**英文版《软件工程》教学内容回顾2018下**

**(下述问题仅是课件中的主要部分回顾，问题答案以课件为主要参考)**

**Chapter01**

1. **SE的定义、目的、方法及作用:**

**SE：**在将有关软件开发与应用的概念科学体系化的基础上，研究如何有计划、有效率、 经济的开发和利用能在就算机上正确运行的软件的理论和技术的工程方法学，一些开发和维护 软件的方法、过程、原则。**是一个系统工程，既有对技术问题的分析与综合，也有对开发过程和参与者的管理。**

**SE的方向：**面向对象模式，结构化模式，基于过程的模式等

**SE的作用：**付出较低的开发成本，达到要求的软件功能，取得较好的软件性能，开发的软件易于移植，需要较低的维护费用，能按时完成开发工作，及时交付使用。

**//** 开发模式（paradiam）

1. **说明错误、缺陷、失败的含义与联系。（请举例说明）**

**错误：** 软件生产中认为的错误（误解需求，错误代码等）

**缺陷：** 功能实现中的问题 （一个错误----几个故障）（静态存在）

**失败：** 运行软件失败（因缺陷）（动态存在）

单个错误可能产生多个故障。故障是系统的内部视图，这是从开发人员的角度看待系统；而失效是系统的外部视图，它是用户所看到的问题。并非每一个故障都对应于一个失效（不执行故障代码就不会是代码失效）。

1. **软件质量应从哪几个方面来衡量？论述之。\***

**产品的质量**

用户（功能齐全，易学易用）

  开发人员（内部特征）

**过程的质量**

**有很多活动会影响到最终的产品质量**。只要活动出了差错，产品的质量就会受到影响。因此，许多软件工程师认为开发和维护过程的质量与产品的质量是同等重要的

**商业应用背景下的软件质量（商业质量）**

机构对软件是否与其战略利益相吻合的一种价值评估，目标**将技术价值与商业价值统一起来**

// 软件系统的系统组成。

****A system = entities(实体) + activities(活动) + relationships(关系) + boundary(边界)****

1. **现代软件工程大致包含的几个阶段及各个阶段文档。**

（1）需求分析：:《SRS》软件需求规格说明书

  （2）系统设计：系统结构图SAD

  （3）程序设计：包括模块功能算法与数据描述

  （4）程序实现：主要包括编程的代码和注释

  （5）单元测试：模块测试与性能测试 测试报告

  （6）集成测试：按照结构图SAD进行测试产生测试报告

  （7）系统测试：按SRS对系统总体功能进行测试

  （8）系统提交：交付产品

  （9）维护：修改软件的过程，为满足改错或新需求

//使现代SE实践发生变化的（七个）关键因素是什么？

1. **什么是软件过程？软件过程的重要性是什么？包含几个阶段？**

**软件过程:** 软件开发活动中的各种组织及规范方法

上述阶段

1. **什么是重用、抽象等现代软件工程主要概念？\*\***

**重用:**

重复采用以前开发的软件系统中具有共性的部件, 用到新的开发项目中去.(注: 这里的重用绝不仅仅是源代码的重用)

**抽象:-**

基于某种层次归纳水平的问题描述。它使我们将注意力集中在问题的关键方面而非细节

**Chaoter02**

1. **什么是软件过程？软件过程的重要性是什么？软件生命周期？\*\***

软件过程： 软件开发活动中产生某种期望结果的一系列有序任务，涉及活动、约束和资源.

上述阶段

通用性 （ 一致性 结构性 ）自我指导性

 重要性：（1）它强制活动具有一致性和一定的结构，使程序的集合组合起来以产生满足目标和标准的产品，（2）过程结构允许我们分析、理解、控制和改进组成过程的活动，并以此来指导我们的行动（3）它能使我们获取经验并把它创收给他人。

1. **瀑布模型及各阶段文档，优缺点？**

需求分析**：**《SRS》软件需求规格说明书

系统设计 ：系统设计文档《SAD》

程序设计：模块功能算法和数据描述文档

编码 ：源程序和注释

单元测试和集成测试：单元测试报告

系统测试 ：系统测试报告

验收测试：验收测试报告

运行与维护：维护报告

优点：

描述规范软件开发活动，每一个过程活动都有与其相关联的里程碑和可交付产品; （项目经理可以衡量进展情况）

简单性：很容易向用户解释

其他复杂模型的基础： 如加入反馈循环及额外的活动

缺点：

面临软件变动时, 该模型无法处理实际过程中的重复开发问题（软件是一个

创造的过程, 不是一个制造的过程）

当时的文档转换有困难

1. **原型的概念与用途。**

一种部分开发的产品，用来让用户和开发者共同研究，提出意见，为最终产品定型

降低开发时的风险和不确定性

1. **论述分阶段开发模型的含义, 其基本分类及特点是什么？**

对原型化模型的改进，系统被设计成部分提交, 每次用户只能得到部分功能, 而其他部分处于开发过程中。

分类：

**增量式开发：**统需求按照功能分成若干子系统，开始建造的版本

是规模小的、部分功能的系统，后续版本添加包含新

功能的子系统，最后版本是包含全部功能的子系统集.

**迭代式开发：**

系统开始就提供了整体功能框架，后续版本陆续增强

各个子系统，最后版本使各个子系统的功能达到最强.

特点：

（1）即使还缺少某些功能，但在早期的发布中就可以开始培训。

       （2）可以及早为那些以前从未提供的功能开拓市场。

（3）快速修复问题未预料到的问题

       （4）针对不同的发布版本，开发团队将重点放在不同的专业领域技术上。

1. **螺旋模型四个象限的任务及四重循环的含义？**

**任务**

计划

目标/可选方案

风险评估

开发和测试

**四重循环：**

操作概念

软件需求

软件设计

系统实现与部署运行

螺旋模型的每一次迭代都根据需求和约束进行风险分析，以权衡不同的选择，并且在确定某一特定选择之前，通过原型化验证可行性或期望度。当风险确认之后，项目经理必须决定如何消除或最小化风险。

// ------ 习题2， 3。

// 在所有的软件开发过程模型中，你认为哪些过程给予你最大的灵活性以应对需求的变更？

1. **什么是UP， RUP，进化式迭代等市场流行的过程模型?**

**UP ：**统一过程 可以用三句话来表达：它是用例驱动的、以基本架构为中心的、迭代式和增量性的软件开发过程框架，它使用对象管理组织（OMG）的UML 并与对象管理组织（OMG）的软件过程工程原模型（SPEM）等相兼容。

**RUP**是IBM提供支持和包装的UP系统。

**进化式迭代开发**（Iterative development）

迭代开发是统一开发过程(RUP)的关键实践

开发被组织成一系列固定的短期小项目

每次迭代都产生经过测试、集成并可执行的局部系统

每次迭代都具有各自的需求分析、设计、实现和测试

随着时间和一次次迭代，系统增量式完善

**Chapter03**

1. **什么是项目进度？活动？里程碑？**

**项目进度：** 是对特定项目的软件开发周期的刻画。包括对项目阶段、步骤、活动的分解，对各个离散活动的交互关系的描述，以及对各个活动完成时间及整个项目完成时间的初步估算。

**活动：** 项目的一部分, 一般占用项目进度计划中的一段时间。

**里程碑：** 指特定的时间点, 标志着活动的结束, 通常伴随着提交物

1. **如何计算软件项目活动图的关键路径？（习题2，3）冗余时间？最早和最迟开始时间（课堂习题讲解）**

关键路径将能够标明或计算出完成整个项目所需的最少时间

// 软件团队人员应该具备的能力是什么？

1. **软件项目团队组织的基本结构？**

主程序员

|

副主程序员

|

高级程序员 资料员 管理机构 测试小组

|

初级程序员

1. **专家估算法的大致含义？算式估算法的大致含义？**

**专家估算法：** 依赖于专家的判断，使用专家的知识和经验，对软件项目的工作量进行评估，预测的精确性基于估算者的能力、经验、客观性和洞察力。是对构建整个系统或其子系统所需的工作量做出经验性的猜测。

**算式估算法**

E = (a + bSc) m(X)

1. **试述COCOMO模型的三个阶段基本工作原理或含义。**

阶段一： 项目通常构建原型以解决包含用户界面，软件和系统交互，性能和技术成熟性等高风险问题

阶段二： 设计人员必须研究几种可选的体系结构和操作概念

阶段三： 项目正在开发，软件部分功能已经实现。

1. **什么是软件风险？ 了解主要风险管理活动？有几种降低风险的策略？**

**软件风险：** 在软件生产过程中不希望看到的、有负面结果的事件

**风险估算： 风险识别 风险分析 风险优先级分配**

**风险控制： 风险降低 风险管理计划 风险化解**

**三种策略来降低风险：**

（1）通过改变性能或功能需求，避免风险

    （2）通过把风险分配到其他系统中，或者购买保险以便在风险成为事实时弥补经济上的损失，从而转移风险。

（3）假设风险会发生，接受并用项目资源控制风险。

1. **找出课后练习题图3.23和3.24的关键路径。**

**最短选大 最长选小**

**Chapter04**

1. **需求的含义是什么?**

是对来自用户的关于软件系统的期望行为的综合描述, 涉及系统的对象、状态、约束，功能等

1. **需求作为一个工程，其确定需求的过程是什么？**

原始需求获取

问题分析

规格说明草稿

需求核准

软件需求规格化说明 SRS

1. **举例说明获取需求时，若有冲突发生时，如何考虑根据优先级进行需求分类。**

绝对要满足的需求（必须的）

非常值得要的但并非必须的需求（值得要的）

可要可不要的需求（可选的）

// 如何使需求变得可测试？（sidebar4.4）

1. **需求文档分为哪两类？**

****需求定义****，它面向的是业务相关的人员，例如：委托人，客户以及用户；

****需求规格说明****，它面向的是技术性人员，例如：设计人员、测试人员以及项目经理。

1. **什么是功能性需求和非功能性需求/质量需求？ 设计约束？过程约束？如何区分？**

**功能需求：**描述系统内部功能或系统与外部环境的交互作用,涉及系统输入应对,实体状态变化,输出结果,设计约束与过程约束等.

**设计约束**：

物理环境：对环境或设备的限制等（安装及环境要求等）

接口：涉及输入输出的限制或约束条件. (输入格式预定等)

用户：使用者的基本情况（限定几种类型的用户）

**过程约束**：

资源 ：材料、人员技能或其它。

文档 ：类型、数量或其它。（涉及其针对性及要求等）

标准 ：比如阅读文档时的用户指派标准。

其他 ：什么原因会导致从工资单列表中删除某雇员？

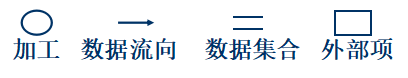
**非功能性需求/质量需求：**

描述软件方案必须具备的某些质量特征，如系统性能，安全性，反应时间。

// 需求的特性？（正确性、一致性、完整性）。

1. **了解DFD图的构成及画法。**

**数据流图DFD**：



描述数据如何流入系统，如何转换以及它们如何离开系统。 （重点始终是数据

流，而不是控制流）

// 在需求原型化方面，什么是抛弃型原型？什么是演化型原型？

// 用DFD图简单描述ATM机的工作原理（主要功能和数据流）（习题7）

**Chapter05**

1. **什么是软件体系结构？设计模式？设计公约？设计？ //概念设计？技术设计？**

**体系结构 :**一种软件解决方案，用于解释如何将系统分解为单元，以及单元如何相互关联，还包括这些单元的所有外部特性。

**设计模式:** 一种针对单个软件模块或少量模块而给出的一般性解决方案，它提供较低层次的设计决策。(此设计决策低于体系结构) (注: 此处为说明,不是定义)

**设计公约:**一系列设计决策和建议的集合，用于提高系统某方面的设计质量

**设计:** 将需求中的问题描述转变成软件解决方案的创造性过程

**概念设计**：告诉客户什么系统会做（软件架构和功能）

**技术设计**：告诉程序员如何系统会做----软件功能和界面 实现方法（编码器的参考 文件）

1. **软件设计过程模型的几个阶段？**

**初始建模:**尝试可能的分解：根据需求描述的系统的关键特性等确定软件体系结构风格，进行系统级别的决策。  
**分析** :主要关注软件系统的质量属性，各种约束等等。

**文档化:** 确定各个不同的模型视图

**复审 :**检查文档是否满足所有功能及质量需求。

**正式的 <SAD>** ：软件体系结构文档

// 三种设计层次极其关系？

//什么是模块化？什么是抽象？

1. **论述设计用户界面应考虑的问题。**

**寓意/比喻** :基本术语、图像和概念等

**思维模型:** 数据、功能、任务的组织与表示

**模型的导航规则:** 如何在数据、功能、活动和角色中移动及切换

**外观:**

**感觉**

**文化差异问题:** （信仰，价值观，道德规范，传统，风俗，传说）

国际设计/无偏差设计 剪裁界面

**用户爱好问题:** 制作替代界面，供有不同偏好的人选择

1. **5.5节----模块独立性----耦合与内聚的概念及各个层次划分？\***

**耦合:** 两个软件部件之间的相互关联程度

**内聚:** 软件部件内部各组成成分的关联程度

1. **举例说明耦合与内聚的基本分类。以及各个分类的含义与特征\***

（非直接耦合） ：模块相互之间没有信息传递

（数据耦合） ：模块间传递的是数据

（特征耦合）：模块间传递的是数据结构

（控制耦合）：模块间传递的是控制量

（公共耦合）：不同模块访问公共数据

（内容耦合）：一个模块直接修改另一个模块(A模块直接调用B模块的私有 数据, 或直接转移到B模块中去)

（偶然性内聚）：不相关的功能, 过程,数据等出现在同一个部件中.

（逻辑性内聚）：逻辑上相关或相似的功能或数据放置在同一个部件内

（时间性内聚）：部件各部分要求在同一时间完成

（过程性内聚）：各部分有特定次序。

（通讯性内聚）：各个部分访问共享数据

（顺序性内聚）：各部分有输入输出关系。

（功能性内聚）：各部分组成单一功能

1. **软件过程中复审的概念，设计复审的重要性。\***

**复审:** 检查文档是否满足所有功能及质量需求。

**Chapter06**

// 什么是面向对象？OO有几个基本特征？如何使用高级语言实现这些基本// 特征？

1. **了解并使用高级语言的OO基本编程方法和技巧。**

是一种软件开发方法，它将问题及其解决方法组织成一系列独立的对象，数据结构和动作都被包括在内。

1. **什么是设计模式？**

是指把系统功能和行为分解成模板的指导方针

1. **OO设计的基本原则？**

单一职责原则 重用原则 开闭原则 替换原则 依赖倒置原则

接口隔离原则 迪米特法则

1. **OO开发有何优势？**

**描述语言角度**

用相同的术语（相同的语义结构）描述问题和解决方案：类，对象，接口，方法，属性和行为

**软件过程的角度**

从制作要求，高级设计，初级设计，编码到测试，所有过程都使用相同的语义结构

1. **OO开发过程有几个步骤？**

OO需求分析

OO设计

OO编码和测试

1. **了解用例图的组成和画法，用例的几个要素的含义。 //掌握用例图的实例解析方法**

**用例：**描述了系统应该执行或展示的特定功能



执行者：与系统交互的实体 （用户或设备）



包含： 是一个已经定义的用例的重用（用以提取公共行为）



**扩展：**扩展用例来说明不同或更深入的视角

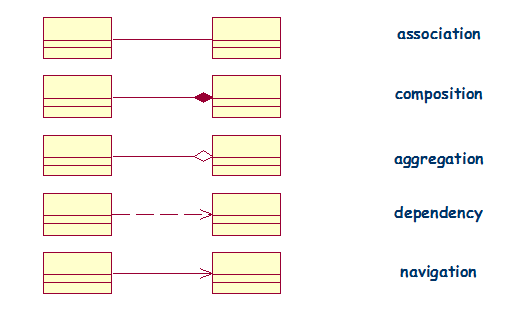


1. **用例图、类图等对面向对象的项目开发的意义是什么？**

阐明需求

有助于发现需求故障

1. **熟悉类图中各个类之间的基本关系分类及其含义，状态图的含义及用途。**



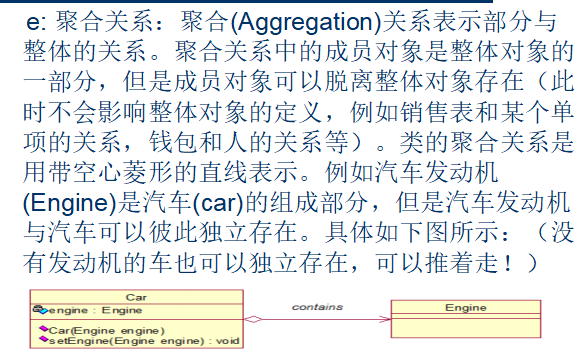
继承/泛化 

关联关系

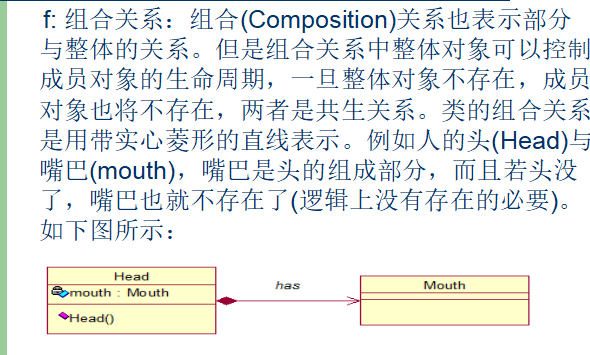
在使用java、C#或C++实现关联时，通常将一个类的

对象作为另一个类的成员变量，如下图：

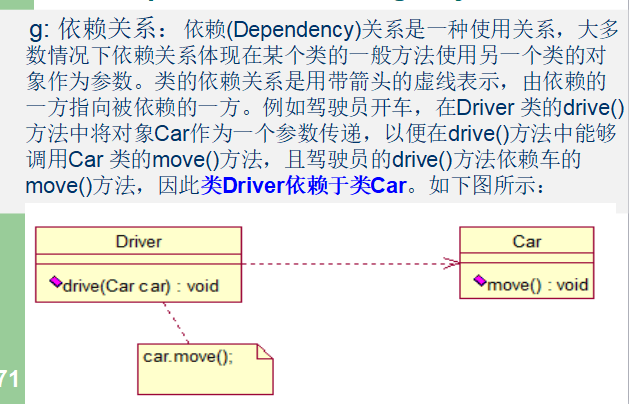
聚合关系：



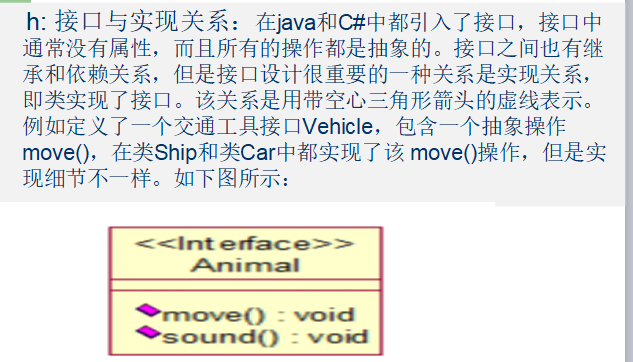
组合关系



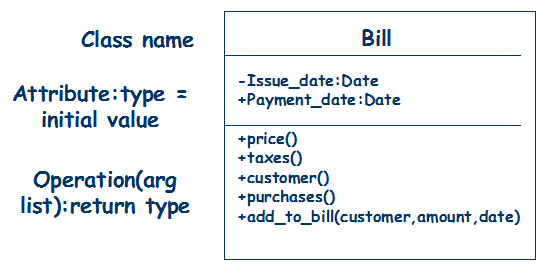
依赖关系



接口实现关系



1. **熟悉用例图、类图、状态图等的组成和画法。**



1. **了解UML其他图示结构的基本用途。**

**活动图：** 描述程序的活动和流程

包图： 显示包或类之间的依赖关系

时序图：

合作图

状态图

UML模型图（5类，10种）：

用例图

静态图（类图，对象图，包图）

行为图（状态图，活动图）

交互图（顺序图，合作图）

实现图（构件图，配置图）

**Chapter07**

//为什么说编码工作是纷繁复杂甚至令人气馁？

1. **一般性的编程原则应该从哪三个方面考虑？**

**编程标准对自身的用处**

**编程标准对他人的用处**

**设计与编程实现相匹配**

//论述编码阶段实现某种算法时所涉及的问题。

1. **在编写程序内部文档时，除了HCB外，还应添加什么注释信息？注意什么？**

其他程序注释

分阶段注释

代码更改伴随注释更新

注释应该有新的信息

写代码时同时写注释、

2.有意义的变量名和语句标记、3.安排格式以增强理解、4.文档化数据

1. **什么是极限编程(XP)? 以及派对编程？**

**极限编程（XP）**是一种轻量级的软件开发方法论，属于敏捷开发方法。XP从实践中来，是对实践的总结，也是经过实践检验的，其主要特征是要适应环境变化和需求变化，充分发挥开发人员的主动精神。XP承诺降低软件项目风险，改善业务变化的反应能力，提高开发期间的生产力，为软件开发过程增加乐趣等等

**派对编程**属于主要的敏捷开发方法，其开发方式是两个程序员共同开发程序，且角色分工明确。一个负责编写程序，另一个负责复审与测试。两人定期交换角色。

**Chapter08**

// 产生软件缺陷的原因？

// 将软件缺陷进行分类的理由？

1. **有几种主要的缺陷类型？**

**算法缺陷**

**计算和精度缺陷**

**文档缺陷**

**过载缺陷**

**能力缺陷**

**时序性缺陷**

**性能缺陷**

**恢复性缺陷**

**硬件和系统软件缺陷**

**代码的标准和规程缺陷**

1. **什么是正交缺陷分类？**

软件开发中的任何地方都存在故障

分类有助于减少数量故障

一个故障属于一个类别

1. **测试的各个阶段及其任务？（图8.3）**

**单元测试：验证组件功能**

**集成测试 验证系统组件共同工作**

**功能测试：验证系统功能需求**

**性能测试： 验证剩余部分功能需求**

**验收测试：客户需求规格说明**

**安装测试：在实际环境中检查系统**

// 测试的态度问题？（为什么要独立设置测试团队？）

1. **掌握测试的方法----黑盒、白盒的概念？**

**黑盒;**  
内容/结构未知，仅测试测试对象的功能。也就是说，测试输入输入到黑盒并

记下产生的输出（组件功能是测试的基础）

(另外：测试时应该考虑让被测模块完成一切应做的事情, 拒绝一切不应做的事情）

(注：黑盒测试的主要参考文档是系统设计和程序设计阶段文档。若是可重用部件，则是类似系统）

**白盒**

使用测试对象的结构以不同的方式测试

1. **什么是单元测试？ 什么是走查和检查？**

将每个程序组件与系统中的其他组件隔离，分开测试。

**走查：**

非正式的代码复审，

**检查**

正式的代码复审，审查小组根据准备好的关注点列表检查代码和文档

1. **黑盒白盒方法各自的分类？测试用例的设计和给出方法（结合补充材料）\*\*\***

**黑盒：**  等价分类法 边界值分析法 错误猜测法 因果图法

**白盒：**  逻辑覆盖法 路径测试法

1. **黑盒白盒方法的分类，各种覆盖方法等。（课件和补充课件）**

**语句覆盖**：使被测程序的每条语句至少执行一次；

**判定覆盖**：使被测程序的每一分支至少执行一次，故又称分支覆盖；

**条件覆盖**：要求判定中的每个条件均按“真”、“假”两种结果至少执行一次。

1. **如何面对一个命题，设计和给出测试用例的问题。（课件）**

------课堂练习的测试题目和讲解内容

1. **集成测试及其主要方法的分类？（驱动，桩的概念）**

**由底向上集成测试**

**自顶向下集成测试**

**莽撞测试**

**混合方式测试**

**整合策略比较**

**驱动模块: 代替上级模块传递测试用例的程序**

**桩模块：代替下级模块的仿真程序**

// 传统测试和OO测试有何不同？OO测试有何困难？

// 测试计划涉及的几个步骤？ （**了解**）

**Chapter09**

1. **系统测试的主要步骤及各自含义？（图9.2）**
2. **功能测试：**
3. **性能测试**
4. **验收测试**
5. **安装测试**

// 什么是系统配置？软件配置管理？ // 基线？（或见课件）

1. **什么是回归测试？**

应用于新版本的测试（验证旧功能和新功能的正确性）（大型系统的配置更动测

试只是“部分”回归测试）

1. **功能测试的含义极其作用？**

根据SRS测试系统功能性需求

1. **功能测试的基本指导原则？**

A: 较高的查错概率

B: 独立的测试团队

C: 了解预期的输出结果

D: 对合法与非法的输入都予以测试（假设是弱健壮等价类）

E: 不能仅仅为了测试的方便而修改系统

F: 停止测试应该有前提条件

1. **性能测试的含义与作用？**

测试系统的非功能性需求根据SRS

1. **性能测试的主要分类？**

**压力测试 / 强度测试：** (在短时间内加载极限负荷, 以验证系统能力)

**巨量数据测试 / 容量测试：** (验证系统处理巨量数据的能力)

**配置测试：**-构建测试用例对系统软硬件的各种配置(最小到最大)进行测试**。**

**兼容性测试 ：-**测试接口等

**回归测试**

**安全性测试**

**计时测试**

**环境测试**

**质量测试**

**恢复测试**

**维护测试**

**人为因素测试**

// 什么是可靠性、可用性和可维护性？

1. **确认测试概念，确认测试分类？（基准测试和引导测试）**

客户检查系统是否满足需求定义的需要

**基准测试**

A：客户准备测试用例（正式案例）

      B：在实验基础上安装

      C：客户评估性能

**引导测试**

1. **什么是alpha测试？β测试？**

**alpha测试：**让来自自己组织机构或公司的用户来测试这个系统，这样的内部测试（由开发人员）

**β测试：**由客户的测试 β测试

1. **什么是安装测试？**

在用户环境中进行集成策略测试的比较（以解决由开发环境和用户环境之间的差异引起的问题）

**注意：**

**每一章节的开头中，大的概念性问题是如何引入的？，其讨论请见课件。**

**试卷答题须知：**

**1．软件工程课程出题覆盖范围比较广泛，考察手段有灵活和多样化特点。**

**2．有考察学生软件工程运用能力的题目存在，难题占的分数很少。其他很多是送分的概念型题目。**

**3．鉴于有难度的题目占极少部分，可以根据时间安排，先回答别的题目。**

**4．所有题目都要写到试卷的指定位置，以免流水阅卷时有遗漏。**