1. 软件工程概念
2. 将系统化的、规范化的、可度量的方法应用于软件的开发、运行和维护的过程，即将工程化应用于软件中；
3. b对a中所述方法的研究
4. 软件工程包含方法、工具和过程三个关键元素
5. 软件工程七条基本原理
6. 用分阶段的生命周期计划严格管理
7. 坚持进行阶段审评

实行严格的产品控制

1. 采纳现代程序设计技术
2. 结果应能清楚的审查
3. 开发小组的人员应少而精
4. 承认不断改进软件工程实践的必要性
5. 软件生存周期（p12-p14）
6. 软件过程4个基本模型
7. 瀑布模型(p15)
8. 原型算法模型
9. 迭代开发模型

螺旋模型：将工程划分为制定计划、风险分析、实施工程、用户评价四个主要活动

螺旋模型任务区

1. 客户交流。建立开发者和客户之间有效通信，正确定义需求
2. 计划。定义资源、进度及其他相关项目信息，即确定软件目标、选定实施方案、 弄清项目开发的限制条件。
3. 风险分析。评估技术的及管理的风险，即分析所选方案、考虑如何识别的消除 风险。
4. 工程。建立应用的一个或多个表示，即设计软件原型。
5. 设计与制作。构造、测试、安装和提供用户支持，即实施软件开发
6. 客户评价。基于对在工程阶段产生的或在安装阶段实现的软件表示的评估，获 得客户反馈，即评价开发工作、提出修正建议
7. 可转换模型
8. 面向对象基本原则
9. 一切事物都是对象
10. 任何系统都是由对象构成的，系统本身也是对象
11. 系统的发展和进化构成都是由系统的内部对象和外部对象之间(也包括内部对象与内部对象之间)相互作用完成的
12. 类概念

类是面向对象的一个基本概念，类封装了客观世界中的实体的特征和行为，即类的属性和方法两个方面

1. 继承与组合概念

继承是一个类从一个基类中继承所有的元素，而组合是一个类中包含另一个类的引用

1. UML定义

统一建模语言UML是一种用于描述、构造可视化和文档化软件系统的语言。

1. UML的11种图形
   1. 用例图
   2. 类图
   3. 行为图
   4. 状态图
   5. 活动图
   6. 交互图
   7. 顺序图
   8. 协作图
   9. 实现图
   10. 构件图
   11. 配置图
2. 用例图
   1. 概念：表示角色和用例之间的关系
   2. 画：<https://www.bilibili.com/video/BV1at411Z7Ni>
3. 类图
   1. 概念:一组静态的描述性模型元素互相连接的集合图
   2. 画：<https://www.bilibili.com/video/BV1ct411U7hk>
4. 需求分析4个过程
   1. 需求获取
   2. 需求分析
   3. 需求规格说明
   4. 需求评审
5. 需求的核心内容
   1. 为什么要设计这个系统
   2. 系统将由谁使用
   3. 系统要做什么
   4. 涉及那些信息
   5. 对解决方案是否由额外的限制
   6. 如何使用该系统
   7. 质量指标约束要达到何种程度
6. 需求按抽象程度分类
   1. 业务需求
   2. 用户需求
   3. 系统需求
   4. 软件设计需求
7. 常用需求获取技术
   1. 面谈
   2. 问卷调查
   3. 群体诱导技术
   4. 头脑风暴
   5. 参与观察
   6. 亲身实践
   7. 原型
   8. 情景分析
   9. 概念建模
   10. A/B测试
8. 系统：是指待开发的任何事物，包括软件、硬件或者过程
9. 参与者：用于表示使用系统的对象，是系统外部的一个实体，以某种方式参与用例的执 行过程
10. 用例：是参与者为达到某个目的而与系统进行的一系列交互，执行结果将为参与者提供 可度量的价值
11. 关系：用来描述参与者和用例间的关联，通常用一条直线表示这种交互关系
12. 用例图常用的4类关系
    1. 通信关系
    2. 泛化关系
    3. 包含关系
    4. 扩展关系
13. 用例建模基本步骤
    1. 找出系统的参与者和用例
    2. 区分用例的优先次序
    3. 详细描述每个用例
    4. 构造用户界面原型
    5. 构造用例图
14. 常用软件体系风格
    1. 以数据为中心的风格
    2. 调用/返回风格
    3. 独立构件风格
    4. 仓库风格
    5. 虚拟机风格
15. SOLID五大设计原则
    1. 单一职责原则(SRP)
    2. 开放-封闭原则(OCP)
    3. Liskov替换原则(LSP)
    4. 接口分离原则(ISP)
    5. 依赖倒置原则(DIP)
16. 常用类关系
    1. 关联关系
    2. 泛化关系
    3. 依赖关系
    4. 实现关系
17. 源程序文档化概念

在编写源程序中要注意标识符命名、注释及源程序的布局等几个方面

1. 程序复杂性概念

程序的复杂性主要是指模块内程序的复杂性，它反映了软件的可理解性、模块性、简洁性等属性

1. 代码审查概念

是一种用来确认方案设计和代码实现的质量保证机制，它通过阅读代码来检查源代码和编程规范符合性以及代码的质量

1. 代码审查主要内容(p136)
   1. 基本规范
   2. 程序逻辑
   3. 软件设计
2. 软件测试不同角度分类p143-p144
3. 测试用例定义

为了一个特定的目标而设计的一组测试输入、执行条件和预期结果，他的目的是测试某个程序的路径是否正确，或者何时程序是满足某个特定的需求

1. 黑盒测试p146
2. 白盒测试p149
3. 单元测试概念

单独测试各模块，验证程序模块和详细设计是否一致，消除程序模块内部逻辑和功能上的错误及缺陷

1. 集成测试概念

将已测试的模块组装进行检测，对照软件设计检测和排除子系统或系统结构上的错误，一般采用黑盒测试

1. 系统测试概念

检测软件系统运行时与其他相关要素的协调工作情况是否满足要求，包括强度测试、性能测试、恢复测试和安全测试等内容

37.回归测试概念

在系统中重新运行先前的测试用例，以确保排除故障的集成测试又称为回归测试

38.团队组织与管理四个方面

* 1. 人力资源规划
  2. 项目团队组建
  3. 项目团队建设
  4. 项目团队管理

1. 团队建设四个阶段
   1. 萌芽阶段
   2. 磨合阶段
   3. 团队规范阶段
   4. 创造阶段
2. 常见沟通方法
   1. 面对面口头沟通
   2. 书面沟通
   3. 电话沟通
   4. 网路沟通
3. 项目估算常用方法
   1. 专家判断发

概念：又称德尔菲法，是一种常见的估算方法，主要是依赖专家的经验和类似项目的历史信息，对具体任务做出时间或者成本的估算

* 1. 参数估算法

1. Scrum框架开发团队的三种角色
   1. 产品负责人
   2. Scrum主管
   3. Scrum团队
2. 燃尽图概念

是以图形方式显示迭代过程中累计剩余的工作量，它是反映工作量完成状况的趋势图

1. 软件配置管理概念

简称SCM，是一种标识、组织和控制修改的技术，它作用于整个软件生命周期，目的使错误达到最小，并且能够有效的提高开发效率

1. 软件配置管理三部分
   1. 版本管理
   2. 问题跟踪
   3. 建立管理
2. 基线概念

是软件生存期中各个开发阶段末尾的特点，又称里程碑