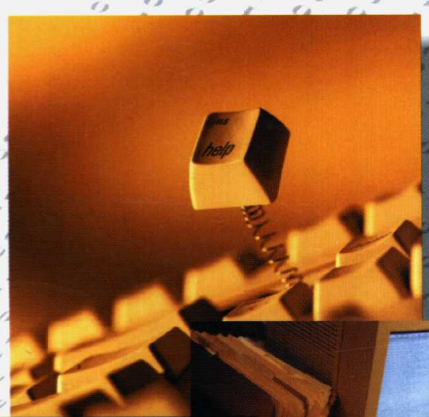
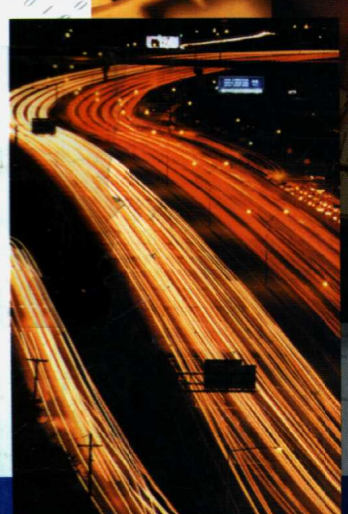
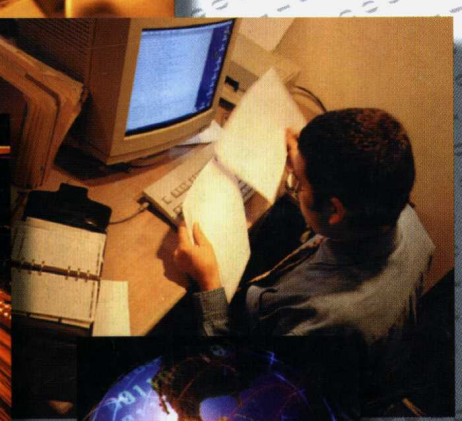
河北省普通高校计算机基础教育教材

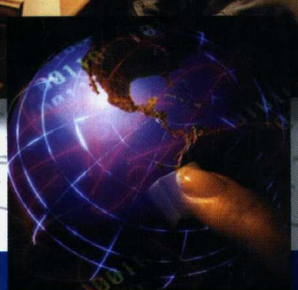
C/C++程序设计

主编 柴 欣









河北大学出版社

C/C++ 程序设计

主　编　柴　欣
副主编　韩　卫
编　委　武优西　史巧硕
　　张春英　阿建卓

河北大学出版社

责任编辑:马　力
封面设计:赵　谦
责任印制:蔡进建

图书在版编目(CIP)数据

C/C++程序设计/柴欣主编. -保定:河北大学出版社,2002.11
ISBN 7-81028-880-6

Ⅰ.C...　Ⅱ.柴...　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(2002)第084136号

出版:河北大学出版社(保定市合作路88号)　　经销:全国新华书店
印制:河北供销印刷厂　　规格:1/16(787mm×1092mm)
印张:20.25　　字数:493千字　　印数:00001~10000册
版次:2002年12月第1版　　印次:2002年12月第1次

ISBN 7-81028-880-6/TP·43　　定价:26.00元

河北省普通高校计算机基础教育
教材编审委员会

内容简介

本书是学习C++语言程序设计的基础教程，全书较为系统地讲述了C++语言的基础知识、基本规则及编程方法，在此基础上，对C++的面向对象的重要特征如类、封装、继承、多态、I/O流操作等进行讲解。

为了加强基础、注重实践，在内容讲解上采用循序渐进、逐步深入的方法，突出重点，注意将难点分开，使读者易学易懂。

本书可作为大专院校各专业程序设计的正式教材，又可作为研究生计算机基础教育的教材，也可借计算机培训班和读者自学使用。

前　言

C++是一种面向对象的程序设计语言，它是在C语言的基础上发展起来的，几乎包括了C语言的全部功能，并克服的C语言的某些不足，是目前使用非常广泛的面向对象的程序设计语言。本书的作者长期从事C++语言程序设计课的教学工作，并利用C++、Visual C++语言开发了多个软件项目，因此有着丰富的教学经验和较强的科研能力，对C++有着较深入的理解。为了使初学程序设计的读者能够掌握C++程序设计语言的使用方法，并初步具备使用C++程序设计语言开发应用程序和解决实际问题的能力，作者精选了C++的内容，本着加强基础、注重实践、勇于创新、突出应用的原则，力求使本教材达到可读性、适用性和先进性的良好融合。为了便于读者自学，在全书的体系结构和内容上注意了由浅入深、深入浅出、循序渐进的方针。为了提高读者编程技巧，在大部分章节中都提供了典型例题。

本书从软件设计思想出发，将全书分为两个部分，第一部分介绍C++的结构化程序设计基础知识，并融和了结构化设计方法。在该部分中，较为系统地讲述了C++语言的基础知识、基本规则及编程方法，其中第1章介绍了程序设计的发展，讲解了结构化程序设计思想和面向对象程序设计思想，还通过一个简单的例子，介绍了C++的基本术语和概念，并对C++程序的调试和运行环境进行了介绍。第2～6章讲述了C++语言的基本内容，包括：常量、变量、运算符和表达式、各种语句、函数和储类、指针和引用、结构体和联合体等，这些内容也是构成C++程序的基础。

第二部分由第7～11章组成，重点介绍C++的面向对象的基本思想及面向对象的设计方法。在这部分中，较系统地讲述了C++语言中面向对象的主要特征，如封装、继承、多态、I/O流操作等，这些都是C++的核心内容，体现了C++语言面向对象的特点。

由于目前的计算机操作系统多为Windows图形界面平台，因此本书在Visual C++ 6.0环境下进行过调试和运行。

教师在选用本书作为大学生软件技术基础课程的教材时，可根据实际授课时数取舍其中的章节。由于授课时数的限制，教师可在规定授课时数内重点讲授第一部分的内容，而在后续的选修课程或研究生课程中介绍第二部分的内容，这样可使学生完整地了解C++的内容。因此，本书适用于本、专科各专业及硕士研究生的计算机基础教育。

本书由柴欣主编，并负责全书的总体策划与统稿、定稿工作，韩卫任副主编。各章编写分工如下：第1、3、6章由柴欣编写，第2、4章由史巧硕编写，第5章由武优西编写，第7～11章由韩卫编写。参加本书大纲讨论及部分编写工作的老师还有张春英、阿建卓等。

本书作为河北省统编教材，在编写之初即得到编审委员会多位专家的大力支持，在编写过程中也一直得到各位专家的热心指导与无私帮助，编者在此一并表示衷心的感谢。此外在本书写作时，还参考了大量文献资料，在此也向这些文献资料的作者深表感谢。

由于时间仓促和水平所限，书中难免有不当和欠妥之处，敬请各位专家、读者不吝批评指正。

编　者

2002年10月

目　录

第1章 绪 论

1.1 计算机语言发展概述

1.1.1 计算机程序设计语言的发展

计算机之所以能自动进行计算，是因为采用了程序存储的原理，计算机的工作体现为执行程序。程序是控制计算机完成特定功能的一组有序指令的集合，编写程序所使用的语言称为程序设计语言，它是人与计算机之间进行信息交流的工具。从1946年世界上诞生第一台计算机起，在短短的50余年间，计算机技术迅速发展，程序设计语言的发展从低级到高级，经历了机器语言、汇编语言、高级语言到面向对象语言的多个阶段，具体过程如下。

1. 机器语言

计算机能够直接识别和执行的二进制指令(也称机器指令)的集合称为该种计算机的机器语言。早期的计算机程序都是直接使用机器语言编写的，这种语言使用0、1代码，因此编写出的程序难以理解和记忆，目前已不被人们使用。

2. 汇编语言

通过助记符代替0、1机器指令以利于理解和记忆，由此形成了汇编语言。汇编语言实际上是与机器语言相对应的语言，只是在表示方法上采用了便于记忆的助记符号来代替机器语言相对应的二进制指令代码，因此也称为符号语言。计算机不能直接识别汇编语言，需要编译后才能识别。这种语言的执行效率较高，但由于难以记忆，因此使用较少。

3. 高级语言

机器语言和汇编语言是面向机器的语言，高级语言采用更接近自然语言的命令或语句，使用高级语言编程，一般不必了解计算机的指令系统和硬件结构，只需掌握解题方法和高级语言的语法规则，就可以编写程序。高级语言在设计程序时着眼于问题域中的过程，因此它是一种面向过程的语言。对于高级语言，人们更容易理解和记忆，这也给编程带来很大方便，但它与自然语言还是有较大差别的。

4. 面向对象语言

面向对象语言是比面向过程语言更高级的一种高级语言。面向对象语言的出现改变了编程者的思维方式，使设计程序的出发点由着眼于问题域中的过程转向着眼于问题域中的对象及其相互关系，这种转变更加符合人们对客观事物的认识。因此，面向对象语言更接近于自然语言，是人们对于客观事物更高层次的抽象。

目前，世界上已经设计和实现的计算机语言有上千种之多，但实际被人们广泛使用的不过数十种。

1.1.2 程序设计的发展历程

回顾程序设计发展的历史，大体上可以划分为如下几个不同的时期。

20世纪50年代的程序都是用指令代码或汇编语言来编写的，这种程序的设计相当麻

烦，编写和调试一个稍大一点的程序常常要花费很长的时间，培养一个熟练的程序员更需经过长期的训练和实习，这种局面严重影响了计算机的普及与应用。

20世纪60年代高级语言的出现大大简化了程序设计，缩短了解题周期，因此显示出强大的生命力。此后，编制程序已不再是软件专业人员才能做的事了，一般工程技术人员花上较短的时间学习，也可以使用计算机解题。这个时期，随着计算机的应用日益广泛地渗透到各学科和技术领域，也发展了一系列不同风格的、为不同对象服务的程序设计语言。其中较为著名的有FORTRAN、COBOL、ALGOL、LISP、PL/1、PASCAL等十几种语言。高级语言的蓬勃兴起，使得编译和形式语言理论相应日趋完善，这是该时期的主要特征。但就整个程序设计方法而言，并无实质性的改进。

自20世纪60年代末到70年代初，出现了大型软件系统，如操作系统、数据库，这给程序设计带来了新的问题。大型系统的研制需要花费大量的资金和人力，可是研制出来的产品却是可靠性差，错误多，且不易维护和修改。一个大型操作系统有时需要几千人年的工作量，而所获得的系统又常常会隐藏着几百甚至几千个错误。当时，人们称这种现象为“软件危机”。

“软件危机”震动了软件行业，程序设计的传统习惯和工作方式导致了不清晰的程序结构，使得程序的可靠性难以保障。另一方面，程序设计工具的严重缺乏也使得大型系统的开发陷入困境。此时人们开始重新审视程序设计中的一些最基本的问题。例如，程序的基本组成部分是什么，应该用什么样的方法来设计程序，如何保证程序设计正确，程序设计的主要方法和技术应如何规范等等。

1969年，E.W.Dijkstra首先提出了结构化程序设计的概念，强调了从程序结构和风格上来研究程序设计。经过几年的探索和实践，结构化程序设计的应用确实取得了成效，用结构化程序设计的方法编写出来的程序不仅结构良好，易写易读，而且其正确性易于证明。

20世纪70年代末结构化设计方法得到了很大的发展，Niklans Wirth又提出了“算法＋数据结构 ＝ 程序设计”的程序设计方法，他将软件划分成若干个可单独命名和编址的部分，它们被称为模块，模块化使软件能够有效地被管理和维护，能够有效地分解和处理复杂问题。到80年代，模块化程序设计方法普遍被人们接受。

虽然几十年来结构化程序设计技术得到了广泛的使用，但有些问题仍未得到很好的解决。由于软件开发是对问题的求解过程，从认识论角度看，软件开发过程包括人们对要解决问题及相关事物的认识和基于认识所进行的描述。而结构化设计方法不能直接反映出人类认识问题的过程。另外，结构化设计方法中，程序模块和数据结构是松散地耦合在一起的，因此，当应用程序比较复杂时，容易出错，难以维护。随着计算机软件的发展，软件系统越来越复杂、庞大，结构化程序设计方法已显得力不从心。

在20世纪80年代，人们又提出了面向对象的程序设计方法。这种方法直接对问题域中的客观事物建造分析模型中的对象，使对象的描述与客观事物一致，保持问题域中的单个事物及事物之间的关系的原貌，而面向对象的语言可以直接描述问题域中的对象及其相互关系。用面向对象的程序设计方法，可以使人们对复杂系统的认识过程与系统的程序设计与实现过程尽可能地一致，这是因为它从客观世界中所存在的事物出发，比较符合人们的思维方式。因此，目前这种面向对象的程序设计模式正逐渐取代“数据结构＋算法”的面向过程的程序设计模式。

1.1.3 结构化程序设计概述

自提出结构化程序设计的概念后，经过十几年的发展，结构化程序设计已经具有了很广泛的内容，大体上可以归纳为以下几点：

1. 结构化程序的基本结构

结构化程序包含有三种基本结构，人们可以用这三种基本结构来展开程序，表示一个良好的算法，从而使程序的结构清晰、易读易懂且质量好、效益高。这三种基本结构为顺序结构、选择结构和循环结构。

(1) 顺序结构

顺序结构是一种最简单、最基本的结构，在顺序结构内，各块按照它们出现的先后顺序依次执行。图 1-1-1 表示了一个顺序结构形式，从图中可以看出它有一个入口 a 点，一个出口 b 点，在结构内 A 框和 B 框都是顺序执行的处理框。

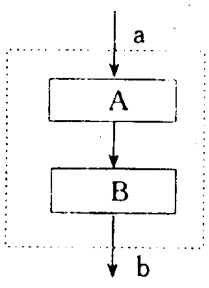


图 1-1-1 顺序结构流程图

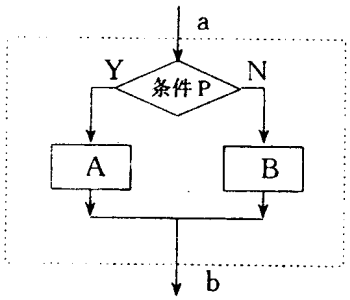


图 1-1-2 选择结构流程图

(2) 选择结构

选择结构中包含一个判断框，根据给定的条件 P 是否成立而选择执行 A 框或 B 框，当条件成立时，执行 A 框，否则执行 B 框。A 框或 B 框可以是空框，即不执行任何操作，但判断框中的两个分支，执行完 A 或 B 后都必须汇合在一起，从出口 b 退出，然后接着执行其后的过程。图 1-1-2 所示的虚线部分就是选择结构，在选择结构中程序产生了分支，但对于整个的虚线框而言，它仍然只具有一个入口 a 和一个出口 b。

(3) 循环结构

循环结构又称重复结构，是指在一定条件下反复执行一个程序块的结构。循环结构也是只有一个入口，一个出口。根据循环条件的不同，循环结构分为当型循环结构和直到型循环结构两种。

① 当型循环的结构，如图 1-1-3 所示，其功能是：当给定的条件 P 成立时，执行 A 框，执行完 A 框后，再判断 P 条件是否成立，如果成立，再次执行 A 框，如此重复执行 A 框，直到判断 P 条件不成立才停止循环。此时不执行 A 框，而从出口 b 脱离循环结构。

② 直到型循环的结构，如图 1-1-4 所示，其功能是：先执行 A 框，然后判断给定条件 p 是否成立，如果不成立，再次执行 A 框；然后再对 P 进行判断，如此反复，直到给定的 P 条件成立为止。此时不再执行 A 框，从出口 b 脱离循环结构。

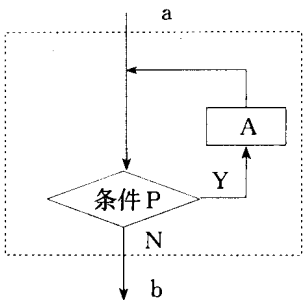


图 1- 1- 3 当型循环结构流程图

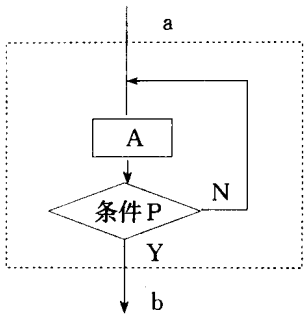


图 1- 1- 4 直到型循环结构流程图

由以上三种基本结构构成的程序，称为结构化程序。一个结构化程序以及三种基本结构中的每一种结构都应具有以下特点：

ⓐ有一个入口。

ⓑ有一个出口。

ⓒ没有死语句，即每一个语句都应该有一条从入口到出口的路径通过它(至少通过一次)。

ⓓ没有死循环(无限制的循环)。

实践证明，任何满足以上四个条件的程序，都可以表示为由以上三种基本结构所构成的结构化程序。反之，任何一个结构化程序都可以分解为若干基本结构。

2. 结构化程序的设计方法

结构化程序主要采用自上而下、逐步细化的设计方法，即先全局后局部、先整体后细节、先抽象后具体的设计方法。由于实际问题往往比较复杂，为了提高效率，在进行结构化程序设计时，常常伴随着使用关键部分优先考虑的设计方法。

3. 结构化程序的组织结构

在结构化程序中常常用模块化结构来组织程序，图 1－1－5 给出了用模块化结构组织程序的示意图。

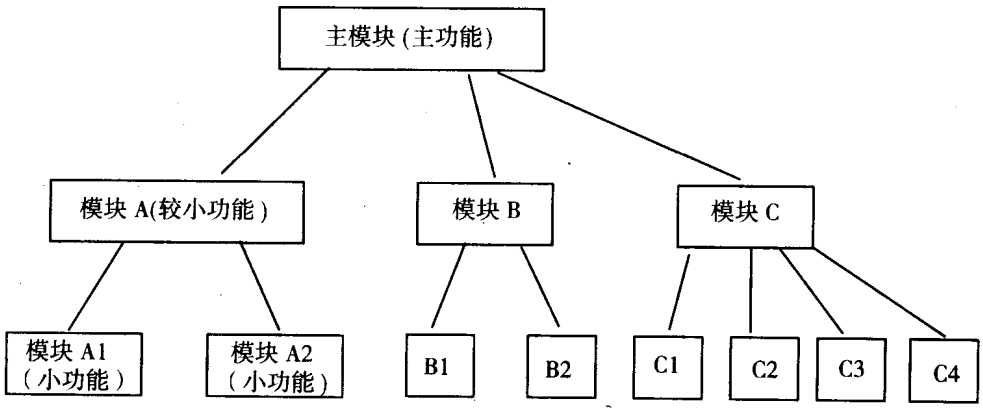


图 1－1－5 程序模块化示意图

1.1.4 面向对象程序设计概述

1. 面向对象方法的起源

结构化程序设计技术虽已使用了几十年，但如下问题仍未得到很好的解决。

首先，面向过程的设计方法与人们习惯的思维方法仍然存在一定的差距，所以很难自然、准确地反映真实世界。因而用此方法开发出来的软件，有时很难保证其质量，甚至需要进行重新开发。

其次，结构化程序设计在方法实现中只突出了实现功能的操作方法(模块)，而被操作的数据(变量)处于实现功能的从属地位，即程序模块和数据结构是松散地耦合在一起的。因此当应用程序比较复杂时，容易出错，难以维护。

由于上述缺陷，结构化程序设计方法已不能满足现代化软件开发的要求，一种全新的软件开发技术应运而生，这就是面向对象的程序设计(Object Oriented Programming，简称OOP)。

20世纪80年代，在软件开发中各种概念和方法积累的基础上，就如何超越程序的复杂性障碍，如何在计算机系统中自然地表示客观世界等问题，人们提出了面向对象的程序设计方法。面向对象的方法不再将问题分解为过程，而是将问题分解为对象。对象将自己的属性和方法封装成一个整体，供程序设计者使用。对象之间的相互作用则通过消息传递来实现。用面向对象的程序设计方法，可以使人们对复杂系统的认识过程与系统的程序设计与实现过程尽可能地一致。目前，这种“对象＋消息”的面向对象的程序设计模式正逐渐取代“数据结构＋算法”的面向过程的程序设计模式。

2. 面向对象语言的发展

早在20世纪60年代，就出现了最早的面向对象语言——Simula 67语言，具有了类和对象的概念。随后又推出了纯面向对象设计语言，如80年代美国Xerox Palo Alto研究中心推出了Smalltalk，它完整地体现并进一步丰富了面向对象的概念，还开发了配套的工具环境，为其最终实用化奠定基础。但由于当时人们已经接受并广泛应用结构化设计理论，而不能一下子完全接受面向对象程序设计的所有思想等诸多原因，这些语言并没有能够广泛流行起来。后来人们开始对流行的语言进行面向对象的扩充，曾经推出过许多种类的版本，而主要成功的代表是在当时流行的程序设计语言——C语言的基础上开发的C++语言。这时面向对象语言已形成几大类别：一类是纯面向对象的语言，如Smalltalk和Eiffel；另一类是混合型的面向对象语言，如C++和Objective C；还有一类是与人工智能语言结合形成的，如LOOPS、Flavors和CLOS以及适合网络应用的JAVA语言等。

C++语言是由AT&T公司贝尔实验室的Bjarne Stioustrup博士开发的，它的创作灵感来源于计算机语言多方面成果的凝聚，特别是BCPL(Basic Combined Programming Language，C语言的基础)和Simula 67(以面向对象为核心的语言)，同时也借鉴了Algol 68语言。C++的名字是由Rick Masenirti提出的，到1983年被确定。

C++是一门高效、实用的混合型程序设计语言，它包括两部分内容：一是C++基础部分，它是以C语言为核心的。另一部分是C++面向对象特性部分，是C++对C语言的扩充部分。这样它既支持面向对象程序设计方法，又支持结构化程序设计方法。同时，广泛的应用基础和丰富的开发环境的支持，也使面向对象设计得到很快普及。

3. C++语言对面向对象方法的支持

C++语言是C语言的扩展，其基础部分除了一些细微的差别外，可以说是C语言的超集，它保留了C语言功能强、效率高、风格简洁、适合于大多数的系统程序设计任务等优点，使得C++与C之间取得了兼容，因此，在过去的软件开发中积累的大量C的库函数和实用程序都可在C++中应用。

另外，C++语言通过对C的扩充，克服了原有C语言的缺点，特别是引进了面向对象语言的要素，使得C++语言成为一种面向对象的程序设计语言，它对面向对象程序设计方法的支持体现在如下几个方面。

(1) 支持数据封装

在C++语言中，类是支持数据封装的工具，对象是数据封装的实现。在封装中还提供了一种对数据访问的控制机制，使得有些数据被隐藏在封装体内，因此具有隐蔽性。封装体与外界进行信息交换是通过操作接口进行的。这种访问控制机制体现在类的成员中可以有公有成员、私有成员和保护成员。

(2) 支持继承性

C++语言允许单继承和多继承。继承是面向对象语言的重要特性。一个类可以根据需要生成它的派生类，派生类还可以再生成它的派生类。派生类继承了基类成员，另外它还可以定义自己的成员。继承是实现抽象和共享的一种机制。

(3) 支持多态性

C++语言支持多态性表现在：

① C++语言允许函数重载和运算符重载；

② C++语言可以定义虚函数，通过它来支持动态联编。动态联编是多态性的一个重要特征。

1.1.5 C++语言开发环境的发展

随着C++语言逐渐成为ANSI标准，这种新的面向对象程序设计语言迅速成为程序员最广泛使用的工具。众多C++语言的开发环境也不断地推出，竞争十分激烈。1986年Borland公司开发了Turbo C++.，而后又推出了Borland C++版本。Microsoft公司于80年代中期在Microsoft C 6.0的基础上开发了Microsoft C/C++ 7.0，同时引进了Mircrosoft Foundation Class(MFC)库1.0版本，完善了源代码。以前这些版本都是依赖于DOS环境，或在Windows下的DOS模式下运行。不久Microsoft公司推出的Microsoft C/C++ 8.0，即Visual C++ 1.0版本，它是Microsoft公司推出的第一个真正的基于Windows环境下的可视化的集成开发环境，它将编辑、编译、链接和执行集成为一体。从Visual C++ 1.5版本以后，Microsoft公司决定不再将更多的努力花在支持16位编程上，虽然Visual C++ 2.0仍提供对16位的支持，但从2.0版本以后，Visual C++更多地用来创建32位程序。在版本上，Microsoft公司没有推出3.0版本，版本号直接从2.0跳到4.0，这样，Visual C++与MFC的版本号取得一致。由于Internet的流行明显地影响了产品的设计，在4.0版本中，Visual C++引进了为Internet编程而设计的新类库。5.0版本也增加了一些新类，但注意力更多地集中在改善产品的界面上，以提供更好的在线帮助系统、更高级的宏能力和对在开发者组内进行类和其他代码共享的支持。6.0版本在功能上有了进一步的改进。Visual C++经历

了从1.0到6.0等版本的发展，软件系统逐渐庞大，功能日益完善。

1.2 C++语言的词法和词法规则

1.2.1 C++语言的字符集

C++字符集是由下列字符组成的(ASCII码字符集)：

①26个小写字母：a～z。

②26个大写字母：A ～Z。

③10个数字：0 ～ 9。

④其他符号：+、-、*、/、=、,、.、_、:、;、?、\、"、'、~、|、!、#、%、&、(、)、{、}、[、]、^、<、>、空格。

用C++字符集中的字符可以构造各种词法符号。

1.2.2 C++语言的词法规则

每种程序设计语言都使用一组字符来构造具有特殊意义的符号——词法符号，C++有这样一些词法符号：标识符、关键字、运算符、分隔符、常量及注释符等，下面分别进行介绍。

1.标识符

标识符是程序员用来命名程序中一些实体的一个字符序列，用来标记变量名、常量名、函数名、对象名、类型名和语句标号名等。C++要求标识符必须符合这样一些语法规定：

①组成标识符的字符有：A～Z、a～z、0～9或_(下划线)。

②标识符必须是以字母或下划线开始，其后跟字母、数字或下划线组成的字符串。例如：name、NAME、_ old、_ 988、double _ list、Total等都是合法的标识符。而下面的标识符是不合法的：2b(不能以数字开头)、D$7(包含非法字符$)、name 1(包含非法字符空格)。

③标识符的长度可以是任意的，但不同的C++编译器能识别的最大长度不同，编译器将忽略多余字符，而不认为是错误。

④标识符中大小写字母表示不同的含义，例如：time、TIME、Time等标识符在同一程序中使用被视为不同的标识符。

⑤标识符不能分行书写。

为了便于读写，标识符的命名最好选择能够代表一定意义的单词或缩写来标识，但不能用关键字，如day表示日期，myage表示年龄等。为了增强程序的可读性，可适当地使用下划线，如load _ num表示取数据。一般变量名、函数名和类型名用小写字母，符号常量用大写字母。

2.关键字

在C++中关键字又称保留字，它是预先由系统定义好的单词，具有特定的含义。表1-2-1列出了C++关键字。ANSI C规定有32个关键字，表中用斜体字表示。ANSI C++在此基础上又补充了29个关键字，表中用黑体字表示。

注意：C++中关键字都是小写的，在程序设计时经常用到，但不可以作为一般标识符使用。另外，C++是一种正在发展的语言，某些关键字可能是一些编译器所不具备的，使用时

会导致语法错误。关于上述关键字的含义，在本书中会逐渐讲述。

表 1-2-1　C++ 关键字表

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*	*bool*	*catch*	*class*	*const _ cast*
delete	dynamic _ cast	explicit	false	friend	inline
mutable	namespace	new	operator	private	protected
public	reinterpret _ cast	static _ cast	template	this	throw
true	try	typeid	typename	using	virtual
wchar _ t					

3. **运算符**

运算符是一些用来进行某种操作的单词，它实际上是系统预定义的函数名。这些函数作用于被操作的对象上将获得一个结果值。运算符是由一个或多个字符组成的单词。

C++ 语言的运算符除了包含了 C 语言中的运算符外，还增加了一些新的运算符。C++ 语言的运算符可以重载。

4. **分隔符**

分隔符被称为程序中的标点符号，它是用来分隔单词与程序正文的，表示某个程序实体的结束和另一个程序实体的开始。

C++ 常用的分隔符有以下几个：

①空格：用作单词之间的分隔符。

②逗号：用作变量之间或对象之间的分隔符，或用作函数的多个参数之间的分隔符。

③冒号：用作语句标号与语句间的分隔符以及 switch 语句中 case ＜整数型表达式＞与语句序列之间的分隔符。

④花括号：用来为函数体、复合语句等定界。

5. **常量**

C++ 语言中，常量有数字常量、字符常量和字符串常量。程序中的常量经常用符号常量来表示。关键字 const 用来定义各种不同类型的常量。

6. **注释符**

在程序中注释起到对程序的注解和说明的作用，目的是为了便于程序的阅读和分析。