04\_容器API-Collection\_4\_笔记

**一、J2SE常用类 ﻿**

* A computer screen with a green and white box

  AI-generated content may be incorrect.
* A computer screen shot of a computer screen

  AI-generated content may be incorrect.

**1. 面向对象概念**

**1）多态特性 ﻿**

* 核心机制：父类引用指向子类对象，通过"一个父类引用指向子类对象"实现程序弹性
* 实现原理：运行时动态绑定，实际调用方法由对象类型决定而非引用类型
* 思维训练：需要反复在脑中构建"父类引用=子类对象"的模型关系

**2）设计优势**

* 可扩展性：只需修改实例化语句（如"换成另外一个子类对象"），其他代码无需变更
* 现实类比：如同装修时铝线换铜线，仅需改动连接处，整体系统即获升级
* 代码弹性：体现为"其他地方不用变"的特性，使系统具备应对变化的能力
* 工业级价值：这种设计模式是专业级代码与业余代码的重要分水岭

**2. 面向对象设计**

**1）类关系组织**

* 设计重心：不同于面向过程，需重点考虑类属性/方法及类间关系
* 常见误区：避免"咣当就往外尿"式的面向过程写法，要建立真正的对象思维
* 架构艺术：类关系的组织方式直接影响系统质量，优秀设计应"很具有技巧性"
* 大师特征：精巧的类设计能实现"平凡与精巧的巨大区别"，需长期修炼

**3. 程序设计技巧**

**1）设计优劣对比**

* 学习路径：建议直接吸收"过来人的经验财富"，避免重复探索
* 复习方法：采用"边走边看"的碎片化复习，重点概念写卡片随时温习
* 质量判断：优秀设计应使修改局限在最小范围（"只用改这一句话"）
* 成长方向：通过观察"大师设计的东西"来提升设计敏感度

**二、知识小结**

|  |  |  |  |
| --- | --- | --- | --- |
| 知识点 | 核心内容 | 考试重点/易混淆点 | 难度系数 |
| 多态（Polymorphism） | 父类引用指向子类对象，实现代码弹性与可扩展性 | 多态的实现原理（父类引用如何调用子类方法） | ⭐⭐⭐⭐ |
| 面向对象设计核心 | 强调类设计（属性/方法）、类间关系与组织技巧 | 与面向过程的区别（避免“伪面向对象”设计） | ⭐⭐⭐⭐ |
| 代码复用与维护 | 通过多态实现“仅修改一处，整体升级”（如：铝线→铜线案例） | 设计模式的应用场景 | ⭐⭐⭐ |
| 学习方法建议 | 反复复习核心概念（如多态）、采纳经验性学习策略 | 实践与理论结合（避免重复探索已知经验） | ⭐⭐ |
| 优秀设计特征 | 精巧的类结构设计 vs 平凡设计的巨大差异 | 大师级设计的评判标准 | ⭐⭐⭐⭐ |