软件工程第四章

1. 设计工程概要介绍
    a. **软件设计的定义：软件系统或组件的架构、构件、接口和其它特性的定义过程及该过程的结果**
        - 软件工程生命周期中的一个活动
        - 进行软件编码的基础
        - 软件需求分析被转化为软件的内部结构
        - 是连接用户需求和软件技术的桥梁
    b. **设计工程的活动**
        - **软件架构设计（又称顶层设计、概要设计）：描述软件的顶层架构和组织，划分不同的组件**
        - **软件详细设计：详细描述各组件以便能够编码实现**
2. 设计过程和质量
    a. 好的设计应该有以下特点
        - 设计必须实现在分析模型中包含的所有明确要求，必须满足客户所期望的所有隐含要求；
        - 设计必须对编码人员、测试人员及后续的维护人员是可读可理解的；
        - 设计要提供该软件的完整视图，从实现的角度解决数据、功能及行为等各领域方面的问题。
    b. 设计质量属性：功能性、易用性、可靠性、性能、可支持性（包括扩展性、适应性、可维护性）
3. 设计技术：数据设计、架构设计、接口设计、组件设计
4. 设计相关概念
    a. 抽象
        i. **含义：是忽略具体信息将不同事物看成相同事物的过程**
        ii. 抽象机制：参数化、规范化
        iii. 规范化抽象
            - 数据抽象：描述数据对象的冠名数据集合（如门，包含们的类型、转动方向、重量和尺寸等）
            - 过程抽象：具有明确和有限功能的指令序列（一系列过程。如走到门前，伸出手并抓住把手，转动把手并拉门，离开打开的门等）
    b. **体系结构**
        i. **定义：软件的整体结构和这种结构为系统提供概念完整性的方式**
        ii. **体系结构设计可以使用大量的一种或多种模型表达：结构模型、框架模型、动态模型、过程模型、功能模型**
    c. 设计模式
        i. **含义：在给定上下文环境中一类共同问题的共同解决方案**
        ii. 微观结构：实体模式、结构模式、行为模式
    d. **模块化**
        i. **定义：软件被划分为命名和功能相对独立的多个组件（通常称为模块），通过这些组件的集成来满足问题的需求**
        ii. **模块化设计标准**

        - 模块化的分解性：可分解为子问题
        - 模块化的组合性：组装可重用的组件
        - 模块化的可理解性：可作为独立单元理解
        - 模块化的连续性：需求小变化只影响单个模块
        - 模块化的保护：模块内异常只影响自身
    e. 信息隐藏
        i. 原则
            - 模块应该具有彼此相互隐藏的特性，即：模块定义和设计时应当保证模块内的信息（过程和数据）不可以被不需要这些信息的其他模块访问
        ii. 特点
            - 抽象有助于定义构成软件的过程或实体
            - 信息隐藏原则定义和隐藏了模块内的过程细节和模块内的本地数据结构
    f. 功能独立
        i. 含义：每个模块只负责需求中特定的子功能，并且从程序结构的其他部分看，该模块具有简单的接口
        ii. 好处
            - 易于开发：功能被划分，接口被简化
            - 易于维护和测试：次生影响有限，错误传递减少，模块重用
        iii. 定性衡量标准：内聚性（模块的功能相对强度），耦合性（模块之间的相互依赖程度），模块独立性强=高耦合低内聚
    g. 精化：逐步求精的过程
    h. 重构：不改变组件功能和行为的条件下，简化组件设计（或代码）的一种重组技术
5. 面向结构化的设计方法：研究、分析数据流图->推导出初始结构图->得到符合要求的系统结构图->修改补充数据字典
6.