**英文版《软件工程》教学内容回顾2019**

**(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**

**SE的定义、目的、方法及作用**

定义：用系统科学的方法解决问题，采用工具、技术等用来解决现实问题的综合过程

目的：生产出高质量的软件进而找到解决方案，并考虑那些对质量有影响的特性

方法及作用：

分析，分析问题，调查软件正反两方面，

设计，给出解决方案

开发团队，描述在团队中的人员的角色和职责

开发，实现解决方案

项目管理。将系统分为小部分，逐步明确过程，控制进度，处理每个改变等等

**说明错误、缺陷、失败的含义与联系。（请举例说明）**

含义：

错误：软件开发过程中人为产生的错误。（比如，需求文档中对错误理解，代码的编写错误）

缺陷（故障）：在开发活动的过程中出错后，就有可能引起故障，比如设计人员错误理解了需求后给出了错误的设计

失败：在运行时，软件违背了它应该有的行为。（比如，使用者发现某个计算功能算不出结果，是动态的）

联系：

人为原因导致错误，错误可能导致软件缺陷，用户在使用这样的软件时，因为软件缺陷而可能导致软件失败。

缺陷是系统的内部视图，是开发人员的视角。

失败是系统的外部视图，是用户的视角。

**软件质量应从哪几个方面来衡量？论述之。**

产品质量，过程的质量，商业环境背景下的软件质量。

1、产品质量：

衡量标准有两方面：用户和开发者。

用户：足够的功能，上手简单，易用。

开发者：从软件的内部特征来衡量，比如缺陷的数量。

2、过程质量（软件生产过程的质量）：

很多活动都会影响过程的质量，并且最终影响到软件的质量。开发和维护过程的质量和产品的质量是同等重要的。

量化方法：CMM等。

3、商业质量：

考虑的是商业环境中，产品的技术价值。

目标：将技术价值和商业价值进行统一。

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 SAD 系统结构图
3. 程序设计 模块功能算法与数据描述
4. 程序实现 源码和注释
5. 单元测试 单元测试报告
6. 集成测试 集成测试报告文档
7. 系统测试 系统测试报告文档
8. 系统交付 用户手册
9. 维护 维护记录文档

**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。可以将一组有序的任务称为过程，它涉及活动、约束和资源使用的一系列步骤，用于产生某种想要的输出。

软件过程的重要性：过程之所以重要，是因为它强制活动具有一致性和一定的结构性和指导性，它允许我们分析、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。

软件过程的阶段：9个（就是软件工程的9个阶段。）每一阶段本身就可以描述为一组活动的过程，并且每一个活动都包括约束、输出、和资源。每个活动涉及输入，子活动，约束，输出，和资源。

**什么是重用、抽象等现代软件工程主要概念？**

抽象：在某种概况层次上的问题的描述，是人们可以集中于问题的关键方面而不必陷入细节。

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。

分析、设计方法和符号描述系统：使用标准来对程序进行描述，利于交流和建模并检查其完整性和一致性，利于重用。

用户界面原型化：建立系统的小型版, 通常具有有限的关键功能,以利于用户评价和选择，证明设计或方法的可行性。

测度和度量：通用的评价方法和体系，有助于使过程和产品的特定特性更加可见，包括量化描述系统、量化审核系统。

工具和集成环境：通过框架比较软件工程环境提供的服务，以决定其好坏。

**Chaoter02**

**什么是软件过程？软件过程的重要性是什么？软件生命周期？**

软件工程：将软件开发中的一组有序的任务称为软件过程，它涉及活动、约束和资源使用的一系列步骤，用于产生某种想要的输出。

重要性：（1）它强制活动具有一致性和一定的结构，是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、控制盒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传授给他人。

软件生命周期：当过程是在构建某些产品时，有时把这种过程成为生命周期。因为软件开发过程描述了软件产品从概念到实现等的整个过程，因此，有时把软件开发过程称为软件生命周期。

需求分析

系统设计

程序设计

软件开发

单元测试

集成测试

系统测试

系统提交

维护

**瀑布模型及各阶段文档，优缺点？**

特点：瀑布模型将开发阶段描述为从一个阶段瀑布般转到另外一个阶段。一个开发阶段必须在另一个开发阶段之前完成。

优点：可强迫开发人员用规范的方法。严格地规定了每个阶段必须提交的文档。要求每个阶段交出的产品都必须经过质量保证小组的仔细验证。

缺点：瀑布模型最大的问题是它并不能反映实际的代码开发方式。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，文档转换有困难。

需求分析——SRS

系统设计——SAD

程序设计——算法和数据描述文档

编码——源程序及注释

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

**原型的概念与用途。**

概念：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，为最终的产品定型。

用途：

1、原型化设计可以使开发者更容易地提高软件质量。

2、原型化设计可以提供多种解决方案供用户选择。

**论述分阶段开发模型的含义, 其基本分类及特点是什么？**

定义：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。

**基本分类：**

增量式开发：系统需求按照功能划分成若干子系统，开始建造的版本是规模小的，部分功能的系统，后续版本不断更新新的功能

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的功能

特点：每个软件版本的周期减少

**螺旋模型四个象限的任务及四重循环的含义？**

任务：确定目标，可选方案及约束， 风险评估，开发和测试，计划

循环：

第一次循环得到操作概念，

第二次循环得到软件需求，

第三次循环中系统开发产生设计，

第四次循环能够进行测试

// ------ 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？

**什么是UP， RUP，进化式迭代等市场流行的过程模型?**

UP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。

RUP模型是IBM提出的提供支持和包装的UP模型。

迭代开发是统一过程模型（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**

**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：特定的时间点，标志着活动的结束，通常伴随着提交物。

**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）**

**// 软件团队人员应该具备的能力是什么？**

**软件项目团队组织的基本结构？**

1、主程序员负责制

由一个主程序员主导系统的设计和开发。

优势：交流最小化，迅速做出决定，效率高。

劣势：主观性强，对主程序员要求高。

2、忘我方法

每个成员平等的承担责任，批评和表扬只针对产品，不针对个人。

一般软件项目的团队组织结构是介于以上两个极端之间的。

**//专家估算法的大致含义？算式估算法的大致含义？**

**试述COCOMO模型的三个阶段基本工作原理或含义。**

COCOMO的估算过程反应了软件开发项目的三个主要阶段。

阶段一，计划阶段，项目通常构建原型以解决包含用户界面等高风险问题，人们对最终产品可能知之甚少，所以使用应用点来估计规模。

阶段二，早期设计阶段，使用功能点作为对规模的度量，比应用点提供更丰富的系统描述。

阶段三，后体系结构阶段，规模可以用功能点或代码行的形式来表述

**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：**

一种具有负面后果，人们不希望发生的事情。

**风险管理活动：**

风险评价：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

**策略：**

避免风险：改变性能或者功能需求

转移风险：把风险分配到其他系统中，或者购买保险

假设风险：接受并用项目资源控制风险

**弄懂活动图基本原理（参考课本），找出课后练习题图3.23和3.24的关键路径。**

**Chapter04**

**需求的含义是什么?**

需求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。

**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：收集用户需求。

问题分析：理解和建模期望的形为

规格说明草稿：文档化要开发的软件系统的行为

需求核准：检查我们的规格说明是否与用户的需求匹配

正式需求文档：SRS软件需求规格说明书

**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）

非常值得要但并非必须的需求（值得要的）

可要可不要的需求（可选的）

受到了技术、开发时间和资源等诸多因素的限制

举例：信用卡记账系统必须能够列出最近的费用，将他们加起来并要求在某个日期前支付，这是必须的需求。但是，该记账系统也可能按照购买类型区分费用，以帮助消费者理解购买的模式，这是值得要的需求。最后，记账系统可能要求用黑色来打印贷方账目，用红颜色打印借方账目，这用需求是有用的，但它是可选的需求。

按照类型对需求进行优先级的分类，能够帮助所有相关人员理解自己到底需要什么。当软件开发项目受到时间或资源的限制时，如果系统的成本太高或者开发的时间太长，就可以去掉可选需求，并对值得要的需求进行分析，考虑是去掉还是延期。还可解决与质量需求之间的矛盾。

// 如何使需求变得可测试？（sidebar4.4）

**需求文档分为哪两类？**

1. 需求定义：是客户想要的每一件事情的完整列表
2. 需求规格说明：将需求重新陈述为关于要构建的系统将如何运转的规格说明

**什么是功能性需求和非功能性需求/质量需求？**

功能性需求：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。

非功能需求/质量需求：描述软件方案必须具备的某些质量特征

**设计约束？过程约束？如何区分？**

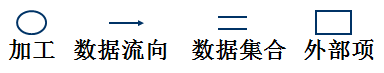
设计约束：是已经做出的设计决策或限制问题解决方案集的设计决策。

过程约束：是对用于构建系统的技术和资源的限制。

// 需求的特性？（正确性、一致性、完整性）。

**了解**DFD图的构成及画法。

DFD：Data Flow Diagrams 数据流图



// 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**

**什么是软件体系结构？**

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。

**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，对解决方案的描述也称为设计。

**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

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和质量需求

最后得到：

SAD，软件体系结构文档

**论述设计用户界面应考虑的问题。**

不同的人有着不同的感知，理解，工作等风格。需要考虑

1. 设计界面要注意解决的要素。

寓意/比喻：基本术语，图形和概念等

思维模型：数据，功能，任务的组织和表示

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：系统向用户提供的外观特性

感觉：向用户提供有吸引力的体验的交互技术。

1. 文化差异问题

信仰，价值观，道德规范，传统等。

1. 用户偏好问题

为具有不同偏好的人选择备选界面。

**5.5节----模块独立性----耦合与内聚的概念及各个层次划分？**

模块的独立性取决于两个部分：内聚和耦合，我们追求的是高内聚低耦合

耦合:两个软件部件之间的相互关联程度

高度耦合 松散耦合 非耦合

内聚：软件部件内部各组成成分的关联程度

**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**

耦合：指构件之间的相互依赖性，可分为

（1）内容耦合：一个构件直接修改了另外一个构件（当一个构件修改了另外一个构件的内部数据项时，或一个构件内的分支转移到另外一个构件中的时候，就可能出现内容耦合）

（2）公共耦合：不同构件访问公共数据。例如，一个公共变量可以被不同的构件修改

（3）控制耦合：某个构件通过传递参数来控制另外一个构件的活动，模块间传递的是控制量。

（4）标记耦合：用一个数据结构来从一个构件到另一个构件传送信息，而且传递的是该数据结构本身。

（5）数据耦合：构件间通过传递数据来完成信息的传递。

内聚：指构件内部的“粘合”程度，可分为：

（1）巧合内聚：构件的各部分互不相关

（2）逻辑内聚：几个逻辑相关的功能或数据元素放在同一个构件中

（3）时态内聚：构件顺序执行若干个功能，但是各功能只和涉及的时间相关

（4）过程内聚：构件中的功能组合在一起只是为了确保这个顺序

（5）通信内聚：将某些功能关联起来，因为它们是操作或生成同一个数据集的

（6）顺序内聚：一个构件的某部分的输出正好是下一部分的输入

（7）功能内聚：每一个处理元素对于执行单个功能来说都是必须的，并且在一个构件内包含了所有必需的元素。

**软件过程中复审的概念，设计复审的重要性。**

检查文档是否满足所有功能和质量需求。

重要性：

1、可以和用户一起检查软件的概要设计。

2、可以向开发者呈现并明确软件的技术设计。

3、程序员通过复审可以在下一阶段的工程实施前得到本阶段工作的反馈。

**Chapter06**

**掌握并使用高级语言的OO基本编程方法和技巧。**

**什么是设计模式？**

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。

**了解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

**了解OO开发有何优势？**

相对于传统的过程式开发，1语言的一致性，2 过程的一致性

**OO开发过程有几个步骤？**

1 面向对象需求

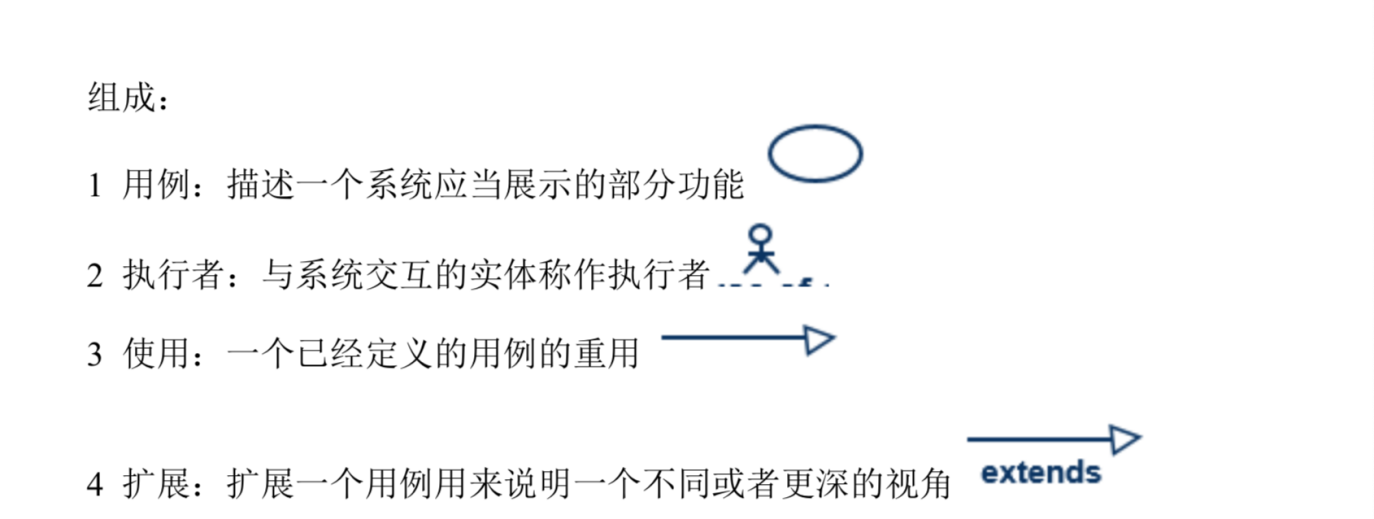
2 面向对象高层设计

3 面向对象底层设计

4 面向对象编程

5 面向对象测试

**掌握用例图的组成和画法，用例的几个要素的含义。**



用例图、类图等对面向对象的项目开发的意义是什么？

用例图的作用:

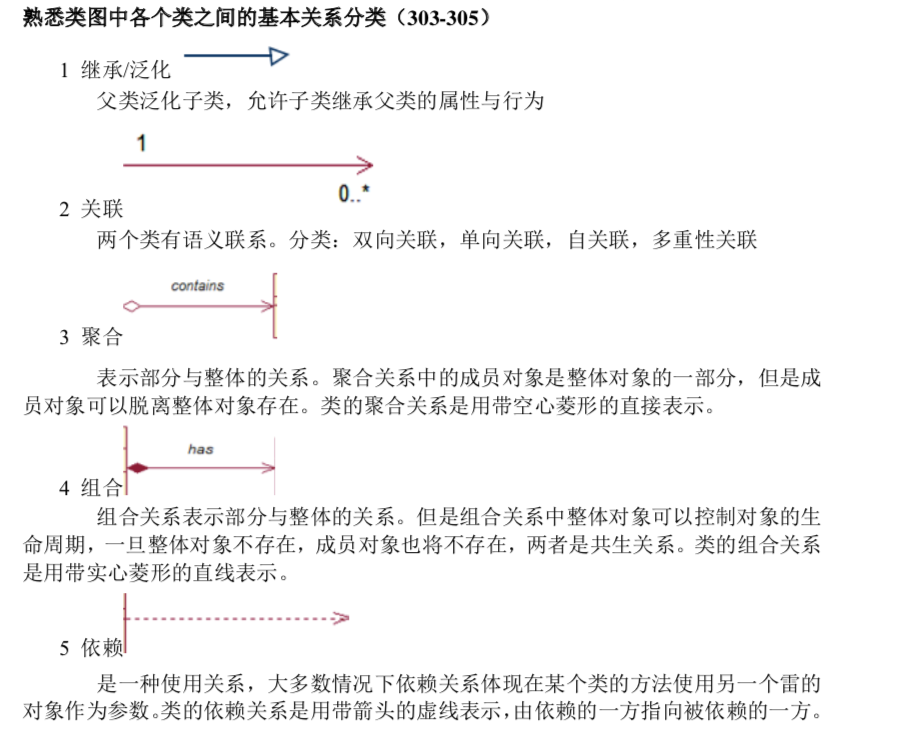
1、阐明需求。

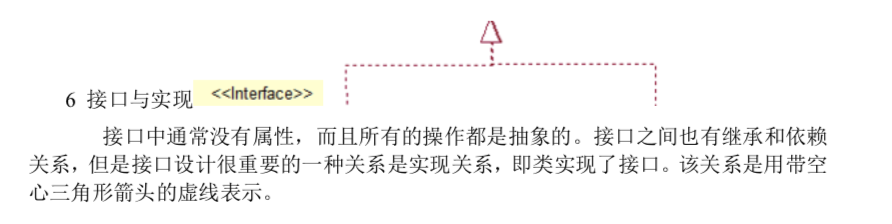
2、对寻找需求错误有帮助。

3、需求本身是复杂且难以描绘的。

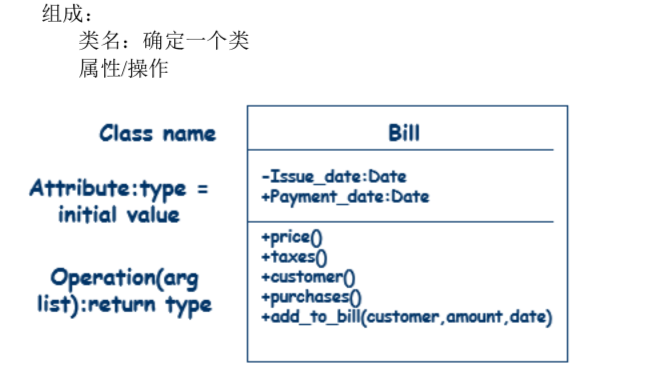
UML 是用来描述面向对象解决方案的一套很受欢迎的设计表示法。经过剪裁，它 可以适应于不同的开发情形和软件声明周期。UML 可以用来可视化、说明或文档化软 件设计。它在描述不同的设计选择，直至最终对设计工件进行文档化时是特别有用的。

**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。





熟悉用例图、类图、状态图等的组成和画法。



**了解**UML其他图示结构的基本用途。

**Chapter07**

**一般性的编程原则应该从哪三个方面考虑？**

1. 结构控制 程序如何传递数据。
2. 算法 程序如何处理数据。
3. 数据结构 程序如何储存数据。

**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

（1）解释性注释：本段源代码是在做什么的注释。

（2）分解性注释：通过注释将代码分解成多个段。

（3）版本注释：随着时间进行修改的记录。

注意的问题：

1、分段注释

2、注释和代码要一并更改。

3、注释要有意义。

4、一边写代码一边写注释，不要写完代码回过头来添加注释。

**什么是极限编程(XP)? 以及派对编程？**

极限编程是一种轻量级的软件开发方法，属于敏捷开发方法。XP的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，为软件开发增加乐趣等等。）

派对编程：是一种敏捷开发方法，两个程序员共同开发程序，角色分工明确。一个负责编写程序，另一个负责复审和测试。两人定期交换角色。

**Chapter08**

了解 产生软件缺陷的原因？

规格说明可能是错误的，或者遗漏了某个需求；

对于指定的硬件和软件，规格说明中可能包含不可能实现的需求

系统设计中可能包含故障。

程序设计中可能包含故障。

程序代码可能是错误的。

// 将软件缺陷进行分类的理由？

**有几种主要的缺陷类型？**

1.算法缺陷

算法的某些处理步骤或逻辑有问题，以至于软件部件对给定的输入数据无法产生正确的输出。

2.计算和精度缺陷

算法或公式在实现时出现错误，或计算结果的精度达不到要求。

3.文档缺陷

文档和程序不一致。

4.过载缺陷

程序运行时，对数据结构的使用超过了其承载能力。

5.能力缺陷

程序活动到达极限时，其性能会变得不可接受。

6.时序性缺陷

多个同时执行或者一个仔细定义的顺序执行的进程之间协调不当。

7.性能缺陷

正常条件下，系统性能达不到要求。

8.恢复性缺陷

系统运行失败后无法恢复。

9.硬件和系统软件缺陷

提供的硬件或者系统软件并没有按照文档中的操作条件或步骤运作。

10.代码的标准和规程缺陷。

代码没有遵循组织机构的标准和过程。

**什么是正交缺陷分类？**

被分类的任何一项缺陷都只属于一个类别

**测试的各个阶段及其任务？（图8.3）**

单元测试：验证构件能否适当运行

集成测试：验证系统构件之间是否能按照系统和程序设计规格说明描述的那样共同工作

功能测试：确定集成的系统是否确实执行了需求规格说明中描述的功能，其结果是一个可运转的系统。

性能测试：测试系统的软硬件性能是否符合需求规格说明文档。 其结果是一个确认的系统。

验收测试：确定系统是按照用户的期望运转的。

安装测试：确保系统在实际环境中按照应有的方式运转

**掌握测试的方法----黑盒、白盒的概念？**

黑盒测试：将测试的对象看作是一个不了解内容的黑盒，向黑盒中提供输入，记录输出。目的是确保针对每种输入，观察到的输出与预期的输出相匹配。一般参考的文档是系统设计和程序设计阶段的文档。

白盒测试：将测试对象看作一个透明的白盒，然后根据测试对象的结构用不同的方式进行测试。

**什么是单元测试？**

将程序构件和系统其他构件隔离，对其本身进行测试。

**什么是走查和检查？**

走查：程序员提供代码和文档，又评审小组评估正确性。以程序员为主导，是非正式的。

检查：与走查类似但是更正式，评审小组事先准备关注问题清单来检查代码和文档。

**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）**

**黑盒测试方法：**

等价分类法：将输入域划分为若干等价类，每个测试用例都代表了一类与之等价的其他例子。

边界值分析法：把测试值选在等价类的边界上进行测试

错误猜测法：猜测程序中哪些地方容易出错，并设计测试用例

因果图法：适用于被此时程序有很多输入条件，程序的输出又依赖输入条件的各种组合。

**白盒测试方法：**

**逻辑覆盖**

语句覆盖：程序的每条语句都要执行一次

判定覆盖：程序的每个分支都要执行一次

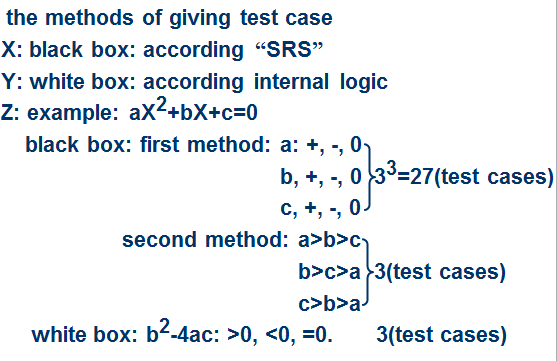
条件覆盖：要求判定中的每个条件均按照“真”“假”两种结果至少执行一次。

条件组合覆盖：要求所有条件结果的组合都至少出现一次。

**路径测试法**

节点覆盖，边覆盖，路径覆盖

**黑盒白盒方法的分类，各种覆盖方法等。（课件和补充课件）**



黑盒测试方法:  
1 等价分类法:把被测程序的输入域划分为若干等价类，把漫无边际的随机测试变成有针对性的等价类测试。分为弱一般等价类，强一般等价类，弱健壮等价类。

2 边界值分析法:在等价分类法中，代表一个类的测试数据可以在这个类的允 许范围内任意选择。而把测试值选在等价类的边界上，往往有更好的效果。同时适用于考察程序的输出值边界。  
3 错误猜测法:猜测被测程序中哪些地方容易出错，并据此设计测试用例。其更多地依赖与测试人员的直觉和经验。  
4 因果图法:借助图形来设计测试用例的一种系统方法，特别适用于被测程序具有多种输入条件，程序的输出又依赖于输入条件的各种组合情况。因果图是一种简化 了的逻辑图，能直观地表明程序输入条件(原因)和输出动作(结果)之间的相互关系。

白盒测试方法:  
1 逻辑覆盖法:一组覆盖方法的总称。按照由低到高对程序逻辑的覆盖程度，又可区分为以下几种覆盖方法:  
1)语句覆盖:使被测程序的每条语句至少执行一次 2)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 3)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 4)条件组合覆盖:最强的一种覆盖。它与条件覆盖的区别是，它不是简单地要求每个条件都出现“真”“假”两种结果，而是要求让这些结果的所有可能都至少出现一次  
2 路径测试法:是借助程序图设计测试用例的一种白盒方法。在本方法中，测试用例是基于流程图来设计的。程序图本质上是一种简化了的流程图，它保持了控制流 的全部轨迹，包括了所有的判定节点。 1)结点覆盖:程序的测试路径至少经过程序图中每个结点一次，相当于逻 辑覆盖法中的语句覆盖。 2)边覆盖:程序的测试路径至少经过程序图中每条边一次。它相当于逻辑 覆盖法的判定覆盖。 3)完全覆盖:同时满足结点覆盖和边覆盖的覆盖。 4)路径覆盖:程序图中每条路径都至少经过一次。

**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 在集成测试及其后的测试阶段，除去很小的程序，都采用黑盒方法。其策略包括: (1)用边值分析法和(或)等价分类法提出基本的测试用例; (2)用猜错法补充新的测试用例; (3)如果在程序的功能说明中含有输入条件的组合，宜在一开始就用因果图法，

然后再按以上(1)、(2)两步进行。  
2 单元测试的设计策略稍有不同。因为在为模块设计测试用例时，可以直接参考模

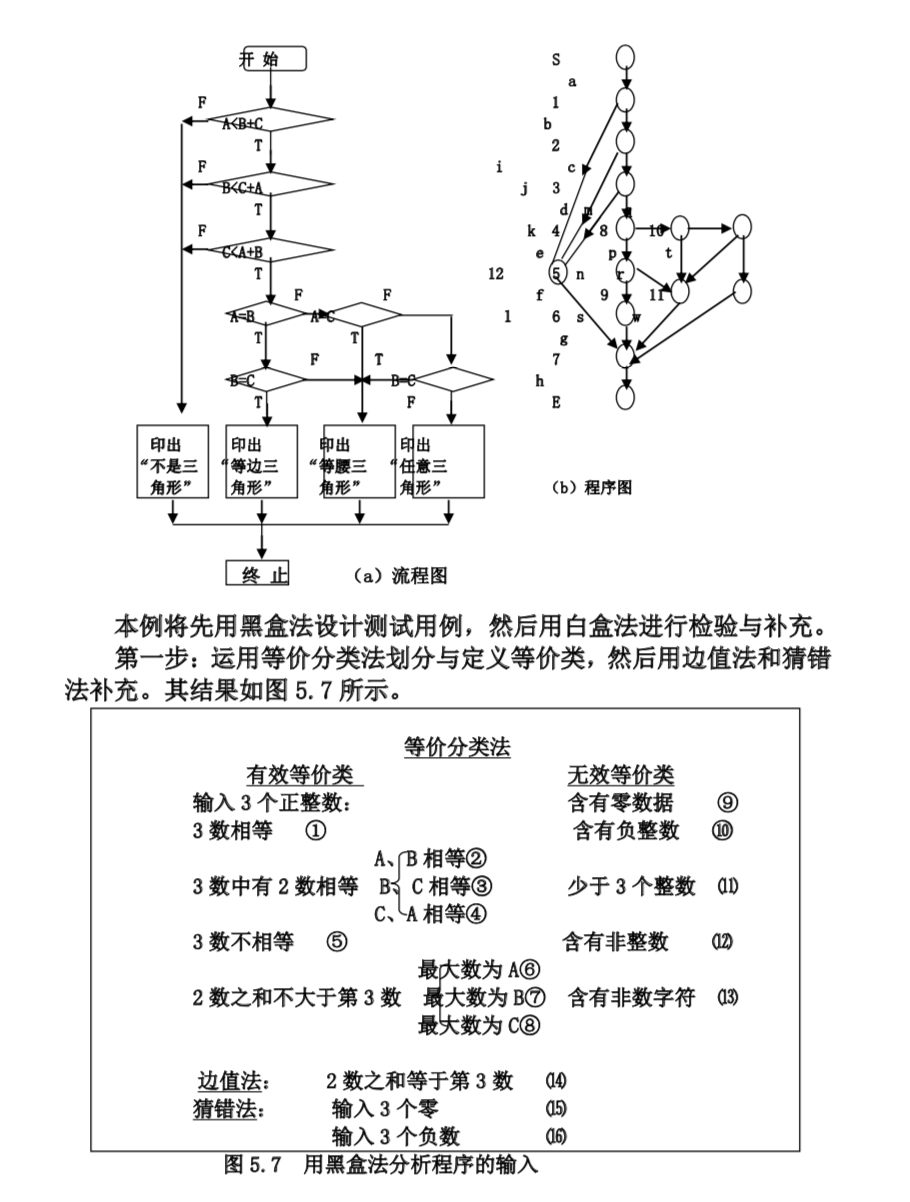
块的源程序。所以单元测试的策略，总是把白盒法与黑盒法结合运用。具体的作法又有 两种:

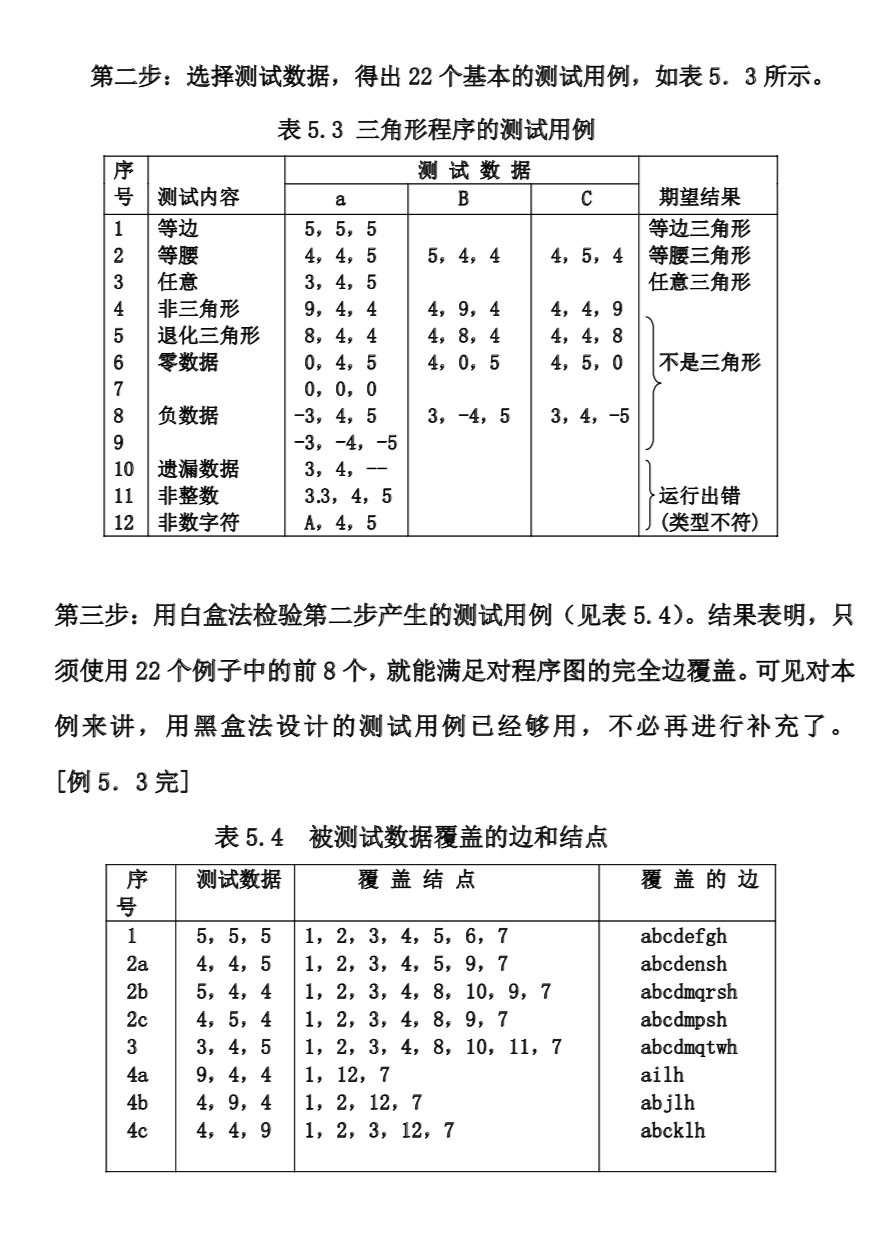
(1)先仿照上述步骤用黑盒法提出一组基本的测试用例，然后用白盒法作验证。 如果发现用黑盒法产生的测试用例未能满足所需的覆盖标准，就用白盒法增补新的测 试用例来满足它们覆盖的标准应根据模块的具体情况确定。对可靠性要求较高的模块， 通常应满足条件组合覆盖或路径覆盖标准。

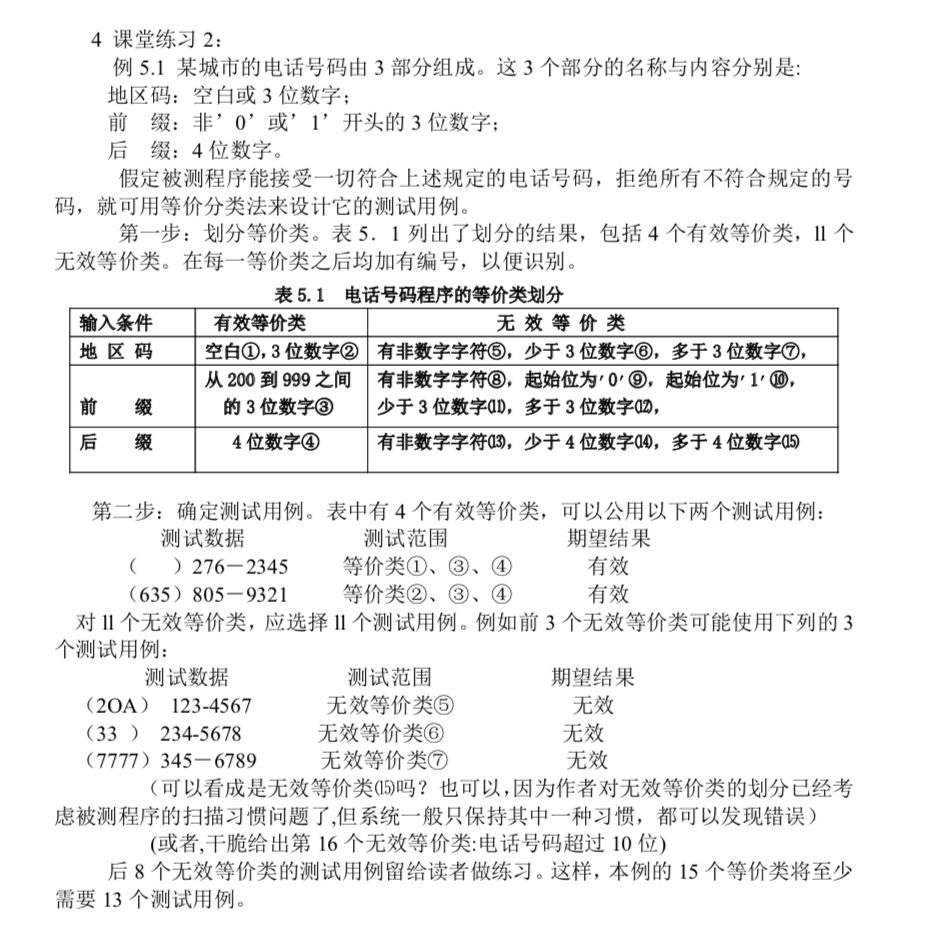
(2)先用白盒法分析模块的逻辑结构，提出一批测试用例，然后根据模块的功能 说明用黑盒法进行补充。

在一般情况下，小型单元测试应该大多以白盒方法为主。  
3 课堂练习:三角形程序设计一组测试用例。这个程序的功能是，读入代表三角形

边长的 3 个整数，判断它们能否组成三角形。如果能够，则输出三角形是等边、等腰或 任意三角形的识别信息。

****

****

****

**集成测试及其主要方法的分类？（驱动，桩的概念）**

1 集成测试:验证系统构件是否能够按照系统和程序设计规格说明中描述的那样共同工作  
2 分类:

1)自底向上测试:每一个出于系统层次中最低层的构件首先被单独测试，接着 测试那些调用了前面已测试构件的构件。反复地采用此方法，知道所有的构件都被测试完毕。**构件驱动程序:调用特定构件并向其传递测试用例的程序。**

2)自顶向下测试:顶层构件，通常是一个控制构件，是独立进行测试的。然后， 将被测构件调用的所有构件组合起来，作为一个更大的单元进行测试。重复执行这种方 法，直到所有的构件都被测试。**桩:一种专用程序，用于模拟(测试时)缺少构件时的 活动。桩应答调用序列，并传回输出数据，使测试过程得以继续运行。**

3)一次性测试:当所有构件都分别经过测试，再讲它们合在一起作为最终系统 进行测试，看看这个系统是否能一次运行成功。

4)三明治测试:将系统看成三层，目标层处于中间、目标上面有一层，目标下 面有一层，在顶层使用自顶而下方法，而在较低的层次使用自底向上方法。测试集中于 目标层，目标层是根据系统特性和构件层次结构来选择的。

**Chapter09**

**系统测试的主要步骤及各自含义？（图9.2）**

功能测试：根据SRS测试系统功能。  
性能测试：根据SRS测试系统性能。  
验收测试：根据客户的需求定义，由客户和用户一起测试。  
安装测试：在用户环境下进行测试。

**//什么是回归测试？**

回归测试是用于新的版本或者改进版本的一种测试，以验证与旧版本相比，软件是否仍然以同样的方式执行同样的功能。

**功能测试的含义极其作用？**

含义:检查集成的系统是否按照需求中指定的那样执行它的功能。

作用:有更高的概率发现缺陷，因为从逻辑上讲，相对于一个大的构件集，在一个 小的构件集中更容易找出问题的原因。

**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

**性能测试的含义与作用？**

测试系统性能和srs的非功能需求描述是否一致

作用：

确保系统的可靠性、可用性和可维护性。

**性能测试的主要分类？**

压力测试 短时间加载极限负荷，验证系统能力

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 使用旧的和新的测试用例

安全测试 测试安全需求

计时测试 相应时间相关的需求

环境测试 在安装场所的执行能力

质量测试 可靠性可维护性等

恢复测试 出现故障或丢失数据是否能够恢复

维护测试 针对帮助人们发现问题的工具和过程的需求

文档测试 确保文档已经编写正确

人为因素测试 检查涉及用户界面的需求

**确认测试概念，确认测试分类？（基准测试和引导测试）**

含义:使客户和用户能确定我们构建的系统真正满足了他们的需要和期望。

由用户和开发人员共同测试系统是否和需求定义预期的行为一致，由用户主导

分类：

基准测试，客户准备一组代表在实际安装后系统运作的典型情况的测试用例，由客户来评估执行的情况。

引导测试，在试验环境中安装系统，依赖系统的日常工作来进行测试。没有正式和结构化的测试用例。包括alpha测试和beta测试

并行测试，新系统和旧版本共同运行，使用户适应新系统

**什么是alpha测试？β测试？**

α 测试:在向客户发布一个系统之前，先让来自自己组织机构或公司的用户来测试这个系统，在客户进行实际的试验性测试之前先“试验”这个系统，这样的内部测试称为 α 测试。

β 测试:当要向大范围的客户发布系统时，客户进行的试验性测试称为β测试。

**什么是安装测试？**

在用户的环境中安装系统，使得用户能够执行系统功能并记录在实际环境中可能引

起的的其他问题。

**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。**