最流行软件丛书

谭浩强 主编

C++语言及程序设计

郭 永 张吉生 王伦津 朱继生 编

丛书总序

电子计算机正以空前的速度发展，微型计算机更是其中的佼佼者，它几乎已深入到社会生活的一切领域。随着微型机的普及应用，众多的软件应运而生，其中有些软件因其功能丰富、实用性强、普及性好而流行于世。要使微型机发挥更大的作用，就必须掌握和熟悉这些软件的使用方法和技巧。为了适应广大初、中级计算机使用者的迫切需要，我们经过反复研究，特组织编写这套《最流行软件》丛书。我们企望尽此绵薄之力推动计算机在我国进一步普及应用。

本丛书采取“一种软件一本书”的模式，分别介绍国内广泛流行和经常使用的软件，力图突出其实用性强、普及面广、内容新颖、品种配套、概念清晰、通俗易懂等特点。

本丛书不同于计算机厂商销售的“使用手册”，也不同于一般教材。现在市面上有些译自国外资料的使用手册，虽然内容详实，但往往由于各种原因而难以阅读和理解，不适合于初、中级计算机使用者学习。考虑到多数读者的实际情况，我们采用循序渐进，深入浅出的编写方式，力求使那些从未接触过该软件的读者也可以做到“学了就能用，用了就见效”。限于篇幅不宜过大，每本书仅介绍该软件最基本、最常用功能的使用方法和技巧，不拟囊括其全部细节，也不列举较大规模的例题。一般也不详细介绍基本原理和名词概念，而以教会如何使用为目的。读者在掌握基本使用方法以后，可以通过实践更深入更巧妙地去使用有关软件。

考虑到国内微型机配置的现实情况，本丛书以 IBM PC 机及与其兼容的长城系列微型机上广泛使用的软件为主，兼顾其他。鉴于软件版本翻新很快，拟以当前广泛流行的版本为基础，并根据发展，不断更新。

本丛书的选题是根据我国软件应用发展状况和广大读者急需来确定的，特约高等院校和科研、设计单位有丰富实践经验的专家参加编撰，拟陆续分期分批奉献于世。“问渠哪得清如许，唯有源头活水来”。我们热切希望专家和读者能及时向我们提供有关信息，以使本丛书在选题、编撰、出版、发行等环节更具针对性和实时性。

本丛书无论在选题策划还是在编写细节上都可能会有不足甚至错误之处，恳切希望大家批评指正。谢谢！

丛书主编

前　言

C++是当今世界上最流行的面向对象程序设计语言。它在C语言的基础上进行了改进和扩充，继承了C语言效率高，可移植性好、功能强和代码精炼等优点，克服了C语言的某些不足，重要的是增加了面向对象程序设计的功能，更适合于编制大型复杂的软件系统。

C++虽然起步较晚，但发展速度很快，世界上许多计算机公司都已推出自己的C++，其中最流行的是BorLand C++。现在，C++的标准化工作正在进行，预计会成为90年代最流行的程序设计语言。

为满足高中以上的计算机应用人员学习C++的需要，在内容选材方面努力体现科学性、实用性、普及性、系统性和先进性等特点，以IBM PC机和DOS系统为基础，主要参考BorLand C++。编写时屏弃繁琐难懂的理论介绍和各种版本的差异，力图深入浅出、简明精炼地介绍C++的基本概念、规则和程序设计方法。书中所选例题简单易懂，并在80386微机和DOS系统下，用BorLand C++调试通过。

全书共分十章。第一章介绍程序设计的基本概念、C++发展概况、程序结构及基本编程过程。第二章到第五章介绍C++的基础知识，包括程序的基本成分、语句和标准数据类型等。第六章和第七章介绍面向对象程序设计的基本概念和方法，如类、对象、继承和多态性。第八章到第十章介绍C++的输入输出、预处理、集成开发环境、内存管理和视频函数。

本书第一章和第五章由王伦津编写，第二、三、四章由张吉生编写，第六、七、八章由郭永编写，第九、十章由朱继生编写。全书由郭永负责统编，王伦津修改，朱继生组织协调，刘进峰调试例题和编写附录。

本书全稿由谭浩强教授审校，并提出许多宝贵意见，表示衷心感谢。

由于水平所限，书中难免有错误或疏漏之处，敬请各位读者批评指正。

编　者

内 容 简 介

《最流行软件》丛书由著名计算机教育专家谭浩强教授主编。本丛书采取“一种软件一本书”的模式，以教会如何使用为目的，分别介绍国内广泛流行和经常使用的软件，具有内容新颖全面、实用性强、品种配套、概念清晰、通俗易懂等特点。

本书是该丛书之一，是C++的普及型读物，介绍当前最流行的面向对象程序设计语言——C++的基本概念、规则和程序设计方法。全书共分十章，分别介绍了程序设计的基本概念，程序的基本成分、语句和标准数据类型，面向对象的程序设计的基本概念和方法，Borland C++的集成环境、输入输出、内存管理和视频函数。

本书的主要读者对象为具有高中以上文化程度的初、中级计算机使用者，也可作为需要、开拓计算机应用面的大、中专师生和科技工作者的自学读物。

目　录

第一章 引 论

自 1946 年研制成第一台计算机至今，计算机技术发展非常迅速。以计算机语言为例，由机器语言、汇编语言发展到高级语言。高级语言的发展又分为面向过程的、结构化的和面向对象的程序设计语言几个阶段。

面向过程的高级语言是最早使用的高级语言，如 FORTRAN、COBOL 和 BASIC 等。其特点是以解决问题的过程为中心，把大的问题划分成许多子模块，解决问题时先确定数据结构和算法，再编制程序。这类语言的缺点是程序结构不够清晰。

结构化程序设计语言继承面向过程语言的模块化等优点，对程序结构作了改进，使程序由顺序、选择和循环三种基本结构组成。对于复杂问题，采用模块化、自顶向下逐步求精的设计原则。因此程序结构清楚、易读，容易维护。典型的结构化语言有 C 和 PASCAL 等。著名的 UNIX 和 Windows 操作系统就是用 C 语言编制的。

随着计算机技术的发展，需要解决的问题越来越复杂。有些大型问题的解决要用很多人、很多时间编程，系统规模很大，各模块间的联系、系统维护、正确性测试都很困难，出现了软件危机。这些问题是结构化语言无法解决的，在这种背景下产生了面向对象的程序设计语言 OOPL(Object-Oriented Programming Language)。

OOPL 继承结构化程序设计语言的精华，克服了数据与代码分离的弱点，它解决问题的过程更符合人的思维方式，更适合于解决大型复杂的问题。程序可靠性高，容易维护和理解，C++就是这种语言的典型代表。

1.1 C++ 概 述

1.1.1 C++的产生和发展

一、C++的发展简史

面向对象的基本思想于 1967 年在 simula 语言中正式出现。80 年代后，出现了 smalltalk 语言，这是一个纯粹的面向对象的程序设计语言，它提出了 OOPL 中的主要概念。由于多种原因，它没有被广泛使用。

C 语言产生于 1972 年，最早用来描写 UNIX 系统，经过完善修改，发展成为一种通用的计算机语言。1983 年进行标准化(ANSIC)，发展更为迅速。现在不仅适用于 UNIX 系统，而且在其它操作系统下，如 DOS 和 Windows 环境中都可使用。使用的机型由原来的小型机发展到大、中、小和微型机。

1985 年，AT&T 贝尔实验室在 C 语言的基础上，吸收了 OOPL 的特点，形成了面向对象的程序设计语言 C++。C++公开发表后，发展更加迅速，很多公司都研制了自己的

C++，版本不断升级。目前，主要的 C++ 有 AT&T C++，Zortech C++、Borland C++ 和 Visual C++ 等。其中，Borland C++ 发展最快，使用广泛。目前，我国使用的大部分是 Borland C++。Borland C++ 是在 Turbo C 的基础上发展起来的，主要版本有 Borland C++ V1.0(也称 Turbo C++)、V.20 和 V3.0 等。Borland C++ 与 C 兼容，它不仅功能完善，运行速度快，而且提供了集成开发环境(IDE)，为用户研制 C++ 软件提供一个有力的工具，很受欢迎。现在 C++ 仍迅速发展，标准化工作正在进行，是一种很有前途的计算机语言。

二、C++ 继承和发展了 C 语言

C++ 是在 C 语言的基础上发展起来的，与 C 兼容，克服了 C 的一些不足和弱点，增加了面向对象的功能。它既是结构化的语言，又是面向对象的语言。

C++ 继承了 C 的大部分语法规则，使 C 成为它的子集。例如，它保留了 C 语言的全部关键字、数据类型、语句和函数等。用 C 编写的源程序不用修改或改动很少，在 C++ 中即可编译运行。

C++ 克服 C 的某些不足，增加了一些关键字和库函数，是一个更好的 C，这体现在如下几方面：

(1)增加了单行注释；

(2)使用 new 和 delete 运算符管理自由存储区更加方便可靠；

(3)提倡用 const 说明的符号常量和嵌入式(inline)函数代替原来的宏，克服宏的副作用；

(4)在说明函数原型(proto type)和定义函数时，参数表中允许有类型说明，使程序在编译时检查函数参数和返回值的类型错误；

(5)增加了引用(reference)数据类型，用它作函数参数和返回值，使程序更清楚；

(6)使用缺省参数，简化程序代码，使函数调用更加方便；

(7)最主要的是增加了类功能，使 C++ 成为面向对象的程序设计语言，这是 C 没有的。

1.1.2 面向对象的程序设计语言

一、OOPL 的基本概念

1. 对象(object)

对象是 OOPL 中最基本、最重要的概念，是程序设计的中心。它有如下特点：

(1)对象是人们要进行研究的任何事物

从最简单的整数到极其复杂的航天飞机、自动化工厂等都可看作对象。对象不仅表示具体的物体，也表示看不见摸不着的事件、规划、规则等。因此对象有极强的描述与表达能力，能做到现实世界中的事物与对象的自然和谐的直接对应。

(2)对象实现了数据与操作的一体化

对象具有状态，一般用数据来描述；对象还有操作，用来改变对象的状态。OOPL 把状态和操作封装于对象体之中，并提供一种访问机制，使对象的“私有数据”仅能由这个对象的操作来执行。这样能灵活而有针对性地描述具体对象的独特行为，实现对复杂对象的直接、准确的模拟。对象的封装技术使对象具有较强的独立性，其内部状态很少受外界影响，

因此可以把对象作为模块化的单位。

2. 对象类(class)

对象类简称类,它将具有相同状态、操作和访问机制的多个对象组合在一起,这组对象就称为对象类。

在定义了类以后,属于这种类的一个对象叫作类实例或类对象。

例如人类由成千成万的具体的人组成。人就是一个对象类,一个具体的人,如张三就是一个类对象。

一个类的定义应包括类名、类的说明和类的实现。类说明也称为类的外部特性或外部接口,它为用户所见,即让用户了解类是什么和能做什么。类实现是类的内部表示,一般由系统开发人员编程实现。用户不必去了解类的实现细节,便于集中精力开发好应用系统。

类也是一种用户定义的数据类型。OOPL 除系统定义的标准数据类型之外,还允许根据需要,定义复杂的数据类型。

3. 类的封装(encapsulation)

封装是把一类对象的状态(用数据表示)和操作(用函数来表示)封闭起来,装在类对象体之中,形成一个能动的实体。

例如,一块手表有许多零件,将这些零件装在表体之内,用户只需用它查看时间,不需了解其内部构造。手表内部的零件装配由制造厂家进行。这种封装技术有如下好处:

(1)对用户来讲,隐藏了表内各零件的制作和装配的复杂细节;

(2)装在表内的零件不容易受外界的破坏;

(3)使用方便,任何人都可以用它查看时间,不需要了解各零件的制造及装配过程。

OOPL 的封装机制模仿现实生活中的装配技术,它把一类对象的数据和函数封闭起来,并提供它们的访问机制。一般把类对象的数据说明为私有的,这样能隐藏类内这些数据的说明和实现的细节,外界不能直接访问这些数据,而把这种对象的函数说明为公有的,它们可使用对象的私有数据,也是类对象与外界交换信息的接口,即外界只能调用类对象的公有成员函数才能和类交换信息。这样可以隐藏类对象内部实现的复杂细节,有效保护其内部私有数据不受外界的破坏,也为外界使用类对象提供一个方便、简洁的接口。

4. 类的继承(inheritance)

类的继承是指新的类继承原有类的全部数据、函数和访问机制,并可以增添新的数据、函数和访问特性。这样产生的新类叫子类或派生类,原来的类叫父类或基类,这种产生新类的方法叫类的派生,也称为类的继承。

一个子类还可以派生新的子类,这种派生过程进行多次,形成不同层次的类,最低层的子类具有它上面各层父类的全部数据、函数和访问特性,这叫类的纵向继承或单向继承。

一个类可从多个父类中继承,这叫类的多重继承。

例如,人可分为工人、农民、干部、学生和教师等,他们都继承了人的特性,并且还可以细分。学生又可分为小学生、中学生、大学生和研究生;教师又可分为助教、讲师、教授和研究生。这样形成三个层次不同的类,其中研究生既是学生,又是教师,他继承了教师和学生的特性,属于多重继承(见图 1-1)。

类继承的作用有以下 3 点。

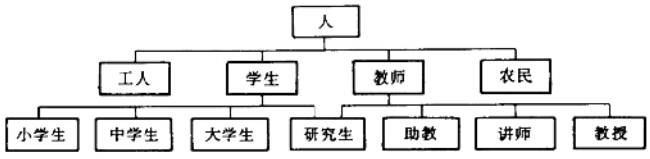

图 1-1　类层次与多重继承

(1)增强了类的共享机制

在类层次结构中，一个类的数据、函数和访问规划不仅为本类的全部类实例共享，而且也可由它的子类、子类的子类、……直至最低层的子类的全部对象所共享。类的共享范围的扩大能显著地减少创建新类与对象的工作量，有效地节约了存储空间。

(2)实现软件的可重用性，提高软件的可靠性

有了类的继承机制，在基类中只定义一次各层次类中共同需要的数据、函数和访问规则。类派生时，只增加与基类不同的数据、函数和访问规则，这样用户可充分利用已有的类，提高了原有类的可重用性。

(3)简化系统开发工作，使系统易于扩充，具有良好的开放性

OOPL 具有许多功能很强、由系统生成的类库。进行系统开发时应充分利用系统提供的标准类库，全力研制新系统专用的新子类。按照这种方法，新系统的开发工作就简单，能显著加速开发进程。

5. 类的多态性(Polymorphism)

所谓多态是指一个名字有多种功能或相同的界面有多种实现方法。

OOPL 的多态性是通过运算符重载、函数重载及虚函数来实现的。

(1)函数重载

函数重载是指一个函数名，可有不同数量的参数、参数类型和返回值类型。

例如，可以定义立方函数 cube，它有整型、浮点型和双精度型三种类型的参数，它们使用同一个函数名，但系统能根据其参数类型，调用相应类型的立方函数，并返回相应类型的函数值。如：

```
int cube(int i);//整数立方
float cube(float a);//浮点数立方
double cube(double d);//双精度数立方
```

当调用 cube(5)时，因参数是整数，系统自动调用第一个立方函数。

(2)运算符重载

运算符重载是指同一个运算符，可以作用于不同类型的数据。

例如运算符加＋，不仅可用于整数，还可用于浮点数和双精度数。在 OOPL 中，经过重新定义(重载)，它不仅可对系统已定义的各类标准数据相加，还可以对用户定义的数据类型相加。例如，可以对复数、字符串相加。

(3)虚函数与动态联编

用关键字 virtual 说明的类的成员函数叫虚函数。

在不同的类层次中，虚函数不仅名称相同，而且函数返回类型、参数个数、类型、顺序

也完全相同，但函数体不同。这在编译时是无法区别的，只有在运行时由系统动态寻找所需的函数体进行匹配，这叫动态联编(Dynamic binding)或迟后联编。

重载函数的调用在编译时就能根据参数确定相应的函数体，这叫早期匹配；而虚函数在运行时才确定匹配的函数体。

二、OOPL 的目标和方法

OOPL 的基本想法是把要解决的问题表示为类对象的集合，它的目标是克服软件的复杂性，并用计算机很自然地表示各种各样的对象。

1. 分类和归纳方法

人们认识各种事物的过程有两种。一种是从一般到特殊。以学校为例，人们对学校最初的认识，是一个学习知识的地方；在对学校分类的过程中，进一步知道学校有大学、中学和小学之分；再进一步又知道大学可分为综合性大学，理、工、农、医师范等大学，每一科又分不同的专业。分类到一定程度，就对各类对象进行状态描述和功能定义，以明确每一类对象的工作及建立各对象之间的联系。

另一种是从特殊到一般。例如，我们今天接触的是长城计算机，明天看到的是一台 IBM 中型计算机，后天看到的是银河大型计算机。这些计算机除内外存大小不相同、结构不同外，它们的共同特点是可用来进行数据处理等。

OOPL 提供了从一般到特殊的分类方法(继承)，也提供了从特殊到一般的归纳方法(类)。

2. 软件集成方法

软件要提高生产率，应走类似硬件的道路，即应有软件的集成模块，把已经成熟的正确的软件形成独立的实体，使它们不加改动或改动很少就可用到新的软件系统中。

OOPL 中的类能起到集成模块的作用。

1.1.3 C++程序示例

C++是一种面向对象的程序设计语言，下面用 2 个例子说明 C++的基本概念。

例 1-1 计算圆的面积。

```
//ex1-1.cpp
#include<iostream.h>
const float pi=3.14;
float area(float);
int main(void)
{
  float r,a;
  cout<<"Enter the radius of a circle:";
  cin>>r;
  a=area(r);
  cout<<"area of a circle is:"<<a;
  return 0;
```

```
}

//area function
float area(float r)
{
  return(pi * r * r);
}
```

程序运行时显示：

Enter the radius of a circle：

这时从键盘输入 5 和回车键，程序又显示

area of a circle is：78.5

程序第 1 行和第 14 行是注释行，程序执行时将忽略它。

第 2 行是预处理包含指令，其中的 iostream.h 是 C++ 提供的标准类库，它提供了输入输出流 cin 和 cout 及输入输出运算符>>和<<的定义。

第 3 行说明了一个浮点型符号常量 pi。

第 4 行是对函数 area 的引用说明。

第 5～13 行定义了主函数 main()，main 是函数名，它是程序执行的起点。

第 6 行和第 13 行是一对花括号，表示函数体的开始和结束。

第 7 行说明了程序中使用的二个浮点型变量。

第 8、11 行是 2 个输出语句，其中的 cout<<表示从屏幕输出用双引号括住的字符串和变量 a 的值。

第 9 行是一个输入语句，表示从键盘输入一个数给变量 r。

第 10 行是赋值语句，右边是函数调用，它调用 area 函数计算半径为 5 的圆的面积。

第 15～18 行是 area 函数的定义，它计算一个半径为 r 的圆的面积。

需要强调的是，运算符<<和>>原来的意思是表示算术左移和右移，在 iostream.h 文件中被重新定义(叫运算符重载)作为输出输入运算符使用，它们和 cin、cout 流类结合使用，分别表示从键盘输入和从屏幕输出。本书后面的很多例题将用它们进行输入输出。

例 1-2 类的应用。

```
//ex1-2.cpp
#include<iostream.h>
class Point
{
  int x
  int y;
 public:
  Point(int initx,int inity)
  {
    x=initx;
    y=inity;
```

```
    }
    int getx()
        {return x;}
    int gety()
        {return y;}
};

int main(void)
{
    Point myPoint(50,100);
    cout<<"x:"<<myPoint.getx()<<"\n";
    cout<<"y:"<<myPoint.gety()<<"\n";
    return 0;
}
```

程序输出：

x:50

y:100

例 1-2 中 main 函数前面定义了一个名为 Point 的类，它有两个私有数据成员 x 和 y。在关键字 public 之后的成员函数是公有的，其中的 Point 函数与类同名，它是类的构造函数。Point 类将私有数据成员和公有成员函数封装于类体之中，它的私有数据成员 x 和 y 只能被该类的成员函数 Point、getx 和 gety 使用，在 main 函数中不能直接使用 x 和 y，只能调用类的公有成员函数来间接访问它。

在 main 函数中，定义了一个 Point 类对象 myPoint，它和 Point 类具有相同的类成员和访问规则。在定义这个类对象时，系统隐含地调用它的构造函数进行初始化，使成员 x 为 50，y 为 100。在输出语句中，调用类对象的公有成员函数 getx 和 gety，返回类对象的私有数据成员 x 和 y 的值，并在屏幕上显示。

这个例题涉及 C++中 OOPL 的一些基本概念，这里仅作简单介绍，后面还要详细讨论。

1.1.4 C++程序结构

C++源程序以文件为单位存放，每个程序由一个或多个文件组成。

C++程序一般由 4 部分组成：预处理指令、全局说明、main 函数和用户定义的函数。图 1-2 给出 C++程序的基本结构。

一、预处理指令

预处理指令以＃号开始。用得最多的是预处理包含指令：

＃include

它包含 C++程序使用的另一个源程序文件。

被包含文件一般为系统提供的头文件或用户编写的另一个 C++源文件。系统提供的

头文件一般包含标准函数的原型、宏定义、全局量说明和标准类库。例如，例 1-2 中的 iostream.h 头文件就是系统提供的标准输入输出类库文件。有时也可包含用户提供的另一个 C++ 源文件。

预处理指令将这些被包含文件嵌入到包含文件中一起编译。

二、全局说明

全局说明一般包括程序使用的全局量，如类定义、全局变量、用户函数的引用说明等。例如，在例 1-1 中说明的全局符号常量 pi、area 函数的引用说明，在例 1-2 中 Point 类的定义都是全局说明。

三、main 函数

main 函数也叫主函数。程序从 main 函数开始执行，也终止于 main 函数。一个程序必须有一个而且只能有一个 main 函数。

main 函数的函数体如果有带表达式的返回语句，其类型应说明为 int，返回值为 0 表示正常结束，其它值表示异常返回。如果 main 函数无返回语句或返回语句无表达式，应说明其类型为 void。

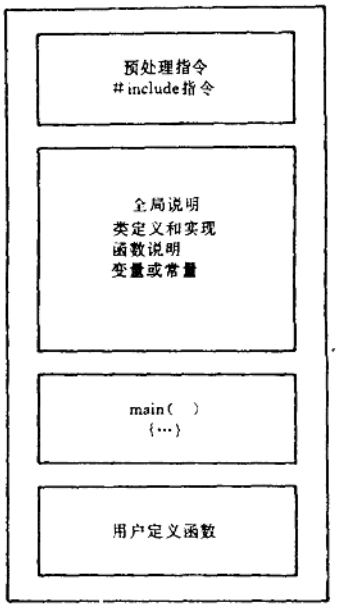

图 1-2 C++ 程序的基本结构

四、用户定义的函数

用户定义的函数是用户根据需要按函数规则编写的函数。一般在 main 函数或其它函数中要调用这个函数，在调用前应对其作引用说明。如例 1-1 中，在 main 函数后定义 area 函数，主函数中调用它，调用前，在主函数外对它作了引用说明。

1.1.5 C++ 的特点

C++ 的特点表现在以下 3 个方面。

一、C++ 是一个更好的 C

(1)运算符和数据类型丰富，语言简洁、紧凑、处理功能强，使用方便灵活。

C++ 保留了 C 的全部运算符，增加了 new、delete 和 :: 运算符，它把括号、赋值号和强制类型转换都作为运算符，使 C++ 语言处理功能比其它语言强，且程序代码简洁、紧凑。

C++ 继承了 C 的全部数据类型，增加了引用类型，扩充了结构体和共用体的概念，这二者和新加的 class 三者构成了用户定义的类，使一些复杂数据结构(如链表、树等)的处理比较容易。

C++ 保留了 C 的按位运算、指针和位段，可以替代汇编语言编写操作系统、大型软件和硬件控制程序。

(2)代码质量可靠、使用面广、可移植性好。C++ 保持 C 高质量代码的特点，程序运行效率高，仅低于汇编语言。可在各种机型和多种操作系统下运行，程序可移植性好。