项目 1　Java 语言概述

Java 语言是由 Sun 公司于 1995 年 5 月 23 日正式推出的纯面向对象的程序设计语言，是当前最流行的一种网络编程语言，它推出不久，就获得了极大的成功。这是因为 Java 与传统的计算机语言相比，更简洁、更安全，易学易用，并独具特色，本项目主要介绍 Java 语言的发展史及特点，然后介绍 Java 虚拟机工作原理和 Java 的开发环境搭建，最后介绍面向对象程序设计的基本概念。

- 了解 Java 的发展与特点
- 理解 Java 的平台无关性
- 掌握 JDK 的安装和环境变量的配置
- 理解面向对象程序设计的基本概念
- 掌握 Java 程序的开发流程

任务 1　Java 发展史与特点

Java 语言的产生背景，如同它的名字一样，散发着淡淡的咖啡香气。Java 的特点是：简单、安全、可移植、面向对象、健壮、多线程、结构中立、解释性和高性能、分布式、动态。

1.1.1　Java 的起源与发展

1. Java 的起源

1991 年，Sun Microsystem 公司的 James Gosling、Bill Joe 等人，为了解决家用消费类电子产品智能化过程中的控制和通信问题，设计出了一个新的软件。这个软件是 James 等人在充分研究了传统计算机语言，尤其是 C/C++语言的特点后，设计出的一种适合开发跨平台嵌入式软件的语言，当时命名为 Oak，这就是 Java 的前身。但由于智能家用电器并未像预期那样快速发展，同时 Sun 公司参与投标的一个交互式电视系统的商业项目也未获成功，Oak 面临夭折。但此时，国际互联网的应用却高速发展，Sun 公司看到了 Oak 的跨平台特性在计算机网络应用方面将大有可为。于是他们对 Oak 系统进行了结构和功能上的改造并加以扩充，将 Oak 重新命名为 Java。

注：印度尼西亚有一个重要的盛产咖啡的岛屿叫 Java，中文译名为爪哇，开发小组在一次聚会中，从咖啡获得灵感想到了“Java”这个名字。

2. JDK1.0 发布

1995 年，Sun 推出的 Java 只是一种语言，而要想开发复杂的应用程序，必须要有一个强大的开发库支持才行。

Sun 在 1996 年 1 月 23 日发布了 JDK1.0。这个版本包括了两部分：运行环境（即 JRE）和开发环境（即 JDK）。在运行环境中包括了核心 API、集成 API、用户界面 API、发布技术、Java 虚拟机（JVM）五个部分。而开发环境还包括了编译 Java 程序的编译器（即 javac）。在 JDK1.0 时代，Java 库比较单薄，不够完善。随着 JDK 的逐步升级，它为开发人员提供了一个强大的开发支持库。

3. Java 2 问世

1998 年 12 月 4 日，Sun 发布了 Java 历史上最重要的一个 JDK 版本：JDK1.2。这个版本标志着 Java 已经进入 Java 2 时代。这个时期也是 Java 飞速发展的时期。

在 Java 2 时代 Sun 对 Java 进行了很多革命性的变化，而这些革命性的变化一直沿用到现在，对 Java 的发展形成了深远的影响。

JDK1.2 自从被分成了 J2EE、J2SE 和 J2ME 三大块，得到了市场的强烈反响。

- J2EE 是 Java 2 的企业版（Java 2 Platform Enterprise Edition），主要用于构建企业级分布式应用系统，定义了企业应用 Java 组件、JSP/Servlet、JDBC、Java 命名和目录服务接口等。这些技术也是 Java 包中的类和接口来实现的，只是有的包已包含在 J2EE 中，有的包还要单独下载和安装。
- J2ME 是 Java 2 微缩版（Java 2 Platform Micro Edition），主要适用于 PDA、移动电话、机顶盒设备及无线通信类电子产品中的 Java 应用软件的开发。微缩版的含义表示其构成相对简单。
- J2SE 是 Java 2 标准版（Java 2 Platform Standard Edition），主要用于桌面操作系统环境下开发 Java 程序。除包含微缩版中的三个包以外，还有 java.math、java.net 和 java.text 等三个包，它们定义了 Java 独立应用程序 Application，和在浏览器中运行的小应用程序 Applet 的开发过程中可应用的类和接口，这是 Java 的核心包。

2000 年 5 月 8 日，Sun 对 JDK1.2 进行了重大升级，推出了 JDK1.3。

2002 年 2 月 13 日，Sun 发布了 JDK 历史上最为成熟的版本 JDK1.4。

4. Java 5.0 发布

2004 年 9 月 30 日，Sun 发布了 JDK1.5，成为 Java 语言发展史上的又一个里程碑。为了表示该版本的重要性，Sun 将版本号 1.5 改为 5.0，就是预示着 J2SE 5.0 较以前的 J2SE 版本有着很大的改进。

在 Java 5.0 中，主要包含以下主要新特性：

- 泛型
- 增强 for 语句
- 可变数目参数
- Annotation 注解
- 自动拆箱和装箱
- 类型安全的枚举
- 静态导入

5. Java 8.0 发布

2014 年 4 月 24 日，Oracle 的 Java 开发团队终于发布了 Java 8 正式版本。Java 8 版本最大的改进就是 Lambda 表达式，其目的是使 Java 更易于为多核处理器编写代码；其次，新加入的 Nashorn 引擎也使得 Java 程序可以和 JavaScript 代码互操作；再者，新的日期时间 API、GC 改进、并发改进也相当令人期待。

注：2009 年 4 月，Oracle 公司以 74 亿美元收购 Sun 公司。

1.1.2　Java 的特点

Java 语言是一种跨平台、适合于分布式计算环境的面向对象程序设计语言。具体来说，它具有平台无关、简单、面向对象、分布式、健壮、多线程、安全及多态性等特点。

1. 平台无关性

Java 有一句著名的口号：“一次编程，四处应用。”Java 可以跨平台使用，这是因为 Java 编译器把 Java 源程序编译成二进制代码后并不是像其他计算机语言那样生成可执行文件后直接执行，而是生成了一种后缀名为.class 的字节码文件。运行时，由一种被称为 Java 虚拟机的运行环境，针对不同的处理器指令系统，把字节码转换为不同的具体指令，然后再执行。所以，Java 程序能够运行安装了 Java 虚拟机的任意类型的操作系统中，如 Linux、Solaris、Windows 95/98/2000/NT/XP 等，实现了平台无关。因而，特别适合于网络应用。

平台无关的特性使 Java 程序可以方便地被移植到网络上的不同机器。同时，Java 的类库中也实现了与不同平台的接口，使这些类库可以移植。另外，Java 编译器是由 Java 语言实现的，这使得 Java 系统本身也具有可移植性。

2. 简单

Java 语言的设计目标是作为一种小型的嵌入式控制语言以解决智能家用电器设备的控制与通讯，随着 Internet 的发展，Java 的目标又定位于运行在浏览器中的小程序（Applet）。所以，Java 语言的出发点就是容易编程，使用简洁。它与 C/C++语言类似，但却要简单得多。它抛弃了 C/C++中的指针、结构体等概念；没有多继承、运算符重载等容易使人迷惑的规则；也没有预处理器；程序员也不必自己释放占用的内存空间，系统将自动回收。因而 Java 程序设计中不用考虑无用内存的释放和无用对象的删除等，极大减少了内存冲突的问题。

Java 虽然语法简单，但功能却十分强大。使用它也可以设计出非常复杂的系统。如构建分布式系统的中间件 Jini，就是完全用 Java 开发的。

3. 面向对象

Java 是纯面向对象的语言。面向对象是把现实世界中的物体（对象），用属性和行为来描述，然后对应到计算机编程中。如汽车，其颜色、形状等是属性；它的功能如加速、刹车等，则是行为；整个汽车就是类，类是抽象的描述，或者说是泛指，具有共性。而具体是特指，如一辆红色面包车则是对象。即对象是类的实例。对应到计算机编程上，属性就是数据，常用变量或常量来表达。行为就是能实现一定功能的一段程序代码，常叫作方法。也就是说，我们用属性与方法就能够描述一个对象。如果抽象地描述同一类型的对象，这就是类。

定义类，是 Java 编程的第一步。在定义类时可以继承原有的一个类，这样新的类就继承

了原有类中的属性与方法，新类的对象就可以在编程中直接调用原有类中的方法，从而实现了代码重用。这是面向对象的一大特征，即继承。封装和多态则是面向对象的另外两大特征。封装就是将对象所具有的描述共性的属性和行为或者说是数据和代码封装于类中，利用类的优点，实现程序的简洁性和易维护性。多态则是指一个接口，可以有多个实现形式，即完成多个功能。

面向对象编程就是定义类与对象，并利用消息触发机制将事件与方法联系起来，它支持封装、多态和继承。

4. 分布式

Java 允许将其编译后的二进制代码分布于网络上。它提供了大量的系统模块支持基于 TCP/IP 协议的编程，这使 Java 建立网络连接十分容易。Java 应用程序可以通过 URL 来访问网络资源，如寻找应用程序所需的类，这同在本地机上寻找一样方便、快捷。

5. 健壮性

Java 在编译和运行程序时，都要对可能出现的问题进行检查，以消除错误的产生。它提供自动垃圾收集来进行内存管理，防止程序员在管理内存时容易产生的错误。通过集成的面向对象的异常处理机制，在编译时，Java 提示出可能出现但未被处理的异常，帮助程序员正确地进行选择以防止系统的崩溃。另外，Java 在编译时还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. 多线程

Java 支持多线程。线程是程序运行的最小单位。所谓多线程，即计算机可以同时运行多个程序段，各自完成不同的任务。这样极大地提高了程序执行的效率和处理能力。Java 提供了有关线程的操作，线程的创建，线程的管理，线程的结束等处理。Java 自身也是多线程的，它可以利用系统的空闲进行一些常规处理。如 Java 虚拟机启动后，就一直运行着一个线程，在后台负责对不在使用的对象的垃圾进行回收工作，即自动释放内存。当然，该线程的优先级最低。

7. 安全

所有 Java 语言对内存进行的访问都是通过对象实例变量实现的。程序运行时，内存由操作系统分配，这样可以防止病毒通过指针侵入系统。Java 对程序提供了安全管理器，使用了公开密钥加密机制，防止程序的非法访问。

8. 动态性

Java 语言的设计目标之一是适应于动态环境的变化。即 Java 程序运行时，是动态地进行加载，当程序运行到要找所需的类时，可在本机上寻找，也可在网上寻找，找到后再动态地加载。这保证了对类库支持的一致性，类库升级也可不必重新编译程序。

习题

1．Java 语言的显著特点是什么？

2．Java 平台由哪几部分组成？各有什么作用。

任务 2　Java 程序工作原理

1.2.1　Java 虚拟机

Java 虚拟机是软件模拟的计算机，可以在任何处理器上（无论是在计算机中还是在其他电子设备中）安全并且兼容地执行保存在.class 文件中的字节码。Java 虚拟机的“机器码”保存在.class 文件中，有时也可以称之为字节码文件。Java 程序的跨平台主要是指字节码文件可以在任何具有 Java 虚拟机的计算机或者电子设备上运行，Java 虚拟机中的 Java 解释器负责将字节码文件解释成为特定的机器码进行运行。Java 源程序需要通过编译器编译成为.class 文件（字节码文件），Java 程序的编译和执行过程如图 1-1 所示。

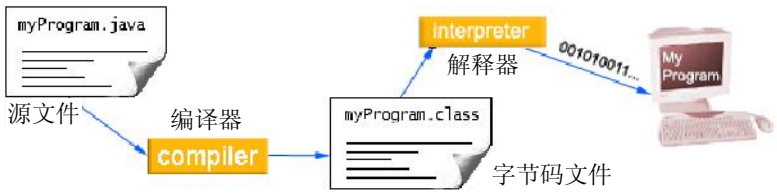

图 1-1　Java 程序编译过程

但是，Java 虚拟机的建立需要针对不同的软硬件平台做专门的实现，既要考虑处理器的型号，也要考虑操作系统的种类。目前在 SPARC 结构、X86 结构、MIPS 和 PPC 等嵌入式处理芯片上、在 UNIX、Linux、Windows 和部分实时操作系统上都有 Java 虚拟机的实现，如图 1-2 所示。

图 1-2　不同平台 Java 程序实现过程

在 Java 虚拟机中，是利用 Java 解释器对编译后的字节码文件进行解释执行的，由解释器对在不同计算机系统中建立的与内存布局对应的字节码文件进行内存映射。解释器对 Java 字节码文件的解释执行通常分为以下三步：

（1）装入代码：由解释器申请内存空间，读入字节码文件进行分析，建立内存布局。其具体功能是由类装载器来完成的。

（2）校验代码：由字节码校验器检验内存布局，发现错误。

（3）运行代码：解释读取字节码含义，执行代码，完成相应功能。

计算机语言的执行方式一般分为解释执行和编译执行两种。Java 虚拟机自推出以来，一直是对字节码文件采用的解释执行方式，因而运行速度较慢，这也曾是 Java 的缺点。但是现在，Sun 公司已经解决了这一问题，推出了 Java 字节码编译器，大大提高了 Java 的执行速度。

1.2.2 垃圾回收机制

在 C++中，对象所占的内存在程序结束运行之前一直被占用，在明确释放之前不能分配给其他对象；而在 Java 中，当没有对象引用指向原先分配给某个对象的内存时，该内存便成为垃圾。JVM 的一个系统级线程会自动释放该内存块。垃圾回收意味着程序不再需要的对象是“无用信息”，这些信息将被丢弃。当一个对象不再被引用的时候，内存回收它占领的空间，以便空间被后来的新对象使用。

事实上，除了释放没用的对象，垃圾回收也可以清除内存记录碎片。由于创建对象和垃圾回收器释放丢弃对象所占的内存空间，内存会出现碎片。碎片是分配给对象的内存块之间的空闲内存洞。碎片整理将所占用的堆内存移到堆的一端，JVM 将整理出的内存分配给新的对象。

垃圾回收能自动释放内存空间，减轻编程的负担。这使 Java 虚拟机具有一些优点。首先，它能使编程效率提高。在没有垃圾回收机制的时候，可能要花许多时间来解决释放无用内存的问题。在用 Java 语言编程的时候，靠垃圾回收机制可大大缩短时间。其次是它保护程序的完整性，垃圾回收是 Java 语言安全性策略的一个重要部分。

垃圾回收机制是一个系统级线程，它给程序员带来好处的同时，也存在着影响系统性能问题，因为它要必须追踪运行程序中有用的对象，而且最终释放没用的对象。这一个过程需要花费处理器的时间。其次垃圾回收算法的不完备性，早先采用的某些垃圾回收算法就不能保证100%收集到所有的废弃内存。当然随着垃圾回收算法的不断改进以及软硬件运行效率的不断提升，这些问题都可以迎刃而解。

程序员可以调用 System.gc()这个方法通知 Java 虚拟机释放无用资源，但 Java 虚拟机会选择在合适的时候释放无用资源，具体释放的时间，不是程序员调用 System.gc()的时刻，而是 Java 虚拟机决定的，程序员不能精确控制和干预。

习题

1．简述 Java 虚拟机的工作原理。

2．请描述什么是 Java 的垃圾回收机制。

任务 3　面向对象基础

Java 语言是一个面向对象的程序设计语言，因此在系统学习 Java 语言之前，首先需要对面向对象的程序设计思想有一个了解，在以后学习过程中将不断加深理解并掌握面向对象的方法。

1.3.1 什么是面向对象程序设计

面向对象程序设计是一种新的程序设计范型（Paradigm）。程序设计范型是指设计程序的规范、模型和风格，它是一类程序设计语言的基础。一种程序设计范型体现了一类语言的主要特征，这些特征能用以支持应用领域所希望的设计风格。不同的设计范型有不同的程序设计技术和方法学。

面向过程程序设计范型是使用较广泛的程序设计范型，这种范型的主要特征是，程序由

过程定义和过程调用组成，即**程序=过程+调用**。基于面向过程程序设计范型的语言称为面向过程性语言，如 C、PASCAL、Ada 等都是典型的面向过程性语言。函数式程序设计范型也是较为流行的程序设计范型，它的主要特征是，程序被看作“描述输入与输出之间关系”的数学函数。LISP 是支持这种范型的典型语言。除了面向过程程序设计范型和函数式程序设计范型外，还有许多其他的程序设计范型，如模块程序设计范型（典型语言是 Modula）、逻辑式程序设计范型（典型的语言是 PROLOG）、进程式程序设计范型、类型系统程序设计范型、事件程序设计范型、数据流程程序设计范型等。

面向对象程序设计是一种新型的程序设计范型。这种范型的主要特征是：

程序=对象+消息

面向对象程序的基本元素是对象，面向对象程序的主要结构特点是：第一，程序一般由类的定义和类的使用两部分组成，在程序中定义各对象并规定它们之间传递消息的规律。第二，程序中的一切操作都是通过向对象发送消息来实现的，对象接收到消息后，启动有关方法来完成相应得操作。一个程序中涉及到的类，可以由程序设计者自己定义，也可以使用现成的类（包括类库中为用户提供的类和他人已构建好的）。尽量使用现成的类，是面向对象程序设计范型所倡导的程序设计风格。

需要说明的是，某一种程序设计语言不一定与一种程序设计范型相对应。实际上存在有具备两种范型或多种范型的程序设计语言，即混合型语言。例如 C++就不是纯粹的面向对象程序设计范型，而是面向过程程序设计范型和面向对象程序设计范型的混合型程序设计语言。

1.3.2　面向对象的基本概念

1. 对象

首先需要搞清楚的第一问题是，什么是对象？对象具有两方面含义，即在现实生活中的含义和在计算机世界中的含义。

在我们所生活的现实世界中，“对象”无处不在。在我们身边存在的一切事物都是对象，例如一粒米、一本书、一个人、一所学校，甚至一个地球，这些都是对象。除去这些可以触及的事物是对象外，还有一些无法整体触及的抽象事件，例如一次演出、一场球赛、一次借书，也都是对象。

一个对象既可以非常简单，又可以非常复杂，复杂的对象往往可以是由若干个简单对象组合而成的。

所有这些对象，除去它们都是现实世界中所存在的事物之外，它们都还是有各自不同的特征。例如一粒米，它首先是一粒米这样一个客观存在。再例如一个人，首先他是一个客观实体，具有一个名字来标识，其次它具有性别、年龄、身高、体重等这些体现他自身状态的特征；再其次他还具有一些技能，例如会说英语、会修电器等。

通过上面的这些举例我们可以对“对象”下一个定义，即对象是现实世界中的一个实体，它具有如下特性：

- 有一个名字以区别于其他对象。
- 有一个状态用来描述它的某些特征。
- 有一组操作，每一个操作决定对象的一种功能或行为。
- 对象的操作可分为两类：一类是自身承受的操作，一类是施加于其他对象的操作。

在面向对象程序设计中，对象是描述其属性的数据以及对这些数据施加的一组操作封装在一起构成的统一体。对象可以认为是：数据+操作。对象所能完成的操作表示它的动态行为，通常也把操作称为方法。

为了帮助读者理解对象的概念，图 1-3 形象地描述了具有 3 个操作的对象。

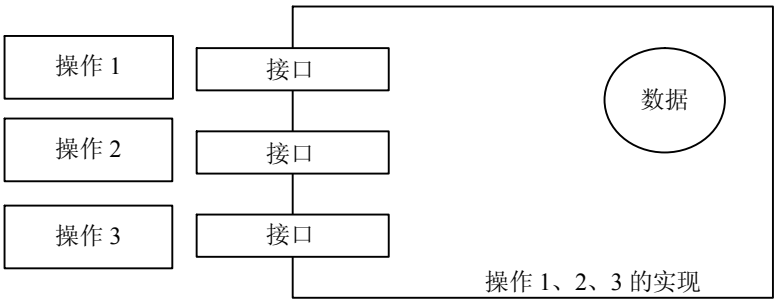

图 1-3　具有 3 个操作的对象

下面我们用一台录音机比喻一个对象，通俗地说明对象的某些特点。

录音机上有若干按键，如 Play（播放）、Rec（录音）、Stop（停止）、Rew（倒带）等，当人们使用录音机时，只要根据自己的需要如放音、录音、停止、倒带等按下与之对应的键，录音机就会完成相应的工作。这些按键安装在录音机的表面，人们通过它们与录音机交互。我们无法操作录音机的内部电路，因为它们被装在机壳里，录音机的内部情况对于用户来说是隐蔽的，不可见的。

一个对象很像一台录音机，当在软件中使用一个对象的时候，只能通过对象与外界的接口操作它。对象与外界的接口也就是该对象向公众开放操作。使用对象向公众开放的操作就好像使用录音机的按键，只需知道该操作的名字（如录音机的键名）和所需要的参数（用于提供附加信息或设置状态，好像听录音前先装录音带并将录音带转到指定位置），根本无需知道实现这些操作的方法。事实上，实现对象操作的代码和数据是隐藏在对象内部的，一个对象好像是一个黑盒子，表示它内部的数据和实现各个操作的代码，都被封装在这个黑盒子内部，在外面是看不见的，更不能从外面去访问或修改这些数据或代码。

使用对象时只需知道它向外界提供的接口形式而无需知道它的内部实现算法，不仅使得对象的使用变得非常简单、方便，而且具有很高的安全性和可靠性。可见面向对象程序设计中的对象来源于现实世界，更接近人们的思维。

2. 类

在现实世界中，“类”是一组相同属性和行为的对象的抽象。例如，张三、李四、王五……，虽然每个人的性格、爱好、职业、特长等各不同，但是他们的基本特征是相似的，都具有相同的生理构造，都能吃饭、说话、走路等，于是把他们统称为“人”，而具体的每一个人是人类的一个实例，也就是一个对象。

类和对象之间的关系是抽象和具体的关系。类是多个对象进行综合抽象的结果，一个对象是类的一个实例。例如“狗”是一个类，它是由千千万万个具体的不同狗抽象而来的一般概念。同理，鸡、鸭、牛、羊等都是类。

类在现实世界中并不真正存在。例如，在地球上并没有抽象的“人”，只有一个个具体的

人，如张三、李四、王五……。同样，世界上没有抽象的“学生”，只有一个个具体的学生。

面向对象程序设计中，“类”就是具有相同的数据和相同的操作的一组对象的集合，即类是对具有相同数据结构和相同操作的一类对象描述。例如，“学生”类可以由学号、姓名、性别、成绩等表示其属性的数据项和对这些数据的录入、修改和显示等操作组成。

在面向对象中，总是先声明类，再由类生成对象。类是建立对象的“模板”，按照这个模板所建立的一个个具体的对象，就是类的实际例子，通常称为实例。比如，手工制作月饼时，先雕刻一个有凹下图案的木模，然后在木模上抹油，接着将事先揉好的面塞进木模里，用力挤压后，将木模反扣在桌上，一个漂亮的图案就会出现在月饼上了。这样一个接着一个的，就可以制造出外形一模一样的月饼。这个木模就好比是“类”，制造出来的月饼好比是“对象”。

3. 消息

现实世界中的对象不是孤立存在的实体，他们之间存在着各种各样的联系，正是它们之间的相互作用、联系和连接，才构成了世间各种不同的系统。同样，在面向对象程序设计中，对象之间也需要联系，我们称为对象的交互。面向对象程序设计必须提供一种机制允许一个对象与另一个对象的交互。这种机制叫消息传递。即对象之间进行通信的结构叫做消息。在对象的操作中，当一个消息发送给某个对象时，消息包含接收对象去执行某种操作的信息。发送一条消息至少要包括说明接受消息的对象名、发送给该对象的消息名（即对象名、方法名）。一般还要对参数加以说明，参数可以是认识该消息的对象所知道的变量名，或者是所有对象都知道的全局变量名。

在面向对象程序设计中的消息传递实际是对现实世界中的信息传递的直接模拟。以实际生活为例，我们每一个人可以为他人服务，也可以要求他人为自己服务。当我们需要别人为自己服务时，必须告诉他们我们需要的是什么服务，也就是说，要向其他对象提出请求，其他对象接到请求后，才会提供相应的服务。

一般情况下，我们称发送消息的对象为发送者或请求者，接收消息的对象为接收者或目标对象。对象中的联系只能通过消息传递来进行。接收对象只有在接收到消息时，才能被激活，被激活的对象会根据消息的要求完成相应的功能。

消息具有三个性质：

- 同一对象可接收不同形式的多个消息，产生不同的响应。
- 相同形式的消息可以送给不同对象，所做出的响应可以是截然不同的。
- 消息的发送可以不考虑具体的接收者，对象可以响应消息，也可以对消息不予理会，对消息的响应并不是必须的。

在面向对象程序设计中，消息分为两类：即公有消息和私有消息。到底哪些消息是公有消息，哪些消息是私有消息，需要有一个明确的规定。若有一批消息同属于一个对象，其中有一部分是由外界对象直接向它发送的，称之为公有（public）消息；还有一部分则是它自己向本身发送的，这些消息是不对外开放的，外界不必了解它，称之为私有（private）消息。

4. 方法

在面向对象程序设计中，要求某一个对象做操作时，就向该对象发送一个相应的消息，当对象接收到发向它的消息时，就调用有关方法，执行相应的操作。方法就是对象所能执行的操作。方法包括界面和方法体两部分。方法的界面也就是消息的模式，它给出方法的

调用协议；方法体则是实现某种操作的一系列计算步骤，也就是一段程序。消息和方法的关系是：对象根据接收到消息，调用相应的方法；反过来，有了方法，对象才能响应相应的消息。所以消息模式与方法界面应该是一致的。同时，只要方法界面保持不变，方法体的改动不会影响方法的调用。

1.3.3 面向对象编程的特征

面向对象的程序设计方法与传统方法相比较有着自身鲜明的特点，主要概况为抽象、封装、继承和多态四大基本特征。

1. 抽象

抽象就是忽略一个主题中与当前目标无关的那些方面，以便更充分地注意与当前目标有关的方面(就是把现实世界中的某一类东西提取出来,用程序代码表示,一般叫做类或者接口)。抽象并不打算了解全部问题，而只是选择其中的一部分，暂时不用部分细节。抽象包括两个方面，一是数据抽象，二是过程抽象。

数据抽象就是用代码的形式表示现实世界中一类事物的特性，就是针对对象的属性。比如建立一个鸟这样的类，鸟都有以下属性：一对翅膀、两只脚、羽毛等。抽象出来的类都是鸟的属性，或者成员变量。

过程抽象就是用代码形式表示现实世界中事物的一系列行为，就是针对对象的行为特征。比如鸟会飞、会叫等。抽象出来的过程一般都是鸟的方法。

2. 封装

封装性就是尽可能地隐藏对象内部细节，对外形成一道边界，只保留有限的接口和方法与外界进行交互。封装的原则是使对象以外的部分不能随意地访问和操作对象的内部属性，从而避免了外界对对象内部属性的破坏。

Java 使用类机制体现其封装性，可以通过对类的成员设置一定的访问权限，实现类中成员的信息隐藏。在实际开发中，类用来构建软件系统的模块，模块之间只能通过接口进行交互，使它们的耦合和交叉大大减少，从而降低开发过程的复杂性，减少可能的错误，使得 Java 程序具有良好的可维护性。

3. 继承

一个类继承另一个类，继承者可以获得被继承类的所有方法和属性，并且可以根据实际的需要添加新的方法或者对被继承类中的方法进行覆写，被继承者称为父类或者超类，继承者称为子类或导出类，继承提高了程序代码的可重用性，Java 中一个子类只能继承一个父类，Object 类是所有类的最终父类。一个父类可以被多个子类继承，此时父类是所有子类的公共属性和方法的集合,而每个子类则是父类的特殊化,是在公共属性和方法的基础上的扩展和延伸，子类是可以定义属于自己的特有的属性和方法。

4. 多态

对象的多态性是指在父类中定义的属性或方法被子类继承之后，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方法在父类及其各个子类中具有不同的语义。例如：“几何图形”作为父类有“绘图”方法，“椭圆”和“多边形”都是“几何图形”的子类，其“绘图”方法功能与父类不同。

Java 的多态性体现在两个方面，一个是由方法重载实现的静态多态性（编译时多态），另

一个是方法重写实现的动态多态性（运行时多态）。

编译时多态：在编译阶段，具体调用哪个被重载的方法，编译器会根据参数的不同来静态确定调用相应的方法。

运行时多态：由于子类继承了父类所有的属性（私有的除外），所以子类对象可以作为父类对象使用。程序中凡是使用父类对象的地方，都可以用子类对象来代替。一个对象可以通过引用子类的实例来调用子类的方法。

习题

1．面向对象编程与面向过程编程有哪些不同？
2．什么是类？什么是对象？两者关系是什么？
3．面向对象编程四个基本特征是什么？
4．什么是继承？什么是多态？

任务 4　Java 的开发和运行环境

如果要开发 Java 应用程序，必须建立 Java 开发环境。Java 语言自身提供了开发环境 JDK（Java Development Kit，Java 软件开发工具集），当前的最新版本是 JDK 8.0。之前的版本可以使用，本书使用 JDK 版本相对稳定的 JDK 7.0 版本。

1.4.1　JDK 简介

JDK 是一种用于构建 Java 平台上编译和发布 Java 程序的开发和运行环境。它由两部分组成，下层是处于操作系统层之上的运行环境，上层由编译工具、调试工具和运行 Java 应用程序所需的工具组成，其核心 Java API 是一些预定义的类库，开发人员需要用这些类来访问 Java 语言的预定义的功能。

JDK 所包含的工具当中，主要有以下几种：

- javac：Java 编译器，将 Java 源代码转换成字节码。
- java：Java 解释器，直接从类文件执行 Java 应用程序字节代码。
- appletviewer：小程序浏览器，一种执行 HTML 文件上的 Java 小程序的 Java 浏览器。
- javadoc：根据 Java 源码及说明语句生成 HTML 文档。
- jdb：Java 调试器，可以逐行执行程序，设置断点和检查变量。
- javah：产生可以调用 Java 过程的 C 过程，或建立能被 Java 程序调用的 C 过程的头文件。
- javap：Java 反汇编器，显示编译类文件中的可访问功能和数据，同时显示字节代码含义。

JDK 包含以下常用类库：

- java.lang：系统基础类库，其中包括字符串类 String 等。
- java.io：输入输出类库，例如进行文件读写需要用到。
- java.net：网络相关类库，例如进行网络通信会用到其中的类。
- java.util：系统辅助类库，编程中经常用到的集合属于这个类库。

- java.sql：数据库操作类库，连接数据库、执行 SQL 语句、返回结果需要用到该类库。
- java.servlet：JSP、Servlet 等使用到的类库，是 Java 后台技术的核心类库。

1.4.2 JDK 的安装

首先，在 Oracle 公司的官方网站（www.oracle.com）下载 JDK 的安装包，根据自己电脑的操作系统选择正确的版本下载。另外，JDK 的使用也不是版本越高越好。在企业级开发中，通常一个项目中的开发人员统一使用一个稳定版本的 JDK，避免因为各版本 JDK 的差异带来问题。

JDK 下载结束后，用鼠标左键双击 JDK 安装包，系统将显示 JDK 安装向导画面。JDK 的安装过程很简单，一直单击“下一步”按钮即可。默认情况下，JDK 及其所包含的 JRE 将被安装到 C:\Program Files\Java 文件夹中。将 JDK 安装到默认目录下后，会形成如图 1-4 所示的目录结构。

图 1-4 JDK 目录结构

JDK 目录的主要内容如下：

- bin：存放 javac、java、appletviewer 等命令程序。
- db：包含使用嵌入式数据库 Derby 开发所需要的资源及一些案例。
- include：存放与 C 程序相关的头文件。
- lib：附加库，是开发工具所需的其他类库和支持文件。

1.4.3 JDK 配置

在使用 Java 来编译与运行程序之前，必须先设置系统环境变量。所谓系统环境变量就是在操作系统中定义的变量，可供操作系统上的所有应用程序使用。为此，需要设置三个环境变量：JAVA_HOME、Path 和 CLASSPATH。

1. JAVA_JOME 环境变量配置

在 Windows 7 系统中，右击“计算机”图标，选择“属性”→“高级系统设置”→“环境变量”命令，打开对话框如图 1-5 所示。在“系统变量(S)”中新建 JAVA_HOME 环境变量，“变量值”输入“C:\Program Files\Java\jdk1.7.0_75”，如图 1-6 所示。

2. Path 环境变量配置

在图 1-5 所示“环境变量”对话框中的“系统变量(S)”中找到变量 Path，然后单击“编辑”按钮，打开“编辑系统变量”对话框。在“变量值”编辑框最前面加上“%JAVA_HOME%\bin;%JAVA_ HOME%\jre\bin;”，如图 1-7 所示。

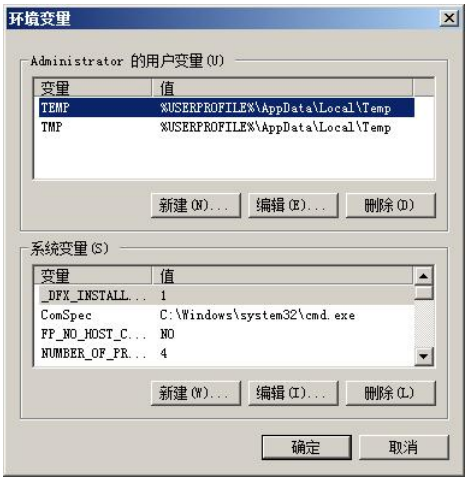

图 1-5　环境变量对话框

图 1-6　新建环境变量

图 1-7　编辑环境变量

3. CLASSPATH 环境变量配置

与新建 JAVA_HOME 环境变量类似，新建 CLASSPATH 环境变量，“变量值”输入“.;%JAVA_HOME%\lib;%JAVA_HOME%\lib\tools.jar”（注意最前面有一点号），如图 1-8 所示。

图 1-8　新建环境变量

4. 验证 JDK 是否成功

单击“开始”按钮，在“搜索程序和文件”框中输入“cmd”，按“Enter”健，打开仿真 DOS 窗口，输入“java –version”命令，如果能看到如图 1-9 所示的显示版本信息，则说明安装和配置成功。

图 1-9　JDK 显示版本信息

1.4.4 第一个 Java 程序

1. 编辑 Java 程序

JDK 中没有提供 Java 编辑器，需要使用者自己选择一个方便易用的编辑器或集成开发工具。Java 程序的源程序是以“.java”为后缀的文本文件。作为初学者，可以使用记事本、EditPlus、UltraEdit 作为 Java 编辑器，编写第一个 Java 程序。下面以记事本为例，使用它编写 HelloWorld 程序。

打开“记事本”，按照示例程序 1.1 输入代码（注意大小写和缩进结构），完成后将其保存为 HelloWorld.java 文件（注意如果保存为 HelloWorld.txt 要修改后缀名为.java）。

例 1.1 第一个 Java 程序 HelloWorld。

```
public class HelloWorld
{
    public static void main(String args[])
    {
        System.out.println(“欢迎来到 Java 世界”);
    }
}
```

说明：

（1）“public class HelloWorld”表明要建立一个类，类名为 HelloWorld。定义类必须使用关键字 class。Java 应用程序必须以类的形式出现，一个程序中可以定义若干个类。public 表明该类是公共类，可以被所有类访问。虽然一个程序文件中可以定义多个类，但只能有一个 public 类。如果一个文件中包含一个 public 类，文件名字必须和该类名相同。

类的内容，即类中的属性和方法在后面的一对花括号中列出。类的属性用变量描述，称为成员变量。对应地，类中的方法也称为成员方法。

（2）“public static void main(String args[])”建立一个名为 main 的方法。一个应用程序中可以有多个方法，但只能有一个 main()方法。main()方法是程序入口点，若无此方法，程序无法运行。

public 表明该方法是一个公共方法，所有的类都可以调用该方法。static 表明该方法可以通过类名 HelloWorld 访问。void 表明该方法没有返回值。String args[]表明该方法有一个字符串数组类型的参数。

（3）main()方法中包含一条语句“System.out.println("欢迎来到 Java 世界");”，其功能是输出括号中字符串的内容，即输出“欢迎来到 Java 世界”。

其中 System 是 Java 类库中的一个类，利用此类可以获得 Java 运行环境的有关信息和输入、输出信息。out 是 System 类中的一个对象。println 是 out 对象的一个方法，其功能是向标准设备即显示器输出括号中的字符串。

初次接触 Java 程序，可能觉得程序结果比较复杂，但读者很快就会发现，对于一般的 Java 程序，其基本框架和这里给出的结构是相同的，所不同的只是类中的属性（变量）和方法。

2. 编译 Java 源文件

打开一个命令提示符窗口，进入保存 HelloWorld.java 的目录，输入如下命令开始编译源程序，Java 源程序必须带上扩展名“.java”，否则编译程序会提示出错。

```
javac HelloWorld.java
```

Java 源程序经过编译后得到的字节码文件为“.class”文件，一个源程序可以编译成一个或多个字节码文件，每个字节码文件对应源程序中定义的一个类，文件名与对应的类名相同。例 1.1 源程序编译后生成的文件为 HelloWorld.class，该文件被自动保存在源程序所在的目录下。

3．运行 class 文件

运行编译后的 Java 字节码文件需要调用 Java 的解释器 java.exe。在编译后使用如下命令运行已生成的 HelloWorld.class 文件。注意：Java 字节码文件的扩展名.class 在运行时不必列出。运行结果如图 1-10 所示。

```
java HelloWorld
```

图 1-10　编译和运行 Java 程序

习题

1．简述 J2SE 的下载及安装步骤。
2．运行一个 Java 应用程序需要哪些步骤？具体如何实现？
3．参照例 1.1 编写一个简单的 Java 程序，在屏幕上打印出自己的名字。
4．上网搜索 Java 有哪些常用的集成开发环境，分析它们各自的优点。

项目总结

1．介绍 Java 的起源与发展史，Java 语言的主要特点。
2．Java 语言程序的运行机制，虚拟机工作原理及垃圾回收机制。
3．面向对象程序设计基本概念及四个基本特征。
4．Java 运行环境的安装与配置，Java 程序编辑、编译和运行过程。