**（一）简答题**

# 如果要为某企业开发一个业务信息系统，系统需求来自于一线业务工作人员，但是用户并不明确要做成什么样的信息系统，需要不断地交流和反馈，针对这一软件需求，请从瀑布模型和原型模型中选择一个软件过程模型指导软件开发过程，并说明原因。（8分）

答：应该选择原型模型，因为瀑布模型只有等到后期阶段才能产生可运行的软件系统，用户才能接触和适用软件，在需求不明确的情况下，很容易出现和用户需求不一致的问题；而原型模型可以根据用户的初步需求构建软件原型，并交给用户使用，帮助用户导出需求，发现开发人员和用户之间的需求认知偏差。

# 关于“空巢老人看护”软件需求的如下两段叙述中，哪段属于软件质量需求？ 哪段属于软件开发约束需求？（8分）

需求描述a: “医生通过手机操控机器人移动时，操作延迟须控制在1秒内。”

需求描述b : “软件应当采用Java语言实现。”

答：需求描述a属于软件质量需求。它强调了系统性能方面的要求，即操作延迟的时间限制，这直接影响用户体验和系统的响应速度。

需求描述b 属于软件开发约束需求。它规定了开发过程中必须遵守的技术选择或开发环境约束，即使用Java语言进行开发。这种需求主要是为了确保开发过程的一致性或与现有系统的兼容性。

# 请解释模块的内聚与模块间的耦合，模块化设计应当遵循的两个原则是什么？（8分）

答：内聚是指一个模块内部各个元素之间的紧密程度。一个高内聚的模块，其内部元素之间的关系紧密，功能集中，通常围绕一个单一的任务或功能。内聚可以分为不同的层次，从低到高包括：偶然内聚、逻辑内聚、时间内聚、过程内聚、通信内聚、顺序内聚和功能内聚。功能内聚是最高层次的内聚，表示模块内的所有元素都围绕单一功能进行。

耦合是指不同模块之间的依赖程度。低耦合的模块相对独立，变化在一个模块中不会对其他模块产生大的影响。耦合也可以分为不同的层次，从高到低包括：内容耦合、公共耦合、外部耦合、控制耦合、标记耦合、数据耦合和无耦合。数据耦合是最低层次的耦合，表示模块之间仅通过简单的数据传递进行交互。

在模块化设计中，需要遵循的两个重要原则：

1. 高内聚原则：模块内部应当保持高内聚，即模块内的元素应紧密相关，围绕单一功能进行组织。这种设计使得模块功能明确，易于理解和维护。

2. 低耦合原则：模块之间应当保持低耦合，即模块之间的依赖关系应尽可能少。这种设计使得模块独立性强，便于模块的独立开发、测试和重用，同时提高系统的灵活性和可维护性。

# 测试(Testing)和调试(Debugging)是软件开发过程中的两项重要工作，请说明这两项工作有何区别？二者之间有何关系？（8分）

答：软件测试通过运行程序代码的方式来发现程序代码中潜藏的缺陷；调试是发现产生缺陷原因，定位缺陷位置，进而对代码缺陷进行修复。软件测试的目的是为了发现软件中的缺陷。它只负责发现缺陷，不负责修复和纠正缺陷。因此，软件测试的结果是要报告通过测试所发现的软件缺陷集合，一旦通过软件测试发现了软件中的缺陷，程序员则需开展程序调试，寻找软件缺陷的原因以及产生缺陷的代码位置，修复并溢出软件缺陷。

# 简述软件维护的定义和分类，并以电商网站软件为例进行说明。（8分）

答：软件维护的定义：软件在交付使用后，由于应用需求和环境变化以及自身问题，对软件系统进行改造和调整的过程。

软件维护的分类:

纠正性维护：纠正软件中的缺陷和错误，确保软件能够正常运行。

适应性维护：对软件进行改造以便适应新的运行环境和平台。

完善性维护：对软件进行改造以增加新的功能、修改已有的功能。

预防性维护：对软件结构进行改造以便提高软件的可靠性和可维护性等。

举例:

纠错性维护：修复登录功能中的一个bug,用户无法正常登录。

适应性维护：将网站从Windows服务器迁移到Linux服务器。

完善性维护：增加一个商品推荐功能,根据用户浏览记录推荐商品。

预防性维护：重构数据库设计,提高查询效率。为了防止未来可能出现的安全漏洞，对网站的代码进行了安全审计，发现并修复了一些潜在的安全问题。

**（二）分析题**

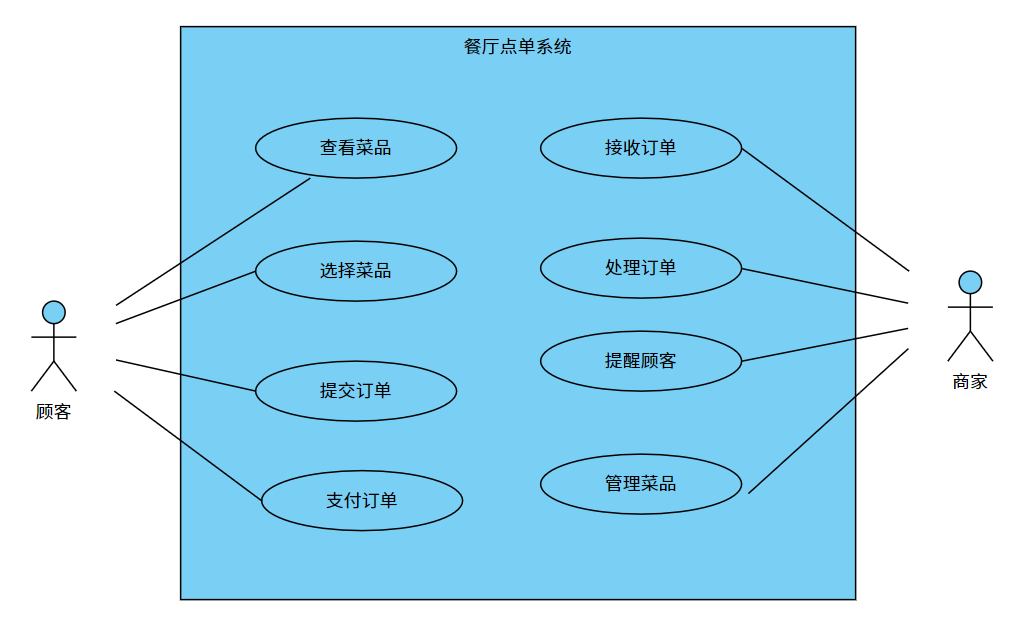
# 需求分析题（20分）

请阅读以下对软件功能和非功能性需求的描述，绘制出该软件的用例图（6分），并绘制出该软件的类图（7分），并针对要求的功能场景，绘制出顺序图（7分）。

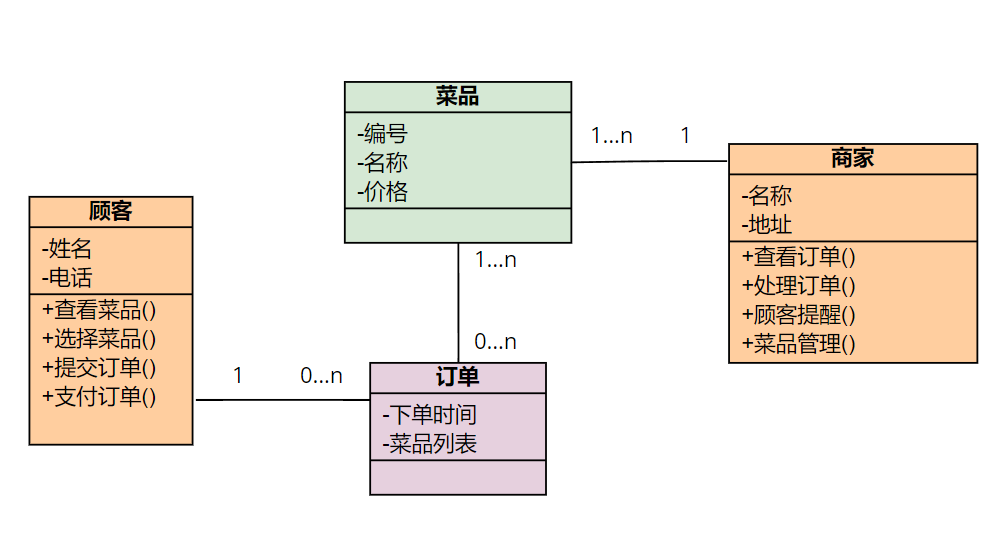
描述：*“在餐厅点单系统中，顾客可以进行菜品信息查看，并选择想要购买的菜品，选择完成后将订单提交，最后进行付款完成点单。顾客信息包括顾客姓名、顾客电话。商家接受订单后，将会对订单进行处理，处理完成后提醒顾客订单已完成，同时商家还可以对菜品信息进行管理。菜品信息包括菜品编号、菜品名称、菜品价格。订单信息包括下单时间、菜品列表。”另外，请针对其中顾客点单的场景绘制顺序图。*

答：

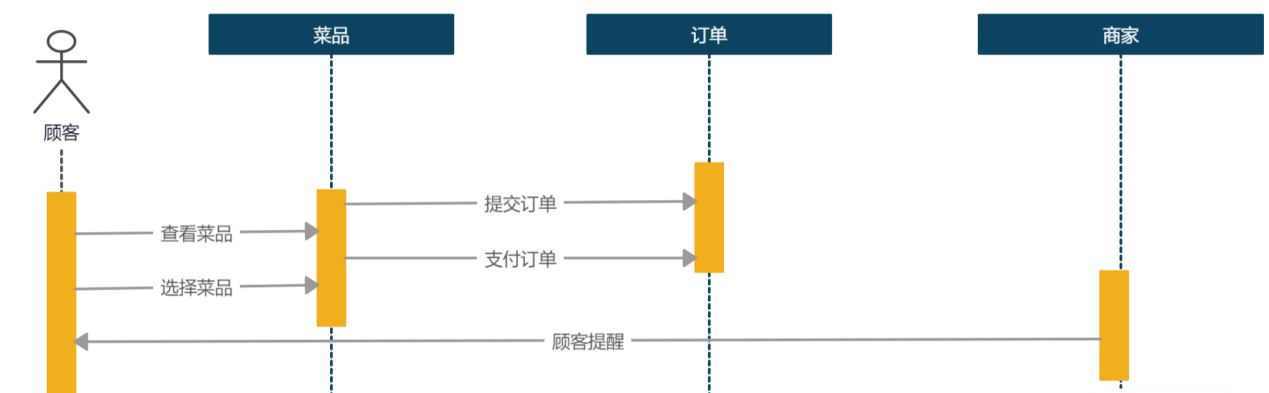
1. 用例图：



（2）类图：



（3）顺序图：



# 软件设计题（20分）

请阅读以下对软件功能和非功能性需求的描述，选择一个合适体系结构风格，请列出选择该风格后，该软件的构件有哪些（5分），连接件有哪些（5分）。请用图形的方式绘制出该软件的体系结构示意图（5分）。请简述该体系结构风格的优缺点（5分）。

描述：*“TCP/IP协议是在网络中最基本的通信协议。在网络通信的过程中，源主机中的应用进程先将数据交给应用层，应用层加上必要的控制信息形成报文流，向下传给传输层。传输层将收到的数据单元加上本层的控制信息，形成报文段、数据报，传给网际层。网际层加上本层的控制信息，形成IP数据报，传给网络接口层。网络接口层将网际层交下来的IP数据报组装成帧，并以比特流的形式传给网络硬件（即物理层），数据就离开源主机前往目标主机。”。*

答：

1. 构件有：各层次及其内部包含的组件
2. 连接件有：层次间的交互协议
3. 示意图：



1. 优缺点：优点是设计者可以将大的问题分解为若干渐进小问题，减少了复杂度；修改一层，最多影响两层。缺点是层层相调、影响性能。

# **白盒测试题（20分）**

# 白盒测试中分支覆盖和语句覆盖的概念是什么？（4分）请分别对下述代码给出满足分支覆盖（8分）和语句覆盖（8分）的测试用例集，并填写在表格中（表格行数不代表测试用例数量）。

|  |  |
| --- | --- |
|  | // 判断三边能否构成三角形 |
| 1 | int isValidTriangle(int a, int b, int c) { |
| 2 | if (a <= 0 || b <= 0 || c <= 0) { |
| 3 | return 0; |
| 4 | } |
| 5 | if (a + b > c && a + c > b && b + c > a) { |
| 6 | return 1; |
| 7 | } |
| 8 | return 0; |
| 9 | } |

答：

分支覆盖，也叫判定覆盖，是指运行代码进行测试时，程序中的所有判定语句的真、假分支至少都被执行过一次。

语句覆盖是指设计适当数量的测试用例，使被测程序中的每条语句至少被执行一次。

| **分支覆盖** | **a** | **b** | **c** | **预期结果** | **说明** |
| --- | --- | --- | --- | --- | --- |
| 1 | 3 | 4 | 5 | 1 | 所有边都为正数且能构成三角形 |
| 2 | 1 | 2 | 3 | 0 | 所有边都为正数但不能构成三角形 |
| 3 | 0 | 4 | 5 | 0 | 至少有一边为非正数 |
| 4（与3同类） | -1 | 2 | 3 | 0 | 存在负数边长，不能构成三角形 |
| 5（与2同类） | 10 | 1 | 1 | 0 | 一边远大于其他两边，不能构成三角形 |

| **语句覆盖** | **a** | **b** | **c** | **预期结果** | **说明** |
| --- | --- | --- | --- | --- | --- |
| 1 | 3 | 4 | 5 | 1 | 所有边都为正数且能构成三角形，覆盖除行3，行8外 |
| 2 | 1 | 2 | 3 | 0 | 所有边都为正数但不能构成三角形，覆盖除行3，行6外 |
| 3 | 0 | 4 | 5 | 0 | 至少有一边为非正数，覆盖  if (a <= 0 || b <= 0 || c <= 0)为True结果（行3） |
| 4  （与3同类） | -1 | 2 | 3 | 0 | 存在负数边长，不能构成三角形，覆盖  if (a <= 0 || b <= 0 || c <= 0)为True结果（行3） |
| 5  （与2同类） | 10 | 1 | 1 | 0 | 一边远大于其他两边，不能构成三角形，  覆盖除行3，行6外 |