高等学校计算机语言应用教程

C++ 应用教程

胡 也　童爱红　龙 瑞　编著

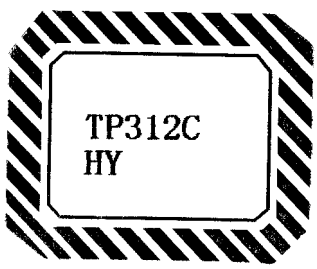
本书配光盘

清华大学出版社 · 北京交通大学出版社

高等学校计算机语言应用教程

C++应用教程

胡 也 童爱红 龙 瑞 编著



清华大学出版社
北京交通大学出版社
·北京·

内 容 简 介

C++语言是一门完全面向对象的程序设计语言，是高校广泛使用的程序设计教学语言之一。本书从教学实践全程的角度对 C++进行了全面的阐述。全书共分 13 章，全面讲解面向对象的程序设计概念、C++的数据类型和表达式、C++的程序设计语句、C++的数组与指针及函数、类、对象、继承、多态性等程序设计的基本要素。本书的重点是 C++的面向对象语言成分，详细深入地探讨类与对象、继承与派生类、多态性与虚函数、模板等的概念及其 C++语言的实现。本书的最后一章还给出了常用数据结构的 C++实现。

本书从教学实践的角度出发，立足于提高学生的程序设计应用能力，全书理论分析透彻严谨，实例丰富生动，内容由浅入深，能快速引导学生进入 C++编程世界。本书可作为理工科本科各专业和信息技术类大专的程序设计教材，也可作为广大希望掌握 C++编程的程序设计人员的参考用书。

图书在版编目（CIP）数据

C++应用教程 / 胡也，童爱红，龙瑞编著. —北京：清华大学出版社；北京交通大学出版社，2005.1
（高等学校计算机语言应用教程）
ISBN 7-81082-427-9

Ⅰ.C… Ⅱ.①胡… ②童… ③龙… 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 097147 号

责任编辑：谭文芳
出 版 者：清 华 大 学 出 版 社　邮编：100084　电话：010-62776969　http://www.tup.com.cn
　　北京交通大学出版社　邮编：100044　电话：010-51686414　http://press.bjtu.edu.cn
印 刷 者：北京东光印刷厂
发 行 者：新华书店总店北京发行所
开　本：185×260　印张：18.75　字数：480 千字　附光盘 1 张
版　次：2005 年 1 月第 1 版　2005 年 1 月第 1 次印刷
书　号：ISBN 7-81082-427-9 / TP・153
印　数：1～5000 册　定价：32.00 元（含光盘）

前　言

程序设计是高等学校理工科各专业学生的一门重要基础课程，目前在高校普遍使用的面向对象的程序设计教学语言主要有C++、Visual C++、Visual Basic、Delphi、Java、Visual C#等。C++是从C语言发展而来的面向对象的程序设计语言，与其他语言相比，具有更广泛的群众基础，是广大学生学习面向对象程序设计的首选语言。有关C++的教材，市场上有很多，但基本上都把侧重点放在面向对象的程序设计概念的论述上，与应用结合不太紧密，从而使学生难以入门。在此背景下，我们编写了这本C++应用教程，目的是为了引导学生快速高效地进入C++编程世界。

本书的编写人员都有多年从事一线教学的经验，从事过本科和大专层次的程序设计教学，对程序设计的教学规律把握较为独到，能够预料到学生在学习中可能遇到的困难并加以解决。同时本书的编创人员均有教材编写的经验，具有很强的敬业精神，编写的教材有助于提高学生的学习效率。

本书的总体编写思路如下。

1．全书分13章，前12章全面深入地讲解C++程序设计语言的主要语言成分及使用方法，第13章给出了常用数据结构的C++语言实现。

2．每章均分4个部分进行编写："理论知识"部分简明扼要地讲解本章的主要理论，并通过小的实例进行深化理解；"典型实例"部分通过分析一些精心挑选和编制的典型实例，强化学生的编程能力；"上机练习"部分挑选一些具有实用价值的上机练习题，加以剖析并给出部分程序代码，引导学生在上机练习中提高应用能力；"课后考场"部分设计了一套试题，方便学习进行自我测试。

3．实例引导。本书的每一章都有着丰富的实例，有的实例具有较强的趣味性，易引起学生的兴趣，激发学生对程序设计的喜好。

与现有的教材相比，本教材的具有以下特色。

1．重点难点突出。针对C++语言的特点，本书没有罗列大量的语言成分，不介绍较琐碎或不太常用的语法成分，而是较详细地介绍了C++的主要语言成分，重点讲述C++程序设计的方法和实际应用。

2．不为写C++语言而写教材。始终贯彻为写程序设计教材而写教材的思路，C++只是选择的一门工具语言。因此本书将重点放在面向对象程序设计的方法上，放在面向对象程序设计教材的共性上。本书力争达到这样的目标：通过本书的学习使学生能够掌握面向对象程序设计的概貌，进入面向对象程序设计的大门。

3．在编写风格上注重学生动手编程能力的培养。针对学生普遍认为程序设计语言难学的特点，本教材不再讲解高深难懂的理论，而是强调通过实例学编程。通过精选有趣的实例，讲解实例的实现过程，激发学生的编程兴趣，引导学生一步一步地步入面向对象程序设计的大门。

本书的配套光盘包含全书的所有例题、习题源代码和可执行文件，所有的程序都在 Visual Studio .NET 环境下调试通过并经过严格测试。本书的电子教案可在北京交通大学出版社网站 http://press.bjtu.edu.cn 下载。

在本书的编写过程中，得到了解放军理工大学计算机与指挥自动化学院黄松副教授的指导和帮助，得到了解放军理工大学工程兵工程学院计算机应用教研室全体老师的指导与帮助，在此表示衷心的感谢。同时对参加本书资料收集、程序测试和文稿校对的李建东、胡光兵、侯太平、汪刚等同志表示衷心的感谢。编者参阅了大量文献资料及网站资料，在此也一并表示感谢。

虽然我们力求完美，力创精品，但由于水平有限，书中难免有疏漏和错误等不尽人意之处，还请广大读者不吝赐教。

编　者

2005 年 1 月

目　录

第 1 章　C++语言概述

本章要点

- ☑ 面向对象程序设计方法的概念和特点
- ☑ C++语言的基本结构和词法
- ☑ Microsoft Visual Studio .NET 开发平台

1.1　理论知识

1.1.1　计算机语言的发展——从面向过程到面向对象

当前的计算机无法直接理解和接受人类的思想，需要借助于计算机语言精确地告诉它做什么、以及如何去做。软件开发的过程就是使用各种计算机语言将人们关心的现实世界映射到计算机世界的过程，如图 1-1 所示。

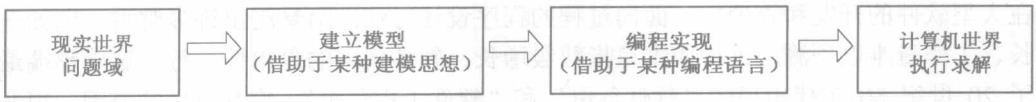


图 1-1　软件开发的过程

纵观计算机发展史，计算机语言的发展如同生物进化，从低级语言逐步发展为高级语言，构成一棵枝繁叶茂的“进化树”。在这棵“进化树”上，可以按照接近人类自然语言和数学语言的程度，将计算机语言划分为三大类：机器语言、汇编语言和高级语言；也可以按照程序设计方法的不同，将计算机语言划分为面向过程的语言和面向对象的语言。

1．面向过程的计算机语言

最早的计算机语言是机器语言，它完全由二进制数字 0 和 1 组成，能够被计算机直接识别和执行，具有灵活、速度快等优点。但是，它的缺点也明显，程序员需要熟记全部的指令及其含义，手工处理每条指令和每个数据。所以在今天看来，采用机器语言编程是一件烦琐的事情。

为了克服机器语言难读、难写和容易出错等缺点，人们发明了汇编语言，采用与指令代码实际含义相近的英文缩写词、字母和数字等符号（助记符）来取代指令代码。例如，在 8086 指令系统中，把寄存器 CL 中的内容加到 BH 中的汇编语言指令为：

```
ADD CL, BH
```

显然，记住该指令比记住机器指令“00000010 11111001”要容易得多。和机器语言一样，汇编语言也是面向机器的，它的指令与机器语言的指令一一对应，紧密依赖于硬件系统，程序的可移植性仍然很差。机器语言和汇编语言被统称为“低级语言”，要求程序员必须十分熟悉计算机的硬件结构及其工作原理。

随着计算机科学的发展，人们开始寻求一些与具体机器指令无关并接近人类自然语言，能被计算机所接受，同时语意确定、规则明确、自然直观的计算机语言，即高级语言。

1957 年，诞生了第一个高级程序设计语言 FORTRAN。它引进了许多现在仍在使用的程序设计概念，如数组、变量、循环、分支等。随后，LISP，COBOL，ALGOL60，BASIC，Pascal 和 PROLOG 等高级语言雨后春笋般地出现，被广泛应用于人工智能、情报检索、商业数据处理和关系数据库等领域。

1967 年剑桥大学的马丁·理查德在 CPL 语言的基础上推出了 BCPL 语言。1970 年美国贝尔实验室的肯·汤普逊对 BCPL 语言进行了简化，并取 BCPL 的第一个字母“B”作为新语言的名称。1972 年，贝尔实验室的布朗·W·卡尼汉和丹尼斯·M·利奇对 B 语言进行了完善和扩充，在保留强大的硬件处理能力的基础上，扩充了数据类型，恢复了通用性，并取“BCPL”的第二个字母作为新语言的名称——这就是著名的 C 语言。此后，两人采用 C 语言重写了 UNIX 操作系统，随着 UNIX 操作系统的广泛应用，C 语言成为一种极受欢迎的计算机语言。1987 年，美国标准化协会（ANSI）制定了 C 语言标准，即 ANSIC。

以上这些计算机语言又被称为面向过程的语言，均直接面对解决问题的过程，将一个问题分为若干个子问题，然后再将这些子问题转换为函数，与问题相关的数据则被独立存储。它们在一定的历史阶段推动了计算机科学的发展，然而随着计算机应用领域的不断扩展，特别是在大型软件的开发和管理中，面向过程的程序设计方法渐渐暴露出许多弊端，例如开发周期长、工作量难以预测、维护成本呈指数级增长、软件代码的复用性差等，这些弊端最终导致了 20 世纪 80 年代出现的“软件危机”和“软件工程危机”。究其背后的原因，根源正是基于“流水线”似的过程化程序设计方法与现实世界之间存在着巨大的认识“鸿沟”。那么，如何缩小这道“鸿沟”呢？面向对象的程序设计方法就是一种跨越“鸿沟”的努力。

2. 面向对象的计算机语言

早在 1967 年，面向对象语言的鼻祖 Simula67 就诞生了。它将 ALGOL60 中块结构的概念向前发展了一大步，提出了对象的概念，并且支持类和类的继承。

随后出现的 Smalltalk 语言继续丰富和发展了 Simula67 中的面向对象的概念，并且实施更加严格的信息隐藏，但由于当时的软件规模还不大，技术也还不太成熟，面向对象的优势并未全部发挥出来。1980 年，Smalltalk-80 问世，面向对象技术开始显示出魅力，成为 C++之前最具有影响力的面向对象语言。

1982 年，美国 AT&T 公司贝尔实验室的 Bjarne Stroustrup 博士在 C 语言的基础上，扩充了 Simula67 语言中面向对象的概念，发明了一种新的语言，这种新语言被命名为 C++。

1990 年，Bjarne Stroustrup 博士出版了 *The Annotated C++ Reference Manual*（ARM），由于当时还没有出台 C++标准，ARM 成为事实上的标准。

1990 年，引入了模板（Template）和异常（Exception）的概念，使 C++具备了泛型编程和更好的运行期错误处理能力。

1993 年，运行时刻类型识别（RTTI）和名字空间（Namespace）的概念被加入到 C++中。

1998 年，ANSI 和 ISO 先后批准 C++语言成为美国国家标准和国际标准，使之成为迄今为止最重要、也是最为成功的一门计算机语言。

C++语言的演化过程如图 1-2 所示。

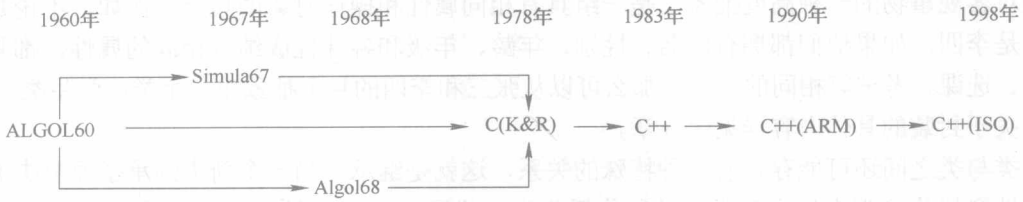


图 1-2　C++语言的发展历程

1.1.2　面向对象的基本概念

编写程序的目的是为了解决问题。先介绍一个例子。

某学校将每位学生的姓名、性别、年龄、年级和各课程的考试成绩输入到一个文件中，到了期末，按照总成绩的高低排出各年级学生的名次。（假设不存在同名同姓的情况。）

解决问题的一种方法就是将一个问题最终分解为多个便于处理的子问题。因此该学生成绩排序问题可以被分解成以下四个子问题：

（1）从文件中读取学生的资料；

（2）汇总各个学生的总成绩；

（3）在每个年级中，按照成绩的高低对学生进行排序；

（4）输出结果。

针对这些子问题设计出相应的函数（也称作过程），再设计出用于保存姓名、性别、年龄、年级和各课程成绩的数据结构，利用函数修改、增加、删除和保存这些数据结构中的数据，最后输出得到的结果。

显然，这种编程方法的工作重点在于问题的分析、分解和解决，即函数的设计与实现，这种方法称为面向过程的程序设计方法。

面向过程的程序设计方法增加了程序开发和维护的成本。例如，当需要奖励单科成绩列前 5 名的学生，或者增加学生的资料（身高、体重、特长等），必须修改原有的程序代码，重新调试、编译并链接，所有的这些工作都是单调枯燥的，而且容易出现错误。出现以上问题的原因在于，面向过程的程序设计方法将思考的重点放在函数上，而彻底忽略了数据，函数和数据被当做没有任何共同点的、相互独立的实体，因此函数就难以理解和适应数据结构的变动。

当进一步研究这个问题的时候，就会发现数据结构中的信息（如学生的资料），对我们来说更加重要，函数只是修改、增加、删除和保存数据的工具而已。

面向对象的程序设计方法则将关注的焦点转换到了数据上，它关心数据和操作的结合，从实际看到、听到、触摸到的人或事出发，将这些客观事物与程序中的对象建立一一对应的映射。因此，面向对象的程序由众多的对象组成，每一个对象是算法和数据的结合，是封闭的、具有一定功能的实体。对象之间通过某种相互操作来传递消息，在消息的作用下完成指

定的功能。

面向对象程序设计方法有三大特点：封装、继承和多态。

封装，就是将用来描述客观事物的一组数据和操作组合在一起，构成一个类。也就是说，类是对客观事物的一种高度抽象，是一组具有相同属性和操作对象的集合。例如，不论是张三还是李四，如果他们都拥有姓名、性别、年龄、年级和各课程成绩等相同的属性，都具有注册、选课、考试等相同的行为。那么可以从张三和李四的身上抽象出一个类：学生类。

关于封装的具体内容详见第 6 章。

类与类之间还可能存在着一种特殊的关系，这就是继承。当一个新类继承了原有类的所有属性和操作，那么这个新类就被称作派生类（或子类），原有类就是新类的基类。派生类和基类之间存在着继承关系。程序员可以通过继承关系建立类的层次结构，并在这个层次结构中建立解决问题的模型。

关于继承的具体内容详见第 7 章。

所谓多态性，是指一个函数具有多个实现。在 C++中，多态性体现为虚函数、函数重载和运算符重载。关于多态的具体内容详见第 9 章。

1.1.3 C++语言的优势

C++语言是对 C 语言的扩充，它除了具备 C 语言的各种优点之外，还支持封装、继承和在类之间进行多态的消息传递，是真正的面向对象的程序设计语言。尽管提及 C++，就会联想到面向对象，但它同时也支持面向过程的程序设计，是一种支持多种程序设计方法的通用语言。

C++之父 Bjarne Stroustrup 博士对 C++的基本定义是：一种经过改进的更为优化的 C，支持面向对象的程序设计，支持泛型程序设计。

简而言之，C++语言有以下四个方面的优点。

- 它允许程序员采用近似自然语言的方式来表达需要解决的问题，降低了程序开发和维护的成本。
- C++不仅有着与 C 相同的控制能力，而且具有和 C 语言几乎一致的运行效率，采用面向对象方法编写的程序可能比用 C 语言编写的程序更加有效率。
- C++允许程序员更加自由地使用各种库，由于程序各部分的名字是相互独立的，所以程序员想用多少库就可以使用多少库，不会再有名字冲突问题。
- C++的异常处理机制能够保证在运行期间检查到错误，并转至相应的处理程序，减少了代码的长度和复杂度。

1.1.4 C++程序的基本结构

下面，通过一个简单的程序，来说明 C++程序的典型结构。

【例 1-1】 从键盘上输入三个整数，然后输出它们的最大值。

```
/* 这是我们的第一个 C++程序
*  输入任意三个整数，输出其中的最大值
```

```
*/
#include "stdafx.h"
#include <iostream>                          // ④
using namespace std;                         // ⑥
int max (int a,int b,int c)                  // ⑤
{   int temp;
    temp = a;                                // ②
    if (temp < b) temp=b;
    if (temp < c) temp=c;
    return temp;
}
int main()
{   int a, b, c;                             // ①
    cin >> a >> b >> c;                      // ③
    cout << "max = " << max (a, b, c) << endl;   // ③
}
```

C++程序的典型结构通常包括四个部分：注释区、声明区、函数定义区和主程序区，如图 1-3 所示。

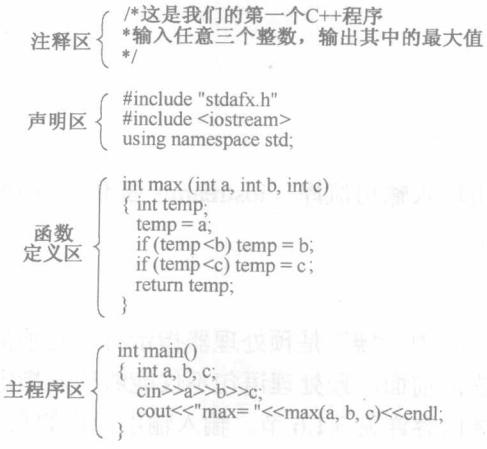


图 1-3　C++程序的典型结构

其中，注释区包括程序的作者、日期、版本和功能等内容，尽管它是可选项，仍然建议大家在注释区内对该程序的相关情况做一个详尽的描述，便于程序的管理和维护。

声明区包括预处理语句、类定义、函数声明、全局变量和条件编译等内容。

函数定义区包括所有函数的具体定义，每个函数由函数头和函数体组成，表示一个或一组算法，关于函数详见第 5 章。

主程序区由主函数 main()组成，程序从 main()函数体的第一条语句开始执行。在本例中，main()主函数的返回类型是 int 型，表明该函数在结束时必须通过 return 语句返回一个整数数值。return 是 C++的语句，能够终止函数的执行并返回一个值，这个值被称作函数的返回值。在本例中，返回值为 0 表示程序成功执行完毕。

下面介绍 main()主函数中的其他语句。

语句①是一个变量声明语句：

```
int a, b, c;
```

其中 int 是整型，a, b 和 c 是变量，用于存储整型数据。变量、函数和对象在使用前都必须事先进行定义或声明。变量的具体内容详见 2.1.3 节。

语句②是一个赋值语句：

```
temp = a;
```

作用是把变量 a 的值拷贝给变量 temp。

语句③是输入输出语句：

```
cin >> a >> b >> c;
cout << "max = " << max (a, b, c) << endl;
```

其中“>>”是输入运算符，作用是从标准的输入对象 cin 中读取数据，“<<”是输出运算符，作用是向标准的输出对象 cout 输出数据。连续出现的输入输出操作可以串联在一起，如果一行写不下，可以分成多行来写。endl 表示输出一个换行符并刷新缓冲区，例如语句：

```
cout << "Hello, world! " << endl << " I come from China! ";
```

的输出结果是：

```
Hello, world!
I come from China!
```

C++的输入输出功能由输入输出流库（iostream）提供，在使用前必须用预处理语句引入头文件 iostream，如语句④：

```
#include <iostream>
```

该语句是预处理语句，其中，“#”是预处理器指示符，必须位于它所在的程序行的最前面，不得有其他字符位于它的前面。预处理语句不仅能够读入指定文件的内容，而且能够定义宏命令和条件编译，具体内容详见 3.1.6 节。输入输出操作的具体内容详见第 10 章。

语句⑤定义了一个函数 max()，它由函数头和函数体组成，有三个整型参数，并返回一个整型数值，具体内容详见第 5 章。

最后，在 C++标准库中定义的对象，如 cout，cin 等，都是在一个名为 std 的名字空间域中声明的。因此，在程序中必须使用 using 指示符，引入标准名字空间域 std，例如语句⑥：

```
using namespace std;
```

名字空间域的具体内容详见 5.1.9 节。

运行该程序，从键盘上输入三个用逗号或空格符分隔的整数，按 Enter 键，屏幕上就会显示出三个数中的最大值。假如输入的三个数为 34，56 和 23，则输出结果如下。

```
34, 56, 23
max=56
```

1.1.5　C++语言的基本词法

1．C++的字符集

C++语言的基本字符集包括 96 个字符，其中 5 个为特殊字符：空格符、垂直表符、水平表符、进纸符和换行符（\n），其他的 91 个为图形化字符，它们被分成三类：

（1）大小写英文字母（52 个）：a～z 和 A～Z。

（2）数字字符（10 个）：0，1，2，3，4，5，6，7，8，9。

（3）其他字符（29 个）：_，{，}，[，]，#，(，)，<，>，%，:，;，.，?，*，+，–，/，^，&，|，～，！，=，,，\，"，'。

2．C++的标识符

标识符是任意长度的字母和数字的组合，用于命名程序中的一些实体，例如变量名、常量名、函数名、类名等。C++语言规定，标识符由下列字符组成。

（1）大小写英文字母（52 个）：a～z 和 A～Z。

（2）数字字符（10 个）：0，1，2，3，4，5，6，7，8，9。

（3）下划线字符：_。

需要注意以下几点。

（1）标识符不能以数字开头，并且对英文字母的大小写敏感。例如：

```
int 5b;                    // 错误！标识符不能以数字开头
int max, Max;              // 正确！但 max 和 Max 是两个完全不同的标识符
```

（2）标识符中只能采用下划线“_”，而不能采用中划线“-”。

（3）虽然 C++标准并没有对标识符的长度进行限制，但是某些编译系统只能识别有限长度的标识符。为了便于程序的阅读和维护，建议尽量采用有意义的单词作为标识符。

（4）某些标识符已经被 C++语言保留，成为关键字，程序员不得另行定义。

3．C++的关键字

所谓关键字，就是指那些已经被 C++语言预定义了特定含义的标识符，可以在程序中被直接使用而无需另行定义。例如：int、float、if、new 等都是常见关键字。

如果程序员定义了和关键字相同的标识符，会导致编译错误。例如：

```
int new;                   // 错误！new 是关键字
```

C++标准中的关键字如表 1-1 所示。

表 1-1　C++的关键字

asm	do	if	return	typedef
auto	double	inline	short	typeid
bool	dynamic_cast	int	signed	typename
break	else	long	sizeof	union
case	enum	mutable	static	unsigned
catch	explicit	namespace	static_cast	using