

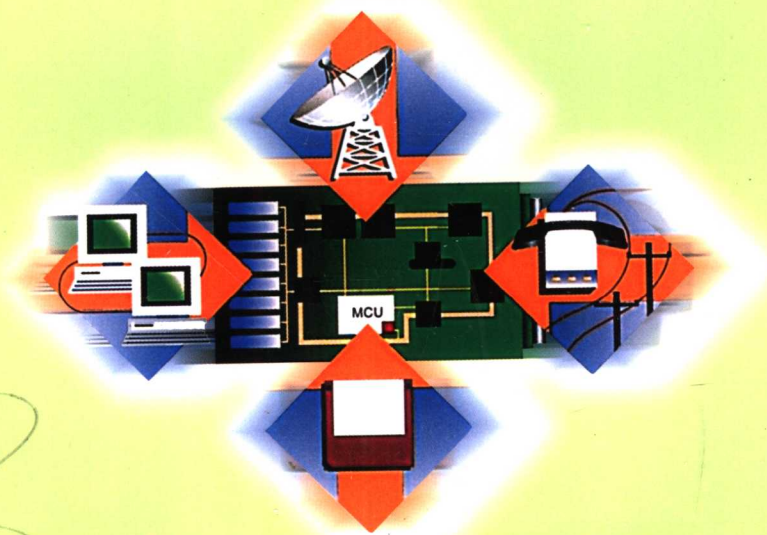


普通高等教育“十一五”规划教材

C++程序设计

主 编 刘於勋 张雪萍

副主编 许 速 马 丁 白 浩



科学出版社

www.sciencep.com

TP312/2542

2007

普通高等教育“十一五”规划教材

C++程序设计

主 编 刘於勋 张雪萍

副主编 许 速 马 丁 白 浩

科学出版社

北 京

内 容 简 介

本书全面、系统地介绍了面向对象程序设计的基本概念和程序设计方法,通过实例详尽分析了 C++ 语言中面向对象的基本特性,包括类、对象、派生类、继承、多态性、虚函数、类模板、异常处理、I/O 流等。全书采用基本概念与实际应用相结合的方式组织教材内容,使读者能够在深刻理解和全面领会面向对象程序设计特点和风格的基础上,切实掌握面向对象程序设计的方法。

本书通俗易懂、结构合理、叙述简洁,每章均有小结,可作为高等院校计算机及其相关专业的学生学习面向对象程序设计的教材,也可以供自学 C++ 的读者参考。

图书在版编目(CIP)数据

C++ 程序设计/刘於勋,张雪萍主编. —北京:科学出版社,2007
普通高等教育“十一五”规划教材
ISBN 978-7-03-019684-2

I. C… II. ①刘… ②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(2007)第 128121 号

责任编辑:马长芳 潘继敏 / 责任校对:鲁 素

责任印制:张克忠 / 封面设计:陈 敬

科 学 出 版 社 出 版

北京东黄城根北街 16 号

邮政编码:100717

<http://www.sciencep.com>

北京市文林印务有限公司印刷

科学出版社发行 各地新华书店经销

*

2007 年 8 月第 一 版 开本:B5(720×1000)

2007 年 8 月第一次印刷 印张:20 3/4

印数:1—4 000 字数:394 000

定价:29.00 元

(如有印装质量问题,我社负责调换〈文林〉)

前 言

本书是专门为高等院校计算机及其相关专业开设的“面向对象程序设计”课程编写的。

C++语言从C语言进化而来,是C语言的超集,C++语言和C语言都是用函数驱动机制实现功能调用,不过C语言是面向过程的程序设计语言,而C++语言容纳多种编程模式,包括面向对象编程、通用编程和传统的过程化编程。本书侧重介绍C++面向对象程序设计基本原理和方法。

本书适合学过C语言的读者使用。因为在C++语言中,数据类型、常量、变量、表达式、标准函数、三种基本程序结构、函数的定义和调用等基本概念都与C语言相类似,故本书没有专门对这些基本概念进行介绍,而是着重介绍C++面向对象的突出特性,以使读者树立面向对象思想,学会使用类编程,领会面向对象程序设计的优势,进而掌握面向对象程序设计的各种方法。

本书与其他教材不同之处在于:

(1) 组织结构新颖。本书第1章先介绍什么是软件、程序设计语言、程序设计方法的演变,后引入C++面向对象程序设计概念,使读者能够首先对计算机程序设计有比较系统、全面的了解和认识,并对C++面向对象程序设计产生兴趣。如果从一开始就直接介绍枯燥无味的C++的语法和规则,会使读者陷入一种痛苦、无味的学习境况。

(2) 程序设计风格规范。书中按照ANSI/ISO C++的规范要求书写程序,使读者通过阅读程序,能够养成编写程序的良好习惯,这对于今后编写复杂的程序非常重要。如果偏重描述复杂算法,对程序书写的规范性强调不够,前后举例书写不统一,会使读者模仿后养成不严谨、随意等不良习惯。

(3) 突出面向对象思想。全书紧紧围绕与面向对象相关的理论知识由浅入深地展开,淡化面向过程程序设计思想,突出C++中类、对象、派生类、继承、多态性、虚函数、模板、异常处理、I/O流概念。如果既介绍面向过程程序设计思想,又叙述对象程序设计思想,往往会使读者很困惑,不知应该重点掌握哪种程序设计思想。

(4) 举例分析通俗易懂。书中所举例子是应用面向对象程序设计方面对读者最容易理解的事例进行分析,便于读者理解、记忆和掌握。

本书由刘於勋、张雪萍、许速、马丁、白浩、易虹、徐振强共同编写。刘於勋、张雪萍为主编,许速、马丁、白浩为副主编,易虹、徐振强参编。最后由刘

於勋统稿。

本书在编写过程中得到了兄弟院校和同行们的支持和关心，他们对本书的编写提出了许多宝贵意见和建议，在此对审定本书编写大纲的专家们及在全书编写过程中给予帮助的同事们表示感谢。

由于编者水平有限，加之时间仓促，书中难免有不妥之处，敬请广大读者多提意见。

编 者

2007 年 6 月

目 录

前言

第 1 章 面向对象程序设计与 C++ 概述	1
1.1 软件	1
1.2 程序设计语言	2
1.3 程序设计方法的演变	4
1.3.1 结构化程序设计	4
1.3.2 面向对象程序设计方法	5
1.4 C++ 程序设计语言	7
1.5 简单的 C++ 程序	8
1.5.1 第一个 C++ 程序	8
1.5.2 第二个 C++ 程序	11
1.5.3 第三个程序	14
本章小结	16
第 2 章 从 C 到 C++	17
2.1 C++ 关键字	17
2.2 const 常量	17
2.2.1 问题的引出	18
2.2.2 const 与 C 语言中 #define 的区别	19
2.2.3 const 定义	19
2.3 函数	20
2.3.1 内联函数 (inline 函数)	20
2.3.2 重载函数	24
2.3.3 函数参数的默认值	27
2.3.4 函数模板	30
2.4 指针	38
2.4.1 存储数据的两种策略	38
2.4.2 声明和初始化指针	41
2.4.3 指针与动态内存分配	43

2.4.4	const 与指针	46
2.5	引用	48
2.5.1	引用的概念	49
2.5.2	引用的操作	50
2.6	名称空间	54
2.6.1	名称空间的意义	54
2.6.2	名称空间定义	55
2.6.3	名称空间成员的存取	56
2.6.4	未命名的名称空间	59
2.6.5	标准名称空间 std	59
本章小结		60
本章作业		62
第3章	类与对象	65
3.1	类	65
3.1.1	类的定义	66
3.1.2	类成员的访问限制	73
3.1.3	友元	75
3.2	对象	84
3.3	类的静态成员	92
3.3.1	静态数据成员	92
3.3.2	静态成员函数	93
3.4	this 指针	96
3.5	类的结构与接口	98
3.6	程序结构	102
本章小结		113
本章作业		114
第4章	构造函数与析构函数	116
4.1	对象的创建	116
4.2	构造函数	117
4.2.1	默认构造函数	119
4.2.2	带参数构造函数	122
4.2.3	拷贝构造函数	123
4.2.4	重载构造函数	128

4.3 析构函数	131
4.4 成员初始化列表及顺序	132
4.5 典型举例	136
本章小结	138
本章作业	138
第5章 继承	140
5.1 类的层次结构	140
5.2 基类与派生类	141
5.3 派生类对象的构造	142
5.3.1 派生类对象的内存空间	142
5.3.2 公有继承的含义	148
5.3.3 派生类对象的构造	149
5.3.4 Protected 成员	154
5.4 组合	155
5.4.1 组合的语法	155
5.4.2 组合和继承的联合	159
5.5 继承的方式和访问控制	160
5.5.1 公有继承	160
5.5.2 私有继承	161
5.5.3 保护继承	161
5.6 多重继承	164
5.6.1 多继承的工作方式	164
5.6.2 多继承中的二义性问题	167
5.6.3 虚拟继承	169
5.6.4 构造函数的调用顺序	173
本章小结	174
本章作业	174
第6章 多态	176
6.1 多态和虚函数	176
6.1.1 什么是多态性	176
6.1.2 绑定和 C++ 中的多态性类型	176
6.1.3 虚函数	179
6.1.4 虚函数的工作方式	187

6.2 纯虚函数和抽象类	189
6.2.1 纯虚函数	189
6.2.2 抽象类	191
6.3 虚析构函数	193
6.4 利用多态性实现工资计算实例	195
本章小结	200
本章作业	200
第7章 类模板	203
7.1 类模板的概念	203
7.2 实例化类模板	206
7.3 类模板的成员函数	207
7.4 顺序表类模板	209
本章小结	220
本章作业	220
第8章 运算符重载	221
8.1 运算符重载的概念	221
8.1.1 运算符的两种重载方式	223
8.1.2 运算符函数的原型	223
8.2 使用成员函数重载二元运算符	224
8.3 使用友元函数重载二元运算符	229
8.4 重载一元运算符	233
8.4.1 重载运算符“++”	234
8.4.2 重载赋值运算符“=”	236
8.5 类型转换运算符与类型转换构造函数	240
8.5.1 类型转换运算符	241
8.5.2 类型转换构造函数	244
本章小结	248
本章作业	249
第9章 异常处理	252
9.1 异常处理的概念	252
9.2 异常处理的基本语法	253
9.3 如何进行异常处理	254
9.3.1 抛出异常	255

9.3.2 try 块	255
9.3.3 捕获异常	256
9.4 抛出异常的对象	263
本章小结	274
本章作业	274
第 10 章 I/O 流	277
10.1 C++ 的流及流类库	277
10.1.1 C++ 的流	277
10.1.2 流类库	282
10.2 输入输出的格式控制	283
10.2.1 用 ios 类成员函数进行格式控制	283
10.2.2 使用操作符进行输入输出格式控制	287
10.3 文件输入输出	289
10.3.1 文件的打开与关闭	290
10.3.2 文本文件的读/写	291
10.3.3 二进制文件的读/写	292
10.3.4 随机访问文件	294
本章小结	296
本章作业	296
参考文献	298
全国等级考试《C++ 程序设计》考试大纲	299
模拟试题 A	302
模拟试题 B	310
模拟试题参考答案	316

第 1 章 面向对象程序设计与 C++ 概述

1.1 软 件

一个完整的计算机系统包括硬件 (Hardware) 和软件 (Software) 两大部分。硬件是指计算机系统中的各种物理元件, 包括控制器、运算器、内存储器、I/O 设备以及外存储器等, 它是计算机系统的物质基础。软件是相对于硬件而言的, 是计算机中能够执行的各种程序以及开发、使用和维护程序所需要的所有文档和数据的集合。在计算机科学中, 数据是指所有能输入到计算机中并被计算机程序处理的符号的总称, 比如文字、数字、图形、图像、声音、影像、动画等等。

计算机的软件系统通常分为两大类: 系统软件和应用软件。

1. 系统软件 (System Software)

系统软件是指管理、控制和维护计算机及其外部设备, 提供用户与计算机之间界面等方面的软件。相对于应用软件而言, 系统软件离计算机系统的硬件比较近, 而离用户关心的问题则远一些, 它们并不专门针对具体的应用问题。通常情况下, 系统软件就是操作系统 (Operating System), 如 DOS, Windows 95, Windows 2000, Windows XP, Windows 2003 Server, Windows Mobile, Windows Vista, UNIX, Linux, RedHat, OS/2, Mac OS, Solaris 等等。系统软件是计算机系统的基本软件, 也是计算机系统必备的软件。系统软件的主要功能是管理、监控和维护计算机资源 (包括硬件和软件), 以及开发应用软件。它包括 4 个方面的软件: 操作系统、各种语言处理程序、系统支持和服务程序、数据库管理系统。

2. 应用软件 (Applications)

应用软件一般是指那些能够在操作系统的支持下运行, 直接帮助使用者完成具体工作的各种各样的软件, 如文字处理软件、计算机辅助设计软件、企业事业单位的信息管理软件以及游戏软件等。应用软件是为解决计算机各类应用问题而编制的软件系统, 它具有很强的实用性。应用软件是由系统软件开发的, 可分为两种:

(1) 用户程序。用户为了解决自己特定的具体问题而开发的软件，在系统软件和应用软件包的支持下开发。

(2) 应用软件包。为实现某种特殊功能或特殊计算，经过精心设计的独立软件系统，是一套满足同类应用的许多用户需要的软件，如表 1-1 所示。

表 1-1 各种应用软件

应用软件种类	应用软件名称
文字处理软件	Microsoft Word, 金山 WPS
表格处理软件	Microsoft Excel, Lotus 1 2 3
演示文稿软件	Microsoft PowerPoint
电子书浏览软件	Adobe Reader
图像浏览软件	AcdSee
图像处理软件	Adobe Photoshop, MAYA
浏览器软件	Internet Exploer, Maxthon
多媒体制作软件	Authorware, 方正奥思
音频视频播放软件	Windows Mediaplayer, Realplayer, 暴风影音
P2P 软件	Bitcomet, Emule, PPLive
游戏软件	Warcraft, 仙剑奇侠传, 帝国时代

一般来说，计算机应用软件的发展体现了用户对计算机的需求，只有通过应用软件，计算机才能够涉及更多的应用领域。

3. 软件开发工具 (Application Development Tools)

系统软件和应用软件这两类软件之间并没有严格的界限。有些软件夹在两者中间，不易分清其归属。例如，目前有一些专门用来支持软件开发的软件系统（软件工具），这些软件通常采用某一种程序设计语言，如 C++，Java 或 Pascal，具有友好的程序设计界面，以及完整的编译器（Compiler）和调试器（Debugger），它们不涉及用户具体应用的细节，但是能为应用开发提供支持。例如：Microsoft Visual Stdio. Net, JBuilder, Borland C++ Builder, Borland Delphi, IBM Websphere, Eclipse, PowerBuilder 等。

1.2 程序设计语言

自 1946 年世界上第一台电子计算机问世以来，计算机科学及其应用的发展十分迅猛，计算机被广泛应用于人类生产、生活的各个领域，推动了社会的进步与发展。特别是随着因特网（Internet）日益深入千家万户，传统的信息收集、传输及交换方式正被革命性地改变，人们难以摆脱对计算机的依赖，计算机已将

人类带入了一个新的时代——信息时代。

人使用计算机,就需要和计算机交换信息。为解决人和计算机对话的语言问题,就产生了计算机语言。计算机语言是随着计算机技术的发展,根据解决实质问题的需要逐步形成的。

计算机程序设计语言的发展,经历了从机器语言、汇编语言到高级语言的历程。

1. 机器语言 (Machine Language)

由计算机硬件系统可以识别的二进制指令组成的语言称为机器语言。这是直接用二进制代码指令表示的计算机语言,是计算机唯一能直接识别、直接执行的计算机语言。因为不同计算机的指令系统不同,所以机器语言程序没有通用性。在计算机发展的初期,软件工程师们只能用机器语言来编写程序。这一阶段,在人类的自然语言和计算机编程语言之间存在着鸿沟。

计算机所使用的是由“0”和“1”组成的二进制数,二进制是计算机的语言的基础。

计算机发明之初,用计算机的语言去命令计算机进行某项运算,需要写出一串串由“0”和“1”组成的指令序列交由计算机执行。

例如:假定 11010110000101110010110010101010 代表 $Z=X+Y$ 这样一个指令。

使用机器语言是十分复杂的事情,特别是在程序有错需要修改时,更是如此。而且,由于每台计算机的指令系统往往各不相同,所以在一台计算机上执行的程序,要想在另一台计算机上执行,必须另编程序,造成了重复工作。但由于使用的是针对特定型号的计算机语言,故运算效率是所有语言中最高的。机器语言,是第一代计算机语言。

2. 汇编语言 (Assembly Language)

用机器语言编写程序几乎是不可行的,因此,人们进行了一种有益的改进:用一些简洁的英文字母、符号串来替代一个特定的指令的二进制串。比如,用“ADD”代表加法,“MOV”代表数据传递等等,这样一来,人们很容易读懂并理解程序在干什么,纠错及维护都变得很方便,而且它和机器语言基本上是一一对应的,更便于记忆。这种程序设计语言就称为汇编语言,即第二代计算机语言。然而,计算机是不认识这些符号的,这就需要一个专门的程序,负责将这些符号翻译成二进制数的机器语言,这种翻译程序被称为汇编程序。

汇编语言同样十分依赖于机器硬件,移植性不好,但效率仍十分高,针对计算机特定硬件而编制的汇编语言程序,能准确发挥计算机硬件的功能和特长,程序精炼而质量高,所以至今仍是一种常用而强有力的软件开发工具。

汇编语言和机器语言都是面向机器的程序设计语言，不同的机器具有不同的指令系统，一般将它们称为“低级语言”。此时，编程语言与人类自然语言间的鸿沟略有缩小，但仍与人类的思维相差甚远。因为它的抽象层次太低，程序员需要考虑大量的机器细节。

3. 高级语言 (High-level Programming Language)

从最初与计算机交流的痛苦经历中人们意识到，应该设计一种语言，这种语言接近于数学语言或人的自然语言，同时又不依赖于计算机硬件，编出的程序能在所有机器上通用。经过努力，1954 年，第一个完全脱离机器硬件的高级语言——Fortran 问世了，40 多年来，共有几百种高级语言出现，有重要意义的有几十种，其中影响较大、使用较普遍的有 Fortran, Algol, COBOL, BASIC, LISP, Pascal, C, Prolog, Ada, C++, Delphi, Java, C# 等。

高级语言屏蔽了机器的细节，提高了语言的抽象层次，与具体的计算机指令系统无关，其表达方式更接近人们对求解过程或问题的描述方式。这是面向程序的、易于掌握和书写的程序设计语言。使用高级语言编写的程序称为“源程序”，必须将其编译成目标程序，再与有关的“库程序”连接成可执行程序，才能在计算机上运行。

1.3 程序设计方法的演变

高级语言的发展经历了从早期语言到结构化程序设计语言，从面向过程到非过程化程序语言的过程。相应地，软件的开发也由最初的个体手工作坊式的封闭式生产，发展为产业化、流水线式的工业化生产。

1.3.1 结构化程序设计

20 世纪 60 年代中后期，软件越来越多，规模越来越大，而软件的生产基本上是各自为战，缺乏科学规范的系统规划与测试、评估标准，其恶果是大批耗费巨资建立起来的软件系统，由于含有错误而无法使用，甚至带来巨大损失，软件给人的感觉是越来越不可靠，以致几乎没有不出错的软件。这一切，极大地震动了计算机界，史称“软件危机”。人们认识到：大型程序的编制不同于写小程序，它应该是一项新的技术，应该像处理工程一样处理软件研制的全过程。程序的设计应易于保证正确性，也便于验证正确性。1969 年，提出了结构化程序设计方法。1970 年，第一个结构化程序设计语言——Pascal 语言出现，标志着结构化程序设计时期的开始。

结构化程序设计的思路是：自顶向下、逐步求精。采用模块分解与功能抽

象，自顶向下、分而治之。程序结构采用模块化的设计方法，按功能划分为若干个基本模块，形成一个树状结构；各模块间的关系尽可能简单，功能上相对独立；每一模块内部均是由顺序、选择和循环三种基本结构组成。其模块化实现的具体方法是使用子程序（函数）。

结构化程序设计的优点是：能够有效地将一个较复杂的程序系统设计任务分解成许多易于控制和处理的子任务，便于开发和维护。但是它的缺点也很明显：可重用性和数据安全性差，难以开发图形界面的应用。因为在程序设计过程中，把数据和处理数据的过程分离为相互独立的实体；当数据结构改变时，所有相关的处理过程都要进行相应的修改；每一种相对于老问题的新方法都要带来额外的开销；图形用户界面的应用，很难用过程来描述和实现，开发和维护都很困难。

在这一阶段，利用结构化程序设计方法所设计出来的程序主要应用于数学计算。

1.3.2 面向对象程序设计方法

80 年代初开始，在软件设计思想方面又产生了一次革命，其成果就是面向对象的程序设计。在此之前的高级语言，几乎都是面向过程的，程序的执行是流水线似的，在一个模块被执行完成前，人们不能干别的事，也无法动态地改变程序的执行方向。这和人们日常处理事物的方式是不一致的，对人而言是希望发生一件事就处理一件事，也就是说，不能面向过程，而应面向具体的应用功能，也就是对象（Object）。其方法就是软件的集成化，如同硬件的集成电路一样，生产一些通用的、封装紧密的功能模块，称之为软件集成块，它与具体应用无关，但能相互组合，完成具体的应用功能，同时又能重复使用。对使用者来说，只关心它的接口（输入、输出）及能实现的功能，至于如何实现的，那是其内部的事，使用者完全不用关心，C++，VB，Delphi 就是典型代表。高级语言的下一个发展目标是面向应用，也就是说：只需要告诉程序你要干什么，程序就能自动生成算法，自动进行处理，这就是非过程化的程序语言。

面向对象程序设计方法的出发点是希望更直接地描述客观世界中存在的事物（对象）以及它们之间的关系。面向对象程序设计语言属于高级语言，它将客观事物看作具有属性和行为的对象，通过抽象找出同一类对象的共同属性和行为，形成类，类通过一个简单的外部接口，与外界发生关系，对象与对象之间通过消息进行通信；通过类的继承与多态实现代码重用；这样的程序能够比较直接地反映问题域的本来面目，软件开发人员能够利用人类认识事物所采用的一般思维方法来进行软件开发；而且程序模块间的关系更为简单，程序模块的独立性、数据的安全性就有了良好的保障，通过继承与多态性，可以大大提高程序的可重用性，使得软件的开发和维护都更为方便。

1. 对象 (Object)

对象是自然世界中一个实际存在的事物。可以是有形的（比如一栋房子），也可以是无形的（比如一个项目）。对象是构成世界的一个独立单位，它具有自己的特征，和自己所表现的行为或具有的功能。

面向对象方法中的对象是系统中用来描述客观事物的一个实体，它是用来构成系统的一个基本单位，对象由一组属性和一组行为构成。

2. 类 (Class)

分类是人类通常的思维方法。可以把自然界的事物分成许多类，比如说生物界的门、纲、目、科、属、种等就是一种分类的方式。

在对事物进行分类的时候，首先要考虑事物共有的特征，提取这些共有特征的方法称之为抽象。抽象就是忽略事物的非本质特征，只注意那些与当前目标有关的本质特征，从而找出事物的共性，把具有共同性质的事物划分为一类，得出一个抽象的概念。例如，山、树木、汽车、房屋、鸟、鱼、虫等都是人们在长期的生产和生活实践中抽象出的概念。

在面向对象程序设计方法中提出了“类”的概念，它实际上就是把我们需要进行编程的对象进行抽象，形成类。类是具有相同属性和服务的一组对象的集合，它为属于该类的全部对象提供了抽象的描述，包括属性和行为两个主要部分。

类与对象的关系：类是静态的，是对对象的抽象，对象是动态的，是通过类创建出来的，类创建出来的一个对象称作类的实例。

3. 面向对象程序设计方法的三大特性

(1) 封装性。把对象的属性和服务结合成一个独立的系统单位，尽可能隐蔽对象的内部细节。对外形成一个边界（或者说是一道屏障），只保留有限的对外接口使之与外部发生联系。封装是一种数据隐藏技术，它通过把一组数据和与数据有关的操作集合放在一起形成对象来实现。对象通过操作接口与外部发生联系，而内部的具体细节则被隐藏起来，对外是不可见的。

(2) 继承性。继承对于软件复用有着重要意义，是面向对象技术能够提高软件开发效率的重要原因之一。通过继承，程序可以在扩展现有类的基础上声明新类。其中新类被称作原有类的子类或派生类，原有类称作基类，又叫父类。

(3) 多态性。多态是指相同的消息为不同的对象接收到时，可能导致不同的动作。在一般类中定义的属性或行为，被特殊类继承之后，可以具有不同的数据类型或表现出不同的行为。这使得同一个属性或行为在一般类及其各个特殊类中

具有不同的语义。比如画图操作 Draw(), 可以绘制平面图形 (矩形、圆形、三角形等), 也可以画立体图形 (球、圆柱、圆锥、三角锥等)。

4. 面向对象在软件工程领域中的应用

(1) 面向对象的分析 (OOA)。直接用问题域中客观存在的事物建立模型中的对象, 对单个事物及事物之间的关系, 都保留它们的原貌, 不做转换, 也不打破原有界限而重新组合, 因此能够很好地映射客观事物。

(2) 面向对象的设计 (OOD)。把 OOA 模型直接搬到 OOD, 作为 OOD 的一部分, 针对具体实现中的人机界面、数据存储、任务管理等因素补充一些与实现有关的部分。

(3) 面向对象的程序设计 (OOP)。OOP 工作就是用一种面向对象的编程语言把 OOD 模型中的每个成分书写出来, 是面向对象的软件开发最终落实的重要阶段。

(4) 面向对象的测试 (OOT)。测试的任务是发现软件中的错误。在面向对象的软件测试中继续运用面向对象的概念与原则来组织测试, 以对象的类作为基本测试单位, 可以更准确地发现程序错误并提高测试效率。

(5) 面向对象的软件维护 (OOSM)。软件交付使用后, 工作并没有完结, 还要根据软件的运行情况 and 用户的需求, 不断改进系统。使用面向对象的方法开发的软件, 其程序与问题域是一致的, 因此, 在维护阶段运用面向对象的方法可以大大提高软件维护的效率。

1.4 C++ 程序设计语言

1. C++ 语言的发展

20 世纪 60 年代, Martin Richards 设计出 BCPL 语言, 它被用于在开发软件系统时作为记述语言。1970 年, Ken Thompson 在 BCPL 的基础上开发了 B 语言, 美国 DEC 公司的 PDP-7 计算机中的 UNIX 操作系统就是用 B 语言开发的。1972 年, Dennis Ritchie 等在为 PDP-11 计算机开发 UNIX 操作系统时, 对 B 语言做了进一步的改进, 推出了更加通用的 C 语言。C 语言只是一种面向过程的编程语言, 已经无法满足运用面向对象方法开发软件的需要。为此, AT&T 贝尔实验室的 Bjarne Stroustrup 博士在 C 语言的基础上对其进行了改进和扩充, 开发出支持面向对象程序设计的 C 语言。1983 年正式取名为 C++。C++ 语言的标准化工作从 1989 年开始, 一直到 1998 年 9 月 ISO (国际标准组织) 出版的编号为 ISO/IEC14882-1998 的文件《Information Technology Programming Lan-