# 第 1 章

# 面向对象程序设计概述

# 1.1 结构化程序设计方法

## 1.1.1 结构化程序设计思想的提出背景

回首计算机的发展历程，人们发现计算机软件的发展速度始终滞后于计算机硬件的发展，它已经成为制约计算机产业整体发展的瓶颈。究其原因可能有很多方面，但下面两点不容忽视。

- 软件产业的个体化
- 受限于程序设计语言

## 1.1.2 结构化程序设计方法

- 自顶向下、逐步求精的开发方法
- 模块化的组织方式
- 结构化的语句结构

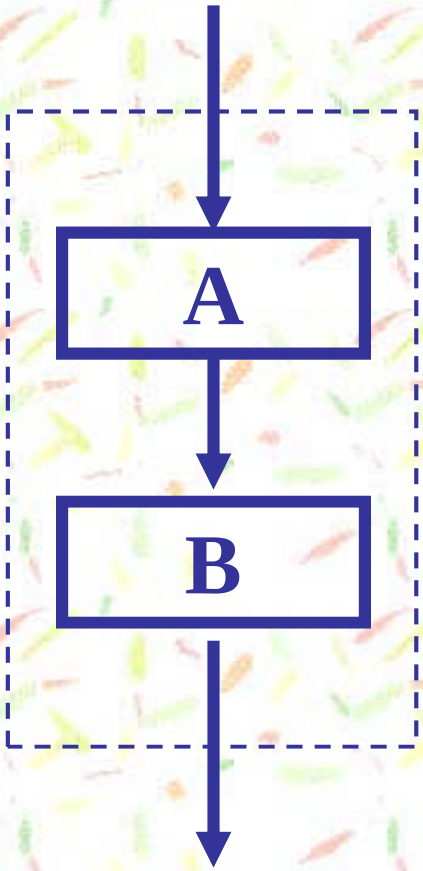
# 自顶向下、逐步求精

将编写程序看成是一个逐步演化的过程。所谓自顶向下是指将分析问题的过程划分成若干个层次，每一个新的层次都是上一个层次的细化，即步步深入，逐层细分。

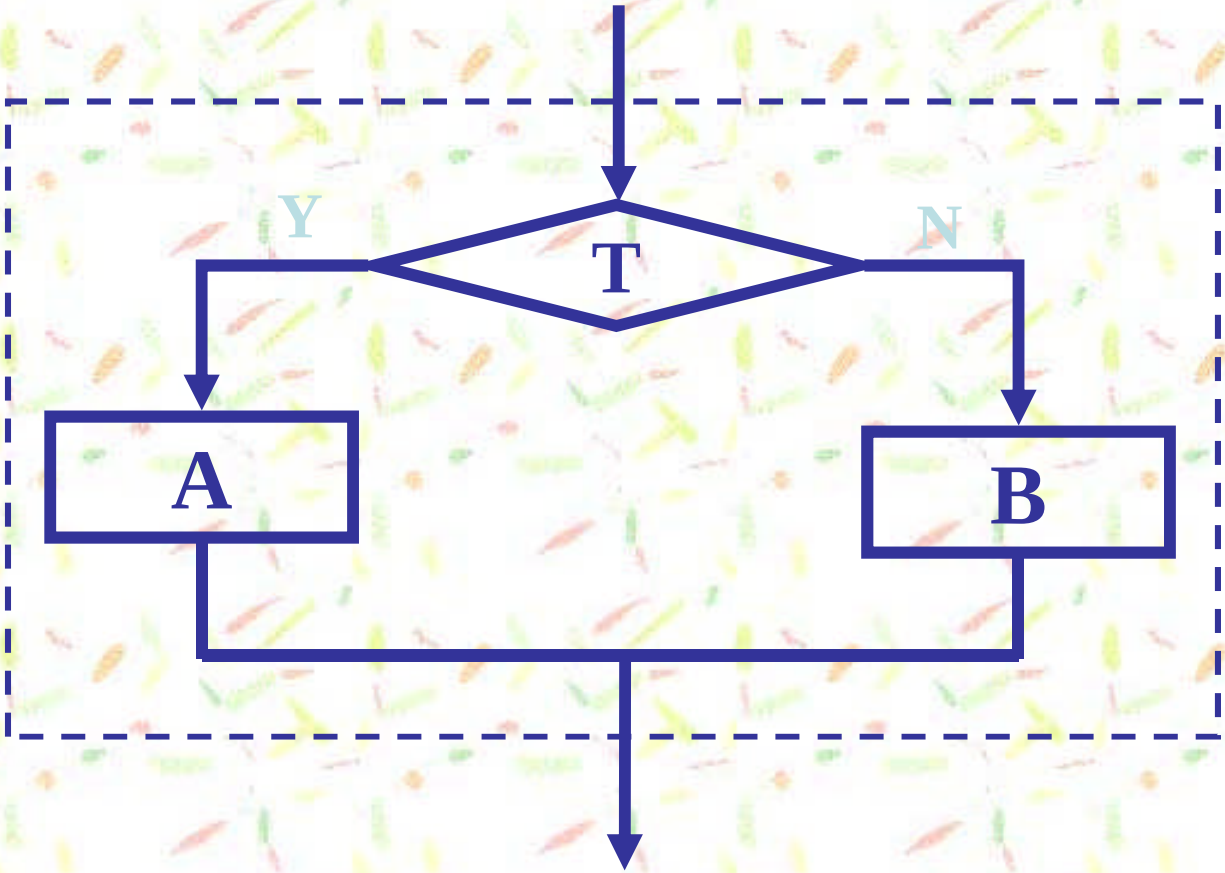
# 模块化

将整个系统分解成若干个模块，每个模块实现特定的功能，最终的系统将由这些模块组装而成。模块之间通过接口传递信息，力求模块具有良好的独立性。

# 语句结构化




顺序结构




分支结构

面向对象程序设计

面向对象程序设计

面向对象程序设计

# 结构化程序设计的特点

程序设计＝数据结构＋算法

程序内容＝过程＋过程调用

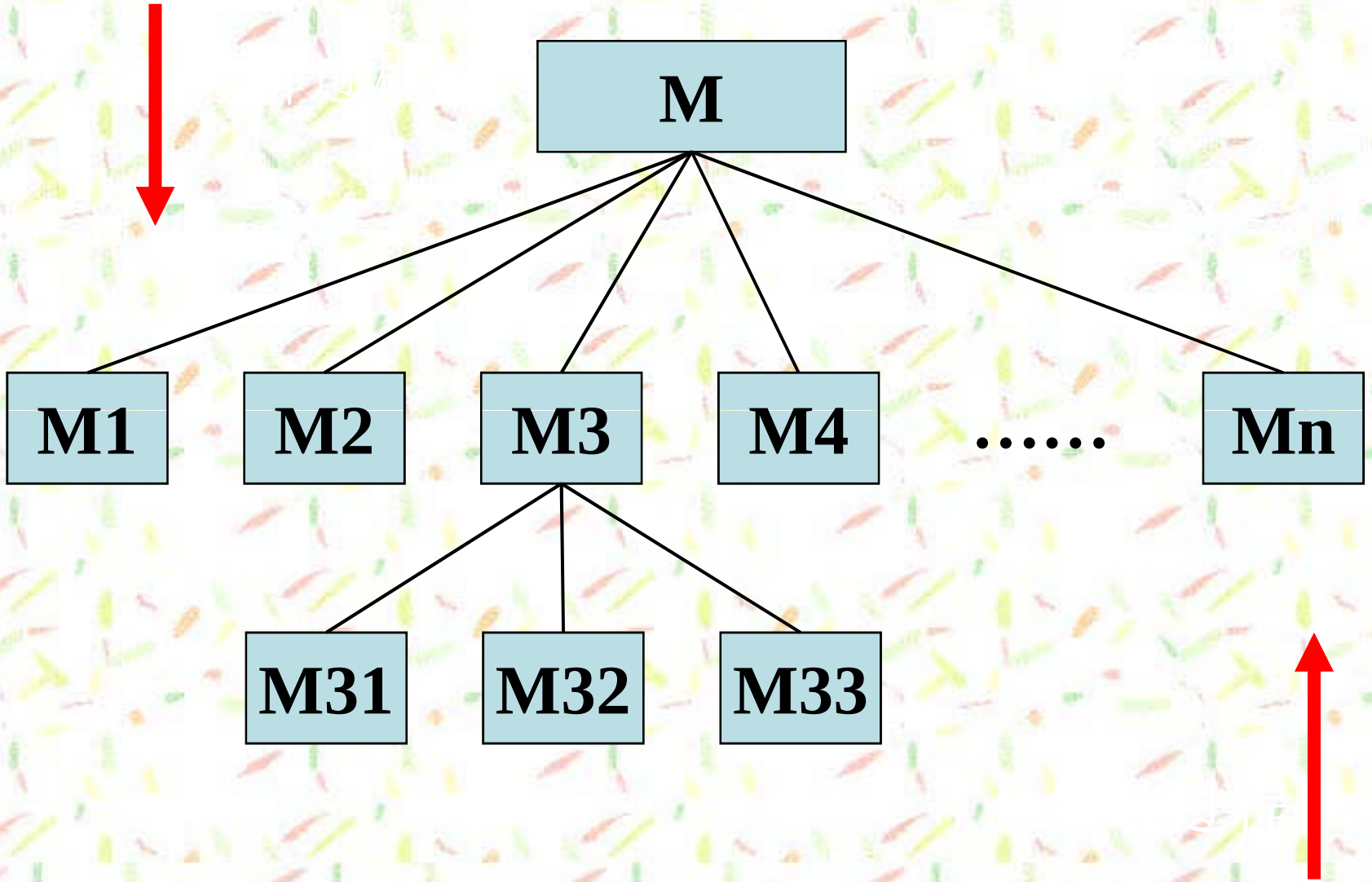
# 结构化程序设计的基本过程

软件开发的基本过程：

- 需求分析
- 系统设计
- 系统实现
- 系统测试
- 系统维护

当结构化思想贯穿于每个过程时，

其基本过程：分解和组装




面向对象程序设计

面向对象程序设计

面向对象程序设计

# 1.2 面向对象程序设计方法

## 1.2.1 面向对象程序设计的产生背景：

- 审视问题域的视角
- 抽象级别
- 封装体
- 可重用性

## 1.2.2 面向对象程序设计

面向对象程序设计方法是指用面向对象的方法指导程序设计的整个过程，所谓面向对象是指以对象为中心，分析、设计及构造应用程序的机制。

# 面向对象程序设计应该具有的特征

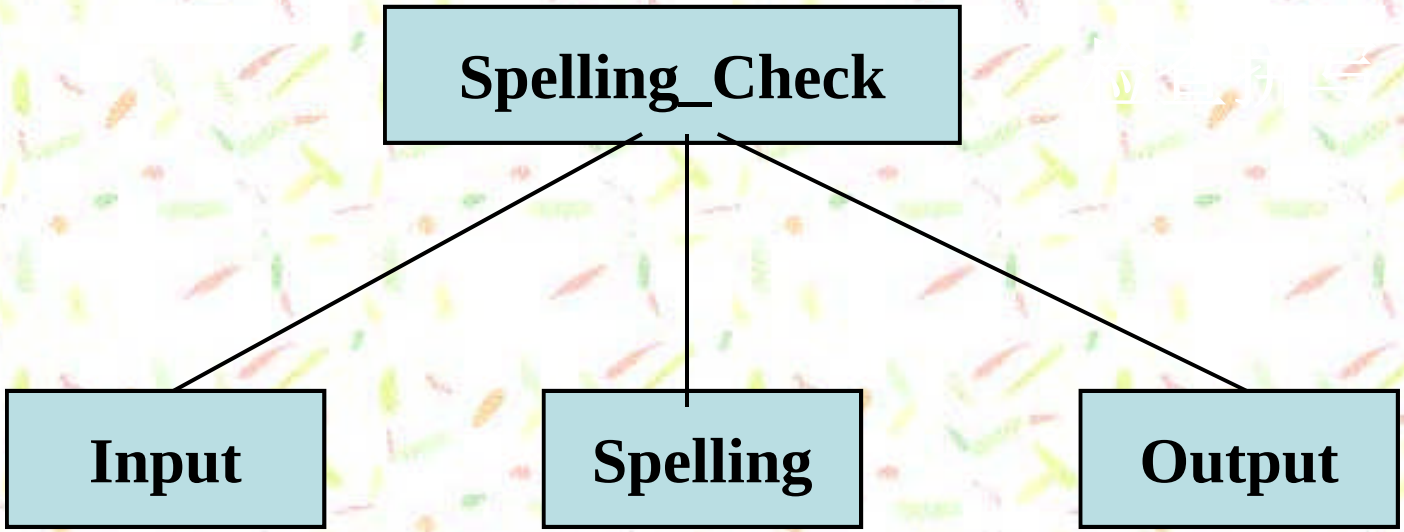
- 所有待处理的内容都表示成对象；
- 对象间依靠相互发送消息或响应消息实现通信；
- 每个对象都有自己的惟一标识，以便区别属于同一个类的不同对象；
- 对象一定属于某个类，我们又将这个对象称为所属类的一个实例；
- 类是将具有共同属性的对象进行抽象的结果，它可以具有层次关系，即一个类既可以通过继承其他类而来，也可以被其他类继承。

举例：快速拼写检查程序。

快速拼写检查程序将对用户提供的单词进行拼写检查，如果在字典中找到，输出“拼写正确”的字样，否则输出“拼写不正确”的字样。

面向对象程序设计

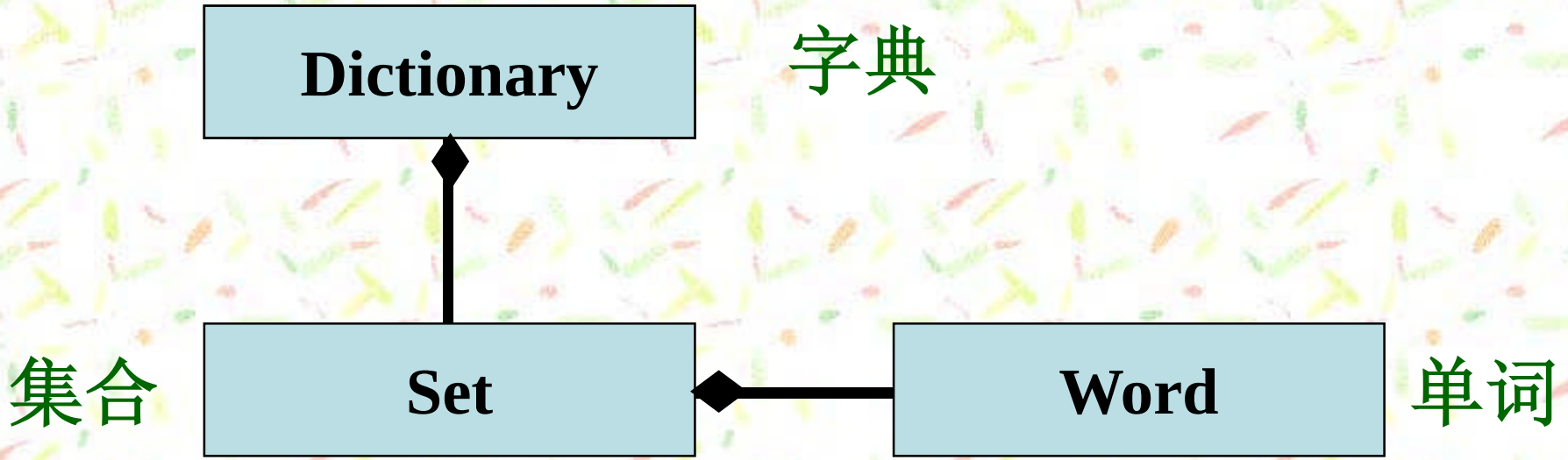
# 结构化程序设计方法



举例：快速拼写检查程序。

快速拼写检查程序将对用户提供的单词进行拼写检查，如果在字典中找到，输出“拼写正确”的字样，否则输出“拼写不正确”的字样。

# 面向对象的程序设计方法




面向对象程序=对象+消息

# 面向对象程序设计的优点

1. 能够实现对现实世界客体的自然描述
2. 可控制程序的复杂性
3. 可增强程序的模块性
4. 可提高程序的重用性
5. 可改善程序的可维护性
6. 可适应新型的硬件环境

面向对象程序设计

# 1.3 基本概念

## 抽象 ——

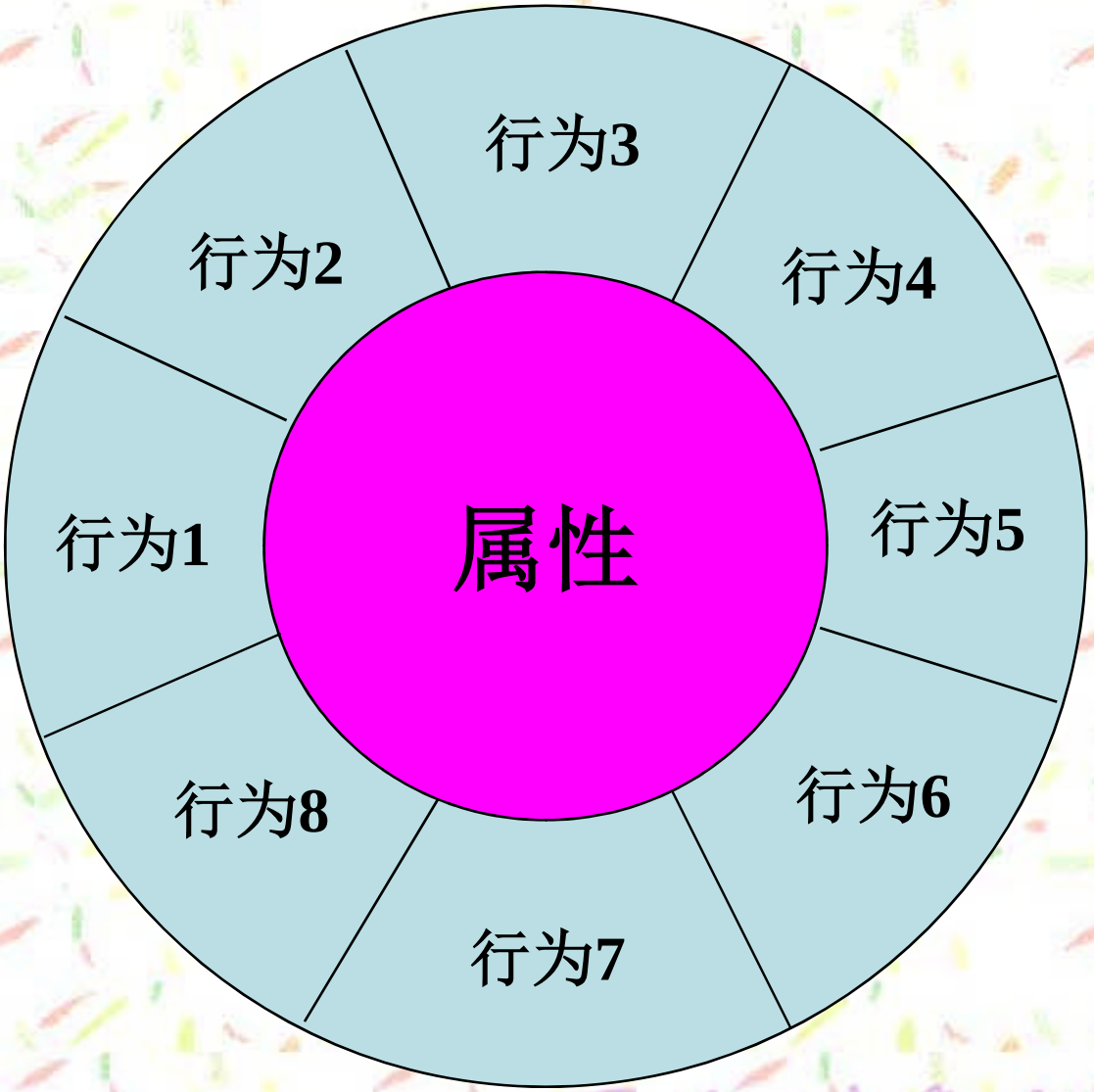
抽象是解决任何问题所采用的基本策略，是人类认识世界的本能方式。所谓抽象是指从许多事物中，舍弃个别的、非本质的属性，抽取出共同的、本质的属性的过程，它是形成概念的必要手段。

抽象主要包括过程抽象和数据抽象两个部分。所谓过程抽象是指功能抽象，即舍弃个别的功能，抽取共同拥有的功能，

数据抽象是一种更高级别的抽象方法，它将现实世界中存在的客体作为抽象单元，其抽象内容既包括客体的属性特征，也包括行为特征，它是面向对象程序设计所采用的核心方法。模块化和信息隐蔽是数据抽象过程的两个主要概念。

## 封装——

封装是指将现实世界中某个客体的属性与行为聚集在一个逻辑单元内部的机制。利用这种机制可以将属性信息隐藏起来，外界只能够通过提供的特定行为接口改变或获取其属性状态。在面向对象的程序设计中，封装是指将对象的属性和行为分别用数据结构和方法描述，并将它们绑定在一起形成一个可供访问的基本逻辑单元。




面向对象程序设计

## 对象——

对象是用来描述现实世界中客体的部件，是面向对象软件系统在运行时刻的基本单位。为了区分属于同一个类的不同对象，每个对象都有一个惟一的标识。

# 对象应该具有下面5个基本特性：

- 自治性，指对象具有一定的独立操作能力；
- 封闭性，指对象具有信息隐蔽的能力；
- 通信性，指对象具有与其他对象通信的能力；
- 被动性，指对象的状态转换需由外界刺激引发；
- 暂存性，指对象的动态创建与消亡。

# 类——

- 类是一组具有相同属性特征的对象的抽象描述，是面向对象程序设计的又一个核心概念。
- 类是对象抽象的结果。有了类，对象就是类的具体化，是类的实例。类可以有子类，同样也可以有父类，从而构成类的层次结构。
- 类之间主要存在三种关系。它们是：关联、聚合和泛化。

## 消息——

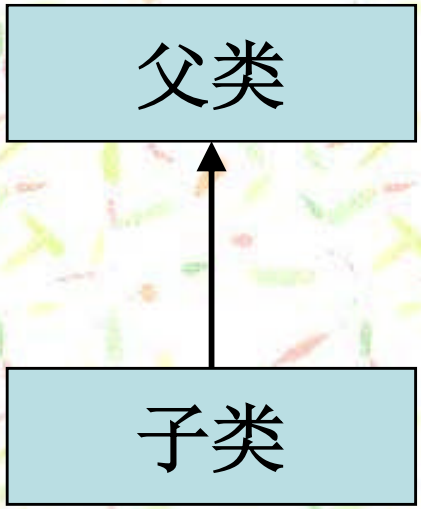
消息是一个对象要求另一个对象实施某项操作的请求。在一条消息中，需要包含消息的接收者和要求接收者执行哪项操作的请求，而并没有说明应该怎样做，具体的操作过程由接收者自行决定。

消息传递是对象之间相互联系的惟一途径。发送者发送消息，接收者通过调用相应的方法响应消息，这个过程被不断地重复，使得应用程序在人的有效控制下运转起来，最终得到相应的结果。可以说，消息是驱动面向对象程序运转的源泉。

面向对象程序设计

# 继承——

继承是类之间的一种常见关系。这种关系为共享数据和操作提供了一种良好的机制。通过继承，一个类的定义可以基于另外一个已经存在的类。继承是面向对象程序设计方法的一个重要标志，利用继承机制可以大大提高程序的可重用性和可扩充性。



## 多态性——

不同的类对象收到同一个消息可以产生完全不同的响应效果，这种现象叫做多态。利用多态机制，用户可以发送一个通用的消息，而实现的细节由接收对象自行决定，这样，同一个消息可能会导致调用不同的方法。

# 面向对象的4个特性

- 抽象性
- 封装性
- 继承性
- 多态性

# 1.4 面向对象程序设计语言

所谓面向对象程序设计语言**OOPL**

（**O**bject-**O**riented **P**rogramming **L**anguage）

是指提供描述面向对象方法所涉及到的类、对象、继承和多态等基本概念的程序设计语言。它应该具有下列特征：识别性、分类性、继承性和多态性。

# 几种有代表性的OOPL

- **Simula67，支持单继承、一定含义上的多态和部分动态联编。**
- **Smalltalk，支持单继承、多态和动态联编。**
- **Eiffel，支持多继承、多态和动态联编。**
- **C++，支持多继承、多态和部分动态联编。**
- **Java，提供了类机制，以及有效的接口模型。支持单继承、多态和动态联编。**


面向对象程序设计