面向对象程序设计的基本概念和基本方法，从软件工程的基本观点和实践方法出发，以内聚和耦合、数据封装和信息隐藏等作为软件设计的指标，讲述结构化程序设计的缺陷，说明为什么要采用面向对象程序设计，以及怎样采用C++语言实现面向对象的程序设计。本书通过大量的例题和练习，介绍了数组、链表、队列、堆栈等基本数据结构，讲解了多种求π、求素数、搜索、排序和递归等常用算法的设计和实现。通过本书的学习，学生能够理解和掌握面向对象程序设计的基本概念和基本方法，具备一定的运用基本数据结构和算法进行程序设计的能力。

全书共分为9章。第1章和第2章讲述了面向过程和面向对象程序设计的基本思想和基本方法，C/C++语言的特点，C/C++程序的结构，基于Visual C++ 6.0集成开发环境的C++程序的编辑、编译和运行步骤，基本的C++语言语法，包括数据类型、运算符、表达式、顺序、选择和循环3种基本程序结构、数组、函数、指针、引用等。这一部分构成了结构化程序设计的基本内容和方法，作为C++程序设计语言的基础；第3章至第8章围绕面向对象程序设计的数据封装、继承性和多态性3个基本特性，讲述类与对象、构造函数与析构函数、继承与派生、虚函数与多态性、友元函数与友元类、静态成员、模板以及输入输出流等内容。这部分主要讲解C++实现数据封装和信息隐藏、软件重用和接口重用等面向对象程序设计的基本特征和基本方法；第9章作为面向对象程序设计思想和方法的一个具体应用，以Visual C++ 6.0作为开发环境，简介在Windows环境下利用MFC类库设计Windows应用程序的基本方法和思想，讲述了MFC应用程序框架、MFC类库的层次结构、Windows消息处理机制和基于MFC的消息映射方式，介绍了鼠标消息、键盘消息和用户自定义消息的处理方法等。本书采用的编写顺序是先过程后对象，先基础后设计再到应用，进而以应用促进基本概念和基本程序设计方法的理解和掌握。书中的语言基础、设计和应用三部分内容相互衔接，前后呼应，便于读者循序渐进地学习。本书每一章都附有大量的、富有启发性的习题，便于读者加深理解和巩固提高。本书可作为高等院校计算机及相关专业本科面向对象程序设计课程的教材，也可作为工程技术人员和广大计算机爱好者自学的参考书。

目　录

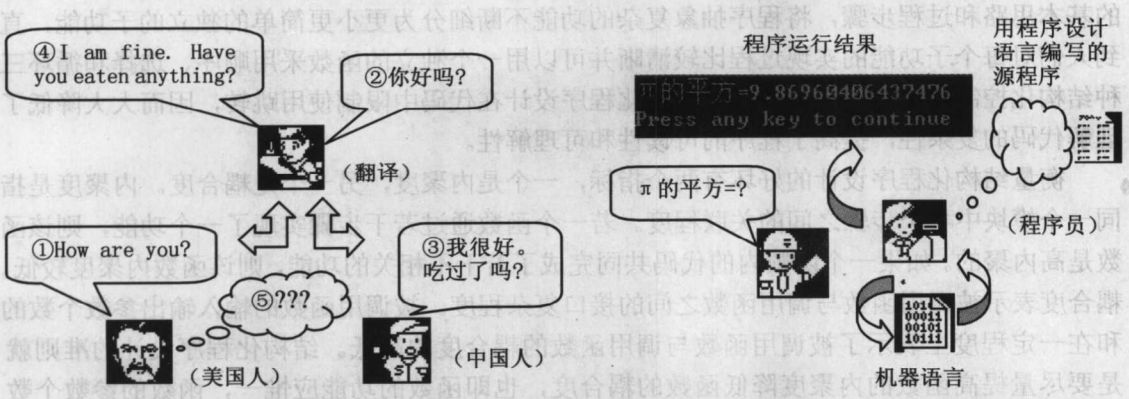
第1章　程序设计与C++语言

1.1　程序设计与程序设计语言

程序设计如同电子、机械和建筑设计一样，也是一门工程设计。其基本方法就是采用程序设计语言编写计算机为了完成某一特定任务而必须执行的一系列指令。

程序设计语言是人与计算机进行交流沟通的一种形式语言，是人们运用计算机分析问题、解决问题的一个基本工具。程序设计语言由一组特定的文字集和一定的语法规则组成。最早，程序员使用的程序设计语言是原始的计算机指令，即一串可被计算机直接识别的二进制数，称之为机器语言。随后，在机器语言的基础上增加了便于人们记忆的助记符，即所谓的汇编语言。再之后，Fortran，Basic，C，C++，Java等上百种高级语言应运而生。

图1-1形象地表示了人与人之间采用不同自然语言交流和人运用程序设计语言与计算机交流的相似与不同之处。



（a）人们采用不同自然语言(英文和中文)进行交流　　（b）程序员运用程序设计语言与计算机进行交流

图1-1　自然语言与程序设计语言

图1-1（a）说明了一个不懂中文的美国人用英语和一个不懂英语的中国人用中文交流的情景片段，两者之间的交流经过一个既懂英语又懂中文的翻译将双方的对话翻译成各自的母语。图中的数字序号表示对话的顺序，可见由于不同的文化背景，双方在交流中可能会发生相互误解的情形：当中国人出于客套，问美国人是否吃过饭了，美国人可能理解这个中国人打算准备请他吃饭吧！

图1-1（b）则形象地表示了程序员将人们的需求和求解问题的算法用程序设计语言加以描述，然后经编译程序（其功能相当于图1-1（a）中的翻译）编译，转换成计算机能够直接识别和执行的机器语言。计算机执行用机器语言表示的人们的指令，并将计算结果通过其输

出设备反馈给人们。

程序设计语言可分为低级语言和高级语言两大类。用汇编语言等低级程序设计语言编写的程序能直接被计算机的控制器所识别，从而控制计算机的运算器进行运算，其工作方式就像使用同一种语言的两个人在交谈一样。而采用高级程序设计语言（如 C++）编写的程序，必须经过编译器编译转换成机器语言之后计算机才能识别与理解，就像只会中文的人和只会英语的人对话一样，必须通过翻译。计算机将高级语言编写的程序翻译成低级语言的过程叫做编译。

程序设计语言的发展是一个从低级到高级、从具体到抽象的过程，语言越高级意味着它越接近自然语言。但目前程序设计语言呈线性等差级数的发展态势，还远远跟不上计算机硬件呈指数等比级数发展的趋势。

1.2 面向过程和面向对象程序设计方法简介

1.2.1 面向过程的程序设计

传统的模块结构化程序设计（或面向过程的程序设计）使用函数作为模块化的单位，其基本思想是自顶向下、逐步求精、模块化设计、结构化编码。首先从问题出发找出解决问题的基本思路和过程步骤，将程序抽象复杂的功能不断细分为更小更简单的独立的子功能，直到人们对每个子功能的实现过程比较清晰并可以用一个独立的函数采用顺序、选择和循环三种结构化控制结构的组合来实现。结构化程序设计在代码中限制使用跳转，因而大大降低了理解代码的复杂性，提高了程序的可读性和可理解性。

衡量结构化程序设计的好坏有两个指标，一个是内聚度，另一个是耦合度。内聚度是指同一个模块中各个步骤之间的关联程度。若一个函数通过若干步骤实现了一个功能，则该函数是高内聚的。如果一个函数内的代码共同完成了若干不相关的功能，则该函数内聚度较低。耦合度表示被调用函数与调用函数之间的接口复杂程度。被调用函数的输入输出参数个数的和在一定程度上表示了被调用函数与调用函数的耦合度的高低。结构化程序设计的准则就是要尽量提高函数的内聚度降低函数的耦合度，也即函数的功能应惟一，函数的参数个数要尽可能地少。但实际上高内聚低耦合是一对矛盾，函数功能的惟一性要求会造成划分出来的函数太多太小，从而导致函数之间的接口关系越来越复杂（要相互调用，或者共享同样的数据等）。

结构化程序设计将程序按过程和功能分解成若干个功能独立的模块之后，通常由几个人来完成，每个人负责几个模块，共同进行程序的开发。由于分开的模块不可能完全独立（例如可能是实现同一个算法的不同步骤等），不同的开发人员必须相互协作，他们需要开会讨论写备忘录、生成设计文档等。这样就会造成某些问题被误解，某些事情被遗忘，某些问题经共同讨论决定下来后可能仍未被及时修改。当隐藏的错误被发现之后，对代码要进行修改。代码某一部分的修改又可能潜在地影响到代码的其他地方，从而造成错误的恶性循环。总之，设计和维护人员在程序的开发和维护过程中需要考虑的因素太多，容易出现难以发现和改正的错误。

综合上述分析，结构化程序设计方法的致命缺点在于数据与处理数据的程序之间的分离，程序设计不是以数据为中心，而是以功能、按步骤来进行。造成的结果往往是在程序中应该放在一个单元内具有高内聚的部分被分开了，这样增加了单元之间进行合作、协调和通信的额外开销。而应该分开的，关系不紧密低耦合的部分却放在了一个单元，增大了软件编码的复杂性以及软件开发与维护的费用。

1.2.2　面向对象的程序设计

访问和修改某个特定数据的一组函数（比如 getData ( ), deleteData ( ) , changeData ( ), addData ( ) , showData ( )等）通常应该放在程序源代码的同一个地方，这样有助于维护人员和新的程序设计人员理解程序员在当初编写程序时的意图。对于小型的程序，这一点不难做到，但对于大型的程序，结构化程序设计将无法保证所有访问和修改某个特定数据的函数都存放在程序的同一个地方。函数模块之间存在着过多的相互联系，函数模块对数据的引用使得单元代码的意义模糊不清。简言之，结构化程序设计无法保证将程序中应该放在一起的内容放在一个单元。

面向对象程序设计的基本思想是将数据和对数据进行操作（输入、访问、修改、输出等）的函数绑定封装在一个称为类的数据结构中。通过类将程序中相关的内容合并在一个单元，解决了结构化程序设计存在的问题。这样在修改某个数据时，只要熟悉描述这个数据的类就行了，而不必担心程序中还有其他函数会访问或修改该数据，这就是面向对象程序设计的封装性；并且还可通过访问控制机制将数据私有化，以保证只有与该数据封装在同一个类的函数才能访问这些数据，这也就是面向对象程序设计的隐藏性。封装性和隐藏性构成了面向对象程序设计的基础，数据与处理数据的函数封装构成了对象，数据受到了保护，要访问对象的数据只能通过调用对象的成员函数来进行。图 1-2 中以一个公司销售、人事和财务三个部门的组织和运作方式举例说明了这一概念。每个对象（部门）有自己的数据和处理数据的方法，每个对象的数据是私有的，在对象外是不可见的。例如，人事部的数据（人事档案）对财务和销售部门而言是不能随便访问的。如果财务部需要销售部的销售数据，财务部经理不能够直接访问销售部的销售数据，而必须首先给销售部经理发一条消息（电话、书面请求、E-mail）请求销售部经理对销售数据进行访问，并处理后将结果返回给财务部经理。

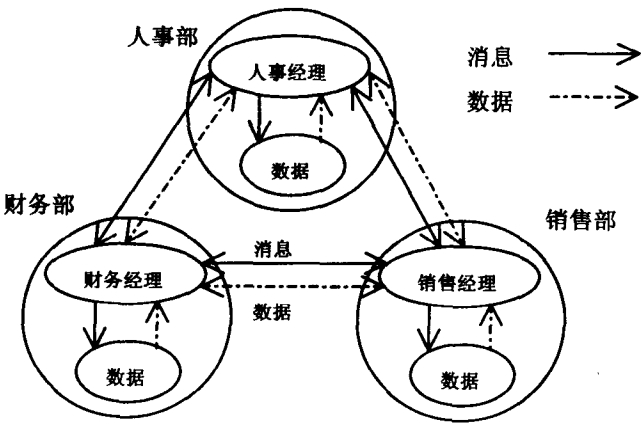


图 1-2　一个公司的组织机构及其运作方式

封装性和隐藏性是衡量一个采用面向对象程序设计方法设计的程序的基准。采用面向对象的程序设计方法，是将程序编写为一组相互协作的对象，而不是一组相互协作的函数。类作为程序的基本单位可减少各个部分之间的相互联系，降低程序代码的耦合性。类的成员函数对同样的数据进行操作，有助于实现模块化，提高程序代码的内聚度。通常一个类可由同一个程序员来开发实现，减少了类之间的相互依赖，就可以减小开发各个类的开发人员之间协调通信容易出现遗漏、误解而导致错误的概率。

面向对象程序设计方法相对于传统的方法，其最大的优点是提高了软件的可维护性。由于数据和对数据的所有操作被封装在类中，数据被设置成私有的只能通过调用类的对数据进行处理的函数来操作。这样，软件维护人员可以通过类的成员函数名清晰地掌握软件设计人员的意图，判断代码的含义。同时，当数据发生改变时，只需要修改相应类中对这些数据进行处理的成员函数就可以了。例如，程序员开始时只使用最后两位数来表示年份，到了2000年后，维护人员需要将用2位数改成用4位数来表示年份。采用传统的方法就需要检查代码中所有涉及到年份的地方，一旦有一处遗漏就会产生“千年虫”问题。而采用面向对象程序设计的方法，只需找到包含有表示年份数据的所有类的描述，然后对类中处理年份数据的函数加以修改即可，而不需要对调用这些类的函数的代码做任何修改。

除了封装性和隐藏性，面向对象程序设计的另一个特性就是继承性。继承性提供类复合的实现机制，有助于实现代码的重用。Microsoft公司就是运用面向对象程序设计方法，将Windows编程用到的几百个应用程序接口函数（API）封装成类，并通过继承机制将其组织成MFC（Microsoft Foundation Class）类库。程序员在编写基于Windows的应用程序时，可通过从MFC类库中派生基本的对Windows底层操作的类，然后对派生类进行修改和扩充来满足应用程序的要求。继承性使得软件的可重用性大大提高，继承性是面向对象程序设计的关键。

多态性是指对象能用不同的接口访问同名的函数。具有不同接口的多个同名函数称为函数的重载，它解决了大型程序中的命名冲突问题。使用传统的程序设计方法必须保证程序中变量和函数命名的惟一性，如果这个程序由多个程序员一起来开发，那么这些开发人员之间必须随时交流，互相关注其他人的命名决定，这样很容易使人们的精力分散，降低工作效率。如果采用面向对象程序设计的方法来进行程序的开发，只有那些要使用你所编写的函数的开发人员才需要知道你对函数的命名，而其他开发人员可集中精力做他们各自所承担的开发工作。多态性还体现在运算符的重载和接口的重用，给运算符赋予新的功能并应用于对象的运算，使程序设计变得简单直观，易于理解阅读，提高程序的可读性和可理解性。

总之，面向对象程序设计提供了一种新的解决程序设计问题的方法，对象是面向对象程序设计的核心，类就是对一组相似对象特征的抽象描述，这些对象具有共同的属性（数据成员）和处理问题的办法（成员函数）。对象是类的一个具体实例，在程序运行时，不是将类装入内存，而是将对象装入内存。同一个类的不同对象具有其自身的数据，处于不同的状态中。

下面通过一个假设简化功能的Motorola－01手机为例来说明面向对象的方法和概念。

所有的Motorola-01手机模型可用一个Motorola-01类来描述，它有如下的属性和操作。

属性：

手机框架尺寸

电路芯片

电源

按钮
键盘
黑白显示幕

操作：

开机
关机
打电话
接电话
调节音量

一部 Motorola-01 手机就是 Motorola-01 类的一个对象，你和你朋友的两部 Motorola-01 手机是 Motorola-01 类的两个不同对象。你在打电话的时候，你朋友可能没有打，或者你在接电话，而你朋友在打电话，这时两个对象处于不同的状态中。打电话就是通过“按钮”向对象（手机）发送一条消息，对象（手机）收到这条消息立即操作对象（手机）的电路芯片使对象（手机）达到你想要的状态。

继承性体现在产品的更新换代上，假设 Motorola－02 手机在 Motorola-01 手机的基础上增加了编辑短信功能，Motorola-02 手机不需要从头开发。描述 Motorola-02 手机模型的 Motorola－02 类可从 Motorola－01 类中派生而来。Motorola－01 手机所有的功能都被 Motorola－02 继承下来，并且 Motorola－02 手机对象增添了新的短信编辑功能且可对从 Motorola－01 手机那里继承的功能进行完善和补充。

多态性体现在不同类型的手机采用不同的界面来实现同一个功能。比如 Motorola－01 手机和 Motorola－02 手机分别使用了不同的电路和芯片来控制音量设置功能，虽然用户在设置音量时的操作界面或功能按键可能不一样，但产生的效果应该是一样的。这样两种不同类的对象产生同样结果的行为就是多态行为。

面向对象程序设计技术也有一定的缺点。由于支持面向对象程序设计的语言通常比较复杂，需要对程序开发人员和管理人员进行培训。使用面向对象的方法要比使用面向过程的传统方法在项目的分析和设计阶段通常要花费更长的时间。尤其对于已经习惯于采用面向过程程序设计方法的有经验的分析员和程序员而言，要改变他们的固定思维模式往往是比较困难的，其结果可能是开发出来的软件不但实现不了面向对象方法的优点，反而使得程序可能比传统的程序更长、更复杂、更慢甚至更难以维护。本书在介绍 C++面向对象程序设计语言的同时，也介绍怎样避免和解决上述问题的方法。

1.3　C 语言和 C++语言的特点

1.3.1　C 语言的特点

C++语言是从 C 语言进化而来的，它是 C 语言的超集。C 语言具有如下一些特征和优点。

（1）语言简洁、紧凑，使用方便、灵活。C 语言只有 32 个关键字，程序书写形式自由。所以 C 语言便于软件开发人员写出优雅美观、可读性好的源代码。

（2）具有丰富的运算符和数据类型，支持结构化的块结构和流控制。C 语言是一个适于实现复杂算法和复杂数据结构的高级语言。

（3）提供了直接访问内存地址的机制，能进行位操作，允许程序员控制计算机中寄存器、端口、标志位屏蔽码等硬件资源，生成的目标代码质量高，程序运行效率高。这一优点使得 C 语言成为一种面向性能的系统程序设计语言。事实上，C 语言就是为了开发 Unix 操作系统而发明的。

（4）可移植性好。C 语言在源代码级上支持程序的可移植性，用 C 语言写的源代码可以不加修改地在不同的编译器上进行编译或者在不同的操作系统或不同的硬件平台上进行编译，经编译后的可执行程序就可以在不同的操作系统或不同的硬件平台上运行。

C 语言也有如下一些局限性。

（1）C 语言数据类型检查机制相对较弱，在表达式和函数调用中可以自动地进行数值类型之间的转换，这使得程序中的一些错误不能在编译阶段被发现。例如，表达式 1/2 的值为 0 而不是 0.5。

（2）数组下标越界既不能在编译阶段被发现，也不能在运行时被发现，从而造成因非法使用下标导致内存被破坏的错误。如类型转换、指针操纵、数组处理、参数传递、变量初始化等都是导致程序出错的根源。

（3）为使程序编写简洁紧凑，对包括标点符号和运算符在内的符号赋予了特定的含义，且这些符号可因为上下文不同，而分别代表不同的含义，使得 C 语言的初学者一时难以理解和适应。

（4）C 语言本身几乎没有支持代码重用的语言结构。因此一个程序员精心设计的程序，很难为其他程序所用。

（5）当用 C 语言编写的程序的规模达到一定程度时，程序员很难控制程序的复杂性。

（6）C 语言是一个结构化程序设计（或面向过程程序设计）语言，不支持面向对象的程序设计方法。

1.3.2 C++语言的特点

C++语言在继承了 C 语言的全部特征和优点的同时，对 C 语言进行了扩充，主要是引进了类这一复合数据类型以支持面向对象的程序设计。C++语言对 C 语言完全向后兼容，即每一个合法的 C 语言程序就是一个合法的 C++程序，这样 C++既支持面向对象程序设计也支持传统的面向过程程序设计，灵活性很大，符合广大程序设计人员逐步更新其程序设计观念和方法的要求，成为当今最流行的程序设计语言之一。C++同时也是学习面向对象程序设计的经典语言。

C++语言的特点具体体现在以下几个方面。

（1）在 C 语言的结构数据类型的基础上增加了类这一复合数据类型，使数据和对数据进行处理的操作可被封装绑定在类这样一个基本单元代码中。数据被限定在类的范围内，使其在类的外部为不可见，从而实现了信息隐藏。C++语言通过类将关系密切的一组函数聚集，并与其共同处理的数据结合在一起。而在 C 语言编写的程序代码中，是无法显示某个函数与其他函数之间的逻辑关系和紧密程度的。

（2）C++程序由对象组成，而任何事物都是对象，复杂的对象可以由比较简单的对象以某种方式组合而成。客观世界由各种对象组成，整个世界就是一个最复杂的对象。因此采用C++语言进行程序设计与现实世界的客观规律和人类习惯的思维方法相一致。

（3）C++提供了类继承和派生的实现机制，高层次的类可以重用低层次的类。类复合和类继承使得程序员可以实现现实世界中复杂的模型以便于软件的重复利用。

（4）C++通过构造函数隐含地对对象进行初始化。在对象定义时，不需要显式地调用初始化函数就可自动地给对象分配计算资源，实现对象的初始化。计算结束后，分配给对象的资源也可在对象消失时自动地回收。

（5）C++支持函数的重载，具有不同接口的函数可以使用相同的函数名。在不同的类的内部也可以使用相同的变量名和函数名，这些机制消除了命名冲突问题。运算符重载使运算符的操作数从基本的数据类型扩充到类对象，提高了软件的可理解性和可读性。

（6）C++提供了虚函数来实现程序设计的多态性，使得基类的函数接口可以在其派生类中继承重用。

（7）C++采用模板，使得函数和类的定义可适用于多种数据类型，使程序设计标准化、通用化，提高了软件开发的效率。

C++语言的缺点表现在以下方面。

（1）C++语言比 C 语言要复杂得多，学习 C++语言要比学习 C 语言难得多。在编写小型程序时，C++可能还不如 C 语言简捷。

（2）使用 C++语言本身并不会给程序的设计与开发带来任何优点，C++语言必须与面向对象程序设计方法相结合，在程序中大量使用 C++语言提供的面向对象的特征和功能，才能体现C++的特点。否则编写程序的效率可能会与使用传统的语言编写程序的一样，还可能更糟糕。

1.4　C++语言初步

1.4.1　C++语言的词法

C++语言的词法包括标识符、分割符、关键字、注释符和运算符等。本节先介绍 C++的标识符、分割符和关键字。注释符和运算符的表示和使用方法将在后续章节中讲解。

C++的标识符是由程序员自定义的，用于表示变量名、类型名和字符常量等程序实体的一个单词。标识符只能以字母或者下划线“_”开头，而不能以数字或其他符号开头。标识符的其余字符可以是英文大小写 26 个字母、数字 0～9 和下划线。分割符有空格符、逗号（,）、分号（;）、冒号（:）、单引号（' '）、双引号（" "）、中括号（[]）和大括号（{}）等。关键字即系统保留字，程序中的其他标识符不能与关键字同名。下面列出的是 C 语言和 C++语言共有的关键字：

```
auto     break    case     char     const    continue default   do     double
else     enum     extern   float    for      goto     if       int    long
register return   short    signed   sizeof   static   struct   switch typedef
union    unsigned void     volatile while
```

以下是C++增加的不属于C语言的31个关键字：

asm	bool	catch	const_cast	class	delete	ynamic_cast	plicit
export	false	friend	inline	mutable	namespace	new	perator
private	protected	public	reinterpret_cast		static_cast	template	this
throw	true	try	typeid	typename	using	virtual	wchar_t

1.4.2 C++程序的框架结构

1. 结构化程序设计的C++程序结构

C++既支持面向对象的程序设计，也支持面向过程按功能划分模块的结构化程序设计。按照结构化程序设计的方法，一个C++的程序由函数来组成，其基本的框架结构和组织形式为：

-预处理程序命令
-全局变量的定义和说明
-用户自定义函数
-主函数main

程序1-1是一个简单的C++程序。程序首先输出“This is my first C++ program”，然后提示用户输入一个角度值，分别计算该角度的正弦和余弦值并输出结果。

```
/* 预处理程序命令 */
#include <iostream>
#include <cmath>
using namespace std;

#define  PI  3.1415926

/* 用户自定义函数 */
void displayGreeting()
{
    cout << "This is my first C++ program" << endl;
}

/* 主函数 */
void main()
{
double  sin_value, cos_value, angle_in_degree, angle_in_radian;

   displayGreeting();

   cout << "Please enter an angle in degree: : ";
   cin >> angle_in_degree;

   angle_in_radian = PI * angle_in_degree / 180.0;
   sin_value = sin(angle_in_radian);
   cos_value = cos(angle_in_radian);
```