HZ BOOKS
华章教育

普通高等院校计算机课程规划教材

Java程序设计教程

余永红　等编著

本书为教师配有电子教案

机械工业出版社
China Machine Press

普通高等院校计算机课程规划教材

Java程序设计教程

余永红　等编著

机械工业出版社
China Machine Press

本书是一本实用的Java程序设计教材，重点突出Java的面向对象编程思想和网络程序设计特征，以及Java程序开发和调试技术等实际开发中所需的知识。本书组织结构合理，语言简练易懂，内容深入浅出，并配有大量的实例分析。

本书从实用的角度介绍了Java语言编程的方法和特征，其主要内容包括：Java语言的面向对象编程特征、Java流与异常处理、图形界面设计、Java Applet、多线程、网络编程、Java与XML、Java数据库访问、实验指导及Java程序调试技术等。

本书可作为高等学校计算机应用及相关专业本科生教材，也可作为Java编程人员的参考资料和相关培训教材。

图书在版编目（CIP）数据

Java程序设计教程/余永红等编著. —北京：机械工业出版社，2008.9
（普通高等院校计算机课程规划教材）

ISBN 978-7-111-24754-8

Ⅰ.J…　Ⅱ.余…　Ⅲ.JAVA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（2008）第141643号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）
责任编辑：杨庆燕
北京牛山世兴印刷厂印刷 · 新华书店北京发行所发行
2008年9月第1版第1次印刷
184mm × 260mm · 19.5印张
标准书号：ISBN 978-7-111-24754-8
定价：33.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换
本社购书热线：（010）68326294

前　言

程序设计是高等院校计算机及电子信息学科类各专业的一门核心课程。面向对象程序设计方法是目前软件开发的主流方法。Java语言是目前功能强、应用广泛的一种完全面向对象的程序设计语言，具有面向对象、与平台无关、多线程以及强大的网络编程功能等特点。Java自问世以来，就以其得天独厚的优势，在IT行业中掀起了研究与开发浪潮。由于Java语言从根本上解决了Internet的异质、代码交换以及网络程序的安全性等诸多问题，因此Java语言完全改变了网络应用程序的开发和使用方式，并成为许多应用领域特别是Internet网络应用领域最受欢迎的开发与编程语言。它已成为长时间以来最卓越的程序设计语言之一，并进入了主流计算模式，对整个计算机软件业的发展产生了极其重大及深远的影响，对传统的计算模型提出了新的挑战。因此开设Java程序设计课程作为程序设计课程和面向对象方法的训练课程是十分恰当和必要的。

目前市场上关于Java程序设计的教材很多，但对Java程序设计技术的介绍大多基于语言本身，而对一些较为实用的技术则浅尝辄止，难以对读者独立开发Java应用程序有实质性的帮助。同时由于教材需求层次多、类型广，因此需要有适应不同需求特色的教材，有鉴于此，作者在实际教学经验基础上编写了本教材。

全书共11章，前9章介绍Java的基础知识，包括Java面向对象程序设计、Java输入/输出、Java图形用户界面、Java网络编程和多线程等，后两章（*注释）介绍Java在XML和数据库方面的应用，可根据课时安排选择是否讲授。各章的具体内容安排如下：

第1章：概要介绍Java语言的特点、面向对象编程思想、网络编程思想以及Java程序的开发和执行环境。

第2章：概要介绍Java程序设计语言基础，包括数据类型、操作和语句等。

第3章：具体介绍Java面向对象程序设计的基本概念和知识，主要讲述Java语言中的类与对象等知识。

第4章：具体介绍Java面向对象程序设计的进阶概念和知识，主要讲述Java语言中的继承、多态、重载以及包和接口等知识。

第5章：具体介绍Java语言的输入/输出、文件及异常处理等知识。

第6章：具体介绍Java语言的图形用户界面的设计与实现，包括各种图形控件、图形类及Swing等知识。

第7章：具体介绍Java Applet的使用，包括Java Applet的运行、绘制、多媒体、网络通信及安全方面的知识。

第8章：具体介绍Java语言的多线程编程，包括线程的基本概念和在Java中如何使用多线程等知识。

第9章：具体介绍Java语言的网络编程特征，包括URL、Socket、UDP Socket编程等知识。

*第10章：具体介绍XML基础知识以及Java语言访问XML的SAX、DOM编程知识。

*第11章：具体介绍Java语言的数据库访问接口JDBC的使用。

---

* 标注的为选修内容——译者注。

此外，本书的后面还提供了两个附录，一是Java语言实验指导，包含与教学章节对应的实验内容，为全书提供实验环节的指导。二是NetBeans IDE 5.5调试工具的介绍，指导读者掌握程序调试方法和技能。

本教材具有如下特点：① 实用性：根据实际开发中所需的Java技能组织内容，重点介绍Java的面向对象编程思想和实际开发中所需的Java高级特性，以及Java程序的开发和调试技术。② 技术先进性：以循序渐进方式介绍Java程序设计的多种实用技术，并尽可能将最新技术反映在教材中；注重理论知识和实用新技术相结合，注重基本知识的理解与基本技能的培养，训练学生既具有扎实深厚的基本功，又具有可扩展素质和较强的创新能力；重点突出Java的网络程序设计特征，在开发环境和工具方面尽可能采用最新技术。③ 可读性：组织结构尽可能合理，语言简练易懂，内容深入浅出，并配有大量的实例分析，可帮助读者理解课程内容。④ 系统性：除重点介绍实用的Java高级特性外，还介绍了Java程序设计语言的基本元素，涵盖了Java从面向对象程序设计思想、设计、实现、调试到应用的过程。

本书可作为高等学校计算机应用及相关专业本科生教材，也可作为Java编程人员的参考资料和相关培训教材。

本书主要由余永红、陈红琳、段爱华编写，全书由南京大学徐永森教授和徐洁磐教授审阅并提出许多宝贵意见，本书编写过程中还得到许多老师的支持和帮助，他们参与了资料的收集、实验及程序的编写和调试工作，在此一并表示衷心的感谢。

在本书的编写过程中，作者参阅借鉴了大量的参考资料，在此谨向诸多学者表示衷心的感谢。由于作者水平有限，虽对本书作反复的审核，书中错误与缺点在所难免，希望读者给予批评指正，多提宝贵意见。

编　者

2008年3月

目　录

前言
第1章　Java语言概述 ……1
1.1　Java语言简介及特点……1
1.1.1　Java简介 ……1
1.1.2　Java语言的特点 ……3
1.1.3　Java语言和C/C++语言的比较 ……4
1.2　Java面向对象编程思想……5
1.2.1　对象与类 ……5
1.2.2　消息与方法 ……7
1.2.3　继承 ……9
1.2.4　多态与动态绑定 ……10
1.2.5　面向对象程序设计 ……10
1.3　Java网络编程思想 ……12
1.3.1　Java Applet ……12
1.3.2　Java网络通信 ……13
1.3.3　Java与XML ……14
1.4　Java程序运行过程与开发环境 ……15
1.4.1　Java程序运行过程 ……15
1.4.2　JDK6的安装与使用 ……16
1.4.3　NetBeans 5.5集成开发环境的安装与使用 ……22
1.5　简单Java应用程序开发步骤 ……29
小结 ……29
复习思考题 ……30
第2章　Java语言程序设计基础 ……31
2.1　Java语言程序结构 ……31
2.2　Java语言数据类型 ……33
2.2.1　常量数据 ……33
2.2.2　变量数据 ……35
2.2.3　数值类型之间的转换以及混合运算…37
2.2.4　数组数据 ……38
2.3　Java语言运算符和表达式 ……39
2.3.1　运算符的使用格式及功能 ……39
2.3.2　运算符的使用说明 ……40
2.4　Java语言流控制语句 ……42
2.4.1　分支结构 ……42
2.4.2　循环结构 ……45
小结 ……48
复习思考题 ……48
第3章　类和对象 ……49
3.1　面向对象基础 ……49
3.1.1　对象及其特点 ……49
3.1.2　抽象 ……50
3.1.3　封装 ……51
3.2　类 ……51
3.2.1　类定义 ……52
3.2.2　类使用 ……53
3.2.3　构造函数 ……55
3.3　方法 ……57
3.3.1　方法定义 ……57
3.3.2　方法调用 ……58
3.4　访问控制符 ……59
3.5　静态修饰符、静态字段和方法 ……62
3.6　抽象类与抽象方法 ……65
3.7　最终类、最终属性与最终方法 ……67
小结 ……68
复习思考题 ……69
第4章　继承 ……70
4.1　继承的概念 ……70
4.1.1　继承的定义 ……70
4.1.2　父类和子类 ……71
4.2　类继承 ……71
4.2.1　继承关系的定义 ……71
4.2.2　属性继承与隐藏 ……72
4.2.3　方法继承、覆盖与重载 ……74
4.2.4　在子类中使用构造函数 ……74
4.2.5　父类对象与子类对象的关系 ……74
4.3　多态与动态绑定 ……78
4.3.1　基本概念 ……78
4.3.2　this和super变量……79

4.3.3 继承与多态的应用 ……80
4.4 构造函数的重载 ……84
4.5 包 ……87
4.5.1 创建包 ……87
4.5.2 定位类 ……88
4.5.3 包的导入 ……89
4.5.4 标记包作用域 ……90
4.5.5 Java API包 ……90
4.6 接口 ……92
4.6.1 接口的概念 ……92
4.6.2 接口声明 ……92
4.6.3 接口实现 ……93
小结 ……96
复习思考题 ……97
第5章 输入输出及异常处理 ……98
5.1 流和文件 ……98
5.1.1 流 ……98
5.1.2 文件 ……98
5.2 常用流类 ……99
5.2.1 字节流 ……99
5.2.2 字符流 ……105
5.2.3 标准输入/输出处理 ……108
5.3 文件处理 ……110
5.3.1 创建或打开、关闭文件对象 ……111
5.3.2 read()方法 ……112
5.3.3 write()方法 ……113
5.3.4 其他操作文件的方法 ……114
5.3.5 随机访问文件 ……115
5.3.6 目录 ……115
5.4 文件处理实例 ……117
5.5 Java异常处理 ……118
5.5.1 异常处理基础 ……118
5.5.2 异常的捕获和处理 ……120
5.5.3 异常抛出 ……122
5.5.4 finally语句 ……123
5.5.5 自定义异常类 ……124
小结 ……125
复习思考题 ……125
第6章 图形用户界面 ……127
6.1 图形用户界面概述 ……127
6.1.1 基本概念 ……127
6.1.2 框架与面板 ……130
6.2 Swing常用组件 ……133
6.3 事件处理 ……143
6.3.1 事件处理原理 ……143
6.3.2 按钮点击事件 ……144
6.3.3 捕获窗口事件 ……147
6.3.4 AWT事件层次结构 ……148
6.3.5 焦点事件 ……150
6.3.6 键盘事件 ……151
6.3.7 鼠标事件 ……153
6.4 布局设计与边界 ……156
6.4.1 布局设计 ……156
6.4.2 边界 ……160
6.5 菜单设计 ……161
6.6 对话框设计 ……165
小结 ……168
复习思考题 ……169
第7章 Java Applet ……170
7.1 Applet简介 ……170
7.2 Applet的执行 ……171
7.2.1 Applet的创建 ……171
7.2.2 Applet类的继承关系 ……172
7.2.3 Applet的生命周期及主要方法 ……174
7.2.4 Applet与HTML ……175
7.2.5 Applet与Application ……177
7.3 Applet的AWT绘制 ……180
7.3.1 AWT绘制方法 ……180
7.3.2 Java.awt.Graphics类 ……181
7.3.3 在Applet中输出文字 ……182
7.4 Applet的多媒体支持 ……184
7.4.1 Applet的图像处理 ……184
7.4.2 Applet的动画处理 ……187
7.4.3 Applet的声音处理 ……188
7.5 Applet的通信 ……189
7.5.1 同页Applet间的通信 ……189
7.5.2 Applet和Browser之间的通信 ……190
7.5.3 Applet的网络通信 ……191
小结 ……192
复习思考题 ……192

第8章 Java语言多线程编程 ……193
8.1 线程的概念 ……193
8.1.1 进程与线程 ……193
8.1.2 线程调度与优先级 ……193
8.1.3 线程的状态与生命周期 ……194
8.2 线程的实现方法 ……197
8.2.1 线程类Thread ……197
8.2.2 继承Thread ……198
8.2.3 实现Runnable接口 ……199
8.2.4 多线程在Applet中的应用 ……200
8.3 线程的同步与死锁 ……201
8.3.1 同步的概念 ……201
8.3.2 Synchronized方法 ……203
8.3.3 线程死锁 ……206
8.4 多线程程序实例 ……207
小结 ……210
复习思考题 ……211
第9章 Java语言网络编程 ……212
9.1 Java网络编程概述 ……212
9.2 URL编程 ……213
9.2.1 URL类 ……213
9.2.2 URL获取网络信息与资源 ……216
9.2.3 URL编程实例 ……218
9.3 Socket编程 ……221
9.3.1 Socket通信的一般结构 ……221
9.3.2 TCP Socket编程 ……223
9.3.3 UDP Socket编程 ……226
9.3.4 Socket编程实例 ……229
小结 ……231
复习思考题 ……231
*第10章 Java与XML ……232
10.1 XML简介 ……232
10.2 XML基础 ……235
10.3 使用JAXP进行SAX编程 ……241
10.3.1 解析文档 ……242
10.3.2 内容管理器 ……243
10.4 使用JAXP进行DOM编程 ……245
10.4.1 解析文档 ……245
10.4.2 基本的DOM类 ……246
10.4.3 存取数据 ……248
10.4.4 修改文件 ……249
小结 ……252
复习思考题 ……252
*第11章 数据库编程接口JDBC ……253
11.1 JDBC概述 ……253
11.2 通过JDBC访问数据库 ……258
11.2.1 创建数据库连接 ……258
11.2.2 访问数据库元信息 ……260
11.2.3 查询数据库 ……263
11.2.4 检索结果集 ……268
11.2.5 数据库更新操作 ……269
11.2.6 处理异常和警告 ……271
小结 ……273
复习思考题 ……274
附录A Java实验指导 ……275
附录B NetBeans 5.5程序调试 ……288
参考文献 ……300

第 1 章　Java语言概述

近年来，随着全球Internet的迅猛发展及万维网WWW（World Wide Web）的普及和快速增长，整个计算环境经历了深刻的变革。1989年，超文本标记语言HTML（Hyper Text Markup Language）和万维网WWW的产生是Internet数据描述语言的一次飞跃，万维网把全世界的信息资源用HTML格式统一起来。1995年，Sun公司Java语言的正式发表是一次Internet的技术革命，Java语言的诞生从根本上解决了Internet的异质、代码交换以及网络程序的安全性等问题。首先，Java语言是与平台无关的语言。Java程序编译后，生成字节代码，运行在Java虚拟机（JVM-Java Virtual Machine）上。一个操作系统平台只要提供Java虚拟机，Java程序就可以在上面运行。从理论上讲，Java程序可运行于所有操作系统平台上，这一点从根本上解决了Internet的异质问题。其次，Java语言采用了可移动代码技术，在网络上不仅可以进行无格式的数据信息交换，而且可以进行程序交换。Java语言是比较纯粹的面向对象语言，它的绝大多数程序实体都是对象，利用对象的封装性可以大大降低网络上程序交换的复杂性。再次，Java语言可以和HTML有效地集成在一起，把静态的超文本文件变成可执行的应用程序，极大地增强了超文本的交互操作性。最后，Java是一种更安全的程序设计语言，它消除了C/C++中众多的不安全因素，提供了诸多安全保障机制。Java语言从根本上改变了网络应用程序的开发和使用方式，并成为在许多应用领域特别是Internet网络应用领域最受欢迎的开发与编程语言。

Java不仅仅是编程语言，也是一个开发平台。Java为程序员提供了许多开发工具，包括编译器、解释器、文档生成器和文件打包工具等。同时Java还是一个程序发布平台，目前Sun公司把Java平台划分成J2EE、J2SE、J2ME三个平台，针对不同的市场目标和设备进行定位。J2EE即Java Enterprise Edition（可扩展的企业应用Java2平台），主要是为企业计算提供一个应用服务器的运行和开发平台。J2SE即Java Standard Edition，主要是为台式机和工作站提供一个开发和运行的平台。J2ME即Java Micro Edition，主要是面向消费电子产品，为消费电子产品提供一个Java的运行平台，使得Java程序能够在手机、机顶盒、PDA等产品上运行。

1.1　Java语言简介及特点

1.1.1　Java简介

在经历了以大型机为代表的集中计算模式和以PC机为代表的分散计算模式之后，互联网的出现使计算模式进入了网络计算时代。网络计算模式的一个特点是计算机是异构的，即计算机的类型和操作系统是不一样的，例如SUN工作站的硬件是SPARC体系，软件是Unix中的Solaris操作系统，而PC机的硬件是INTEL体系，操作系统是Windows或者是Linux，因此相应的编程语言基本上只是适用于单机系统，例如Cobol、Fortran、C、C++等；网络计算模式的另一个特点是代码可以通过网络在各种计算机上进行迁移，这就迫切需要一种跨平台的编程语言，使得用它编写的程序能够在网络中的各种PC机上正常运行，Java语言在这种需求下应运而生。正是因为Java语言符合了互联网时代的发展要求，才使它获得了巨大的成功。Java语言已经成为最卓越的程序设计语言之一，并进入了主流计算模式，它的出现对整个计算机软件业的发展产生了重大及深远的影响，对传统的计算模型提出了新的挑战。

1. Java语言发展史

Java语言来自于SunMicrosystems公司的一个叫“Green”的项目，该项目组是1991年由JamesGosling领导成立的，其原先的目的是为家用消费电子产品开发一个分布式代码系统，这样可以把E-mail发给电冰箱、电视机等家用电器，对它们进行可编程控制，并和它们进行交互式信息交流。开始该项目组成员准备采用C++语言，但感觉到C++语言太复杂，安全性也差，无法满足项目设计的需要，最后决定基于C++语言开发一种新的编程语言，Gosling临时为它起名叫Oak语言，Oak语言是一种用于网络的精巧而安全的语言，但当时并未取得成功。1994年随着WWW在Internet上的迅速发展，Sun公司发现Oak语言所具有的跨平台、面向对象、高安全性等特点非常符合互联网的需要，于是改进了该语言的设计，以期达到如下几个目标：创建一种面向对象的程序设计语言，而不是面向过程的语言；提供一个解释执行的程序运行环境，使程序代码独立于平台；吸收C和C++的优点，使程序员容易掌握；去掉C和C++中影响程序健壮性的部分，使程序更安全；实现多线程，使得程序能够同时执行多个任务；提供动态下载程序代码的机制；提供代码校验机制以保证安全性。

按照上述设计目标，Oak项目组成员决定用该语言开发一个新的不依赖于任何硬件平台和软件平台并具有实时、安全、交互功能的HotJava浏览器，此举得到了Sun公司首席执行官ScottMcNealy的大力支持。HotJava浏览器在1995年发布后，立刻引起了产业界的轰动，Java语言得到了充分肯定，并进一步促进Java语言进军Internet广域网，SUN公司的远见卓识，造就了一代成功的编程语言Java。Java语言的取名也有一段趣闻，由于Oak与别的语言同名，所以在Java语言成员组喝着Java（爪哇）咖啡讨论给这个语言取什么名字时，有人灵机一动说就叫Java吧，Java这个名字就这样传开了。

2. Java语言对软件开发技术的影响及应用前景

Java语言是Sun公司推出的新一代面向对象的程序设计语言，特别适合于Internet应用程序的开发，它的硬件和软件平台的无关性直接威胁到Windows和Intel的垄断地位。一时间，“连Internet，用Java编程”，成为技术人员的一种时尚，并对未来软件的开发产生了重大影响。Java语言对软件开发技术的影响可以归纳为如下几个方面：

1) 软件的需求分析：可将用户的需求进行动态的、可视化描述，以提供设计者更加直观的要求。而用户的需求是各种各样的，不受地区、行业、部门、爱好的影响，这些需求都可以用Java语言描述清楚。

2) 软件的开发方法：由于Java语言是纯面向对象的程序设计语言，所以完全可以用面向对象的技术与方法来开发软件，这符合最新的软件开发规范要求。

3) 动画效果：Java语言的动画效果远比GUI技术逼真，尤其是利用WWW提供的巨大动画资源空间，可以共享全世界的动态画面资源。

4) 软件最终产品：用Java语言开发的软件具有“可视化、可听化、可操作化、交互、动画、动作”等特点。

5) 其他：使用Java语言对开发效益、开发价值都有比较明显的影响。

Java语言有着广泛的应用前景，大体上可以从以下几个方面来考虑其应用：

1) 所有面向对象的应用开发，包括面向对象的事件描述、处理、综合等。

2) 计算过程的可视化、可操作化的软件开发。

3) 动态画面的设计，包括图形图像的调用。

4) 交互操作的设计（选择交互、定向交互、控制流程等）。

5) Internet的系统管理功能模块的设计，Web页的动态设计、管理交互操作设计等。

6) Intranet上的软件开发（直接面向企业内部用户的软件）。

7) 与各类数据库连接查询的SQL语句实现。

8) 其他应用类型的程序（智能Web服务、移动计算、嵌入式Java技术、实时Java等）。

1.1.2 Java语言的特点

Java语言是一种简单的、面向对象的、分布的、解释的、健壮的、安全的、可移植的、多线程的以及可扩展的动态的程序设计语言。

1. Java语言的简洁和小巧性

Java语言最初是为对家用电器进行集成控制而设计的一种语言，因此它必须简洁明了，易于学习。Java语言的简洁首先体现在精简的系统上，力图用最小的系统实现最大的功能。其对硬件的要求不高，Java语言的翻译器只需占用几百Kb，因此对于内存很小的计算机，Java语言是一种很理想的编程工具。其次，Java基于面向对象技术，为程序开发者提供了丰富的类库，使程序编写变得容易、简单。Java语言通过提供最基本的方法来完成指定的任务，只需理解一些基本的概念，就可以用它编写出适合于各种情况的应用程序。

2. Java语言的面向对象特性

面向对象是Java语言最为重要的特性。Java语言是由C++发展而来的，是一种彻底的纯面向对象（Object-Oriented）的程序设计语言，完全具备面向对象的封装、继承、多态和动态绑定四大特点，非常适用于大型软件的开发。除简单的数字和布尔类型外，Java语言中的大部分都是对象，通过封装性、继承性和多态性实施开发。封装性实现了模块化和信息隐藏，继承性实现了代码的重用，用户可以建立并运行自己的类库、构造方法重载等。

3. Java语言的分布式特性

Java语言是面向网络的语言，Java语言支持网络应用程序的特性，使它成为一种分布式程序设计语言。Java语言包括一个支持HTTP和FTP等基于TCP/IP协议的子库，它提供一个Java.net包，通过它可以完成各种层次上的网络连接。因此Java语言编写的应用程序可以凭借URL打开并访问网络上的对象，其访问方式与访问本地文件系统几乎完全相同。Java语言另一个Socket类提供的可靠流式网络的连接，使程序设计者可以非常方便地创建分布式应用程序。

4. Java语言的编译和解释特性

Java开发环境把Java源程序编译后生成一种称为字节代码（bytecode）的中间代码，字节代码非常类似于机器指令代码，但并不是二进制的机器指令代码，且字节代码并不专对一种特定的机器，所以Java程序不需重新编译便可在众多不同的计算机上执行，只要该机器上预先装有Java语言运行系统，这是其编译特性。

Java程序编译后产生字节代码，其运行要借助于Java解释器，Java解释器直接对Java字节代码进行解释执行。以字节代码形式发布的Java程序运行在JVM环境上，JVM将字节代码翻译成具体的CPU机器指令，因此Java解释器是与硬、软件平台有关的，在不同的平台上用不同的JVM实现（与平台相关部分的工作由JVM而不是Java编译器来完成，因为平台的种类比起应用软件的数量要少得多）。Java解释器使Java程序在某一特定硬、软件平台环境中直接运行目标代码指令，这种连接程序通常比编译程序所需资源少，所以程序员可以花上更多的时间用在创建源程序上，而不必考虑运行环境，这是其解释特性。

5. Java语言的健壮性

Java语言致力于检查程序在编译和运行时的错误，以消除错误的产生。Java语言是一种比C++还强的强类型语言，Java语言要求用显式的方法声明所用对象的类型，从而保证编译器可

以发现错误，保证程序更加可靠。同时Java语言自己操纵内存，减少了内存出错的可能性，它提供自动废区收集来进行内存管理，防止程序员在管理内存时产生错误。通过集成的面向对象例外处理机制，Java语言提示可能出现未被处理的例外，帮助程序员正确地进行选择，以防止系统的崩溃。另外Java语言在编译时可捕获类型声明中的许多常见错误，防止动态运行时不匹配问题的出现。

6. Java语言的安全性

作为网络程序设计语言，Java必须有足够的安全保障，并且要防止病毒的入侵。Java在运行应用程序前，要经过多次测试，包括代码校验、检查代码段的格式、检测指针操作、对象操作是否适当以及试图改变一个对象的类型等，Java语言的这种字节代码的验证方式可以确保应用程序不含计算机病毒。同时Java的健壮可靠性，JVM的内存自动管理机制，提供的加密技术和支持多种安全网络协议等特性进一步增强了其安全性。

7. Java语言的可移植性

Java程序编译后生成的字节代码是一种与软、硬件平台无关的代码，该代码在虚拟机上运行，由与机器相关的运行调试器执行。任何一种特定的软、硬件平台，只要在该平台上实现了Java虚拟机，Java程序就可以在它上面运行，这是Java语言应用软件便于移植的良好基础。如果基本数据类型设计依赖于具体的计算机和操作系统，会给程序的移植带来很大不便。Java语言通过定义独立于软、硬件平台的基本数据类型及其相关运算，确保数据在任何硬件平台上保持一致。Java语言的类库中实现了与不同平台的接口，使这些类库可以移植。目前，Java编译器本身是用Java语言编写的，Java虚拟机也是由Java语言实现的，这使得Java语言系统本身也具有可移植性。

8. Java语言的多线程性

多线程机制使应用程序能够并行执行，而且同步机制保证了对共享数据的正确操作。通过使用多线程，程序设计者可以分别用不同的线程完成特定的行为，而不需要采用全局的事件循环机制，这样就很容易实现网络上的实时交互行为和实时控制性能。Java语言线程通常被映射为实际的操作系统线程，只要底层操作系统支持这种映射。在多线程环境中的编程通常是有困难的，Java语言提供了易于使用的同步特性，使编程更为方便。

9. Java语言的自动废区回收性

在用C及C++编写大型软件时，编程人员必须自己管理所用的内存块，这项工作非常困难并往往成为出错和内存不足的根源。在Java环境下编程人员不必为内存管理操心，Java语言系统有一个叫做“无用单元收集器”的内置程序，它扫描内存，并自动释放那些不再使用的内存块。Java语言的这种自动废区收集机制，对程序不再引用的对象自动取消其所占资源，彻底消除了出现存储器泄漏之类的错误，并免去了程序员管理存储器的繁琐工作。

10. Java语言的动态性与可扩展性

Java语言可以动态地从本地或网上加载类，这种动态特性使它适合于一个不断发展的环境。在类库中可以自由地加入新的方法和实例变量而不影响用户程序的执行，并且Java语言通过接口来支持多重继承，使之比严格的类继承具有更灵活的方式和可扩展性。

1.1.3 Java语言和C/C++语言的比较

一般程序设计语言都具有数据结构、对数据结构的操作以及流程控制结构等基本成分，数据结构体现为语言提供的对数据类型的支持，对数据结构的操作体现为在数据类型上的运算类型，流程控制结构体现为顺序、分支和循环三种结构。

C语言是结构化程序设计语言，其数据结构和对数据结构的操作是分开的。C++语言是在C语言的基础上，增加了类和继承等面向对象的编程概念，强调数据和操作的封装性和类的可重用性，同时C++兼容C语言，因此它不是纯面向对象的程序设计语言。Java语言是在C++语言基础上发展而来的，是一种完全的纯面向对象的程序设计语言。

从语言产生基础看，三者之间具有一定的传承性，如在变量声明、参数传递、操作符(运算类型)、流控制结构等方面，C语言、C++语言和Java语言是相同的。

从语言特征看，三者之间存在很大的不同，C语言是结构化的程序设计语言，C++语言兼容结构化程序设计和面向对象程序设计的特征，而Java语言是纯面向对象的程序设计语言。C/C++语言中基本的数据类型相同，指针和对内存的控制是其特色，但也是其最复杂和最容易出错的地方，同时其对平台的依赖性较强。Java语言吸取了C/C++语言中的许多优点，同时为了实现其简单、安全、可移植等特性，摒弃了C/C++语言中许多不合理和容易出错的内容，如指针和内存管理、运算符重载、多重继承等模糊的概念，同时增加了一些很有用的功能，如固定数据类型在内存中的字节长度，自动废区收集、安全检查等，大大简化了程序设计者的内存管理工作，提高了程序的安全性和可移植性。

1.2 Java面向对象编程思想

Java语言是由C++语言发展而来的，它是一种纯面向对象的程序设计语言，具备面向对象的对象、类、继承、多态和动态绑定等基本概念，非常适用于大型软件的开发。在深入学习Java之前，有必要了解面向对象编程的一些基本概念和基本方法。

1.2.1 对象与类

1. 对象

对象含义应用广泛，难以精确定义，不同的场合有不同的含义。一般来说，任何事物均可看作对象。任何事物均有各自的自然属性和行为，当考察其某些属性与行为并进行研究时，它便成为有意义的对象。采用面向对象方法进行软件开发时，需要区分三种不同含义的对象：客观对象、问题对象和计算机对象。客观对象是现实世界中存在的实体；问题对象是客观对象在问题域中的抽象，用于根据需要完成某些行为；计算机对象是问题对象在计算机系统中的表示，它是数据和操作的封装体。三种对象间的关系如图1-1所示。

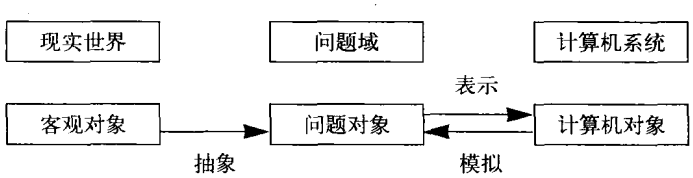

图1-1 三种对象间的关系

对象的表示应包括属性与行为（数据与操作），且对象之间并非彼此孤立，可以通过通信互相交互，因此计算机对象可以表示为一个三元组：对象≡（接口，数据，操作），也即对象是面向对象系统中运行时刻的基本成分，它是属性和行为（数据和操作）的封装体，其中还包括和其他对象进行通信的设施。

下面从不同的角度来考察对象的概念。首先从宏观上看，对象是客观对象在计算机系统中的表示。计算机对象是用来模拟现实世界中的客观对象的，模拟的重点是其中的信息处理

和传递。客观对象自身的信息处理过程是由对象内部状态的变化来模拟的，客观对象间的信息传递则是由对象间的通信来模拟的。

其次从微观上看，对象是由能对外通信的数据及其上的操作组成的封装体。对象由一组数据及其上的操作组成，且能接收其他对象发送的消息，以及向其他对象发送消息。如果把模块间的相互过程调用看作一种模块间的通信，那么，就可以把对象近似地理解成模块的运行实例。不过，两者并不完全相同，模块中数据可以移出，从而对这部分数据不加隐蔽，而对象中的数据则完全是私有的。

最后从形式描述上看，对象是具有输入和输出的有限自动机。对象是一个通信自动机，即以其他自动机的输出作为输入，自己的输出作为其他自动机的输入的自动机。这种自动机的输入输出方式反映了对象的通信能力；而把对象看作某种自动机则反映了对象自身具有独立的计算能力，体现为状态及状态转换机制。

从上述对象概念的描述中，我们可以归纳出对象具有如下特点：

- 自治性：对象的自治性是指对象具有一定的独立计算能力。给定一些输入，经过状态转换，对象能产生输出，说明它具有计算能力。对象自身的状态变化是不直接受外界干预的，外界只有通过发送的消息对它产生影响，从这个意义上说，对象具有自治性。
- 封闭性：对象的封闭性是指对象具有信息隐蔽能力。具体说来，外界不直接修改对象的状态，只有通过向该对象发送消息来对它施加影响。对象隐蔽了其中的数据及操作的实现方法，对外可见的只是该对象所提供的操作（即能接收和处理的消息）。
- 通信性：对象的通信性是指对象具有与其他对象通信的能力，具体说来，就是对象能接收其他对象发来的消息，同时也能向其他对象发送消息。通信性反映了不同对象间的联系，通过这种联系，若干对象可协同完成某项任务。
- 被动性：对象的被动性是指对象的存在和状态转换都是由来自外界的某种刺激引发的。对象的存在可以说是由外界决定的，而对象的状态转换则是在它接收到某种消息后产生的，尽管这种转换实际是由其自身进行的。
- 暂存性：对象的暂存性有两层含义。一是指对象的存在可以动态地引发，而不是必须在计算的一开始就存在；二是指对象随时可以消亡（而不是必须存在到计算结束）。虽然可以在计算过程中自始至终保存某些对象，但从对象的本质或作用来说，它具有暂存性。

五个性质分别刻划了对象不同方面的特点。自治性，封闭性和通信性刻划的是对象的能力；被动性刻划的是对象的活动；暂存性刻划的是对象的生存特性。自治性反映了对象独立计算的能力；封闭性和通信性则说明对象是既封闭又开放的相对独立体。

2. 类

对象是系统运行时的基本成分，它们在程序中如何反映呢？事实上，系统中往往存在多个具有共同特性的对象。例如，张三、李四、王五、赵六……都具有姓名、身份证号、性别和年龄等特性，则具有姓名、身份证号、性别和年龄等特性的抽象对象便是张三、李四、王五、赵六……对象的一个抽象。在语言中，这样的抽象称为“类”，类刻划了一组具有共同特性的对象。比如上述张三、李四、王五、赵六……可以抽象为一个具有姓名、身份证号、性别和年龄等特性的“人”的对象，称为类“人”。

类的作用可归纳为两种：一是作为对象的描述机制，刻划一组对象的公共属性和行为；二是作为程序的基本单位，它是支持模块化设计的设施，并且类上的分类关系是模块划分的规范标准。

与对象的组成部分相对应，类也有三个组成部分：数据、操作和接口。数据刻划对象的

状态，操作刻划对象的行为，类中所有数据均为私有，接口使操作对外可见。从类自身的内容看，它描述了一组数据及其上的操作，这些数据为类所私有，只有操作对外可见。

类的概念可从下面四个方面去理解：

- 类是面向对象程序惟一的构造单位：类是构造面向对象程序的基本构件，也是惟一的构件。面向对象程序设计的任务是设计类及由类组装程序，一个面向对象程序就是一组相关的类。面向对象程序由一个或多个类组成，运行时需要创建类的对象（实例）。因此类是静态概念，对象是动态概念。程序中只出现类，不出现对象，类可看作抽象数据类型的实现。
- 类是面向对象程序设计语言的基本成分：类是面向对象程序设计语言中的基本概念。在面向对象程序设计的五个基本概念中，对象和类是基础，继承是特色，多态与动态绑定与它们密切相关。作为语言成分的类是一种程序单位，类内的成分是不能单独构成程序的，程序至少应包含一个完整的类。
- 类是抽象数据类型的具体实现：类体现了一种数据抽象（抽象数据类型是和具体实现无关的，而类则是联系以相应语言系统的实现），所以类是抽象数据类型在面向对象程序设计语言中的具体实现，它给出了具体的数据结构表示，并用面向对象语言中的语句给出了操作的实现。
- 类描写了一组相似对象的共同特性：面向对象程序执行时体现为一组对象状态的变化，这里的对象便是由类来刻划的。类刻划了一组具有相似特性的对象，在运行时可根据类中的描述动态地创建对象。

3. 类与对象的关系

类是面向对象程序中的概念，而对象则是面向对象程序运行时的概念。程序由一组相关的类构成，所以类是静态的。程序的执行体现为一组相互通信的对象的活动，所以对象是动态的。

类描述了一组相似对象的共同特性，这一组相似的对象被称为该类的实例。类作为一种模式，对象是具有这种模式的具体例子。类与对象的关系很象类型与值的关系。事实上，完全可以把类理解成对象所具有的类型，而把对象理解成是相应类作为类型的值。具体说来，若类C描述了一组对象O_1，O_2，…，O_n则O_1，O_2，…，O_n都称为C的实例。把C理解成某种类型，则该类型对应的值集就是{O_1，O_2，…，O_n}。因此，当说明了一个具有类型C的变量后，该变量就可以取且只能取C的值集中的值。

1.2.2 消息与方法

1. 消息

上面介绍了对象是一个相对独立的具有一定计算能力的自治体，对象之间不是彼此孤立而是互相通信的，面向对象程序的执行体现为一组相互通信的对象的活动。那么面向对象程序是如何实施计算（运行）的呢？计算是由一组地位等同的被称作对象的计算机制合作完成的，合作方式是通信即相互交换信息，这种对象与对象之间互相传递的信息称为消息。消息可以表示计算任务，也可以表示计算结果。

面向对象计算中，每一计算任务都表示为一消息，实施计算任务的若干相关联的对象组成一个面向对象系统。提交计算任务即由任务提交者（系统外对象）向承担计算任务的面向对象系统中的某对象发送表示该计算任务的消息。计算的实施过程是面向对象系统接收到该消息后所产生的状态变化过程，计算的结果通过面向对象系统中的对象向任务提交者回送的

消息体现。

消息一般由三个部分组成：

- 接受消息的对象。
- 接受对象应采用的方法。
- 方法所需要的参数。

计算任务通常先由某一对象“受理”（该对象接收到某种消息），然后，通过对象间的通信，计算任务就分散到各个有关对象中，最后，再由某些对象给出结果（通过发送消息）。发送消息的对象称为发送者，接受消息的对象称为接受者。消息中包含发送者的要求，它告诉接受者需要完成哪些处理，但并不指示接受者如何完成这些处理。消息完全由接受者解析，接受者独立决定采用什么方式完成所需处理。一个对象能够接受不同形式、不同内容的多个消息，相同形式的消息可以发往不同的对象，不同的对象对于形式相同的消息可以有不同的解析，能够作出不同的反应。对于传来的消息，对象可以返回相应的应答消息。

对象可以动态地创建，创建后即可以活动。对象在不同时刻可处于不同状态，对象的活动是指对象状态的改变，它是由对象所接收的消息引发的。对象一经创建，就能接收消息，并向其他对象发送消息。对象接收到消息后，可能出现：① 自身状态改变；② 创建新对象；③ 向其他对象发送消息。

从对象之间的消息通信机制可反映出面向对象计算具有如下特性：

- 协同性：协同性表现在计算是由若干对象共同协作完成的。虽然计算任务可能首先由面向对象系统中的某个特定对象“受理”（即接收到表示该任务的消息），但往往并不是由该对象独立完成的，而是通过对象间的通信被分解到其他有关对象中，由这些对象共同完成，对象间的这种协同性使计算具有分布性。
- 动态性：动态性表现在计算过程中对象根据通信关系组成的结构会动态地改变，新对象会不断创建，老对象也会不断消亡。面向对象系统最初由若干初始对象组成，一旦外界向这些初始对象发送了表示计算任务的消息，面向对象系统即活动起来直至给出计算结果。在此过程中，面向对象系统的组成因创建新对象而不断地改变。
- 封闭性：封闭性表现在计算是由一组相对封闭的对象完成的。从外界看，一个对象只是一个能接收和发送消息的机制，其内部的状态及其如何变化对外并不直接可见，外界只有通过给它发送消息才能对它产生影响。对象承担计算的能力完全通过它能接受和处理的消息体现。
- 自治性：自治性表现在计算是由一组自治的对象完成的。对象在接收了消息后，如何处理该消息（即：自身状态如何改变，需创建哪些新对象，以及向其他对象发送什么消息），完全由该对象自身决定。在面向对象计算中，数据与其上的操作地位等同，两者紧密耦合在一起形成对象，亦即数据及其上的操作构成对象。因此，面向对象计算中，数据与其上的操作之间的联系处于首要地位，何时对何数据施行何种操作完全由相应数据所在对象所接收到的消息及该对象自身决定。由于对象的封装性和隐蔽性，对象的消息仅作用于对象的接口，通过接口进一步影响和改变对象状态。

2. *方法*

方法反映对象的行为，是对象固有的动态表示，可审视并改变对象的内部状态。一个对象往往可以用若干方法表示其动态行为，在计算机中，方法也可称为操作。它的定义与表示包含两部分：一是方法的接口，它给出了方法的外部表示，包括方法的名称、参数及结果类型；二是方法的实现，它用一段程序代码表示，这段代码实现了方法的功能。

把所有对象抽象成各种类，每个类都定义一组方法，代表允许作用于该类对象上的各种操作。方法描述了对象执行操作的算法及响应消息的方法。

1.2.3 继承

类之间的继承关系是现实世界中遗传关系的模拟，它表示类之间的内在联系以及对属性和操作的共享。继承是类与类之间的一种关系，它使程序人员可以在已有类的基础上定义和实现新类。继承是实现利用可重用软件构件构造系统的有效语言机制。

继承能有效地支持软件构件的重用，从而使得当需要在系统中增加新特征时所需的新代码最少，并且当继承和多态、动态绑定结合使用时，为修改系统所需变动的原代码最少。

当类Y继承了类Z时，称Z是基类，Y是Z的子类。在这种情形下，Y由两部分组成，继承部分和新增部分。继承部分是从类Z继承得到的，新增部分是专为Y所编写的新代码。从Z中特征到Y中继承部分相应特征的映射是由语言规则和程序人员在Y中所编写的代码共同确定的。这一映射可以是一种简单的恒等变换，使得Y的继承部分同Z完全一样；也可以是重命名、重定义等变换，对继承特征进行调整。每种面向对象语言都有自己所允许的继承映射集合，在语言一级提供了关键字来指出所需要的映射类型，有些映射要求提供额外信息。例如，如果Z的一个特征映射到Y时要重定义，则必须给出相应的实现。

类与类之间的继承关系反映了相应概念间的包含关系，这是因为，当类Y是类Z的后继时，Y通过继承Z而获得了Z的所有特征，所以从概念上讲，Z包含Y。因此，继承关系用来表示问题域中已有的抽象结构。

一个常用的例子是学校组成人员之间的关系，本科生是一类特殊的学生，这一关系很容易用如图1-2的继承关系来刻划。

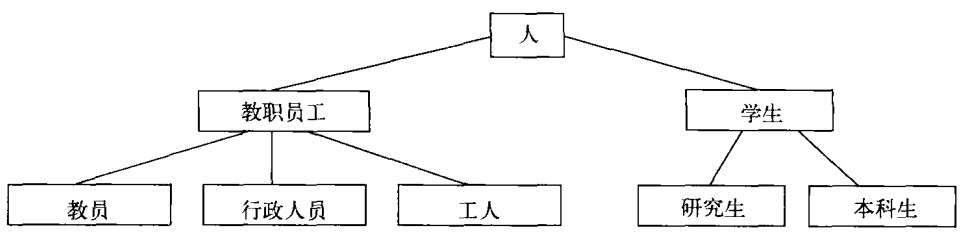

图1-2 学校组成人员间的继承关系图

作为学生的后继，本科生具有学生的所有特征。另外，由于学生是人的后继，本科生也继承了人的特征。在最高抽象层上，所有学校组成人员都具有身份号、姓名、年龄和性别等属性，也有一些相同的操作，这些特征在根类“人”中定义。学生类增加了成绩、课程等属性。由于继承的关系，因此本科生也具有身份号、姓名、年龄、性别、成绩、课程等属性。继承的使用大大地减少了添加新的特征所需的工作，同样在面向对象开发过程中，继承也方便了原型速成的开发方法。

继承机制的强有力之处还在于它允许程序员可重用一个未必完全符合要求的类，允许对该类进行修改而又不至于在该类的其他部分引起有害的副作用。另外，在面向对象设计中，还注重在较高抽象层次上提取和封装公共性质。如果把这些高层次的类保存在类库中，那么，对于用户所需要的类，在类库中可能都有一个更一般的类与之对应。

如果一个软件系统是用面向对象方法进行分析和设计，并用面向对象程序设计语言实现的，那么在分析阶段所识认的对象和分类关系就能在设计阶段得到保留和丰富，而且可以直