全国计算机等级考试教材系列

National Computer Rank Examination

二级

Java语言程序设计教程

杨 昭 主 编

陈祥 杨丽波 张文科 副主编

National Computer Rank Examination

中国水利水电出版社

www.waterpub.com.cn

最新大纲

全国计算机等级考试教材系列

二级 Java 语言程序设计教程

杨 昭 主 编

陈 祥 杨丽波 张文科 副主编

中国水利水电出版社

内 容 提 要

本书是根据教育部考试中心最新制定的《全国计算机等级考试大纲（2004 年版）》对二级 Java 的考试范围要求组织有多年等考培训经验的老师编写的。

全书共 11 章，大致可划分为四大部分：第一部分介绍 Java 语言的入门知识，包括语言概述、语言基础、控制语句等；第二部分介绍 Java 语言的核心编程知识，包括类及其方法、继承与多态、包和接口、异常处理机制等；第三部分对多线程编程、Applet 编程、输入/输出等作了介绍，并对 Java 中一些常用的工具包和类作了较为详细的说明；第四部分为附录部分，包括最新二级 Java 考试大纲、全真模拟试卷、参考答案等。

本书内容翔实、逻辑清晰、讲解透彻、涉及面广，具有极强的可操作性和针对性。通过对本书的学习可轻松掌握有关 Java 编程的基本知识，达到教育部对二级 Java 语言的掌握要求。本书适合作为全国计算机等级考试二级 Java 的培训和自学教材，也可作为高等院校计算机基础课教材和 Java 编程爱好者的自学教材。

图书在版编目(CIP)数据

二级 Java 语言程序设计教程 / 杨昭主编. —北京：中国水利水电出版社，2006

（全国计算机等级考试教材系列）

ISBN 7-5084-3703-9

Ⅰ. 二… Ⅱ. 杨… Ⅲ. Java 语言—程序设计—水平考试—教材 Ⅳ.TP312

中国版本图书馆 CIP 数据核字（2006）第 029793 号

书　　名	二级 Java 语言程序设计教程
作　　者	杨 昭 主 编 陈 祥 杨丽波 张文科 副主编
出版 发行	中国水利水电出版社（北京市三里河路 6 号 100044） 网址：www.waterpub.com.cn E-mail：mchannel@263.net（万水） sales@waterpub.com.cn 电话：（010）63202266（总机） 68331835（营销中心） 82562819（万水）
经　　售	全国各地新华书店和相关出版物销售网点
排　　版	北京万水电子信息有限公司
印　　刷	北京市天竺颖华印刷厂
规　　格	787mm×1092mm 16 开本 21.75 印张 530 千字
版　　次	2006 年 4 月第 1 版 2006 年 4 月第 1 次印刷
印　　数	0001—4000 册
定　　价	32.00 元

前　言

本书是根据教育部计算机等级考试中心对二级 Java 的大纲要求编写的，内容紧扣新大纲的要求，完全适合考生的需要。

自从 1995 年被正式推出之后，Java 语言就以其独特的优势迅猛发展，经过短短七八年的时间，成为迄今为止最优秀的面向对象语言。Java 也从当初的一种语言逐渐形成一种产业，基于 Java 语言的 J2EE 架构已成为微软.NET 平台的强大竞争对手。万维网（www）的创始人 Berners-Lee 曾经说过：计算机产业发展的下一个浪潮就是 Java，并且将会很快发生。现在看来，这一预言已成为不争的事实。比尔·盖茨也不无感慨地说过：Java 是长时间以来最卓越的程序设计语言。

Java 最大的特点是跨平台性，它将是未来网络世界中的“世界语”。有人预言，今后所有用其他语言编写的软件统统都要用 Java 语言来改写。此外，Java 语言还具有简单性、面向对象性、分布性、动态性、鲁棒性、多线程性、可移植性等诸多特点。Java 产业在国内目前正在如火如荼地发展着，基于 Java 的高级技术如 J2ME、J2EE、Java Web 服务等也开始大量进入实际应用领域。为了满足广大参加计算机等级考试的考生和非计算机专业读者学习 Java 语言的需要，我们编写了这本介绍 Java 语言知识的图书。

全书共包括 11 章，大致可划分为四大部分：第一部分为第 1 章至第 3 章，介绍 Java 语言的入门知识，包括语言概述、语言基础、控制语句等；第二部分为第 4 章至第 7 章，介绍了 Java 语言的核心编程知识，主要包括类及其方法、继承与多态、包和接口、异常处理机制等；第三部分为第 8 章至第 11 章，对多线程编程、Applet 编程、输入/输出等作了介绍，并对 Java 中一些常用的工具包和类作了较为详细的说明；第四部分为附录部分，包括最新的二级 Java 考试大纲、全真模拟试卷、习题答案等。全书内容翔实、逻辑清晰、讲解透彻，易于读者快速掌握和深入学习。

本书由杨昭任主编，陈祥、杨丽波、张文科任副主编，参加编写工作的还有杜维兴、黄立超、李强、黄卓、王敬栋、杜波、李鑫、王进、童剑、林晓珊、方春明、王克杰、张晋宝、张勇等。

由于编者水平有限及时间仓促，书中谬误之处实属难免，敬请读者不吝指正，以期日后修订时改进。如果读者在阅读本书的过程中遇到问题或有其他意见和建议，请发电子邮件至：xinyuanxuan@263.net，我们将竭诚为您提供帮助并努力改进今后的工作，奉献给读者高品质的图书。

编　者

2006 年 1 月

目　　录

第 1 章　Java 语言概述

本章与二级 Java 考试大纲内容相关的要点包括：

- 掌握 Java 语言的特点、实现机制和体系结构。
- 掌握 Java 语言中面向对象的特性。
- JDK 的目录结构、API 结构、开发环境设置。

Java 语言是由 Sun 公司于 1995 年推出的一种新的编程语言，它是一种跨平台、适合于分布式计算环境的纯面向对象语言。Java 语言及其扩展正在逐步成为互联网应用的规范，掀起了自 PC 机以来的又一次技术革命。本章主要介绍 Java 语言的起源、特点、简单示例等。

1.1　Java 语言的起源和发展

一般认为，B 语言导致了 C 语言的诞生、C 语言演变出 C++语言，而 Java 语言则明显带有 C++语言的特征。本节将对 Java 语言的起源和发展作简要介绍。

1.1.1　几种典型语言的发展历程

Java 总是和 C++联系在一起，而 C++则是从 C 语言派生而来的，所以 Java 语言继承了这两种语言的大部分特性。Java 的语法是从 C 继承的，Java 的许多面向对象特性都受到 C++的影响。事实上，Java 中几个自定义的特性都来自于或可以追溯到这些前驱语言。略有不同的是，Java 语言完全面向对象，从而摒弃了二者的不足之处。Java 语言的诞生与过去约 30 年中计算机语言的不断改进和发展密切相关。基于这些原因，下面我们将简要介绍一下这个发展历程。

1. 现代编程语言的诞生：C 语言

C 语言产生于 20 世纪 70 年代，它的诞生震撼了整个计算机界。其影响力在于：从根本上改变了以往的编程思路和方法。C 语言的产生是编程人员不断追求程序代码过程化、结构化、高效率、易掌握的直接结果，使用它可以替代汇编语言进行系统程序的开发，在很大程度上提高了编程总体效率。

一般说来，在设计一种计算机编程语言时，经常要从如下几方面进行权衡：① 功能与易用性；② 效率与安全性；③ 稳定与可扩展性。在 C 语言出现以前，程序员们不得不经常在某些方面有优点、但在某些方面又有不少缺陷的语言之间艰难抉择。例如，尽管大家公认的 FORTRAN 语言在科学计算、数值处理领域可以编写出相当高效的程序，但它并不适合于编写系统程序。BASIC 语言虽然容易学习，但功能不够强大，而且也谈不上结构化，这使它在大规模程序中应用的可靠性受到怀疑。当然，使用汇编语言能够编写出各种高效率的程序，但是学习和使用这种低级语言却是不容易的，而且调试汇编程序也比较困难。

另一个复杂的问题是，早期设计的编程语言（如 BASIC，COBOL，FORTRAN 等）没有

考虑结构化设计原则，普遍使用 GOTO 语句作为对程序进行控制的主要方法。这样做的结果是，使用这些语言编写的程序往往成了“意大利面条式的程序代码”，一大堆混乱的“跳蚤”在程序中跳来跳去，跳转语句和条件分支语句使得程序几乎不可能被读懂。Pascal 语言虽然是结构化的语言，但它的设计效率比较低，而且缺少几个必需的特性，因而无法在大的编程范围内使用。特别是，一般的 Pascal 标准语言在某些特定情况下很有用，但将 Pascal 作为系统级编码语言则是不切实际的。

基于以上情况，在 C 语言产生以前，没有任何一种语言能完全满足人们的需要，但人们对这样一种语言的迫切需要是不言而喻的。在 20 世纪 70 年代初期，计算机革命开始了，人们对软件的需求量日益增加，使用早期的编程语言和编程方法进行软件开发根本无法满足这种需求。计算机学术界做出了很多的努力，试图开发出一种更好的计算机语言。毫无疑问，C 语言便是这种努力的结果之一。但是，促使 C 语言诞生的另一个（也许是最重要的）因素是计算机硬件资源的富余所带来的机遇。计算机不再像以前那样被紧锁闺房，程序员们可以随意使用计算机，可以随意进行各种尝试，因而也就有了可以开发适合自己使用的工具的机会。在这种背景下，计算机语言向前飞跃发展，C 语言诞生的时机已经成熟。

在 Dennis Ritchie 第一个发明并实现在 DEC PDP-11 上运行 UNIX 操作系统时，一种更古老的由 Martin Richards 设计的 BCPL 语言导致了 C 语言的产生。受 BCPL 语言的影响，由 Ken Thompson 发明的 B 语言，在 20 世纪 70 年代逐渐向 C 语言发展演变。在此后的许多年里，由 Brian Kernighan 和 Dennis Ritchie 编写的“The C Programming Language”（Prentice-Hall，1978）被认为是 C 语言事实上的标准，该书认为 C 只是支持 UNIX 操作系统的一种语言。1989 年 12 月，美国国家标准化组织（ANSI）制定了 C 语言的标准，C 语言被正式标准化。

许多人认为 C 语言的产生标志着现代计算机语言时代的开始，它成功地处理了长期困扰早期语言的矛盾属性。C 语言是功能强大、高效的结构化语言，简单易学，而且它还包括一个无形的特征：它是程序员自己的语言。在 C 语言出现以前，计算机语言要么被作为学术实验而设计，要么由官方委员会定制。而 C 语言不同，它的设计、实现、开发由真正从事编程工作的程序员来完成，反映了现实编程工作的方法。它的特性被实际运用该语言的人们不断地提炼、测试、思考、再思考，使得 C 语言成为程序员们喜欢使用的语言。这样，C 语言迅速吸引了许多狂热的爱好者，很快受到许多程序员的青睐。简言之，C 语言是由程序员设计并真正由他们自己使用的一种语言。后面将会看到，Java 语言也继承了该思想。

2. 对编程方法的新需要：C++语言

到了 20 世纪 70 年代末及 80 年代初，C 语言成为主流计算机编程语言。时至今日，它仍被广泛地使用着。也许读者会问，既然 C 是一种成功且有用的语言，为什么还需要新的计算机语言？答案就在于复杂性（Complexity）。应用软件越来越工程化、程序代码越来越复杂这一事实贯穿编程语言的整个历程，而 C++语言正是适应了这一需求。下面说明为什么对程序复杂性的更优管理是 C++产生的基本条件。

自从计算机发明以来，编程方法经历了戏剧性的变化。例如，当计算机刚发明出来时，编程是通过面板触发器用人工打孔的办法输入二进制机器指令来实现的。对于只有几百行的程序，这种办法是可行的。随着程序规模的不断增大，人们发明了汇编语言，它通过使用符号来代替机器指令，这样程序员就能处理更大、更复杂的程序。随着程序的进一步增大，高级语言产生了，它给程序员提供了更多的工具来处理复杂性问题。

第一个被广泛使用的高级语言当然是 FORTRAN。尽管 FORTRAN 最初给人留下了深刻的印象，但它无法开发出条理清楚、易于理解的程序。20 世纪 60 年代提出了结构化编程方法。这种结构化的编程思想被像 C 这样的语言所应用，第一次使程序员可以相对轻松地编写适度复杂的程序。然而，当一个工程项目达到一定规模后，即使是使用结构化编程方法，编程人员也无法对其复杂性进行有效的管理。20 世纪 80 年代初期，许多工程项目的复杂度都超过了结构化方法的极限。为解决此问题，面向对象编程（Object-Oriented Programming，OOP）新方法诞生了。简单说来，面向对象的编程是通过使用继承性、封装性、多态性以及重载来帮助组织复杂程序的编程方法。

总之，尽管 C 是当时计算机领域最伟大的编程语言之一，但它处理程序复杂性的能力有限。一旦某个程序的代码超过 25 000～100 000 行，就很难从总体上对它的复杂性进行有效控制了。C++则突破了这个限制，帮助程序员理解并管理规模更大的程序。

1979 年，Bjarne Stroustrup 在新泽西州 Murray Hill 的贝尔实验室工作期间发明了 C++，他最初把这种新语言称为“带类的 C”，1983 年更名为 C++——通过增加面向对象的特性对 C 进行了扩充。由于 C++产生于 C 基础之上，因而它具有了 C 所有的特征、属性和优点。这是 C++作为语言成功的一个关键因素。C++的产生不是企图创造一种全新的编程语言，而是对一种已高度成功语言的改进。C++在 1997 年 11 月被标准化，目前的标准是 ANSI/ISO。

3. 时机的到来：Java 语言的出现

在 20 世纪 80 年代末和 90 年代初，使用面向对象编程的 C++语言占主导地位。在较长一段时间里，程序员们似乎都认为已经找到了一种完美的编程语言，因为 C++既有面向对象的特征，又有 C 语言高效、简洁的优点，因此被作为一种可以广泛应用的编程语言。然而，就像过去一样，信息技术的发展往往是日新月异的。在随后的几年里，Internet 和万维网（WWW）飞速发展，从而促成了 Java 编程语言这一 IT 技术革命。

1.1.2　Java 语言的起源

Java 是由 James Gosling、Patrick Naughton、Chris Warth、Ed Frank 以及 Mike Sheridan 等人于 1991 年在 Sun Microsystems 公司设计出来的，开发第一个版本花了 18 个月时间。该语言最初名叫“Oak”，后来发现“Oak”已经是 Sun 公司另外一种语言的注册商标，于 1995 年更名为“Java”，即太平洋上一个盛产咖啡的岛屿的名字。从 1992 的秋天 Oak 问世，到 1995 春天公开发布 Java 语言，许多人都对 Java 的设计和改进做出了贡献。

说起来多少有些不可思议，Java 的最初推动力并不是因特网，而是源于对体系结构中立语言的需求，这种语言要求能够创建嵌入微波炉、遥控器等各种家用电器设备的软件。1991 年，Sun 公司为了开拓消费类电子产品（如交互式电视、电烤面包箱等）市场，成立了一个开发小组，命名为 Green。该小组的负责人是一位杰出的程序员 James Gosling。在研究开发过程中，Gosling 深刻体会到了消费类电子产品的特点，它们要求高可靠性、低费用、标准化，并且使用简单。为了使整个系统与平台无关，Gosling 开始着手改写 C 编译器。

很快发现，仅仅靠改写 C 编译器是无法满足需求的。用作控制器的 CPU 芯片是多种多样的，但 C 和 C++以及其他绝大多数语言的缺点是只能对特定目标进行编译。尽管为任何类型的 CPU 芯片编译 C++程序是可能的，但这样做需要一个完整的以该种 CPU 为目标的 C++编译器，而创建这种编译器是一项既耗资巨大又耗时很长的工作。因此，需要另外找到一种简单而

经济的解决方案。为此，Gosling 和小组其他人员开始一起致力于开发一种可移植、跨平台的语言，该语言能够生成运行于不同环境、不同 CPU 芯片上的代码。他们的努力最终促成了 Java 的诞生。“一次编写，永远运行”的 Java 程序如图 1-1 所示。

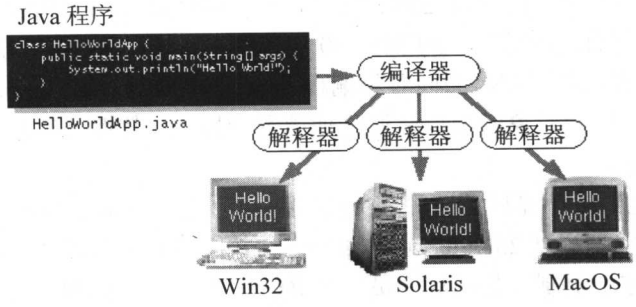

图 1-1 “一次编写，永远运行”的 Java 程序

1.1.3 Java 语言的发展

自从 1995 年被正式推出之后，Java 语言就以其独特的优势迅猛发展，经过短短八九年时间，成为迄今为止最为优秀的面向对象语言。Java 也从当初的一种语言而逐渐形成一种产业，基于 Java 语言的 J2EE 架构已成为微软.NET 平台的强大竞争对手。万维网 WWW 的创始人 Berners-Lee 曾经说过：计算机产业发展的下一个浪潮就是 Java，并且将会很快发生。现在看来，这一预言已成为不争的事实。比尔·盖茨也不无感慨地说过：Java 是长时间以来最卓越的程序设计语言，并确定微软整个软件开发的战略从 PC 单机时代向着以网络为中心的计算时代转移，而毫不犹豫地购买 Java 使用权则是其重大战略决策的实施。Java 最大的特点是跨平台性，将是未来网络世界中的“世界语”。有人预言，今后所有用其他语言编写的软件统统都要用 Java 语言来改写。

当初，Java 语言最初的发布不亚于一场革命，但是它并不标志着 Java 快速革新时代的结束。与其他大多数编程语言、软件系统的小规模改进不同，Java 发布后就继续以爆炸式的速度向前发展。在 Java 1.0 发布后不久，Java 的设计者就已经制定出了 Java 1.1。从版本号看来可能只增加了 0.1，但 Java 1.1 新增的特性远比普通意义上的版本修订有意义，内容也要丰富许多。Java 1.1 增加了许多新的库元素，重新定义了小应用程序处理事件的方法，并且重新设置了 1.0 版中库的许多特性。同时，它也放弃了原来在 Java 1.0 中所定义的若干过时的特征。因此，Java 1.1 不但增加了 Java 1.0 中没有的特性，同时也抛弃了一些原有的特性。

Java 语言的第二个主要发布版本是 Java 2。Java 2 是一个分水岭，它标志着这个语言快速演变时代的开始。Java 2 的第一版本的版本号是 1.2，初看起来这似乎有点奇怪。其实，这是因为它参考了原来 Java 库的版本。对于整个版本来说，它本身没有特别大的变化。Java 2 增加了很多对新特性的支持，例如 Swing 和类集框架，并且提高了 Java 虚拟机和各种编程工具的性能。Java 2 也包含了一些不赞成继续使用的内容，主要是不赞成线程类中 Suspend()、Resume()和 Stop()等这些方法的使用。

Java 的最新版本是 Java 2 的 1.4 版。该版本是对 Java 2 原有版本的一次重要升级。这个版本增强了 Java 大部分现有的功能，并抛弃了部分过时属性。总的来说，版本 1.2、1.3 和版本

1.4 的程序源代码都是兼容的。尽管与前面三个版本相比，版本 1.4 作了一些改变，但这主要是在类库的扩充上。由于本书定位于对 Java 2 语言基础的介绍，因此书中的内容、程序代码对这些版本基本上都是适用的。

1.2　面向对象的程序设计

面向对象的编程思想由来已久，但真正意义上的纯面向对象编程语言目前只有 Java。本节将结合几种高级语言对面向对象程序设计思想进行简要介绍。

1.2.1　面向对象技术的提出

我们知道，所有的计算机程序均由两类元素组成：代码和数据。如何实现这两类元素的有效结合而形成可运行的程序，是多年来程序设计人员所探索的问题。最初，程序的构筑一般围绕“正在发生什么”而编写代码，这种方法被称为面向过程的编程。使用这种方法编写的程序都具有线性执行的特点。面向过程的编程模型可认为是代码作用于数据，像 Pascal、C 这样的过程式语言采用此模型是相当成功的。然而，使用面向过程的方法对小程序的编写可能是比较有效的，但当程序变得非常大且更为复杂时，就会出现种种问题，直至失去对代码的有效控制。由此对软件工程中的编程方法问题提出了新的要求。

为了管理不断增加的复杂性，另外一种编程方式被提了出来，即面向对象的编程（Object-Oriented Programming，OOP）。这种编程方式围绕“谁将受到影响”进行，即以代码的相关数据为核心点进行程序编写。面向对象的编程着眼于它的数据（即对象）和为此数据严格定义的接口来组织程序，程序实际上是用数据控制对代码的访问。这种方式的最大特点是代码与其相关数据被分离开来进行处理，有利于程序规模的扩大，而程序的可维护性得到增强。

1.2.2　面向对象的编程思想

前面提到的面向过程程序，它遵循面向过程的问题求解方法，其中心思想是用计算机能够理解的逻辑来描述和表达待解决的问题及其具体的解决流程。数据结构和算法是面向过程问题求解的核心所在，而面向对象技术则代表了一种全新的程序设计思路，其观察、表述、处理问题的方法与传统的面向过程的编程方法不同。面向对象的程序设计和问题求解力求符合人们日常自然的思维习惯，尽量分解、降低问题的难度和复杂性，从而提高整个求解过程的可监测性、可控制性和可维护性，以此达到以较小代价和较高效率获得较满意效果的目的。

面向对象编程一个实质性的要素是抽象。人们通过抽象（Abstraction）来处理编程过程中遇到的复杂性问题。例如，当看到一辆汽车，人们一般不会把它想象成由几万个互相独立的零件所组成的一套动力装置，而是把整个汽车看成一个具有自己独特行为（停止、启动、运行、加速、减速、换向等）的对象。这种抽象使人们可以很容易地将一辆汽车开到目的地，而不会因组成汽车的各部分零件过于复杂而不知所措。他们可以忽略发动机的工作原理，可以忽略汽车引擎、传动及刹车系统的工作细节，而将汽车作为一个整体来加以利用。

使用层级划分是管理抽象的一个有效方法，它允许根据物理意义将复杂的系统分解为更多更易处理的小块。从外表看，汽车是一个独立的对象。一旦到了内部，你会看到汽车由若干子系统组成：驾驶系统、制动系统、音响系统、安全保障系统、供暖制冷系统、便携电话等。

再进一步细分，这些子系统由更多的专用元件组成。例如，音响系统由一台收音机、一个 CD 播放器或许还有一台磁带放音机组成。从中得到的重要启发是，我们要通过层级抽象对复杂的汽车（或者其他任何复杂的系统）进行管理和控制。

毫无疑问，复杂系统的分层抽象也能被运用于计算机程序设计。在传统的面向对象的程序中，数据经过抽象可以用若干个组成对象表示，程序中的过程步骤则可看成是在这些对象之间进行消息收集。这样，每一个对象都有它自己的独特行为特征。用户可以把这些对象当作具体的实体，让它们对告诉它们做什么事的消息作出反应。这便是面向对象编程的本质。

面向对象的概念是 Java 的核心。对程序员来讲，重要的是要理解怎么将这些概念转化为程序代码。在软件工程理论中，任何软件都不可避免地要经历概念提出、成长、形成、衰老这样一个生命周期。采用面向对象的程序设计方法，可以使软件在生命周期中的每一个阶段都处变不惊，有足够的应变能力。例如，一旦用户定义好了某个对象以及指向这些对象的有效的、可靠的接口，就可以很从容自信地解除或更替旧系统中的某些组成部分，而对整体不产生重大影响。这正是面向过程的程序设计方法所不容易做到的。

最早具有面向对象特征的程序设计语言是于 1966 年推出的 Simula I。而于 1980 年提出的 Smalltalk-80 已经是一种比较成熟有效的面向对象的语言了。其后，先后产生了 Lisp、Visual Basic、Eiffel、Object Pascal、C++等多种面向对象程序设计语言。目前，在使用上最成功的面向对象语言是在 C 语言基础上发展起来的 C++语言和 90 年代新出现的纯面向对象程序设计语言 Java。Java 的核心是面向对象编程。事实上，所有的 Java 程序都是面向对象的，用户别无选择。这一点与 C++不同，因为在 C++里用户可以选择使用面向对象编程，也可以不选择面向对象编程。Java 则与面向对象编程密不可分，因而哪怕是编写最简单的 Java 程序，也必须理解面向对象编程的基本特征和原则。

1.2.3 面向对象编程的基本原则

一般说来，面向对象的系统至少需具备三大特性：封装性、继承性、多态性。将封装、继承、多态（包括重载）等面向对象方法应用于程序的开发工具和开发过程中不仅可以加快开发的速度，还可极大地增强程序的可维护性和可扩展性，提高代码重用率。因此，在面向对象编程过程中需要遵循这三项原则。

1. 封装性

封装（Encapsulation）是将代码及其处理的数据绑定在一起的一种编程机制，该机制保证了程序和数据都不受外部干扰且不被误用。我们知道，一个对象的基本要素包括属性和作用在属性上的操作（方法或事件）。对象的使用实现了数据抽象，它将一组数据和对这组数据的操作结合成一个内在的整体，不允许外界对这组数据任意进行访问，这里就用到了封装的原理。封装的目的是为了实现数据隐藏和数据保护，为对象提供一个对外操作的接口，外界只能从对象所提供的操作接口来认识和操作该对象。

理解封装性的一个方法就是把它想象成一个黑匣子，它可以阻止外部定义的代码随意访问内部的代码和数据。对黑匣子内部代码和数据的访问通过一个适当定义的接口严格控制。如果想与现实生活中的某个事物作对比，可考虑汽车上的自动传送。自动传送中包含了有关引擎的几百个比特的信息，例如你正在以什么样的加速度前进、行驶路面的坡度如何，以及目前的档位情况。作为驾驶员，你影响这个复杂封装的方法只有一个：移动档位传动杆。例如，你不