第 1 章
面向对象方法基础

在 20 世纪 70 年代至 90 年代初期，计算机的主导软件开发方法是传统的面向过程的方法——软件生命周期模式（瀑布型开发方法）。此方法为计算机软件工程的实施发挥了重要的作用。但是，面向过程的软件开发方法存在如下缺点：

①在开发的初始阶段，建立系统逻辑模型的完整性描述极为困难。

②开发过程顺利与否强烈依赖于完整的软件逻辑模型描述。

③所设计的系统与最终模型相差太大，开发中途做出的大量文档的使用价值不大、约束力不强。

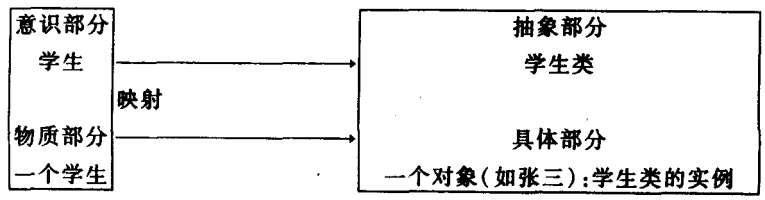
④开发速度太慢、周期太长，要到试运行时用户才能看到软件的原型。

⑤开发人员自始至终都要了解业务过程所有的细节，负担太重；若与用户配合不当或需求情况了解不透，则所实现的系统往往不好使用，很难达到预期的目标。

因此，传统的面向过程的软件生命周期模式难以适应大型、复杂软件系统的设计与开发。为解决这些问题，人们于 20 世纪 80 年代提出了面向对象的软件开发方法，并于 20 世纪 90 年代中、后期以来逐步得到广大系统分析员、软件工程师的普遍认同和应用。

1.1 面向对象方法的思想

面向对象方法是一种观察现实世界的有力手段，它可以帮助人们理解和感知现实世界的问题及其复杂的关系。面向对象的思想是，将现实世界问题空间（物质和意识）经过近似、分类和模拟映射成计算机软件的解空间（抽象和具体）。而物质和意识、具体事物（对象）和抽象概念(类)的关系如下：



面向对象方法强调对世界的宏观思维方式。它的设计方法是基于 Parnas 教授提出的信息隐蔽和 Cuttag 教授提出的抽象数据类型概念，把系统中的所有资源［包括数据、模块（方法）以及系统］都看成对象，每个对象将一种数据类型和一组过程（操作或方法）封装在一起，使得这组过程了解这一数据类型的处理，并在定义对象时可以规定外界（其他对象）在其上运行的权限。

类似于公式“计算机程序 = 数据结构 + 计算机算法”，可以将面向对象方法归纳成如下公式：

面向对象方法＝数据抽象 + 信息隐藏 + 继承性 + 动态链接性

面向对象方法具有如下特点：

1)模块独立性　每个对象是一个功能（操作）和数据独立的实体，对象之间只能通过传递消息进行通信。模块（对象）的内部状态以及实现功能的细节对外界（其他对象）隐蔽，不受外界影响；而且一个模块的内部状态的改变也不会影响其他模块的内部状态。

2 封装功能　将对象的属性（数据）和操作（方法即子程序）封装、隐蔽在一起，用户（包括其他对象）不需了解一个对象的内部细节，只需了解其功能描述即可。

3)程序重用　对象具有层次性，下一层次的对象自动继承上一层次对象的某些属性，而且将具有某些相同属性的对象归纳成一个类。通过继承性和类比性实现了软（构）件的重用。

4)动态链接性　对象与对象之间所具有的一种统一、方便、动态地链接，以及传递消息的能力与机制称为动态链接性。各种对象以及它们之间的相互链接和作用即构成各种不同的应用软件系统。

5 可靠性和易维护性　对象实现了抽象与封装，使得对象其中可能出现的错误仅限制在对象自身之内，不会向其他对象传播，因此增强了对象、系统的可靠性和易维护性。

6)灵活性（多态性）对象的功能是在执行程序时通过传递消息来动态确定的，所以，它支持对象的主体特征，使得对象可以根据自身的特点进行功能实现。

7)可扩展性　通过动态地增加新的类和对象，可以不断扩充系统的功能而不影响原有软件系统的运行。

为什么需要推广和应用面向对象的方法来设计、开发计算机软件系统呢？

面向对象(Object-Oriented ,简称 OO)方法创始人、面向对象语言 Simula 设计者之一、挪威奥斯陆大学信息科学系和信息研究所的 Kristen 教授说“：编写程序就是去理解客观事物”,而面向对象语言 Smalltalk 的设计者、美国犹它州立大学博士、施乐公司的计算机专家 Alan Kag 教授说“：客观现实本身就是面向对象的,所以我们编程也应该如此”。

1.2　面向对象程序语言及开发工具

面向对象思想萌芽的历史最早可以追溯到 1948 年。1948 年 2 月,Kristen 教授及其领导的工作小组开始从事挪威国防部的一个计算模拟项目。对于这个国防科研项目，他们需要处理一些复杂事物的组织、行为及其彼此之间的动态交互关系。而 20 世纪 50 年代的计算机程序语言提供的抽象类型不能完全地表示模拟系统中的具体现象。在这样的背景下，他们决定新开发一种模拟程序语言。经过多年的努力， Simula 语言于 1961 年诞生。1965 年,Simula 的

第一个编译程序进入试用阶段。

由此可知，任何一种程序语言的产生都是由待求解问题的需求驱动的。面向对象 Simula 语言的产生是为了帮助人们理解、描述一个复杂的系统，并基于计算机模型分析和现在及将来的问题。面向对象程序语言，简称对象语言，它打破了传统的、面向过程的程序语言的框架，建立了独特的编程风格，更为有效地帮助人们描述现实世界的复杂问题。

Alan Kag 早年在犹它州立大学攻读博士学位期间，在使用 Simula 编译程序时，发现这是一种独特、优秀的程序语言。他十分推崇 Simula 的面向对象的思想，并将这一思想和应用结合起来，发明了独树一帜的、面向对象的 Smalltalk 程序语言。

20 世纪 70 年代至 80 年代，美国国防部推出了另一面向对象的程序语言 ADA，这一语言在国防领域软件的开发中得到广泛的应用。

20 世纪 90 年代以来，面向对象思想和方法得到人们普遍的认同和接受。许多新的面向对象程序语言、面向对象的开发工具、支持面向对象方法的数据库不断涌现，并在软件开发和程序设计应用中大显身手，取得了积极的成果。它们是 C ++，Visual C，Visual BASIC，DELPHI，Power Builder，Visual FoxPro，Oracle，Informix，SyBASE，Lotus Notes 和 Domino，Jasmine 等。

特别值得一提的是，自从 SUN Microsystem 公司免费推出纯面向对象的 JAVA 程序语言以来，面向对象的软件开发方法不单更加实用，而且已成为软件工程界普遍接受的、主流的软件开发方法之一。

1.3 面向对象的方法学

中国科学院和中国工程院资深院士、我国科学泰斗钱学森教授指出：“思维科学的目的在于研究人们认识客观世界的规律和方法”，“思维科学又细分为抽象（逻辑）思维学、形象（直觉）思维学和灵感（顿悟）思维学三部分”。

上述可以归纳为：面向对象方法强调对世界的宏观思维，而面向过程方法强调对事物的微观理解和认识。面向对象方法学属于思维科学中的一项技术科学，面向对象技术是属于思维科学中的一项工程技术。

众所周知，抽象思维以抽象的概念为基础，形象思维以具体的形象为基础。人们认识世界和获取知识的认知过程，无论是抽象思维，还是形象思维，主要是通过从一般到特殊（从抽象的类到具体的对象、实例）的演绎方法，以及从特殊到一般的归纳方法来进行的。

抽象思维中的形式逻辑、辩证逻辑和数理逻辑建立了演绎和归纳的较完整的理论和方法体系。

在形象思维中，这种演绎或归纳都是在形象间的“相似”关系上进行的。客观事物发展过程中存在着同和变异。只有‘同’，才能有所继承；只有‘变异’，事物才能发展。人们利用形象思维去认识客观事物和改造客观事物时，首先按照相似原理和关系，把所研究的问题区分成一定的相似系统与类别。分类之后，进一步对事物进行详细的剖析（这相当于面向对象程序设计中的实例化过程）。剖析分析后再进行“综合优化”，并在事物运动中和运动的相互关系（相当于在对象之间传递消息）中去考察事物的静态相似与动态相似的关系、宏观相似与微观相似的关系、纵向相似与横向相似的关系。

综上所述，面向对象的方法学认为：

①客观世界是由各种“对象”组成。任何事物都是对象，每一个对象都有自己的运动规律和内部状态，每一个对象都属于一种对象“类”，都是该对象类的一个元素，复杂对象由相对较简单的各种对象以某种方式构成。不同对象的组合及相互作用即构成客观系统。

②通过类比，发现对象之间的相互性，即对象间的共同属性，这就是构成对象类的根据。按“类”、“子类”、“父类(超类)”的概念去构成对象类之间的层次关系。一般地，下一层次的对象可自然地继承上一层次对象的属性。

③对于已分成类的各个对象，可通过定义一组“方法”来描述该对象的功能（即定义允许作用于该对象上的各种操作）。对象之间的联系通过传递“消息”来完成（消息就是通知对象去完成一个允许作用于该对象的操作）。至于该对象将如何完成这个操作的细节，则是封装在相应的对象类的定义之中的，它对其他对象隐蔽。

通过对比，可以发现传统的面向过程的软件开发方法学，是软件人员从开发软件的立场出发，并不是从人们认识客观世界的过程和方法出发，因而不能很好地反映出人类认识问题的宏观层次。结构化软件设计方法以“过程”和“操作”为中心构造系统、设计程序，而不是以“对象”和“数据结构”为中心。因此，其思维成果的“可重用性”很差。

为什么用面向过程方法去开发软件得到的思维成果的“可重用性”差，而用面向对象方法去开发软件却能得到可重用性较好的思维成果呢？

伟大的导师恩格斯在《路德维希·费尔巴哈和德国古典哲学的终结》一书中指出：“必须先研究事物才能研究过程。必须先知道一个事物是什么，然后才能觉察这个事物中所发生的变化”。认识和研究客观世界时应从‘对象’人手，然后再转向‘过程’”。“过程”和“操作”是不稳定的、多变的，而“对象”和“数据结构”则相对稳定。因此，如果开发软件时从“对象”和“数据结构”入手，那么软件的主体结构就比较稳定，其所得的思维成果的可重用性就较好。

在实际应用中，用面向对象方法设计、开发一个软件的过程分为以下三步：

1)面向对象的分析(OOA)了解问题论域内该问题所涉及的对象、对象间的关系和作用(操作)，然后构造该问题的对象模型，并且力争这个“模型”能真实地反映出所要解决的“实质问题”。在这一过程中，“抽象”是最本质、最重要的方法，针对不同的问题性质选择不同的抽象层次。即定义各个对象“是什么”。

2）面向对象的设计(OOD)设计软件的对象模型。根据所应用的面向对象软件开发环境的功能强弱，在对问题的对象模型的分析基础上，以最少地改变原问题论域内的对象模型为原则对它进行一定的改造。然后在软件系统内设计各个对象、对象间的关系（层次关系、继承关系等）、对象间的通信方式(传递消息)等。即设计各个对象“应做什么”。

3)面向对象的实现(OOI)软件功能的实现，包括每个对象的内部功能的实现。确立对象哪些处理能力应在哪些类中进行描述（描述方法），确定并实现系统的界面、输出形式和其他控制机理。即实现各个对象应完成的任务。

1.4 面向对象的计算机体系结构

客观世界的一切事物和活动都是独立、并发进行的。例如，工人在工厂里装配产品，农民在田野上劳动，军人在操场上练兵，干部在办公室里批阅文件，教师在教室里授课，学生在实验室里做实验等，他们以及他们的活动都是独立、并行地进行的。再如，飞机在蓝天飞翔，火车在铁轨上运行，轮船在大海航行，汽车在公路奔驰等，它们以及它们的活动也都是独立、并行地进行的。

因此，对象具有内在的并发性。面向对象的程序由一组对象组成，即一组对象描述一个并行程序，一个对象内可以包含有多个进程。

既然面向对象程序具有并发性，那么处理这些程序的计算机体系结构应当体现并行性。显然,传统的冯·诺依曼结构的顺序计算机难以充分表达、描述面向对象的程序。必须开发新的计算机结构，以多个处理器构成的并行计算机，或者通过物理网络、并行操作系统支持的并行计算环境，将是支持面向对象程序设计的一种理想的计算机体系结构。

1.5 类、对象、方法、消息、协议和封装的概念

对象和传递消息是表现事物及事物间相互联系的概念。类和继承是适应人们一般思维方式的描述范式。方法则是允许作用于某类对象上的各种操作。

基于对象、类、消息和方法的程序设计范式的基本点在于对象的封装性和继承性。通过封装能将对象的定义和对象的实现分开，通过继承能体现类与类之间的关系以及由此带来的动态链接性、实体的多态性和软构件的重用性，从而构成面向对象的基本特征。

类封装了数据和方法。类实质上定义的是一种对象类型，它描述了属于该类型的所有对象的性质。例如,“integer”是一个类，它的性质是该类的所有实例（对象）都是整数。

实例（对象）是程序在执行过程中，由其所属的类动态生成的。一个类可以生成多个不同的实例(对象),同一个类的所有实例(对象)具有相同的性质。

类和实例的关系是抽象与具体的关系。实例是类的具体事物，类是多个实例的综合抽象。例如“马”是一个类“一匹白马”是“马”类的一个实例,其颜色为白色。

从动态的观点看，对象的操作就是对象的行为。现实世界中问题空间对象的行为是极其丰富多彩且多变的，而软件解空间对象的行为则是静态的、呆板的。因此，需要借助于极其复杂的、高效的算法才能操纵软件解空间的对象，从而获得问题的解。面向对象程序语言提供了“对象”概念，程序员可以根据需要去定义软件解空间的对象。

从存储角度考虑，“对象”是一块连续的私有存储区，其中包括数据和方法（子程序或单个程序语句）。其他对象的方法不能直接操纵该对象的私有数据，只有对象私有的方法才可操纵它。

从对象的现实机制看，“对象”是一台自动机。其中私有数据（实例变量、成员变量）表示对象的状态，该状态只能由私有的方法改变它。每当需要改变对象的状态时，只能由其他对象向该对象发送消息，对象响应消息后，按照消息模式找出匹配的方法并执行该方法，从而完成

指定的功能。

消息用于请求对象执行某一处理或回答某些信息的要求。消息统一了数据流和控制流。某个对象在执行相应的处理时，如果需要的话，可以通过传递消息请求其他对象完成某些操作或回答信息。因此，面向对象程序的执行是通过在对象之间传递消息来完成的。

发送消息的对象称为发送者，接收消息的对象称为接收者，消息中只包含发送者的要求。一个对象可以接收不同形式、不同内容的消息，相同形式的消息可以发往不同的对象。不同的对象对形式相同的消息可以有不同的解释并作出不同的反应，从而实现多态性，使得面向对象程序更加灵活。

消息形式用消息模式刻画。一个消息模式定义一类消息，它可以对应内容不同的消息。例如，定义“ + an integer”为实体“ 3”的一个消息模式，则“ + 4”，“ + 6 ”等均属于该消息模式中的消息。

综上所述，设计、执行一个面向对象程序的过程是：首先，将客观世界的事物进行抽象、分类，定义出各种各样的“ 类 ”，并根据需要动态地创建对象（对类实例化）然后，程序处理信息或响应来自用户的输入，将消息从一个对象传递到另一个对象（或由用户从键盘输入到对象），从而完成一定的程序功能；最后，当不需要该对象时，则删除它并回收其存储空间（这一工作在 Java平台上由系统自动完成，在 C ++平台上则需显式地由应用程序指定）。

一个复杂的对象由一些简单对象构成。这种复杂对象可能实现了许多消息，其中一些消息是针对这一复杂对象本身的，可由外界（其他对象）发送，通常称这些消息为公有消息；对于其中一些只是针对该复杂对象的某些简单对象的消息，这些消息只能被复杂对象其中的一部分发送与接收，通常称这样的消息为私有消息。

对象之间传递的消息是按照的一定的协议进行的。通常称一个对象所能接受的所有公有消息的集合为协议。面向对象程序语言以对象协议作为对象的外界面。协议指明该对象所接收的消息，在对象的内部每个消息响应一个方法，方法实施对数据的运算。对方法的描述是协议的实现部分（类体）的描述。例如，计算机网络系统中的通信管理机制就是对象、消息和协议的一个应用实例。其中，对象指的是网络结点（计算机），协议指的是网络通信协议（如 TCP/IP 协议），消息指的是结点根据通信协议发出的信息包。

封装则是一种信息隐蔽技术。封装的目的是，将对象的使用者和对象的设计者分开，使用者不必知道对象行为（数据定义和操作实现）的细节，只需使用设计者提供的消息访问对象即可。通过封装，可以实现模块（对象）的独立性，并可减少错误在对象之间扩散，使得面向对象程序更加安全、可靠。

封装的基本单位是对象，这个对象的性质由它的类说明，这个性质被同类的其他对象共享。例如，集成电路的一块芯片由陶瓷封装起来，其内部电路不可见，使用者也不关心。芯片的使用者只关心芯片的引脚个数、引脚的技术参数以及引脚所提供的功能。硬件工程师通过“ 引脚 ”将不同的芯片连接起来组装成产品。软件工程师当然也期望通过使用“ 类（对象） ”来达到组装软件产品的目的。

下面，具体讨论类的描述、协议描述和实现描述，以及对象的创建、实例的访问、初始化及其删除等问题。

类的描述包括协议描述和实现描述（私有数据的定义和方法描述）两部分。

协议描述包括消息模式、消息注释和消息分类三部分，其中消息模式定义一条消息的格式（消息名、参数个数及类型）；消息注释给出相应消息的用法、意义和返回对象；消息分类将协议中有关的消息分成若干类。

实现描述包括定义某个类的私有数据和对这些数据进行的操作（方法）。方法为允许作用于某类对象上的各种操作，是实现每条消息具体功能的手段。方法与消息一一对应，有一条消息就必然要有一个方法实现之，同样，有一个方法就应该对应一条消息。

类中的方法用来描述如何实施对应消息所要求的操作，当某个类的实例接收到它能接受的消息时，即调用相应的方法执行。类的实现描述包括类名、实例可访问的变量（数据）和实例可使用的方法三部分。

对象是通过对类实例化来创建。向类发送消息（由一元消息 new 实现）即可以产 生新的对象。一个新的实例被创建之后，即自动共享类中的数据与所有方法。执行创建操作后，系统即在内存中申请一块连续空间用以存储实例对象的数据及方法。

一个类可以禁止或允许其他对象访问它的实例（对象）的成员变量和方法。访问（引用）对象的变量或方法与引用 C 语言的结构元素相似。注意，方法的引用实际上是给指定的对象发送消息。

新创建的实例（对象）是通过一种特殊的称为构造方法的方法进行初始化。一个类可以提供多个构造方法，以便对新的对象实现不同的初始化。创建对象和初始化可同时进行。

在程序执行过程中，不需要的实例应将其删除，以回收存储空间。Java 不需要在程序中显式地删除对象，它的垃圾收集程序会自己判断某个对象是否有用，若无用则自动删除（释放所占据的资源）。它的工作原理是：当一个对象被引用时，自动地对此对象所占据的存储空间作标记；当一个对象运行结束时，又自动地将标记清除，那些没有作标记的对象作为垃圾被收集。Java 垃圾收集程序操作一般是在系统空闲时以低优先级运行，但也可以在应用程序中通过调用方法 System.gc （）在任一时刻请求进行收集垃圾。

1.6 继承机制与软件重用

类具有层次性。即某一个类的上层可以有父类（又称超类），其下层可以有子类。类的这种层次结构的一个重要特点是继承性。一个类可以（直接）继承其父类的全部描述，且这种继承具有传递性。

因此，属于某一个类的对象（实例）除了具有该类所描述的特性外，同时还具有该类的上层父类所描述的全部特性。

一个类可以拥有多个子类，如图 1.1 所示：一个类也可以具有多个父类，如图 1.2 所示。

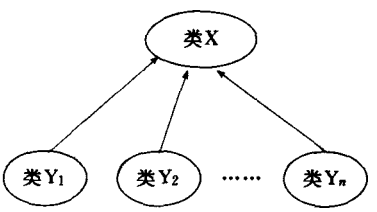


图 1.1 X 类有多个子类

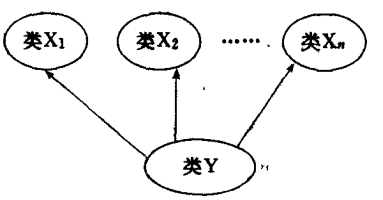


图 1.2 Y 类具有多个父类

例如“数据”类拥有文字数据、声频数据和视频数据子类，如图 1.3 所示；“计算机科学与技术学科”是一个理工结合的学科，具有理科和工科父类的性质，如图 1.4 所示。

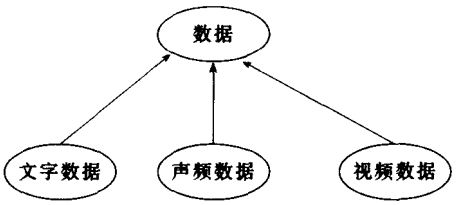


图 1.3　数据类拥有文字数据、声频数据和视频数据子类

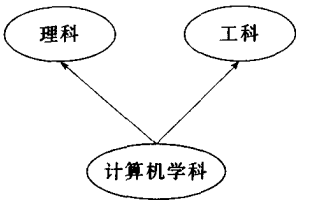


图 1.4　计算机学科类拥有理科和工科父类

若限制一个类最多只能有一个父类，则它最多只能直接继承一个父类的性质，称为单重继承，如图 1.5 所示。例如，人的性别只能要么继承男性，要么继承女性，如图 1.6 所示。

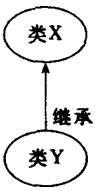


图 1.5　单重继承

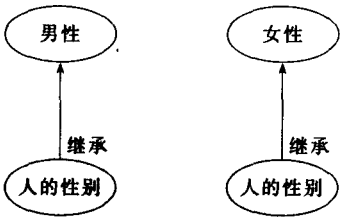


图 1.6　人性别的单重继承例

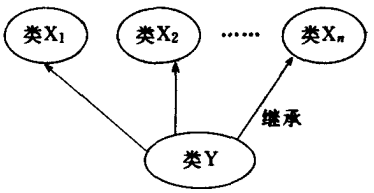


图 1.7　多重继承

若允许一个类可以直接继承多个不同父类的性质，则称为多重继承，如图 1.7 所示。例如，人的性格特征可能继承了父亲性格、母亲性格、祖父性格、祖母性格、外祖父性格、外祖母性格、叔伯性格、姑姑性格、舅舅性格、姨妈性格等，如图 1.8 所示。

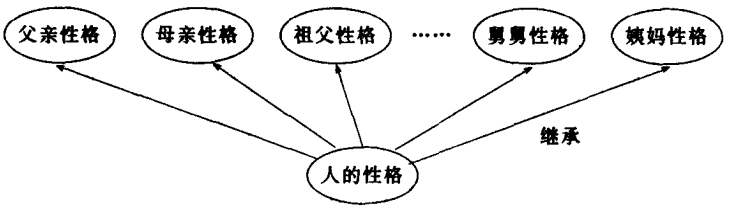


图 1.8　人性格多重的继承

JAVA 仅支持单重继承，而 C ++则支持多重继承。

下面，较为详细地讨论继承机制。

继承性（Inheritance）是一种自动地共享类、子类及对象中的数据与方法的机制。如果没有继承机制，那么，一方面，面向对象程序中的类（对象）的数据与方法就可能出现大量的重复；另一方面，大量重复的数据和方法增加了程序的复杂性，导致软件系统不稳定性和出错的可能性增加。

当类 Y 继承类 X 时，表明 Y 是 X 的子类，而 X 则是 Y 的父类。类 Y 由两部分组成，一部分为继承 X 的部分，另一部分为 Y 自己新增加的部分。新增部分是专门为 Y 类定义的数据和方法，如图 1.9 所示。

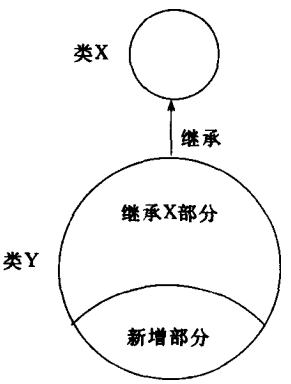


图 1.9　类的继承机制

可以将继承描述为一种映射（Mapping)关系。这种映射可以是简单的“等同”,即类 Y 的继承部分完全等同于类 X(数据与方法),也可以是其他形式的“等同”。例如,定义 Y 到 X 的继承时，可以对 Y 的性质重新命名、重新实现 Y 的方法等。

从语义上看，继承关系是一种“is a”关系。当类 Y 继承类 X 时,通常说 Y 已具备 X 的性质，也就是说 Y 即是 X。当然,Y 可以比 X 包含更多的性质。例如，图形的继承关系，如图1.10所示。

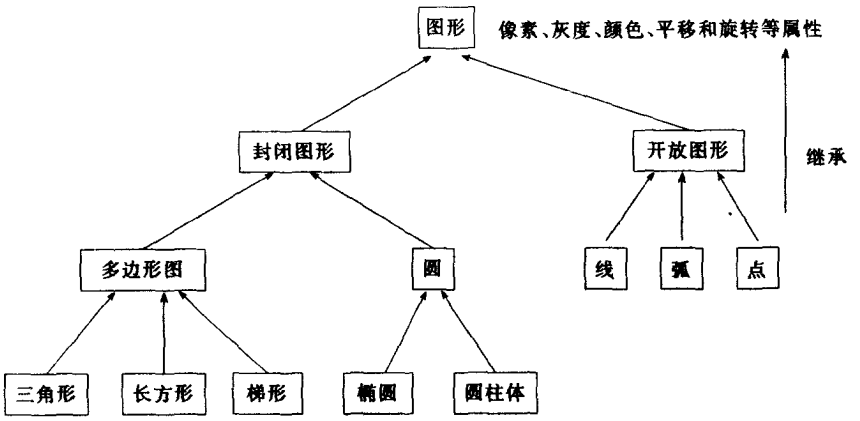


图 1.10　图形的继承关系

除了前面介绍的多重继承和单重继承之外，还有如下几种继承关系：

①全部继承：一个类可以继承某个类的所有性质。

②部分继承：一个类只能继承某个类的部分性质。

③取代继承：一个类完全取代另一个类。例如，如果徒弟掌握（继承）了师傅的所有技术，那么任何需要师傅的时间与场合，都可以被徒弟取代。

④包含继承：一个类包含了另一个类的所有性质。例如，苹果是一类特殊的水果，苹果继承了水果的所有特征，除此之外，苹果可能还具有其他一些自己的特征。

⑤受限继承：一个类继承另一个类时,受到一定的限制,可以继承一些性质,而另一些性质则不能继承。例如，鸵鸟是一类特殊的鸟，它们不能继承鸟会飞的特征。

⑥特化继承：一个类继承、拥有另一个类不具备的一些特有性质。例如，高级工程师是一类特殊的人，他们比一般人具有更多的特有信息（知识和技能等）。

综上所述，单重继承和多重继承用于描述继承源，而取代继承、包含继承等则用于描述继承内容。

在面向对象程序设计中，继承是一种静态地共享程序代码（方法）的手段。通过子类创建的实例对象可以接受某一个消息启动由该类的父类所定义的程序代码，从而使得子类和父类共享了这一程序代码，实现程序的重用。

封装机制除了实现面向对象程序的模块化和独立性之外，它还具有一定的“继承”性。即继承与封装具有一定的联系。封装基础上的消息提供了一种动态地共享程序代码的手段。通过封装，可将一段程序代码定义在一个类中，由另一个类定义的方法通过创建该类的实例对象并向它发送消息而启动这一段程序代码，从而也达到共享、重用程序的目的。

如果父类、子类都定义有相同名字的某一个方法，那么在程序执行过程中若用子类的方法

去代替父类的同名的方法，称为功能（方法）重载。

一个类中可以定义多个同名的功能（操作）不同的方法，通过程序的动态调用该类（对象）的同名的不同方法（某一时刻调用一个方法，另一时刻用不同的参数调用此方法），可以体现类的多态性，进而体现面向对象程序的多态性、灵活性。

由此可见，继承性具有可重用性，它支持采用可重用成分（软构件）开发软件，这有助于开发系统的原型，促进软件的可扩充性。

1.7 面向对象方法的应用

面向对象方法与技术自 20 世纪 90 年代以来已经成熟并得到广泛的应用，正逐步成为软件开发的主流技术。面向对象程序语言（简称对象程序语言）具有许多优点，许多领域的问题都可以采用对象程序语言进行开发。以下介绍对象程序语言和人工智能（AI)程序语言相结合在人工智能领域应用的情况。

人工智能 AI 程序语言具有如下的特点：

①支持符号计算，而不仅仅是数值计算。

②支持知识的不精确表示与处理，即可以实现模糊推理。

③具有清晰和良好定义的语义基础，结构严谨。

④支持启发式搜索和增量式程序设计，系统扩充容易。

除了传统的逻辑程序语言 LISP 和 PROLOG 等适合于 AI 领域的程序开发外，面向对象程序语言和函数式程序语言也可应用于人工智能领域的编程与开发。

一方面，面向对象程序设计将计算过程看做为分类过程加上状态变换过程，而逻辑程序 LP 设计将计算过程看做为推理（推演）过程，即将具有初始状态的输入在一系列条件的约束下，采用推理算法和搜索手段进行匹配，演算过程有利于描述启发式知识。

另一方面，面向对象方法模拟人类知识问题的较高和较广层次的分类过程，属于战略性方法；逻辑程序设计适合于模拟人的逻辑思维，处于人类认识问题的较深层次过程，属于战术性方法。

因此，在进行软件系统开发和程序设计实践过程中，最好采用面向对象方法与逻辑程序设计相结合的技术方式。

20 世纪 90 年代以来，面向对象数据库(OODB ,简称对象数据库)技术正逐步成为主流数据库技术，是面向对象方法最成功的一个应用领域之一。例如，ORACLE 8 , INFORMIX 8 , SYBASE , JASMINE , LOTUS DOMINO 等数据库管理系统均支持面向对象技术和快速原型方法开发应用系统。

面向对象数据库系统具有下列特点：

①支持面向对象数据模型（简称对象数据模型），而不是关系数据模型。提供复合对象、对象关系元组、对象标识、类、数据封装、继承性、多态性等概念和机制 支持分布式处理，支持版本控制与管理，支持数据库模式进化，保持数据的同时，允许数据库管理员修改数据库模式(即数据库结构)。

②紧密集成面向对象程序语言，允许 JAVA 和 C ++ 等面向对象程序语言方便地访问对象

数据库的数据。

③支持对象查询语言 OQL(Object Query Language）。支持对象导航，支持对象间的连接，支持对象复合查询，支持对象方法调用。

④支持永久数据机制。将数据库系统和应用程序分配相同的工作区，应用程序不仅能操纵暂时性数据，而且也能直接操纵永久性数据，对永久性数据的访问仍然是通过提交事务实现，但提交事务后，对数据库数据的修改会立即成为永久性的。

除了在数据库领域和人工智能领域得到广泛应用外，面向对象方法也在可视化设计、电子商务、分布式网络事务处理、虚拟现实与仿真、计算机辅助教学等领域得到应用。

练　习　一

1. 面向对象方法有何特征？

2. 为什么说面向对象方法更符合人们认识客观事物的方式与过程？

3. 面向对象的软件开发过程有哪些步骤？

4. 何谓类？类的描述有哪两部分？方法在类描述中起何作用？类在面向对象程序设计中起何作用？

5. 何谓实例（对象）？它与类的关系如何？

6. 何谓继承？继承在面向对象程序设计有何作用？

7. 何谓单重继承？何谓多重继承？请各举一例。

8. 试用图描述学生类、小学生类、中学生类、中专生类、专科生类、本科生类和硕士生类和博士生类的继承关系。

9. 继承与封装有何关系？

10. 何谓功能重载？何谓多态性？它们的作用是什么？

11. 面向对象程序的定义和执行过程是怎样的？

12. 为什么说面向对象程序的可重用性较好？

13. 面向对象程序设计和逻辑对象程序设计相结合有何意义？

14. 面向对象数据库有何特征？

15. 为什么说面向对象程序更适合于并行处理？

第 2 章
JAVA 语言简介

2.1 JAVA 语言的发展历程

综观计算机科学与技术发展的历史，无论是 FORTRAN 语言、ALGOL 语言、BASIC 语言、PASCAL 语言、C 语言、LISP 语言、PROLOG 语言、C ++语言、ADA 语言、Visual BASIC 语言、Visual C 语言、DELPHI 语言，还是 JAVA 语言等等，任何一种程序设计语言的发明与发展都有其一定的背景。

1990 年，SUN Microsystem 公司感受到 PC微机对工作站计算机市场的巨大压力，为此，他们决定开发一些新产品，并将目标定位为寻求消费类电子产品市场的开拓。经研究，SUN Microsystem 公司迅速成立由 Jame Gosling 博士领导的 Green开发小组。开始时，此小组试图采用 C ++语言开发消费类电子产品。但是，他们很快发现用 C ++语言编写的源程序必须针对具体计算机进行编译，所开发的软件的可移植性和可重用性很差。于是，他们独立地新开发出一种称为 Oak(橡树)的程序语言(Java 语言的原型) 。不久，发现已有产品取名为“ Oak”，因此，他们灵机一动将新语言命名为“ JAVA”。

1991—1992 年，Green 开发小组就用 JAVA 语言开发出代号为“ * 7 ”的具有卡通界面的手持机顶盒，此产品用于控制家庭环境中的电视、电灯、电话和电脑等设备，并且在这基础上，又开发出视频点播系统 VOD。遗憾的是，当时没有形成市场。

1992 年 10 月 1 日，Firstperson 公司将交互式有线电视盒投放市场。

1993—1994 年，国际互联网 Internet 和 WWW 技术蓬勃发展。NCSA 公司适时推出 Mosaic 浏览器，大大方便人们使用 Internet 浏览和获取互联网上的信息。这期间，SUN 创始人之一的 Bill Joy 博士介入 Green 开发小组，他作出一个关键性的决定：将 JAVA 定位到 Internet 的 WWW 应用开发中去，同时决定任何用户均可免费使用 JAVA 语言。Green 开发小组很快在互联网 Internet 上实现了简单文件到可执行文件的传输。

1994 年 5 月 23 日，SUN 公司正式发布 JAVA 与 Hot JAVA (一种浏览器) 标准。

1995 年 5 月，Netscape(网景)公司宣布支持 JAVA，使得 JAVA 可以嵌入 Navigator 2.0 版浏览器。

1995 年冬，IBM 公司、Microsoft 公司、Novell 公司、ORACLE 公司、SGI 公司和 Borland 公司等相继宣布支持 JAVA。

1995 年 12 月，SUN 公司和 Netscape 公司联合推出开放式、跨平台、分布式环境下的对象描述语言 JAVA Script。

1996 年，JAVA 语言及其技术进入中国，许多研究机构和高等院校开始学习、研究和使用 JAVA 技术。

1997 年，SUN 公司申请 JAVA 标准成为国际标准，但未获通过。这一年，我国掀起学习和应用 JAVA 技术热潮。

1998 年 4 月，JAVA 标准获准成为国际标准。

1999 年 10 月，中国计算机学会教育专业委员会和全国高等学校计算机专业教学指导委员会将《JAVA 语言程序设计》课程列入其制定的“计算机学科教学计划 2000”之中。

2.2 JAVA 语言对软件开发技术的影响

自从 JAVA 语言及其计算技术一问世，就以其新颖性、开放性、安全性、纯面向对象、支持分布式网络计算等吸引大批业界人士，受到人们的普遍欢迎。

可以预见，JAVA 语言及其计算技术的出现和普及，将使得面向对象方法与技术在以下几方面产生深刻影响：

①表达宏观思维的面向对象方法逐步成为软件设计、程序设计的主流方法之一。

②面向对象的软件需求分析、面向对象的软件设计和面向对象的程序设计技术更加实用化。

③软件产品可视化、可听化和可操作化程度更高，动画效果逼真。

④软件产品进一步实现跨平台、可移植、可重用。

经过这些年来的实践与应用，计算机工业界对 JAVA 的评价是 因为 JAVA 考虑跨平台，所以其处理速度有所降低，虽然如此，JAVA 语言及其计算环境仍将是现在和将来分布式网络计算的主要工具与技术。

2.3 JAVA 语言的特点

JAVA 语言诞生于 20 世纪 90 年代初，发展、成熟于 20 世纪 90 年代中、后期。与其他程序语言相比，JAVA 语言具有如下特点：

1)简单性　JAVA 语言的思想源于 C ++语言，但比 C ++语言简单。

2)纯面向对象　除数值、布尔和字符三种基本数据类型外，其他数据均作为对象处理。通过 CORBA 对象系统标准的支持，人们可以使用 JAVA 语言及其计算技术方便地建立分布式系统应用。

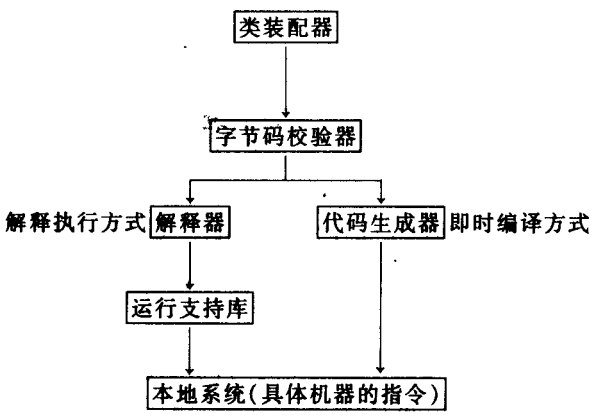
3)分布式处理　JAVA 语言及其计算环境所提供的网络类库支持 TCP/IP 协议，通过 URL 统一资源定位器可以访问 Internet 互联网上其他的对象(资源、数据等)。

4)健壮性　JAVA 语言及其计算环境提供自动管理内存和健全的异常处理机制。

5)安全性　JAVA 语言取消“指针”这一数据类型，内存分配由 JAVA 运行系统决定，提供字节码校验。在分布式网络环境下，JAVA 源程序的建立、编译、传输、安全检验、解释和执行的过程为：

6)平台无关　JAVA 编译器生成的是与具体计算机体系结构无关的字节码指令。这些字节码传输到本地机器后，再由 JAVA 虚拟机和运行时系统进行解释成可在本地机器上执行的代码。

其中，JAVA 运行时系统包括类装配器、字节码校验器、解释器、代码生成器和运行支持库。对于 JAVA 应用程序(JAVA Application)，JAVA　运行时系统主要为解释器在起作用；对于 JAVA 小应用程序(JAVA Applet)，JAVA 运行时系统主要指与 JAVA 兼容的 Web 浏览器。JAVA 运行时系统的结构如下：



7)可移植性　JAVA 编译器由 JAVA 语言本身实现，JAVA 运行时系统则由标准 C 语言实现。

8)高性能　JAVA 系统采取半编译、半解释的执行方式。

9)多线程机制　JAVA 语言支持语言级多线程机制。在底层操作系统为支持多线程的操作系统环境下，可将 JAVA 的线程映射到实际的操作系统线程，从而可在多处理机环境、网络并行计算环境下开发、实现 JAVA 应用程序的并行执行。

10) 动态性　可以动态地在 JAVA 类库中加入新方法和实例变量，而且可以通过接口机制支持多重继承。

2.4　JAVA 平台及其工具

自从 SUN 公司推出 JAVA 语言以来，其核心版本已从 JDK　1.0.2　发展到现在 JAVA 2：

① 1996 年 5 月推出 JDK　1.0.2

② 1997 年 7 月推出 JDK　1.1.2

③ 1998 年 10 月推出 JAVA 2

其中，JAVA 2 不仅仅是一种程序语言，它已经发展成为一种分布式集成计算环境。

支持 JAVA 语言的几种主要操作系统平台是：

① SPARC Solaris （工作站平台上的 Unix）

② X86/PII/PIII Solaris（PC 微机的 Unix）

③ Windows 95/98/2000 和 Windows NT

④ Macintosh

以 JDK为核心的常见的 JAVA开发工具有：

① SUN Microsystems 公司推出的 Visual Cafe

② Symantec 公司推出的 Cafe

③ Symantec 和 Netscape 公司推出的 Visual Cafe

④ Microsoft 公司推出的 Visual J ++

⑤ Borland 公司推出的 JBuilder

上述这些 JAVA 开发工具虽然各有特色，但他们的基本功能是类似的。读者可以任选一种作为上机、编程实习的环境。

JAVA 语言作为一种新型的程序语言，除了提供程序语言的一般功能外，还提供了丰富的类库以供程序员开发使用。这些类库包括：

①基本程序语言类库 java. lang

②实用程序类库 java. util

③文件输入／出类库 java. io

④网络类库 java. net

⑤ 抽象窗口工具箱类库 java. awt

⑥ JAVA 小应用程序类库 java. applet

在编程开发时，可用 import 语句装载指定的类库以便继承引用。

为了帮助程序员方便、快捷地编程、调试和开发 JAVA应用，JAVA 语言提供了下列工具：

① JAVA 编译器 javac，它将后缀为‘ java’的 JAVA 源程序文件进行编译成中间字节码。

② JAVA 解释器 java，它将后缀为‘ class ’的 JAVA 类文件进行解释成可执行代码。注意，JAVA 应用的主方法所在类的类名应与 Java 源程序文件名主名同名。

③ JAVA 调试器 Jdb，它提供调试 JAVA 应用程序的功能。

④ JAVA 反编译器 Javap，它将 JAVA 应用的汇编程序形式翻译成 JAVA 源代码形式。

⑤ JAVA 文档管理器 Javadoc，它提供自动生成 JAVA 应用文档功能。

⑥ JAVA Applet浏览器 applet viewer 它使得机器在未进入 WWW（非 Web 浏览器方式）的条件下，运行包含可视化内容的 JAVA小应用程序 Applet。

2.5 JAVA 应用程序的基本结构

一个 JAVA 应用程序（JAVA Application）由若干个抽象的类组成，每个类封装了若干个数据（成员变量）和若干个方法，其一般结构形式为：

```
class  类 1
 {
  成员变量定义;
    ⋮
  方法 1(…)
    { …… }
  方法 2(…)
    {……}
    ⋮
  方法 n(…)
    {……}
 }
class 类 2
 {……}
  ⋮
class 类 n
 {…… }
public 主类
  {
    成员变量定义;
    ⋮
    public static void main(…)
      {… }
  }
```

其中，作为一个可独立运行的 JAVA 应用程序至少需要一个“主类”,且在存储 JAVA 应用程序时，其文件名主名应与“主类”的类名相同。其他类可视实际需要而定义。

下面,给出一个简单的 JAVA 应用程序例，它显示两行信息‘ Welcome to the world of 2000 times !’和‘ Welcome to the world of JAVA !”:

```
public class example
  {
   public static void main (String arg[ ])
     {
      System. out. println( " Welcome to the world of 2000 times !" );
      System. out. println( " Welcome to the world of JAVA !" );
     }
  }
```

用 JAVA 编辑器将上述源程序编辑好，然后给它取个文件名存盘，接着对此程序文件进行编译、解释之后即可运行。

2.6 JAVA 小应用程序的基本结构

JAVA 小应用程序(JAVA Applet) 是 JAVA 语言与 WWW 技术相结合后引入的一个重要概念。JAVA 不仅可以传输静态的超文本标记语言 HTML 文档，而且还可以传输被称为 Applet 的动态可执行内容。Applet 是一种用 JAVA 语言创建的、嵌入到 Web 页面中用于产生特殊、动态页面效果的小程序。

当用户在客户机的 Web 浏览器中访问嵌入了小应用程序 Applet 的 Web 页面时，指定的 Applet 代码将被下载到客户机的 Web 浏览器中执行。

一方面，通过应用 Applet,程序员可以设计出具有动画、声音、图像和其他特殊效果的 Web 页面；另一方面，在 Web 页面中使用 Applet,可以使 Web 页面能与用户动态地进行交互，以便在 Web 页面中实现需要高度人机交互功能的诸如游戏等方面的应用。

JAVA 小应用程序(JAVA Applet)不能单独运行,它必须嵌入到 Web 页面中,由 Web 浏览器控制执行。JAVA Applet 也封装了若干个数据（成员变量）和若干个访问权限为公共的方法，其一般结构形式为：

```
public class  小应用程序名  extends  java.applet.Applet
 {
   成员变量定义；
      ⋮
   public void 方法 1( …)
     {…… }
   public void 方法 2(…)
     {……}
      ⋮
   public void 方法 n(…)
     {……}
 }
```

创建 JAVA 小应用程序 Applet 的过程为：

①用 JAVA 语言创建 Applet 源程序,并用 JAVA 编译器将其编译成字节码。

②将 Applet 字节码嵌入到 Web 页面中，即建立嵌入了 Applet 资源的 HTML 文件。

③与 JAVA 兼容的 Web 浏览器(如 IE,Navigator,Hotjava 等)根据 HTML 文件中的 APPLET 标志指定的路径及 Applet 字节码文件名下载相应的 Applet 字节码,然后由 Web 浏览器控制执行 JAVA 小应用程序完成所需的功能。

下面，给出一个简单的 JAVA 小应用程序以及嵌入了 Applet 资源的 HTML 文件,它们实现在行号和列号分别为 20 的屏幕上开辟一个大小为 200×40 像素的窗口，在此窗口内显示一行信息‘ Welcome to the world of JAVA !”：

```
// JAVA小应用程序
import java.awt.*;
```