考点：

1. 软件工程概述，软件生命周期

⒉按照需求画数据流图

3.画软件结构图

4.画状态转换图

5.画流程图/盒图/pad图，或者给出伪码

6.白盒测试/黑盒测试

7.画用例图

8.画实体类图

9.画事件跟踪图(简要时序图)

10.软件项目管理问答

软件项目管理

所谓管理就是通过计划、组织和控制等一系列活动，合理地配置和使用各种资源，以达到既定目标的过程。软件项目管理先于任何技术活动之前开始，并且贯穿于软件的整个生命周期之中。

**软件管理的功能：**软件工程管理是对软件项目的开发管理，是对整个软件生存期的所有活动进行管理。软件管理的主要功能包括：

1. 制定计划：规定待完成的任务、要求、资源、人力和进度等；
2. 建立组织项目：为实施计划，保证任务的完成，需要建立分工明确的责任机构；
3. 配备人员：任用各种层次的技术人员和管理人员

④指导：鼓励和动员软件人员完成所分配的情况。

1. 软件项目管理始于项目计划。为了估算项目工作量和完成期限 ，首先需要预测软件规模。 2. 管理者必须制定出一个足够详细的进度表，以便监督项目进度 并控制整个项目。

3. 软件质量保证是在软件过程中的每一步都进行的活动。

4. 软件配置管理是应用于整个软件过程中的保护性活动，是在软 件整个生命期内管理变化的一组活动。

5. 能力成熟度模型（CMM）是改进软件过程的有效策略。

1. 估算软件规模（代码行、功能点估算）
2. 估算工作量
3. 进度计划（估算开发时间，向一个已经延期的项目增加人力，只会增加延期时间，通信开销组内成员协调，刚入组并不是生产力学习耽误时间）
4. 人员组织（小组2-8人）
5. 质量保证（软件质量保证的措施主要有：基 于非执行的测试（也称为复审或评审），基于执行的测试（即以前讲过 的软件测试）和程序正确性证明。）复审主要用来保证在编码之前各阶段产生的文档的质量，（走查、审查，程序正确性证明）
6. 软件配置管理（软件过程的输出信息可以分为3类： （1）计算机程序（源代码和可执行程序）。 （2）描述计算机程序的文档（供技术人员或用户使用）。 （3） 数据（程序内包含的或在程序外的）。

• 上述这些项组成了在软件过程中产生的全部信息，人们把它们统称为软 件配置，而这些项就是软件配置项）

1. 能力成熟模型

１．什么是软件工程？软件工程的目的是什么？

　答：软件工程是将系统的、规范的、可度量的工程化方法应用于软件开发、运行和维护的全过程及上述方法的研究。

　　　目的：是在给定成本、进度的前提下，开发出具有可修改性、有效性、可靠性、可理解性、可重用性、可适应性、可移植性、可追踪性和可互操作性并满足用户需求的软件产品。

２．什么是软件危机？软件危机的表现有哪些？产生的原因是什么？

　答：软件危机是指在计算机软件的开发和维护过程中所遇到的一系列严重问题。

表现在：（１）对于软件开发的成本和进度的估计很不准确。

　　　　（２）开发的软件产品不能完全满足用户要求，用户对已完成的软件系统不满意的现象常常发生。

　　　　（３）开发的软件可靠性差。

　　　　（４）软件通常没有适当的文档。

　　　　（５）软件的可维护性差。

　　　　（６）软件开发生产率提高的速度，无远远跟不上计算机应用普及深入的趋势。

原因：（１）软件的规模愈发庞大；

　　　（２）软件开发的管理困难。

（３）软件本身的独有特点确实给开发和维护造成一些客观困难。

　　　（４）软件开发和维护中千金错误认识和方法的形成可以归结与计算机发展早期软件开发的个体化特点。

　　　（５）软件开发技术落后。

　　　（６）生产方式落后。

　　　（７）开发工具落后，生产率提高缓慢。

**消除软件危机的途径：**

(1) 对计算机软件有一个正确的认识(软件≠程序）

(2) 必须充分认识到软件开发不是某种个体劳动的神秘技巧，而应该是一种组织良好、管理严密、各类人员协同配合、共同完成的工程项目

(3) 推广使用在实践中总结出来的开发软件的成功技术和方法

(4) 开发和使用更好的软件工具

（1）使用好的软件开发技术和方法；

（2）要有良好的组织、严密的管理，各类人员协同配合，共同完成任务；

（3）使用好的软件开发工具，提高软件生产率；

（4）建立严格的文档资料，重视软件开发过程的阶段评审。

瀑布模型：开发的系统功能和性能明确、完整、无重大变化。在消除非结构化软件、降低软件的复杂性、促进软件开发工程化方面起了很大作用。缺乏灵活性，特别是无法解决软件需求不明确或不准确的问题。开发人员熟悉

增量模型：人员分配灵活，适合于软件需求不明确、设计方案有一定风险，工期紧张人员不足有中高风险的软件项目。熟悉

快速原型模型：有开发工具，不能预先确切定义需求的软件开发。快速建立产品雏形，开发与用户交流以便准确获取需求。不熟悉

螺旋模型：适用于大项目，需要风险评估技术支持

4．说明软件开发的生命周期模型，软件开发的几个生命周期，每个周期里的主要任务是什么？

　答：软件开发的生命周期模型是描述软件开发过程中各种活动如何执行的模型。

　　　软件开发的生命周期：一个软件从提出开发要求开始直到该软件报废为止的整个时期，具体分为以下四个时期：

（１）软件分析时期；

任务：确定软件项目的目标，软件应具备的功能和性能，构造软件的逻辑模型，并制定验收标准。在此期间，要进行可行性论证，并做出成本估计和经费预算，制定进度安排。进行可行性研究和项目开发计划，需求分析。

（２）软件设计时期；

任务：a.设计软件的总体结构；

　　　b.设计软件具体模块的实现算法；

　　　c.软件设计结束之前，也要进行有关评审，评审通过后才能进入编码时期。

（３）编码与测试时期；

任务：组织程序员将高驻地的软件“翻译”成计算机可以正确运行的程序；并且要经过按照软件分析中提出需求要求和验收标准进行严格的测试和审查。根据具体软件的特点，决定是否划分成一些阶段，如编码、单元测试、集成测试、验收测试等等。

（４）运行与维护时期。

任务：软件运行过程中可能由于各方面的原因，需要对它进行修改。

6．如果要开发一个xx系统，具备哪些条件，你会选择用瀑布模型进行开发？会选择原型模型？

　答：如果要开发的系统功能和性能明确、完整、无重大变化，例如编译系统、数据库管理系统和操作系系统等，则会选择瀑布模型；

采用原型模型适合如下条件的软件开发：

1.有快速建立系统原型系统的软件工具与环境。

2.那些不能预先确切定义需求的软件开发。

3.那些项目组成成员不能很好协同配合、交流或通信上存在困难的情况。

**七、什么是软件生命周期模型？试比较瀑布模型、快速原型模型、增量模型和螺旋模型的优缺点，说明每种模型的使用范围。**

软件生命周期模型是跨越整个生存期的系统开发、运作和维护所实施的全部过程、活动和任务的结构框架。

* 瀑布模型优点：

它提供了一个模板，这个模板使得分析、设计、编码、测试和支持的方法可以在该模板下有一个共同的指导。虽然有不少缺陷但比在软件开发中随意的状态要好得多。

缺点：

(1) 实际的项目大部分情况难以按照该模型给出的顺序进行，而且这种模型的迭代是间接的，这很容易由微小的变化而造成大的混乱。

(2) 经常情况下客户难以表达真正的需求，而这种模型却要求如此，这种模型是不欢迎具有二义性问题存在的。

(3) 客户要等到开发周期的晚期才能看到程序运行的测试版本，而在这时发现大的错误时，可能引起客户的惊慌，而后果也可能是灾难性的。

(4) 会经常在过程的开始和结束时碰到等待其他成员完成其所依赖的任务才能进行下去，有可能花在等待的时间比开发的时间要长。称之为“堵塞状态”。

它是软件工程中应用最广泛的过程模型，在软件工程中占有肯定和重要的位置。

* 快速原型模型

在进行了基本需求分析之后，快速开发出产品的原型，然后基于这个原型，同客户沟通、交流，更好地了解客户需求，不断修改这个原型，到了双方认可的程度，再做详细地分析、设计和编程，最终开发出令客户满意的产品。

优点：使用户能够感受到实际的系统，使开发者能够快速地构造出系统的框架。

缺点：产品的先天性不足，因为开发者常常需要做实现上的折中，可能采用不合适的操作系统或程序设计语言，以使原型能够尽快工作。

* 增量模型

优点：

(1) 人员分配灵活，刚开始不用投入大量人力资源，当核心产品很受欢迎时，可增加人力实现下一个增量。

(2) 当配备的人员不能在设定的期限内完成产品时，它提供了一种先推出核心产品的途径，这样就可以先发布部分功能给客户，对客户起到镇静剂的作用。

缺点：

(1) 至始至终开发者和客户纠缠在一起，直到完全版本出来。

(2) 适合于软件需求不明确、设计方案有一定风险的软件项目。

该模型具有一定的市场。

* 螺旋模型

优点：对于大型系统及软件的开发，这种模型是一个很好的方法。开发者和客户能够较好地对待和理解每一个演化级别上的风险。

缺点：(1) 需要相当的风险分析评估的专门技术，且成功依赖于这种技术。

(2) 很明显一个大的没有被发现的风险问题，将会导致问题的发生，可能导致演化 的方法失去控制。

(3) 这种模型相对比较新，应用不广泛，其功效需要进一步的验证。

该模型适合于大型软件的开发

**3、瀑布模型、原型模型、增量模型的特点，如何选择这些模型？**

（1）瀑布模型：Page 25-27（特点：Page 28第二点；使用场合：特点的最后一点）

（2）原型模型：Page 27-28（特点：Page 27； 场合：Page 28三点）

（3）增量模型：Page 28）

参考答案：

**瀑布模型**：

（1）定义：是将软件生命周期各活动规定为依线性顺序联接的若干阶段的模型，是一种整体开发模型。里程碑或基线驱动或者说文档驱动。过程逆转性很差，或者说不可逆转。

（2）优点：严格按照生命周期的各个阶段来进行开发，强调了每一阶段的严格性。这样就能解决在开发阶段后期修正不完善的需求说明将花费巨大的费用的问题。在消除非结构化软件、降低软件的复杂性、促进软件开发工程化方面起了很大作用。

（3）缺点：它是一种理想的线性开发模式，缺乏灵活性，特别是无法解决软件需求不明确或不准确的问题。故适用于功能明确、完整、无重大变化的软件开发。如：编译系统、数据库管理系统和操作系统。

（4）适用场合：

● 在开发时间内需求没有或很少变化。

● 分析设计人员对应用领域很熟悉。

● 低风险项目（对目标、环境很熟悉）。

● 用户使用环境很稳定。

● 用户除提出需求以外，很少参与开发。

**原型模型**：

（1）定义：以某个软件原型为参照模型的开发方法，叫做原型法。（原型驱动）

（2）原理：在初步需求分析之后，马上向客户展示一个软件产品原型，对客户进行培训，让客户试用，在试用中收集客户意见，修改原型，再让客户试用，反复循环几次，直到客户确认为止。

（3）适用场合：

● 已有产品/产品原型，只需客户化的项目。

● 简单而熟悉的行业或领域。

● 有快速原型开发工具。

● 进行产品移植或升级。

**增量模型**：

（1）定义：增量模型将软件产品看作一组增量构件，每次设计、实现、集成、测试和交付一块构件，直到所有构件全部实现为止。

（2）特点：●任务或功能模块驱动，可以分阶段提交产品；

●有多个任务单，这些多个任务单的集合，构成项目的一个总任务书(总用户需求报告)。

（3）适用场合：

● 在开发过程中，客户接受分阶段交付。

● 开发人员对应用领域不熟悉，难以一步到位。

● 工期过紧的中等或高风险项目。

● 用户可参与到整个软件开发过程中。

● 使用面向对象语言或第四代语言。

● 软件公司自己有较好的类库、构件库。

白盒测试方法（3种）

● 逻辑覆盖

**语句覆盖**：-在测试时，设计若干测试用例，运行被测程序，使程序中的每个可执行语句至少执行一次。所有语句

**判定覆盖（又称为分支覆盖）**：-在测试时，设计若干测试用例，运行被测程序，使程序中的每个判定的分支至少遍历一次。判定语句结果的两种可能

**条件覆盖**：-在测试时，设计若干测试用例，运行被测程序，使程序中的判定表达式中的每个条件的可能取值至少满足一次。判定语句中每个条件

条件组合覆盖：每个判定表达式中每个条件的各种可能组合都至少出现一次

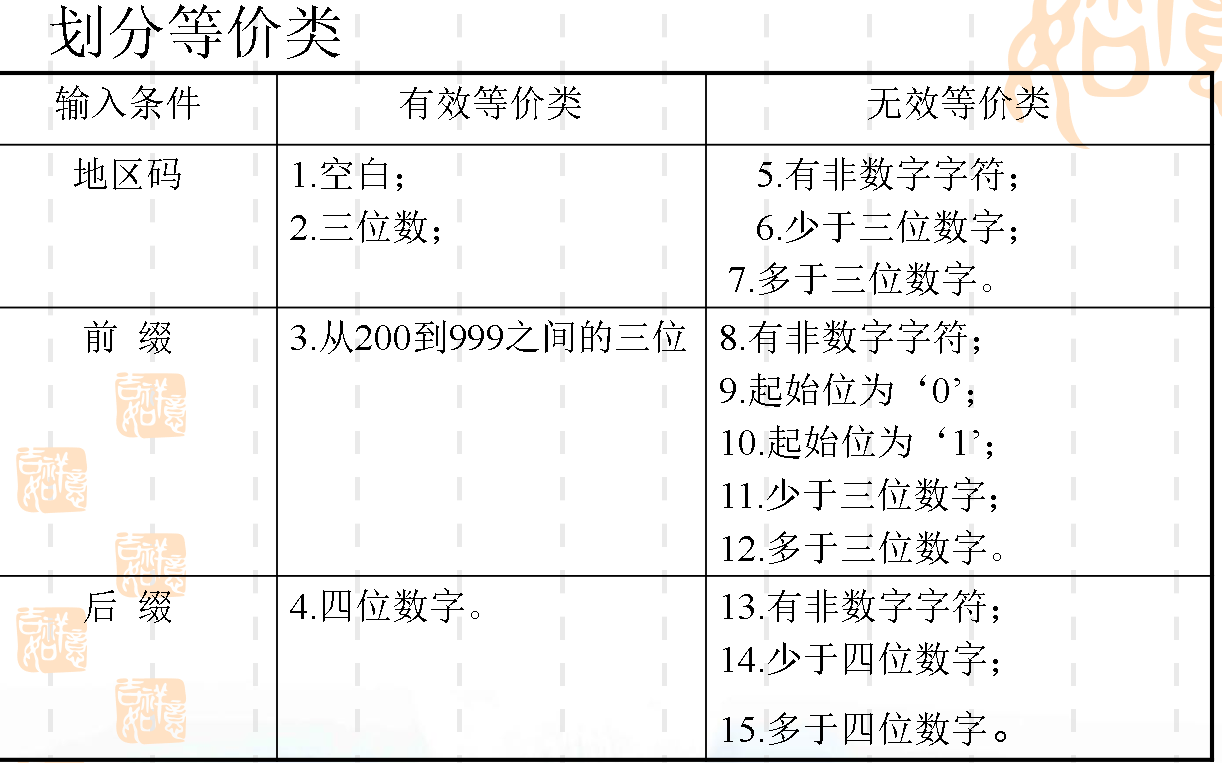
**条件分支覆盖（判定条件覆盖）**：-在测试时，设计足够的测试用例，使得判断中每个条件的所有可能取值至少出现一次，并且每个判断本身的判定结果也至少出现一次。

**路径覆盖**：- 设计足够多的测试用例，要求覆盖程序中所有可能的路径。

● 循环覆盖

● 基本路径覆盖

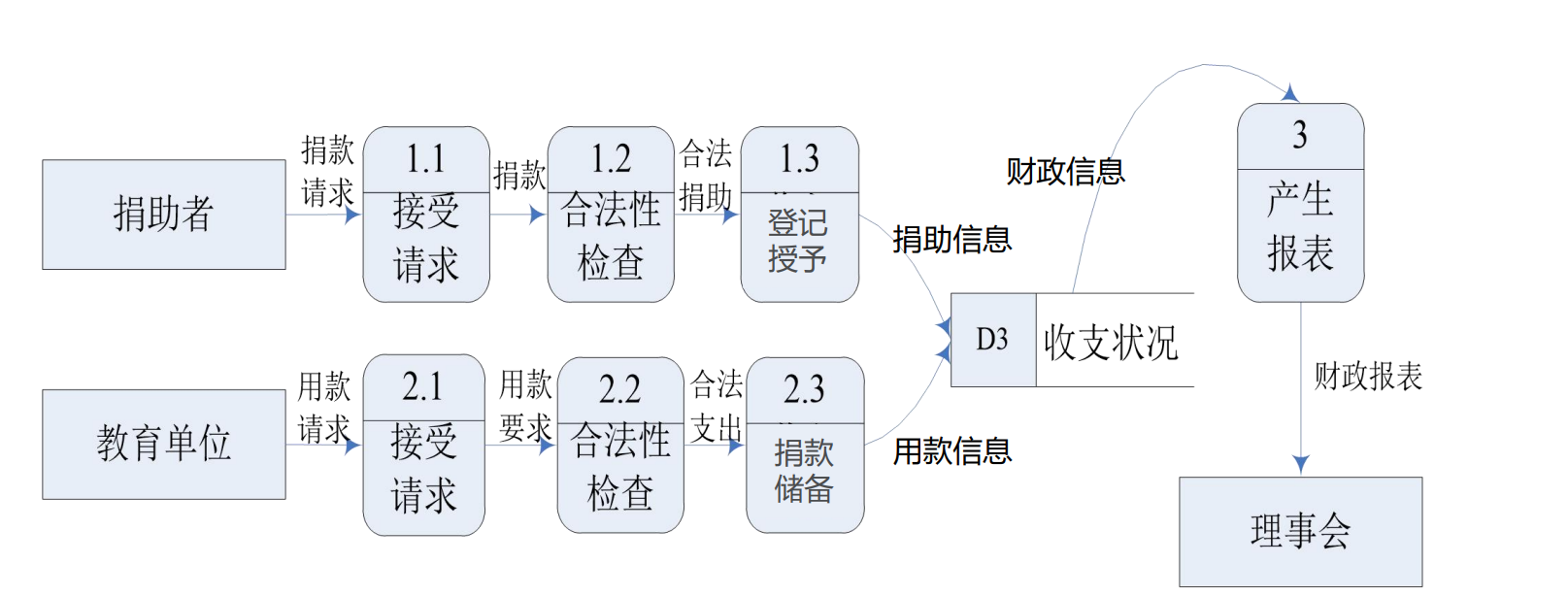
黑盒测试：



测试方案：

方案号：覆盖编号：输入：预期输出：

一个测试方案可包含多个有效等价类，只可包含一个无效等价类

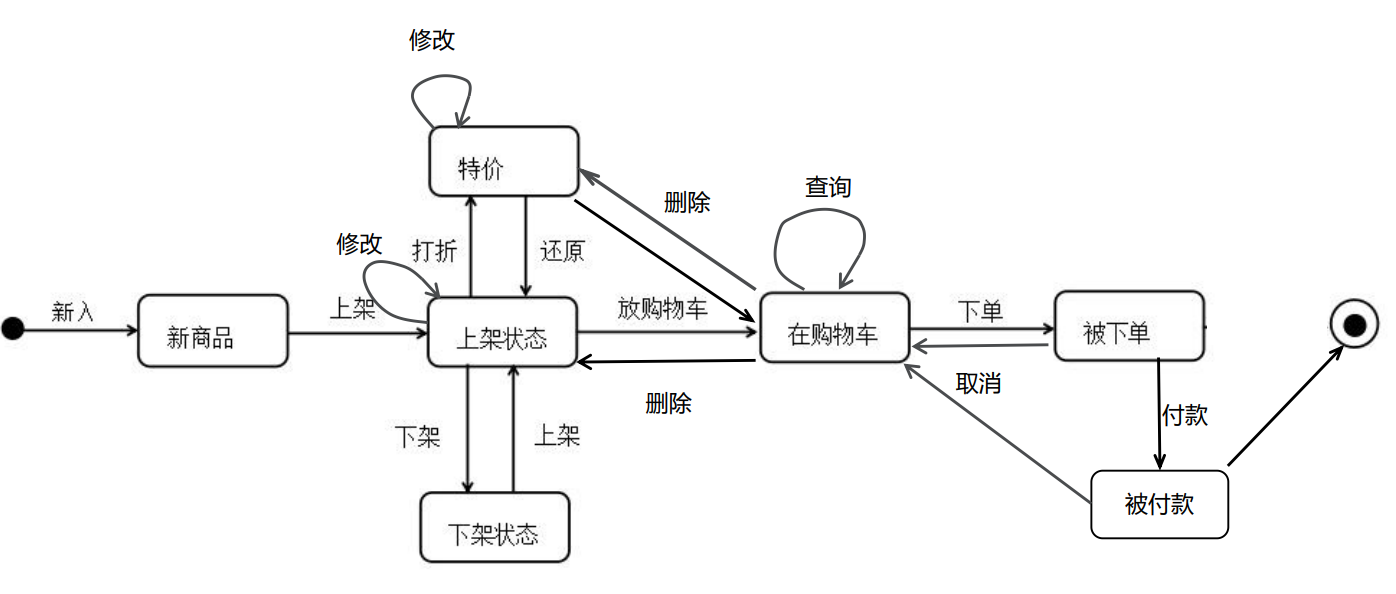


数据流图：源点终点：系统平台之外的类（人）方框

数据处理：找动词 圆圈

数据存储：信息库等字眼 两条线或右开口方框

信息流：各种信息 报表 箭头上面

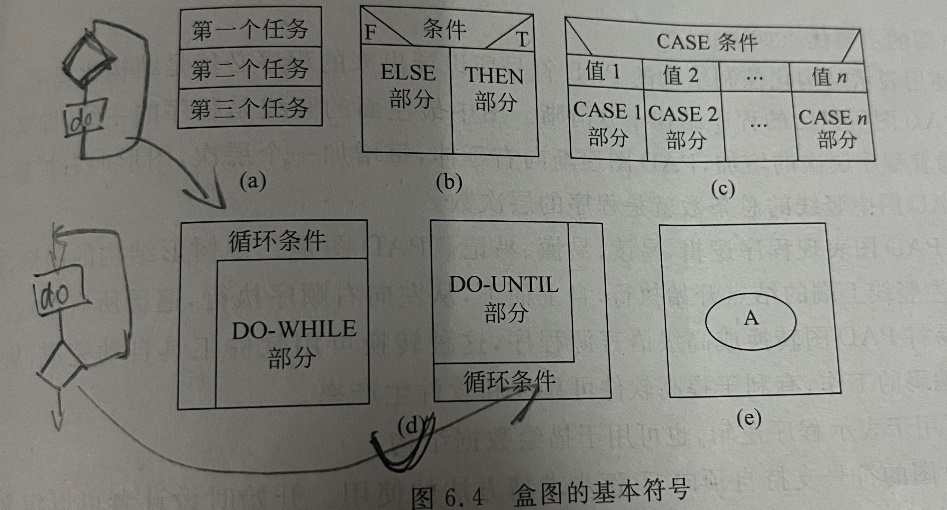


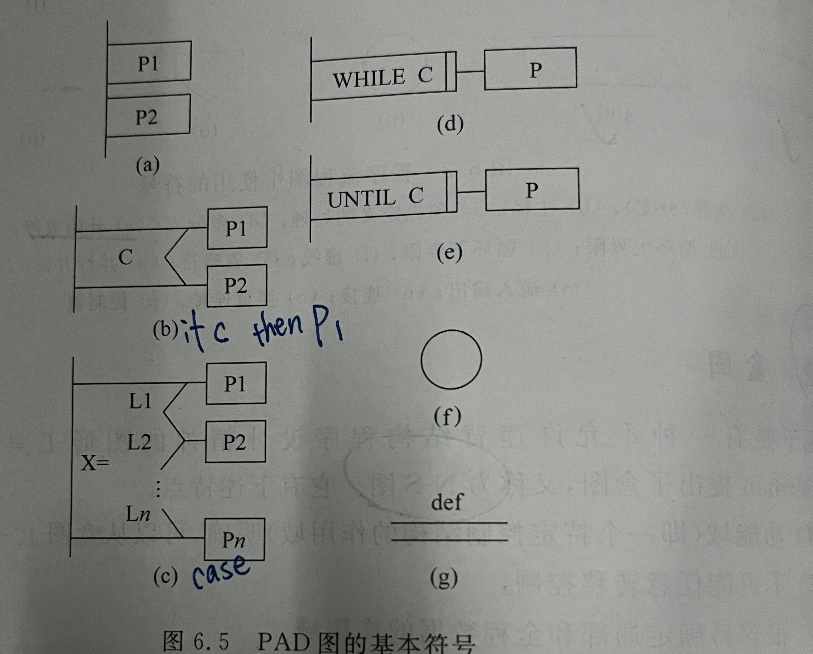
状态图：

初态实心圆，终态内实心的同心圆

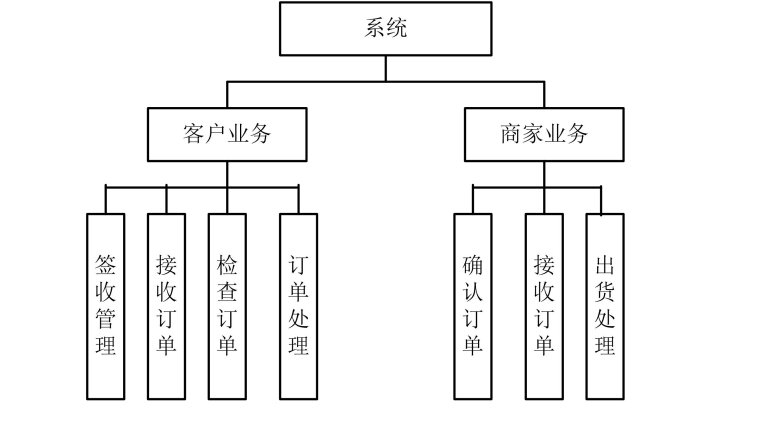
可只写状态 圆角方框

箭头上 触发事件 动作



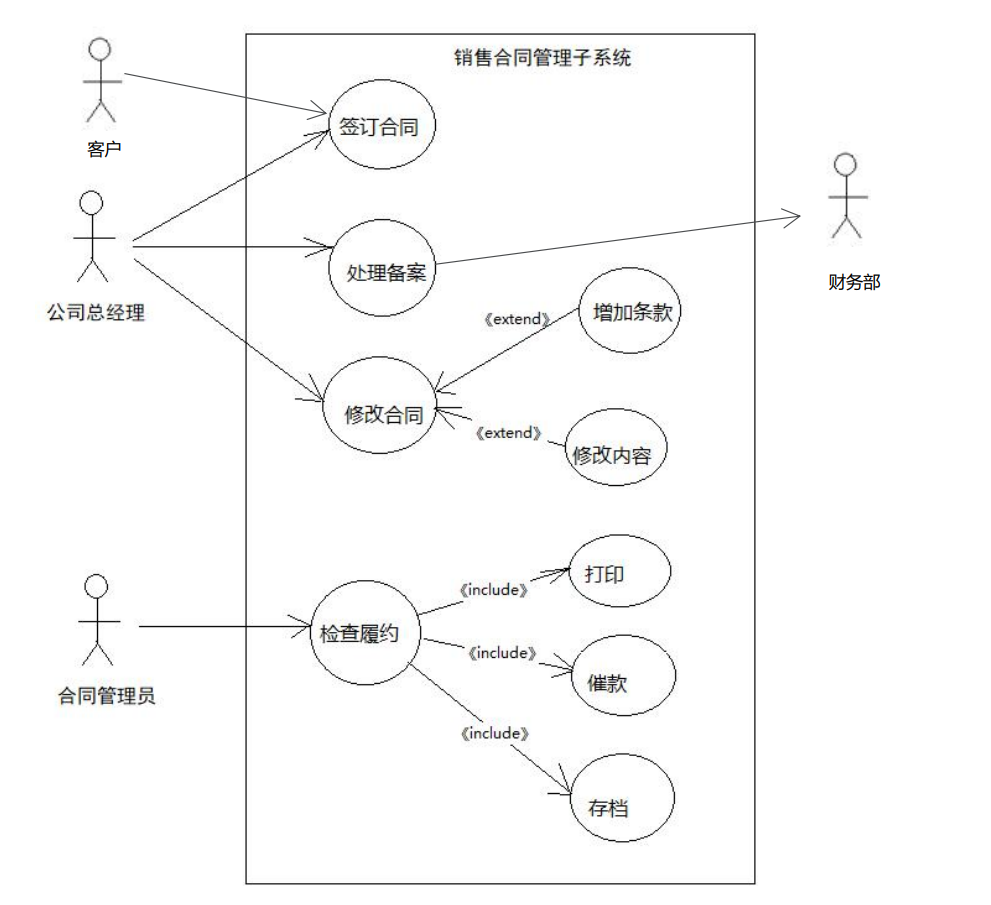


注意盒图如何画 while 和until



软件结构图：考试中数据流动箭头可以不画，直线连接即可、

上层模块调动下层模块



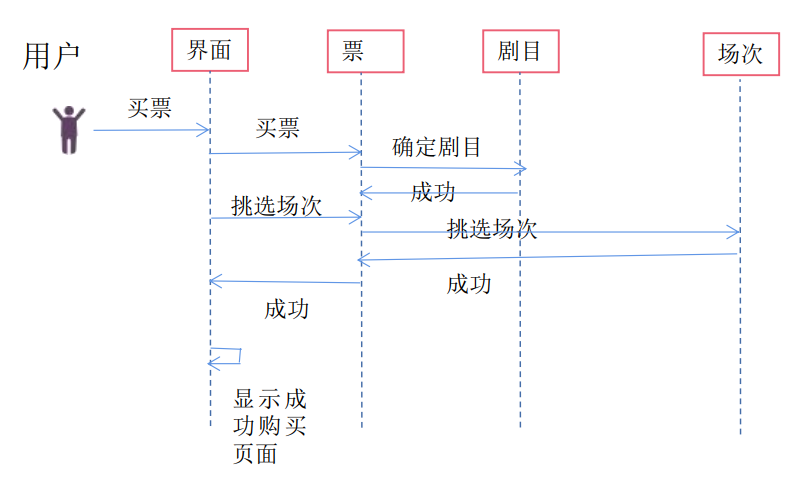
用例图：不需要体现过程，步骤先后顺序

考试中用例关系可不写

行为者：使用系统的人或外部系统 火柴人

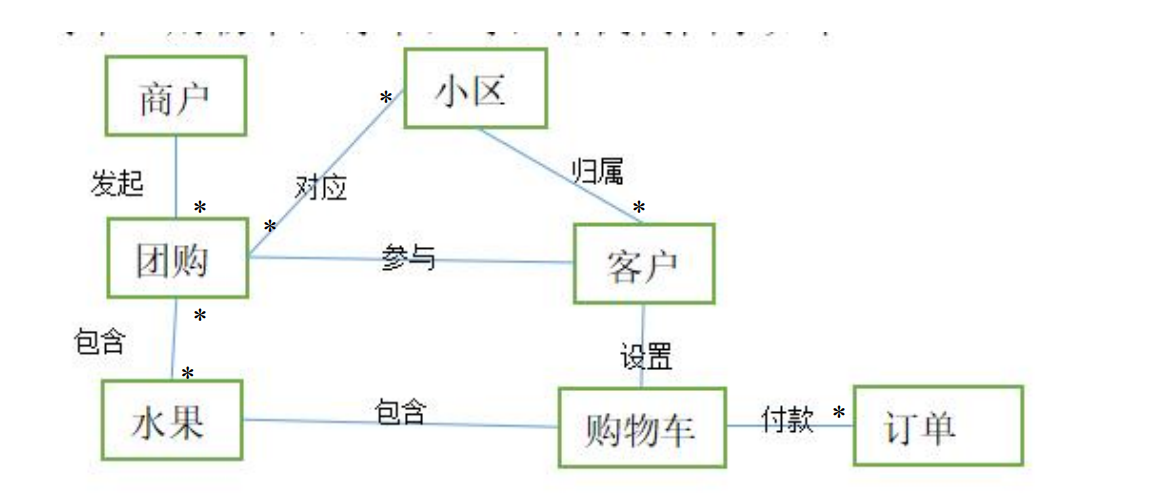
相关用例：可被行为者察觉到的一系列动作，一类功能 圆圈

记得画大方框表示系统 箭头连接



事件跟踪图

交互的类：申请界面类 控制类（没有数据库 系统类，不要出现这些字眼的类）



类图：静态模型

考试中可不写属性服务 类之间关系不用很详细

直线连接