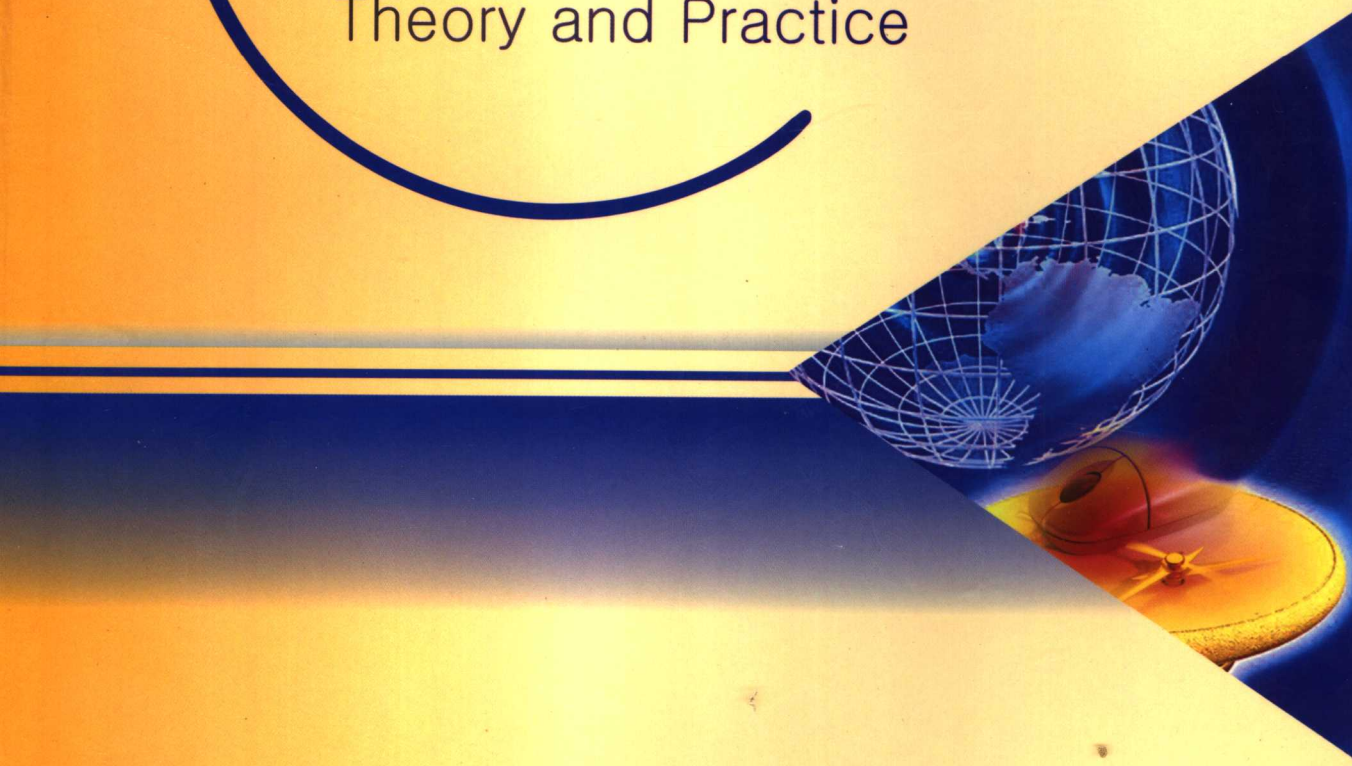
文登敏　张丽梅　编著

# 面向对象理论与实践

## Object-Oriented Theory and Practice



西南交通大学出版社
Http://press.swjtu.edu.cn

# 面向对象理论与实践

# Object-Oriented Theory and Practice

文登敏　张丽梅　编著

西南交通大学出版社
·成　都·

# 内 容 简 介

本书共分五章，包括面向对象理论（一、四、五章）和实践（二、三章）两大部分。第一章讲解了面向对象的基本概念和机制，并结合实际介绍了面向对象的思维方法。第二章讲解了 C++ 中的面向对象特征，针对学习面向对象程序设计过程中遇到的重点和难点问题进行详细的介绍。第三章以 Visual C++ 6.0 为基础，介绍了微软基础类库 MFC 以及应用 VC 环境编写应用程序和软件组件的方法。第四章介绍了 OMT 等经典的面向对象分析和设计方法论以及统一模型语言 UML，并在此基础上给出了目前常用的分析模式和设计模式及其相关内容。第五章结合现代构件理论和软件构架的理念介绍了构件的相关研究现状，以及未来软件构架的发展趋势。

本书较全面地讲述了面向对象理论和方法，并以 C++ 作为实践环节，便于读者在掌握理论的同时获得较强的操作能力。

**图书在版编目（CIP）数据**

面向对象理论与实践 / 文登敏，张丽梅编著. —成都：西南交通大学出版社，2005.6
ISBN 7-81104-079-4

Ⅰ. 面… Ⅱ. ①文…②张… Ⅲ. 面向对象语言—程序设计 Ⅳ. TP312

中国版本图书馆 CIP 数据核字（2005）第 052121 号

面向对象理论与实践

文登敏 张丽梅 编著

*

责任编辑 唐元宁 王庆峰
责任校对 李 梅
封面设计 王 可

西南交通大学出版社出版发行
（成都二环路北一段 111 号 邮政编码：610031 发行部电话：028-87600564）
http: //press.swjtu.edu.cn
E-mail: cbsxx@swjtu.edu.cn
成都蜀通印务有限责任公司印刷

*

成品尺寸：185 mm × 260 mm 印张：16.25
字数：397 千字 印数：1—3 000 册
2005 年 6 月第 1 版 2005 年 6 月第 1 次印刷
**ISBN 7-81104-079-4/TP · 022**
定价：26.00 元

# 前　言

长期以来的教学经验告诉我们，很多人都知道面向对象是个“好东西”，而且非常有必要学，有必要用。但是，也有不少人问到我，该如何学？从哪里下手？为什么总是自学了一半就不得不放弃，下次重新拿起来又会重蹈覆辙呢？

面向对象确实是个好东西。正如早些年有本书上说的：用结构化方法需要开发一年的程序，用面向对象技术只需要一个星期！的确很诱人的。这话说得固然有些绝对和片面，但体现了其中的优势和可取之处。

面向对象技术已经产生很多年了，目前使用计算机的人员，无论是专业的还是非专业的，都在不知不觉中使用着这项技术。它不仅应用在成熟的工具内核中，软件编写人员也在应用。

其实，面向对象的思想在现实生活中无处不在。人们每做一件事情，每思考一个问题，每认识和掌握一项新的技术，几乎都在使用面向对象方法进行着思维。那么，为什么一涉及编写计算机程序，就需要重新认识语言、认识语法、认识语义以及其中的编程规则呢？其主要原因就在于没有充分认识到这种思维方式的内在本质，以及这种思维方式在具体问题上的抽象化方法。

因此，学习面向对象技术，应该首先对其有一个全面的认识。这种认识应该从思维方式入手，而不是从编程入手。到目前为止，面向对象技术不再仅仅应用于面向对象程序设计，而更多的是讲求面向对象的分析与设计过程。

但是，面向对象编程技术仍然是其中的一个重要的组成部分，从编程入手掌握其中的基本概念和机制依旧是目前常用的学习方法。毕竟现在社会的计算机已经成为一种最常用的工具，任何一种思想、算法和方案也都需要通过计算机进行模拟和验证。

本书根据历年来的教案编写，并结合面向对象技术的最新发展，力求有更高适用性和更持久的借鉴价值。书中不仅包含了适合面向对象编程的实践部分，同时也包含了面向对象的分析与设计等方面的理论，面向对象技术的未来趋势，以及软件组件、软件构件理论和软件构架等方面的知识。

全书共分五章。第一章介绍面向对象的基本概念和机制，不仅引入类、对象、继承、封装、多态等基本概念，同时还将这些概念和现实生活中人们认识世界的方法，包括归纳、演绎、抽象思维等方法进行比较和认识，以求从根本上理解面向对象的实质。

为了使读者对第一章接触到的知识有一个感性的认识，本书第二章介绍了 C++ 语言中的面向对象特征，包括 C++ 中的类/对象定义，继承、封装和多态的实现机制和实现方式等。除此而外，第二章中还对 C++ 在 C 的基础上增加的非面向对象特征（如内联函数、引用机制等）进行了必要的介绍，对于学过 C 语言再进一步学习 C++ 的读者来说也会有一个完整性的认识。

掌握了 C++ 中的面向对象特征之后，就需要根据这些特征以及学到的面向对象基本概念和机制进行实践，而实践的环境也是非常重要的。Visual C++ 是编写 C++ 应用程序的常用工

具，MFC 类库又是开发环境提供的基础类库。因此，在第三章中给出了关于 Visual C++ 的基本情况介绍，MFC 类库的内容，以及在该环境下编写控制台程序、编写通用 Windows 应用程序（基于对话框、单文档、多文档）、专用 Windows 应用程序（如数据库编程、多媒体编程、网络编程等）以及编写 DLL 动态链接库/ActiveX 组件/COM 组件的方法和步骤，真正达到学以致用。

第四章和第五章属于面向对象的高级课题。第四章中较全面地介绍了面向对象的系统分析与设计思想以及经典的和现代的常用方法，包括 OMT 方法、Booch 方法和 UML 等。同时，分析模式和设计模式是前人在面向对象分析和设计过程中的经验总结和抽象，合理借鉴和复用对于提高新系统的正确性、合理性和可靠性等方面有着极好的帮助。第四章介绍了几种常见的分析模式，并对设计模式的相关问题做了较为详尽的阐述。

第五章涉及构件理论和软件构架方面的知识，是面向对象技术的更高级别的抽象。构件的发展仍然处于研究阶段，很多现实的问题都还没有得到很合理的解决。软件构架是建立在构件基础上的新的软件开发模式，也是未来软件生产的趋势。

第一、三、四、五章由文登敏编写，第二章由张丽梅编写，全书由文登敏统稿。

由于编写时间仓促，错误在所难免，望业界专家和读者多多指正。

**编　者**

2005 年 3 月

# 目　录

# 第 1 章　面向对象基本概念和机制

**本章提要**

- 类和对象是面向对象技术的核心
- 面向对象技术针对现实思维方式而产生
- 继承、封装和多态是面向对象技术的三大基本机制
- 类的结构，类之间的结构关系构成了复杂的结构体系
- 归纳和演绎是方法学
- 继承和抽象是思维模式
- 人们认识世界和发展世界的过程是归纳和演绎的过程
- 面向对象技术的历史、现状和未来发展

面向对象的理论和方法已经在许多领域得到了广泛的应用，包括面向对象的程序设计、面向对象的分析与设计方法等。要真正了解和掌握面向对象技术，其本质是了解其中的思想，并探讨其中的分析问题和解决问题的思维方式。本章首先就面向对象的基本概念和机制、面向对象思想的基本出发点以及现实世界中与人们的生活实际的相关联系等方面进行深入分析，同时引入对象和类、逻辑实体和物理实体、归纳和演绎、继承和抽象、封装和多态、泛化—特化/整体—部分结构等基本概念。

## 1.1　从“类”谈起

世间万物构成整个世界。所谓“万物”，即是指世间真实存在的个体。这种个体通常也称之为“实体”。为了进一步明确实体的概念，也可以将其划分为“物理实体”和“逻辑实体”两种。

“物理实体”是指在现实生活中看得见摸得着的、实实在在存在的、具有一定物理尺寸和外形的个体；而“逻辑实体”则是指一个“事件”，一个“案例”，一个“事实”，或者一种“现象”，类似于人们常说的“摆事实讲道理”中的“事实”，“透过现象看本质”中的“现象”，为了解释一个问题而举的例子等。

为了能够充分认识“实体”的特征，或者挖掘出“实体”中所蕴含的深层次的意义，从

而进一步发展世界，就必须对这些零散的、众多的实体进行归类，或者简称为“分类”。

“物以类聚，人以群分”。凡具有共性的实体集合，都可以称之为一类。凡是能成为一类的，一般都不会只有一个独立的个体。“类”是“种类”的简称，而“种类”又和“区分”、“划分”等行为密不可分。

对现实世界中的个体，划分种类的方法很多，根据不同的作用、目的和研究方向等，都有不同的分类方法。动物、植物、微生物，学生、干部和工人，文科、理科和工科，机械、建筑和土木，等等，都是典型的分类方法。单就分类学而言，现在已经构成了一个独立研究的学科，对于进一步研究事物的特征起到了明显的促进作用。

分类不仅可以构成层次，而且可能存在交叉。就人类而言，从性别上可分为男性和女性，而他们又都可分别细分为老年、中年、青年和少年儿童。而如果一开始就从年龄划分为老年、中年、青年和少年儿童，则每种年龄段的群体又可细分为男性和女性。每种分类的结果都可以再次细分，细分的层次可能还非常多，交叉的现象也极其复杂。

对于某一个类，其共有特征往往都是很抽象的。我们都了解梧桐树，问到梧桐树具有什么样的特征，可能都可以说出个大概。但如果问起梧桐树有多高，叶子有多大，恐怕也只能说在什么样的范围之内，而不能说出一个具体的数值。同样，人的特征非常明显，每个人都有名字、性别、身高、年龄和体重等，但人和每个人的概念是截然不同的。人是一个大类，每个人是一个个体。我们只能说这个人叫什么名字，是男是女，而不能说人叫什么名字，是男是女，或者再具体点，运动员属于一个特殊的群体，我们同样不能说运动员叫什么名字，是男是女，等等。

可见，类是一个群体，是一些具有某些共性的个体构成的集合。从不同的研究角度，可以对同一个群体进行不同形式的分类，以研究其不同的共性。一个类还可以划分为不同的子类，每个子类还可以根据需要进一步细分。现实世界中的任何一个实体都一定具有某个类的特征。实体是一个对类赋予了生命以后的独立个体。

几乎所有的名词都代表了一个“类”。苹果、水果、动物、猫、桌子、梧桐树、分子、星球，等等，都是“类”，都代表了具有某种相同特征的一类实体的共性。而凡是在“类”名词上加“这个”、“那个”等限定词构成的短语，几乎都代表了特定的类的实例，代表了一个存在的实体，即对象。而如果加上“这些”、“那些”等限定词，则构成子类。

类是分等级的。不同等级的类涵盖了不同的范围。动物类涵盖的范围大于猫科动物类涵盖的范围，而猫科动物类涵盖的范围又大于猫类涵盖的范围。相对而言，涵盖范围大的类称之为泛化类，涵盖范围小的类称之为特化类。但就一个类所包含的特征来说，泛化类具有比特化类更少的共同特征，因为泛化类的共性是所有特化类共性的交集。

类和类之间存在着一定的关系。上述的泛化类和特化类是日常生活中经常遇到的类关系，被称为“泛化—特化”关系。除此而外，“整体—部分”关系也是非常重要的一种类之间的关系，如“衣服”和“纽扣”的关系，“汽车”和“轮胎”的关系等。

在面向对象概念中，类是具有共同特征的一组对象实体的抽象，而对象是类的实例。严格来说，面向对象的技术研究应该称之为面向类的技术研究。但正如在日常生活中人们在不会引起混淆的前提下将类和对象等同对待一样，在面向对象技术领域的许多文献中，也同样将类和对象视为同一概念，当然是在不会引起误解的情况下。

万物称之为“实体”，包括“物理实体”和“逻辑实体”。“实体”即为“对象”。

“类”是对具有共同特性“实体”的一种抽象描述。

“类”和“类”之间存在一定的关系，包括“泛化—特化”关系或“整体—部分”关系。

## 1.2　面向对象的思维方法

人们怎样认识世界？一个新生儿从看到现实世界的第一眼开始，到逐步进入人生旅途，是怎样的一个过程？

从“呱呱”坠地，一个新的生命诞生，母亲便带着他一步步认识这个世界。某一天，小孩第一次看到了一只小白兔，很喜欢，但又不认识。妈妈耐心地告诉他，这是小白兔，是一只兔子，有四条腿和长长的耳朵，等等，使小孩首次建立了一个特定动物的印象。

又有一天，遇到了一只小灰兔。是小白兔吗？当然不是。是兔子吗？是的。相同点在哪里，区别又在哪里？通过妈妈的耐心教导，长见识了。

接下来的日子里，这个小朋友会不断地看到他以前看到过的，没看到过的，大大小小的，相同的、不同的个体，包括动物、植物、人等。在不断接触的过程中，他不断地增长知识，也在不断地对自己已经掌握到的知识进行筛选和整理，并自觉不自觉地进行归类。凡是兔子都具有什么特征，凡是树都具有什么特征，等等。

终于有一天，上幼儿园了。老师在教绘画，并要求每个人画一只小鸟。这个小朋友一方面参照老师的示范，另一方面在自己的脑海里搜索自己曾经见过的小鸟的样子，终于，一只不太完美的小鸟绘画问世了，尽管不是很像。

再长大，小朋友成了大朋友，从他的手中需要诞生“外星人”的形象。“外星人”是什么样子？既然能够称之为“人”，那就应该具有人的特征，两条腿，两只眼睛，还有鼻子、嘴巴，身高也和地球人不相上下吧。但又由于是外星人，也必然要和地球人有所区别，那就把各个部位的比例设计得奇怪些，脸型设计得更怪异些，行动方式也和地球人不同吧！

另外，“鬼”又是什么样子？“牛头马面”、“阎王殿”等，尽管是封建社会用来迷惑劳苦大众的产物，但它又是人们在对现实世界进行了充分的认识之后，发挥想像并通过合理的构思产生的“虚物”。

从认知学的观点分析，从见到的一个个独立的个体进行总结，在脑海里进行归类的过程称为“归纳”。归纳已经成为一个独立的方法学，在很多领域中得到了广泛的应用。“数学归纳法”就是通过前提和递推关系进行数学证明的一种经典的理论和方法。

从归纳的结果，从整体的共性，进行合理想像，产生前人从未见过的，从未实现过的新类的过程称为“演绎”。有了演绎，社会才能进步，人类才能发展。新事物的出现，新产品的研制，新理论和方法的诞生，无一不是通过经验积累和不断演绎的结果。

“归纳”和“演绎”是人们认识世界和发展世界的两种最基本的方法学。

“归纳”的过程也称为“抽象”过程。所谓抽象，一方面表示对一组特定的实体进行共性抽取的过程，另一方面也表示了对这些实体进行共性抽取之后所得到的结果。

“演绎”的过程，有时也称为“派生”过程。抽取共性是针对已经存在的实体进行的。在

这些共性的基础上，如果增加已经存在的实体的个性，即为该实体在抽取共性之前所具有的完整特征，如果通过合理演绎增加一些其他特征，就有可能产生新的实体特征。从一个特征集增加新的成分构成新特征集的过程，就是“派生”的过程。

“归纳”和“抽象”，是从特化类到泛化类的过程。
“演绎”和“派生”，是从泛化类到特化类的过程。

在面向对象理论中，“派生”和“继承”几乎是同一个概念。“派生”是以泛化类为视点，派生出特化类，而“继承”则以特化类为视点，是从泛化类继承而来。

以上谈到的归纳和演绎，也就是抽象和继承，表达了人们认识世界和发展世界的思维方式之一。随着人们知识水平的不断提高，从一个类甚至多个类演绎其他类的单一方式已经不能满足社会的需要。那么，彼此之间没有任何关系的两种类型的实体之间，能否进行必要的组合成为一种新的种类？从而引申出面向对象中的组合机制。

组合产生在彼此看似互不关联的类之间。如发动机、轮胎及必要的机械部件可以构造一部汽车，棉布、纽扣等可以缝制一件衣服，CPU、内存等可以组装一台计算机，等等。还有现代社会中的移动通信技术、发电技术，日常生活中的微波炉、电磁炉、空调、电视等，都是通过对已有技术进行组合而产生的对人们生活具有极大促进作用的“组合”的例子。类似的例子还有很多，举不胜举。

继承关系是“泛化—特化”关系，组合关系是“整体—部分”关系。
继承和组合是面向对象技术中两种最基本的类关系。

就继承关系和组合关系而言，虽然是面向对象技术中的两种最基本的类关系，但在实际运用当中，尤其是面向对象技术发展到今天，组合关系对社会的促进作用已经达到甚至超过继承关系，这是因为：

◆ 继承关系多数都是针对现有类层次进行的一种划分，是对已有类之间关系的一种重新认识和分类。一旦分类关系确定，则彼此之间将构成一种静态关系。

◆ 目前人们对于已有类之间关系的认识已经达到了相当的层次，分类学也已经形成了一门独立学科，创建了相当完善的理论和方法。

◆ 组合关系则千变万化。类和类之间的任意组合关系可能产生更多的促进社会发展的新生事物，有些甚至是意料之外的事物。同样的组合原类，通过不同的组合顺序，不同的组合比例，可能产生出不同的事物。因此，与“继承”关系相比，“组合”关系具有更大的灵活性和多样性，具备一种动态特性。

继承关系可能是单继承，也可能是多继承。在某种意义上说，多继承和组合具有相似的意义，都是从多个类产生一个新类的机制。关于这一点，在本章稍后内容中会做进一步的说明。

从面向对象的三大机制（继承、封装和多态）理论上来看，这里所说的“继承”关系和“组合”关系均属于面向对象机制中的“继承”机制。在后面的描述中，在不引起混淆的情况下，统称为继承。

继承关系是静态关系，而组合关系是一种动态关系。
继承和组合都属于面向对象技术中的“继承”机制。

## 1.3　关于“抽象”

从概念上来理解，抽象既表示一种行为，也表示一个结果。抽象的结果通常也称为“共性类”。从众多的实例中抽取其公共特性的过程，称之为“抽象”。而作为抽取出的共同结果，也可称之为众多实例的“抽象”。

抽象也是一种分析问题的方法。抽象的过程是一种“归纳”的过程，是从许多实例中发现、总结、提取其共性作为“类”来进行分析和研究。

继承是面向对象机制之一，是一种“演绎”的过程，是在已经抽象出的共性的基础上增加个性特征，从而构成新类。

单就抽象的结果而言，根据不同的问题也有不同的情况。从传统的面向对象观点来看，“抽象”是对现有实例（或类）中的所有共性一次性提取，从而构成一个“共性类”的过程，抽象的结果是一个类，该类中包含了所有被抽象类的共性。而从某些复杂大系统的求解过程（如综合集成法、数据挖掘等）来看，抽象的结果不只有一个类，而可能是根据具体情况抽象出一组“共性类”，其中每一个共性类中只含有部分共性，而不是所有共性。也就是说，被抽象的类含有多种（不仅仅是多个）共性。这些共性不是简单地构成一个共性类，而是需要对这些共性进行划分，构成不同的共性子类。所有这些共性子类的“并集”涵盖了被抽象类共性的全体。对这些不同的共性类进行重新组合，即可产生新的个性类。

如果抽象的结果是唯一的共性类，则可以通过增加新的个性特征产生新类。而如果抽象的结果是多个共性类（每一个共性类称为共性子类），那么构成新类的方式即成为对不同的共性子类进行组合的过程。后者将不允许再通过增加所有共性子类中不存在的特征来构成新类，除非对原有抽象过程进行重新调整以求获得所需要的特征，或者对原有抽象出的共性特征进行重新“划分”。

共性类是对被抽象类的共性进行抽象的结果。
共性类可能是唯一的，也可能是通过“划分”得到的一个共性类集合。
可以在单一的共性类的基础上增加新的特征构成新类。
可以通过对共性类集合中的子集进行组合构成新类。

## 1.4　分析问题的过程

解决问题必须首先分析问题，而分析问题则必须首先发现问题。也就是说，只有提出问题，才能有针对性地分析和解决。那么，问题是什么？简单说，问题就是需求，就是需要达到的目的。

有效地分析问题，对问题论域和应用论域进行全面的了解和把握，有助于对问题的解决。在现代软件工程中，需求分析越来越重要，其规范性也越来越强。

对一个问题进行有效的分析，基本素材是必不可少的。这些素材可以是间接的，也可以是直接的。在典型的 MIS 系统中，报表、工作流程、管理规程、座谈交流纪要等都属于直接

素材，而行业规范、知识库等属于间接素材。在分析过程中，素材可以认为是一个一个的实体，是看得见摸得着的。这些素材之间必然存在着某种联系和关系，对这些素材之间的关系进行分析、整理、归纳和演绎，力求发现其中的问题并着手解决这些隐含的问题，是分析阶段的主要内容。

著名科学家钱学森提出的“综合集成方法”，为有效求解大型复杂问题提供了一条可能的途径。其基本思想也就在于首先针对已经存在的“事实”进行归纳，而后针对具体的尚未解决的问题“实例”进行对比分析，以求达到最终的求解。

数据挖掘是现代科学中的一种新生学科。参与挖掘的基础素材作为最原始的“实体”，采用特定的算法（如聚类算法）对其共性进行研究，这也是一种典型的解决问题的过程和方法。

综合集成方法，以及数据挖掘理论，通常都用于解决用单一算法和模型难以求解甚至无法求解的“大型”问题。但我们可以通过这种求解问题的思路看出，大型问题都需要从基本素材入手，而基本素材所具有的共性抽取是一个很复杂的问题，包括共性分类、分层、细化，自学习、自调整，以及素材之间的共性抽取，素材归类原则等。

Coad-Yourdon 方法和 LIA（基于信息的分析技术）被公认为是一种最容易理解的分析问题的方法。其基本思想可归纳如下：

- 收集有用的个体，即一个个的素材。凡其中的主要名词将被作为可能的数据特征或分类名称，凡核心动词将被作为可能的行为和动作（LIA）。
- 按词性、词义及个人经验进行归类和划分，构成一个个的独立单元（对象—类层）。
- 对类和类之间的关系进行分析，进一步进行抽象（归纳）和必要的组合（结构层）。
- 按照类关系的密切程度进行分组（主题层）。
- 定义每个类中的静态特征（属性层）。
- 定义每个类中的动态特征（服务层）。

分析过程是一个迭代的过程。每个过程都可能直接或间接地进入下一级或返回上一级，甚至到设计阶段，都可能因为分析过程的疏漏而产生迭代和反复。

长期以来，以面向对象理论和方法分析问题和解决问题一直是业界关注的焦点和研究的核心。从面向对象基本概念的建立，到面向对象分析（OOA）、面向对象设计 OOD、面向对象实现 OOI 和面向对象测试 OOT 等一系列的规范化过程和方法，研究专家都给出了相应的值得推荐应用的理论和方法，包括 OMT 方法、BOOCH 方法、OOSE 方法等。

在分析问题和解决问题的过程中，关于问题本身的描述和表达、求解过程的文档等，同样需要相应的标准化规范，这些规范通常都是以图、表、流程、公式以及必要的文字说明来实现。更加严谨的表述是形式化语言，如 Z 语言等。

UML 是目前业界公认的“统一模型语言”，采用完全的面向对象方法，提供完整的理论基础、分析和设计机制，更重要的是提供了统一的表达规范，为进一步推进面向对象思想和方法在各个行业中的应用奠定了良好的基础。

*解决问题必须首先分析问题。分析问题是问题驱动的。*

*分析问题的过程是一个迭代的过程。*

*抽象是分析问题的一种最基本的方法。*

## 1.5　面向对象方法简介

20 世纪 80 年代末以来，随着面向对象技术成为研究的热点，相继出现了几十种支持软件开发的面向对象方法。其中，Booch、Coad/Yourdon、OMT 和 Jacobson 的方法在面向对象软件开发界得到了广泛的认可。特别值得一提的是统一建模语言 UML（Unified Modeling Language），该方法结合了 Booch、OMT 和 Jacobson 方法的优点，统一了符号体系，并从其他方法和工程实践中吸取了许多经过实际检验的概念和技术，并逐步成为面向对象方法的标准。

面向对象方法不仅仅是编写计算机软件的一种理想的方法，同时也作为一个工程、一个项目，乃至一个大型系统构造中的分析和设计模型。

对于面向对象方法，根据不同的模型有不同的划分形式：

- 静态模型和动态模型，逻辑模型和物理模型。
- 对象模型、动态模型和功能模型。
- 对象关系模型、对象交互模型和对象行为模型。

其中静态模型、对象模型、对象关系模型，以及 Coad/Yourdon 方法中关于 OOA 的五个层次的描述，都代表着类和对象以及类和类之间的关系。而动态模型则描述了对象和类随着时间的变化以及外界触发事件的产生而产生的内部状态变化。

面向结构的分析方法中，最常用的三视图 ——实体关系图、数据流图和状态迁移图，与面向对象方法有着非常密切的关系。OMT 方法强调对象模型、动态模型和功能模型，分别与三视图模型有着相同或相似的描述。

研究和分析类的内部结构，以及类与类之间的关系，甚至类的动态行为，也只能说是面向对象方法中的一小部分内容。而构造类，从实际问题中抽象出类，抽象出类的属性和方法，才是分析问题的最重要的环节。

在分析和解决实际问题的过程中，为了发现对象和类，开发人员要在系统需求和系统分析的文档中查找名词和名词短语，包括可感知的事物（汽车、压力、传感器）、角色（母亲、教师、政治家）、事件（着陆、中断、请求）、互相作用（借贷、开会、交叉），以及人员、场所、组织、设备、地点等。同时还要在相关文档中寻求必要的动词，如升降、点击、碰撞、运转等。名词或者名词短语可以作为类或者类中的属性，动词可以作为类中可能的方法。

在具体发现对象和类的过程中，相关的文档是最基本的。这些文档包括显式文档和隐式文档。显式文档是指用户需求分析报告、会议纪要、访谈记录，以及终端用户可以提供出来的图表、打印文件、管理规章等。隐式文档则包括与问题领域相关的知识、开发环境和工具、人力资源的知识阶层及素质，甚至是行为习惯和操作模式等。

面向对象方法强调解决问题过程的规范化和标准化。

不同的面向对象方法其本质是相同的，不同的是其表示法。

发现对象和类是采用面向对象方法解决问题的关键环节。

UML 力求统一面向对象的方法，不仅包括过程，而且包括表示法。

## 1.6 关于面向对象编程

初学面向对象技术的人，往往都是从学习面向对象编程（或面向对象程序设计）开始，而且是基于某一种具体的面向对象程序设计语言（如 C++ 等）。只有对面向对象的基本理念深入了解和掌握之后，才能在任何领域甚至是解决具体问题的时候采用面向对象方法进行思维。

但普遍存在的一个问题是，尽管没有意识到，我们始终以面向对象的思维方式来解决生活中的一些具体问题。而在编写计算机程序时，却必须从头开始了解一种编程语言中的语法和语义，以便掌握面向对象方法中的编程约定和相应的规范，但这有时很困难，即便是熟练掌握结构化编程技术的专业人员。其主要原因还是没有从根本上掌握面向对象的实质。

这里给出一个简单的例子来说明。

有一个需求如下：

张三的数学书第 38 页上的第一个字符是什么？
李四的语文书第 40 页上的第二个字符是什么？

面向结构的思维方式是：

编写一个通用的函数，其中包括四个参数，分别表示“谁”的，“什么”书，第“多少”页，第“几”个字。函数的返回值是一个字符。用 C++ 编写的形式为：

```
char GetChar(char *who,char *book,int where,int position);
```

而获得结果的方式是通过函数调用：

```
GetChar("张三","数学",38,1);
GetChar("李四","语文",40,2);
```

与此相对应的面向对象思维方式和解决该问题的方法是：

```
张三.数学.page[38].word[1];
李四.语文.page[40].word[2];
```

从这个简单的例子可以看出以下三点：

◆ 面向结构编程所构造出的函数更加抽象，是不具有任何含义的纯逻辑处理过程。而面向对象强调信息封装，每一个问题的解决总是与具体的对象相关。

◆ 面向结构的抽象函数具有更加通用的效果，但后期的维护和升级变得困难。如果张三和李四的处理过程并不完全相同，抽象出来的函数不能通用，则必须修改源代码才能实现。而对于面向对象编程来说，张三的处理过程和李四的处理过程完全独立，彼此之间没有任何的耦合关系。修改张三的处理过程不会影响到其他的独立个体。

◆ 面向结构的编程需要对明确给出的所有问题进行整理和归纳才能抽象出处理函数，而对未明确给出的问题则不能保证适用。但面向对象编程针对每一个具体的对象个体来处理，便于及时开展工作并有效地适合未来的发展，尤其是对于当前未知问题也提供了良好的继承机制。

因此，面向对象编程相对于面向结构编程来说具有很多优势，尤其是针对递增式的用户

需求来说，灵活多样的继承机制和组合机制可以在不修改原有代码的前提下，通过派生增加新的属性方法或者重新定义原有方法来简单实现。

## 1.7　面向对象的基本概念和机制

### 1.7.1　对象的基本概念和组成

简单地说，对象是一个实体，包括逻辑实体和物理实体。而逻辑实体又可以分为显性实体和隐性实体。

物理实体是在现实生活中看得见摸得着的具有一定物理尺寸和大小形状的个体，如一棵树、一本书、一只猫等，大至一个星球，小至一个分子或原子。

显性逻辑实体看得见但摸不着，有规模但无形状尺寸，实实在在存在，正如一个事件，一个案例、一个具体的需求，甚至是有待解决的一个具体问题，以及为了解决一个问题而编写的软件和一个结论等。

隐性逻辑实体不仅摸不着，通常也看不见，但在解决问题的时候却是经常要使用和借鉴的实体，比如一个定律、一个知识点、一个推理逻辑。隐性逻辑实体是人们在长期学习的过程中逐渐积累起来的。

面向对象的思想将对象的内部组成归结为两大部分：属性部分和方法部分。属性部分也称为数据部分或者静态部分，方法部分也称为函数部分或者动态部分。

对象的内部组成成员称为对象成员。属性部分称为数据成员，方法部分称为方法成员或者函数成员。数据成员常常表征了一个对象的物理属性特征，如长宽高/直径，动物的科类/腿的数量，名称等相对固定的数据，可以通过赋值关系赋予一定的数值，方法成员通常是一段程序代码，用于实现特殊的处理过程，如星球的运行轨迹，动物及微生物的运动规律，植物的生长环境及外界条件的影响等。

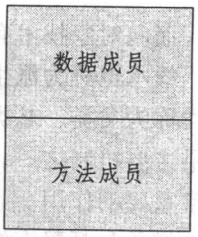


图　1.1

一个对象的简单模型如图 1.1 所示，描述为

对象 = 数据 + 方法

在计算机学科领域，属性也称为数据，一个对象的属性部分也可以称为数据成员或数据部分；方法也称为函数或过程或分程序，一个对象的方法部分也可以称为函数部分或函数成员，即

对象 = 数据成员 + 方法成员

如果从对象构成部分的特征来说明，则

对象 = 静态成员 + 动态成员

对象是由静态数据部分和动态方法部分构成的实体。

### 1.7.2　类与实例

许多不同的对象都可以以非常相似的方式产生作用。这些对象本身存在着许多共同的特

性。为了使得对许多具有共同特性的对象进行更为简便灵活的处理，设计了类的概念。

类是对一组具有相同结构的对象的描述。类由概括了这一组对象中共同性质的方法和属性组成，而从一组对象中抽取公共方法和属性并将它们构成一个类是面向对象功能的核心。

用传统的结构化程序设计的观点来认识类，可以认为类是一个数据类型，而对象则是在类定义之下的实例变量。

如果用存储的观点来认识，类只定义了格式和应包括的成员种类，不占用任何存储空间，而对象则需要占用一块实实在在的存储空间。

如前所述，对象是实体。而类是对对象特征的一种抽象。一个类只能具有某些特征，而每种特征都不可能有具体的“特征值”，最多只能说其特征有一个什么样的取值范围。但一个具体的对象就会有具体的特征值。正如一个人有身高、体重、姓名等特征，正常成年人的身高取值范围一般在 150 ~ 180 cm。而具体的一个人，姓名是“张三”，身高 172 cm，体重 70 kg 等，就是一个典型的例子。

在一般不容易引起混淆的情况下，常常将“类”和“对象”同等对待。正如人们常说的“面向对象”，其实严格来说应该是“面向类”，因为分析一个具体的对象是没有什么实际意义的，也不能做到全面和综合。

因此可以说，类是对象的一种抽象和概括，对象是类的一个实例。

类是对象的特征抽象。对象是类的实例。

### 1.7.3　面向对象机制

面向对象技术有三大机制：继承、封装和多态。

继承也称为派生，是从一个或多个现有类，通过一定的技术产生新类的方法。现有的已知类称为父类、基类或祖先类，新产生的类称为子类或派生类。继承机制作为面向对象技术的最主要机制之一，对于迭代化软件生产，版本升级，原有系统的简单维护和更新等，起着非常重要的作用。

继承包括静态继承和动态继承。静态继承是在父类的基础上通过增加数据成员和方法成员而构成子类的过程，而动态继承则是以父类的实例作为子类的数据成员而存在的一种方法。传统意义上的继承是指静态继承，动态继承是传统意义上的组合，或称为聚合，类似于 C 语言中的结构体成员是结构变量。静态继承中的父类和子类之间是一种泛化—特化的关系，动态继承中的父类和子类之间是一种整体—部分的关系。

封装是面向对象的第二特征。该特征不仅表现在一种形式，而且体现了一种信息的隐藏。将一个对象（类）的数据部分和方法部分组合在一起（即封装在一起）作为一个整体单位来处理，本身就体现了一种封装的概念。同时，作为一个对象内部来说，只要对外接口一定，对象内部的数据成员命名及使用方法、对象内部的代码编写算法等方面的内容都与外界无关。这种无关性也就体现了信息的私有特征，即对外的不可见性，或者称为信息的隐藏性。

多态作为面向对象的机制，是最难以说明的。从一般概念上来说，多态是指同样的消息被不同的对象所接受将会产生不同的结果。但有时多态也可以这样认为，即一个同样的消息，