全国计算机技术与软件专业技术资格（水平）考试

2024 下半年 系统架构设计师真题及答案解析

软考诸葛老师

说明：机考之后，不会再有完整版真题流出，只有考生回忆版、知识点，诸葛老师会根据回忆版还原真题，请知晓。

综合知识

操作系统

下列选项中不能作为预防死锁措施的是____。

A. 破坏“循环等待"条件

B. 破坏“不可抢占”条件

C. 破坏“互斥”条件

D. 破坏“请求和保持”条件

答案：C

解析：所谓死锁，是指两个以上的进程因相互争夺对方占用的资源而陷入无限等待的现象。死锁产生的 4 个必要条件：**互斥（资源互斥）、请求保持（进程占有资源并等待其他资源）、不可剥夺（系统不能剥夺进程资源）和环路（进程资源图是一个环路）**。

但是，**破坏“互斥”条件**不能作为预防死锁的措施。这是因为**互斥条件是资源使用的基本特性，无法进行改变**，它确保了资源的独占性和数据的完整性。在多任务操作系统中，进程或线程需要独占某些资源（如打印机、文件等）以防止数据竞争和不一致。

如果取消互斥条件，允许多个进程同时访问同一资源，虽然可以避免死锁的发生，但会引发其他严重问题，如数据损坏、资源争用冲突和优先级反转等。因此，破坏互斥条件并不是一个可行的死锁预防策略。

为了预防死锁，通常会采取其他措施来破坏产生死锁的其他三个必要条件之一或多个：

请求和保持条件：可以通过实施资源分级策略或静态分配法来破坏。例如，要求进程在开始运行前一次性申请其在整个运行期间所需的全部资源，或者在提出新资源请求前释放已占有的资源。

不可抢占条件：可以允许进程抢占已经分配给其他进程的资源，但这通常需要复杂的管理机制来处理资源的抢占和恢复。

循环等待条件：可以通过全局资源排序策略来破坏，即给系统中的所有资源编号，并要求进程按递增顺序申请资源。

在多道批处理系统中有 P1 和 P2 两个作业，执行顺序如下，P1(计算 40ms，I/O 操作 60ms，计算 20ms)，P2(计算 100ms，I/O 操作 40ms，计算 40ms)，P2 比 P1 晚 20ms 到达，在不考虑调度和切换时间的情况下完成两道作业最少需要_____时间。

A. 240ms

B. 200ms

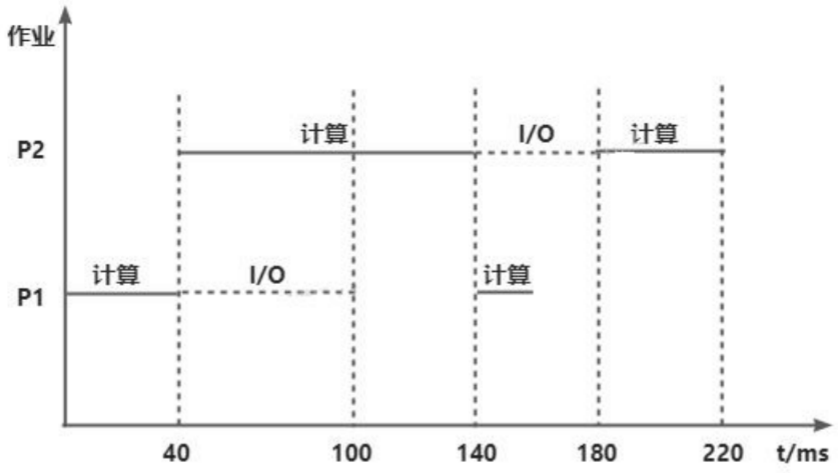
C. 260ms

D. 220ms

答案：D

解析：教材 2.3.2 操作系统，P30。多道批处理操作系统允许多个作业装入内存执行，在任意一个时刻，作业都处于开始点和终止点之间。每当运行中的一个作业由于输入／输出操作需要调用外部设备时，就把 CPU 交给另一个等待运行的作业，从而将主机与外部设备的工作由串行改变为并行，进一步避免了因主机等待外设完成任务而浪费宝贵的 CPU 时间。多道批处理系统主要有 3 个特点：**多道、宏观上并行运行和微观上串行运行**。

简单的多个作业的执行，时空图如下所示：



分段允许程序员把内存视为由这个地址空间或段组成，其中段的大小是_____。

A. 固定的

B. 不可变的

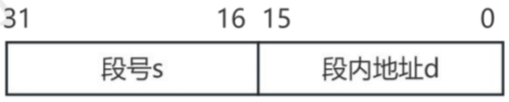
C. 相等的

D. 动态可变的

答案：D

解析：分段存储管理是将进程空间划分成若干个**段**，每段也有**段号**和**段内地址**，每段是一组**完整的逻辑信息**。

分段地址结构：



分页与分段的区别：分页是根据**物理空间**划分，每页大小相同；分段是根据**逻辑空间**划分，每段是一个完整的功能，便于共享，但是大小不同，即根据需要动态改变。

当一个进程被一个更高优先级的进程抢占或其时间片用完时，其状态会从执行态转变为_____状态。

A. 就绪

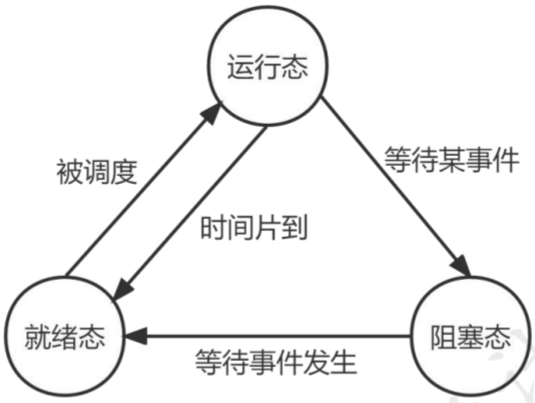
B. 挂起

C. 睡眠

D. 阻塞

答案：A

解析：当一个进程被一个更高优先级的进程抢占或其时间片用完时，其状态会**从执行态转变为绪状态**。



进程的基本状态有：

就绪状态(Ready)：进程已获得除 CPU 以外的所有必要盗源，只要再获得 CPU，就可以立即执行。例如，在多任务操作系统中，多个进程可能同时处于就绪状态，等待 CPU 调度。

执行状态(Running)：进程正在 CPU 上运行，占用 CPU 资源执行指令。

阻塞状态(Blocked)：进程因等待某一事件(如 I/O 操作完成、等待信号量等)而暂时无法继续执行，此时即使 CPU 空闲，它也不能占用 CPU 运行，

低优先级进程状态变化的原因分析：

当低优先级进程正在执行(处于执行态)时，如果高优先级进程进入就绪状态，根据优先级调度算法，操作系统会抢占 CPU，将低优先级进程从执行态变为就绪态。此时低优先级进程仍然具备执行条件，只是因为优先级较低而暂时失去 CPU 使用权。另外，如果采用时间片轮转调度算法，低优先级进程的时间片用光，它也会从执行态变为就绪态，等待下一次调度。这种机制可以保证每个就绪进程都有机会获得 CPU 时间，实现公平的资源分配。

应用程序在用户态使用特权指令进行系统调用，是_____中断。

A.信号中断

B.溢出中断

C.访管中断

D.外部中断

答案：C

解析：信号中断：通常是外部事件或异步信号触发。例如按下 Ctrl+C。 溢出中断：通常发生算术运算时候。访管中断：应用程序主动发起的软件中断。外部中断：通常由外部设备（如键盘、鼠标）等触发。

访管中断是一种自愿性的中断，由用户程序通过执行访管指令来请求操作系统服务。

访管中断是计算机系统中的一个重要概念，它涉及到用户态与核心态之间的切换。在用户态下，程序无法直接执行某些特权指令或访问受限资源，而当需要这些操作时，程序会通过执行一条特殊的访管指令来触发访管中断。这种中断机制允许用户程序以受控的方式进入核心态，从而请求操作系统提供的服务或执行特权操作。

访管中断的处理过程通常包括以下几个步骤：

1）触发访管中断：用户程序执行访管指令，产生一个中断事件（自愿中断），暂停当前用户程序的执行。

2）保存上下文：CPU 保存当前用户程序的执行状态和上下文信息，以便后续恢复执行。

3）切换到核心态：CPU 从用户态切换到核心态，开始执行操作系统的中断处理程序。

4）执行系统调用：操作系统根据访管指令中的操作数和参数，执行相应的系统调用或服务例程，完成用户请求的操作。

5）恢复用户态：系统调用或服务完成后，操作系统恢复用户程序的执行状态和上下文信息，将 CPU 切换回用户态，继续执行被中断的用户程序。

访管中断是实现用户程序与操作系统之间交互的重要机制之一，它确保了用户程序可以在受控的情况下访问受限资源或执行特权操作，同时保护了系统的稳定性和安全性。

数据库设计

数据库的三级模式结构中，_____描述局部数据的逻辑结构和特征。

A.概念模式

B.内模式

C.外模式

D.逻辑模式

答案：C

解析：在数据库的三级模式中，描述局部数据的逻辑结构和特征的是**外模式**。

1. 内模式

也称存储模式，是数据**物理结构**和**存储方式**的描述，是数据在数据库内部的表示方式，定义所有的内部记录类型、索引和文件的组织方式以及数据控制方面的细节。**一个数据库只有一个内模式**。对应数据库中的**物理存储文件**。

2. 概念模式

也称模式，是数据库中全部数据的**逻辑结构**和特征的描述。**一个数据库只有一个概念模式**。对应数据库中的**基本表**。

3. 外模式

也称用户模式或子模式，是用户与数据库系统的接口，是用户需要使用的部分数据的描述。**一个数据库可以有多个外模式**。对应数据库中的**视图**。

外模式称用户模式，是对数据库用户能看见的局部数据的逻辑结构和特征的描述。一个数据库中可以存在多个外模式。外模式可以保证数据库的安全性，每个用户看见只能和访问对应外模式中的数据，而对其余数据是不可见的。

关系操作中，操作的对象和结果都是_____。

A. 元组

B. 记录

C. 集合

D. 列

答案：C

解析：教材 2.2.2 关系运算，P227。

7. 关系运算

关系操作的特点是操作对象和操作结果都是**集合**。关系代数运算符有 4 类：集合运算符、专门的关系运算符、算术比较符和逻辑运算符。

表 9-2　关系代数运算符

运　算　符		含　义	运　算　符		含　义
集合运算符	∪ − ∩ ×	并 差 交 笛卡儿积	比较运算符	> ≥ < ≤ = ≠	大于 大于等于 小于 小于等于 等于 不等于
专门的关系运算符	σ π ⋈ ÷	选择 投影 连接 除	逻辑运算符	¬ ∧ ∨	非 与 或

数据库中有一张人员信息表包含性别属性，要求这个属性的值只能是男或者女，这属于_____。

A. 关系完整性

B. 用户定义完整性

C. 参照完整性

D. 实体完整性

答案：B

解析：教材 6.2.1 关系数据库基本概念，P227。

6. 完整性约束

完整性规则提供了一种手段来保证当用户对数据库做修改时不会破坏数据的一致性。防止对数据的意外破坏。关系模型的完整性规则是对关系的某种约束条件。关系的完整性分为三类：

①**实体完整性：**关系的主属性不能取空。

②**参照完整性：**外键的值或者为空，或者必须等于对应关系中的主键值。

员工（员工号，姓名，性别，参加工作时间，部门号），部门（部门号，名称，电话，负责人）

③**用户定义完整性：**根据语义要求所自定义的约束条件。

给出关系模式 R(A，B，C，D)和其属性之间的函数依赖(A→C，BC→D)，则 R 的码是_____。

A. B

B. C

C. AB

D. A

答案：C

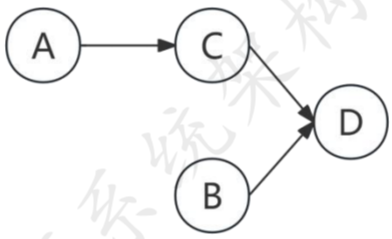
解析：•求候选键最稳靠的办法是**图示法**。图示法求候选键的过程如下：

（1）将关系的函数依赖关系用**“有向图”**的方式表示。

（2）找出**入度为 0 的属性**，并以该属性集合为起点，**尝试遍历有向图**，若能正常遍历图中所有结点，则该属性集即为关系模式的候选键。

（3）若入度为 0 的属性集不能遍历图中所有结点，则需要尝试性的将一些中间结点（既有入度，也有出度的结点）并入度为 0 的属性集中，**直至该属性集合能遍历所有结点**，则该属性集合为候选键。

本题有向图如下所示：



从图中可以看出，入度为 0 的节点是 A 与 B，从这两个节点的组合出发，能遍历全图，所以 AB 组合键为候选码。

有两个关系 R(a，b，c)和 S(b，c，d)，将 R 和 S 进行自然连接，得到的结果包含_____列。

A. 4

B. 6

C. 2

D. 5

答案：A

解析：自然连接：是一种特殊的等值连接，它要求两个关系中进行比较的分量必须是相同的属性组，**相同属性组的值要相等，并且在结果集中将重复属性列去掉**。

此题中，两个关系模式 R 与 S 的相同属性为：b、c，然后针对 R 与 S 做 b 与 c 属性的等值连

接，这样产生的结果记录为：(a，b，c，d)，所以得到的结果包含 4 列。

下面关于 SQL 注入的说法错误的是_____。

A. 使用 ORM 框架可以自动处理查询和参数化输入，但无法减少 SQL 注入的风险

B. SQL 注入是一种常见的攻击方式

C. 使用预处理语句和参数化查询是防范 SQL 注入的有效方法

D. 通过使用 UNION 关键字，攻击者可以将多个查询结果合并，从而获取数据库中的敏感信息

答案：A

解析：教材 13.4.3 ORM、Hibernate 与 CMP2.0 设计思想，P469。**使用 ORM 框架可以减少直接编写 SQL 语句的需求，从而降低 SQL 注入的风险**。ORM（Object-Relational Mapping），它在关系型数据库和对象之间作一个映射，这样，我们在具体操作数据库的时候，就不需要再去和复杂的 SQL 语句打交道，只需像操作对象一样即可。

ORM 的优点：

①使用 ORM 可以大大降低学习和开发成本；

②程序员不用再写 SQL 来进行数据库操作；

③减少程序的代码量；

④降低由于 SQL 代码质量差而带来的影响。

ORM 的缺点：

①不太容易处理复杂查询语句；

②性能较直接用 SQL 差。

计算机网络

路由器在 OSI 模型的_____。

A. 网络层

B. 物理层

C. 传输层

D. 数据链路层

答案：A

解析：P71。

协议层	互连设备	作用
物理层	**中继器** **集线器**	①**中继器**：信号在介质里传输的过程中会有衰减和噪声，使用中继器进行放大和去噪； ②**集线器**：是一种特殊的多路中继器。
数据链路层	**网桥** **交换机**	①**网桥**：连接两个局域网，**检查帧的源地址和目的地址**，若两者在同一网络段上，则不转发，否则转发到另一个网络段上； ②**交换机**：检查以太网帧的**目的地址 MAC**，在自己的地址表（端口号-MAC）中进行查找并转发。
网络层	**路由器**	用于连接多个逻辑上分开的网络，最主要的功能是**选择路由路径**。
应用层	**网关**	将协议进行转换，将数据重新分组，以便在两个不同类型的网络系统之间进行通信。

_____是在传输层定义的协议。

A. NFS 和 Telnet

B. TCP 和 UDP

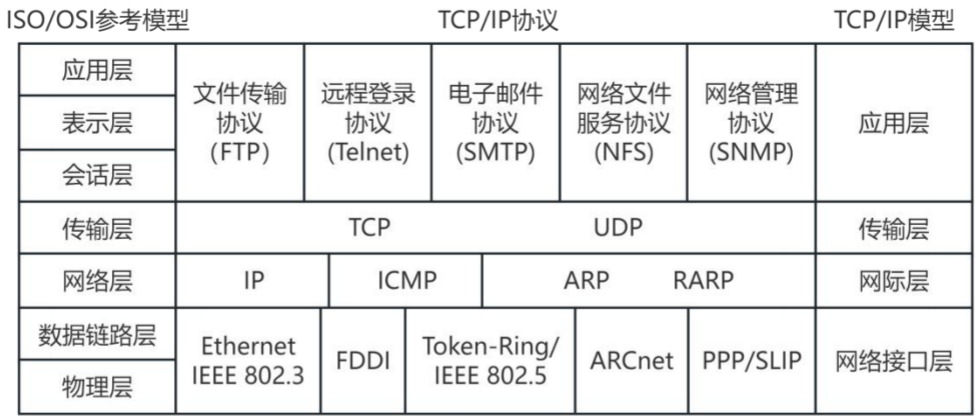
C. FTP 和 SMTP

D. IP 和 ICMP

答案：A

解析：P71。NFS、Telnet、FTP 和 SMTP 属于应用层。IP 和 ICMP 属于网络层协议，TCP 和 UDP 属于传输层协议，正确答案为 B 选项。

TCP/IP 是 Internet 的核心协议，被广泛应用于局域网和广域网中，已成为事实上的国际标准。



ISO/OSI参考模型与TCP/IP模型的对比

如何提高传输速率?正确选项是：_____。

A.提升带宽，降低信噪比

B.提升带宽，提升信噪比

C.降低带宽，提升信噪比

D.降低带宽，降低信噪比

答案：B

解析：带宽越大，理论上传输速率越高，信噪比同样对传输速率有重要影响。较高的信噪比意味着信号强度相对于噪声强度更高，这有助于更准确地接收和解析信号，从而提高传输速率。

软件工程基础知识

螺旋模型是基于_____的改进模型。

A.喷泉模型

B.快速原型模型

C.瀑布模型

D.增量模型

答案：B

解析：教材 5.1.2 软件过程改进，P178。**螺旋模型**（Spiral Model），是在**快速原型**的基础上扩展而成。对于一个复杂的大项目，开发一个原型往往达不到要求。**螺旋模型将生命周期模型（瀑布模型）和原型模型结合起来，加入两种模型均忽略的风险分析**。螺旋模型中的每个螺旋周期分为

四个步骤：**目标设定、风险分析、开发和有效性验证以及评审**。螺旋模型**强调风险分析，与瀑布模型相比，支持用户需求的动态变化。螺旋模型适合用于庞大、复杂且具有高风险的系统**。

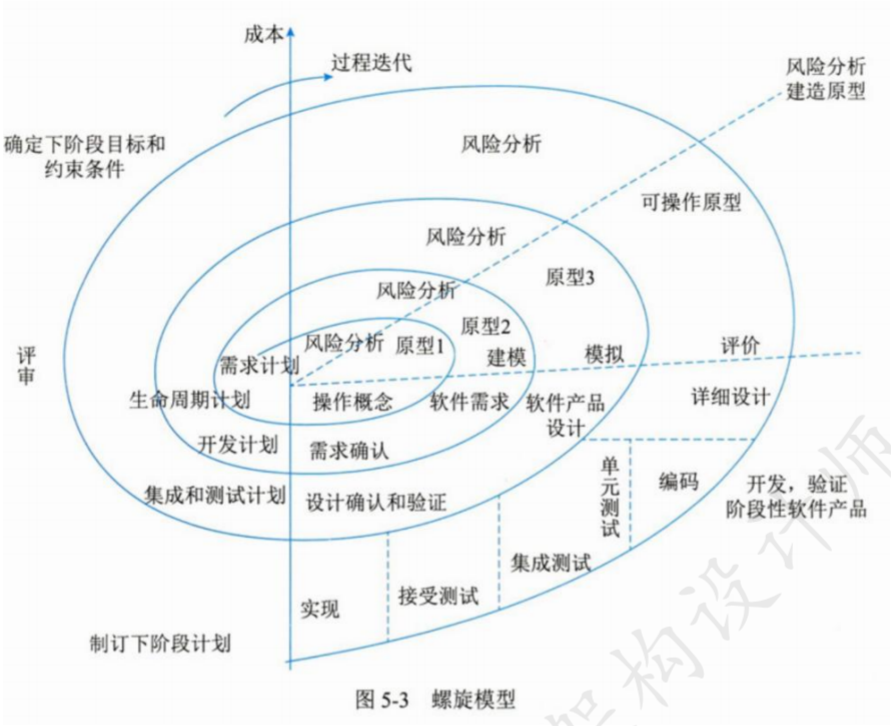


图 5-3　螺旋模型

RUP(Rational Unified Process，统一软件开发过程)是_____的、以体系结构为中心的、选代和增量的软件开发过程。

A.数据驱动

B.用例驱动

C.场景驱动

D.模型驱动

答案：B

解析：教材 5.1.4 统一过程模型（RUP），P183。**RUP 是用例驱动、以体系结构为中心、迭代和增量**的软件开发过程。其特点为：

（1）**用例驱动**：需求分析、设计、实现和测试等活动都是用例驱动的。

（2）**以体系结构为中心**：包括系统的总体组织和全局控制、通信协议等。是一个多维的结构，会采用多个视图来描述。

（3）**迭代与增量**。把整个项目开发分为**多个迭代过程**。在每次迭代中，只考虑系统的一部分需求，进行分析、设计、实现、测试和部署等过程；每次迭代是在已完成部分的基础上进行的，每次增加一些新的功能实现，以此进行下去，直至最后项目的完成。

一个模块内的处理元素是相关的，而且必须以特定次序执行，则称为_____。

A. 通信内聚

B. 逻辑内聚

C. 过程内聚

D. 时间内聚

答案：C

解析：教材 5.3.1 结构化方法，P197。

模块的内聚类型

内聚类型	描述
功能内聚	完成单一功能，各部分协同工作，缺一不可，是**最强**的内聚
顺序内聚	模块内的处理元素都密切相关且按**顺序**执**行**
通信内聚	模块内的所有处理元素集中在一个**数据结构的区域**上
过程内聚	模块内的处理元素都密切相关并按**特定的次序**执行
时间内聚	模块内的任务必须在**同一时间间隔内**执行
逻辑内聚	模块内通过参数确定完成**逻辑上相关**的一组任务
偶然内聚	也称巧合内聚，模块内的处理元素之间没有任何联系，是**最弱**的内聚

以下_____属于是白盒测试。

A.控制流分析

B.数据流分析

C.程序变异测试

D.功能测试

答案：D

解析：白盒测试，也称**结构测试**，根据程序的内部结构和逻辑来设计测试用例，对程序的路径和过程进行测试，检查是否满足设计的需要。常用的白盒测试技术有有**控制流分析、数据流分析、路径分析、程序变异**等。根据测试用例的覆盖程度，分为**语句覆盖、判定覆盖、分支覆盖和路径覆盖**等。

控制流分析：属于白盒测试中的静态分析方法之一，主要通过分析程序的控制流图，来确定

程序的执行路径和逻辑结构，从而设计测试用例以覆盖不同的路径和分支。

数据流分析：也是白盒测试的静态分析方法之一，侧重于分析程序中数据的流动和使用情况，包括数据的定义、引用、赋值等，以发现数据在流动过程中可能出现的问题，如未初始化的变量、数据的别名等。

路径分析：也是白盒测试的静态分析方法之一，是通过分析程序的所有可能执行路径来设计测试用例。路径分析的目标是覆盖程序中的所有可能路径，但由于路径数量可能非常庞大（甚至无限），通常采用一些简化策略，如基本路径测试。基本路径测试通过计算程序的圈复杂度（Cyclomatic Complexity）来确定需要测试的基本路径数量。

程序变异测试：是白盒测试的一种动态分析方法，通过故意引入错误（变异）到程序中，然后运行测试用例来检测这些错误是否被发现，从而评估测试用例的有效性和充分性。

黑盒测试，也称**功能测试**，在不考虑软件内部结构和特性的情况下，根据功能设计用例，测试软件功能。常用的黑盒测试技术有**等价类划分、边界值分析、错误推测、因果图和判定表**等。

下面关于测试覆盖范围关系，_____是不正确的，

A.语句覆盖覆盖是一种最弱的覆盖

B.判定覆盖比语句覆盖强

C.判定/条件覆盖可以代替条件和判断覆盖

D.路径覆盖可以代替判断覆盖和条件组合覆盖

答案：D

解析：常用的白盒测试技术有**逻辑覆盖**、循环覆盖和基本路径测试。逻辑覆盖：**语句覆盖、判定覆盖、条件覆盖、判定/条件覆盖、条件组合覆盖和路径覆盖**。

●**语句覆盖**（SC）：是指选择足够多的测试用例，使被测程序的**每条语句至少执行一次**。语句覆盖是一种很弱的逻辑覆盖。

●**判定覆盖**（DC）：又称分支覆盖，是指设计足够的测试用例，使得**不仅每条语句至少执行一次，而且每个判定表达式的每种可能的结果（分支）都至少执行一次**。判定覆盖比语句覆盖强。

●**条件覆盖**（CC）：是指设计一组测试用例，不仅每条语句至少执行一次，而且使判定表达式中的每个条件的所有可能的值至少满足一次。条件覆盖不一定包含判定覆盖、判定覆盖也不一定包含条件覆盖。

●**判定/条件覆盖**（CDC）：是指同时满足**判定覆盖和条件覆盖**的逻辑覆盖。设计足够的测试用例，使得**判定表达式中每个条件的所有可能的值至少满足一次，而且每个判定本身的所有可能**

结果也至少出现一次。

●**条件组合覆盖**（MCC）：设计足够的测试用例，使得**每个判定表达式中条件的所有值的组合都至少出现一次**。满足条件组合覆盖的测试用例，也一定满足判定覆盖、条件覆盖和判定/条件覆盖。

●**路径覆盖**：设计足够的测试用例，覆盖被测程序中**所有可能的路径**（如果程序中有环路，则要求每条环路至少经过一次）。路径覆盖考虑了程序中各种判定结果的所有可能组合，因此是一种较强的覆盖标准。但**路径覆盖并未考虑判定中的条件结果的组合，并不能代替条件覆盖和条件组合覆盖**。

软件测试过程中的系统测试主要是为了发现_____阶段的错误。

A.编码

B.需求分析

C.概要设计

D.详细设计

答案：B

解析：**系统测试**的对象是完整、集成的计算机系统。**目的是通过与软件需求进行比较，发现所开发的系统与用户需求不符或矛盾的地方。测试依据是用户需求（SRS）或开发合同**。主要包括**功能测试、性能测试**、健壮性测试、安装或反安装测试、用户界面测试、压力测试、可靠及安全性测试等。一般采用黑盒测试法。

基于数据驱动的测试脚本写在_____文件里 。

A. 数据文件

B. 脚本文件

C. 程序文件

D. 系统文件

答案：B

解析：基于数据驱动的测试中，测试数据通常存储在数据文件中，如 Excel、CSV、JSON、YAML 等格式。而测试脚本则写在脚本文件中，通过相应的代码逻辑在脚本中实现从数据文件中读取数据，并根据数据进行测试操作，将测试数据与测试逻辑分离。

在软件的使用过程中，用户往往会对软件提出新的功能与性能要求。为了满足这些要求，需要修改或再开发软件，以扩充软件功能、增强软件性能、改进加工效率、提高软件的可维护性。这种情况下进行的维护活动称为_____。

A. 预防性维护

B. 适应性维护

C. 完善性维护

D. 改正性维护

答案：C

解析：P110。系统维护主要包括硬件维护、**软件维护**和数据维护。

2. 软件维护类型

（1）**正确性维护：**改正在系统开发阶段已发生而系统测试阶段尚未发现的错误。修正 BUG、错误。

（2）**适应性维护：**使应用软件适应环境变化（外部环境、数据环境）而进行的修改。应变。

（3）**完善性维护：为扩充功能和改善性能而进行的修改。占整个维护工作的 50%~60%。**新需求。

（4）**预防性维护：**为适应未来的软/硬环境的变化，应主动增加预防性的新功能，以使应用系统适应各类变化而不被淘汰。针对未来。

不包括配置管理的是_____。

A.UML

B.ISO9000

C.PMBOK

D.CMMI

答案：A

解析：超纲题。UML、ISO9000、PMBOK 和 CMMI 都是与软件工程和项目管理相关的标准或模型，但它们与配置管理的关系如下：

UML（统一建模语言）：是一种用于软件系统建模和其它系统的图形化建模语言。它主要关注于软件开发的分析与设计阶段，并不直接包含配置管理的内容。

ISO9000：是一系列关于质量管理体系的国际标准，它提供了组织如何建立、实施、维护和改进质量管理体系的指导原则。ISO9000 标准中包含了对文件控制和记录保存的要求，这些都是

配置管理的重要组成部分。

PMBOK（项目管理知识体系指南）：是由项目管理协会（PMI）出版的一本指南，它提供了项目管理的一套标准和最佳实践。PMBOK 中包含了项目配置管理的内容。

CMMI（能力成熟度模型集成）：是一个过程改进的框架，它提供了一套过程域，帮助组织改进其软件工程过程。CMMI 中包含了配置管理作为一个关键的过程域。

因此，不包括配置管理的是选项 A。UML 是一种建模语言，不直接涉及配置管理。

不同应用领域中的软件元素，例如数据结构、分类算法和人机界面等属于_____重用。

A.横向重用

B.纵向重用

C.交叉重用

D.可控重用

答案：A

解析：超纲题（2023.11 网络规划设计师真题）。**软件重用**是指在两次或多次不同的软件开发过程中重复使用相同或相似软件元素的过程。按照重用活动是否跨越相似性较少的多个应用领域，软件重用可以区别为**横向重用和纵向重用**。

横向重用（水平式重用）是指重用不同应用领域中的软件元素，例如**数据结构、分类算法和人机界面构建**等。**标准函数是种典型的、原始的横向重用机制**。

纵向重用（垂直式重用）是指在一类具有较多公共性的应用领域之间进行软部件重用。纵向重用活动的主要关键点是域分析：根据应用领域的特征及相似性预测软部件的可重用性。

交叉重用：非标准术语，通常不用于描述软件重用分类。

可控重用：强调对重用过程的管理（如版本控制），而非重用类型本身。

有 ABCD 四项任务，表格展示每项任务的依赖关系，A->B，A->C->D，单位：万。

A	-	3	15	1	25
B	A	7		5	10
C	A	3	5	2	15
D	C	3		2	10

某项目包括 A、B、C、D 四道工序，各各道工序之间的衔接关系如图所示，项目每个项目每天的间接费用是 2 万，求最短工期情况下，需要多少工程费用?

A 122

B.142

C.132

D.130

答案：C

解析：题目不完整。

面向对象技术

在 UML 用例图中，参与者之间存在_____关系。

A. 聚合

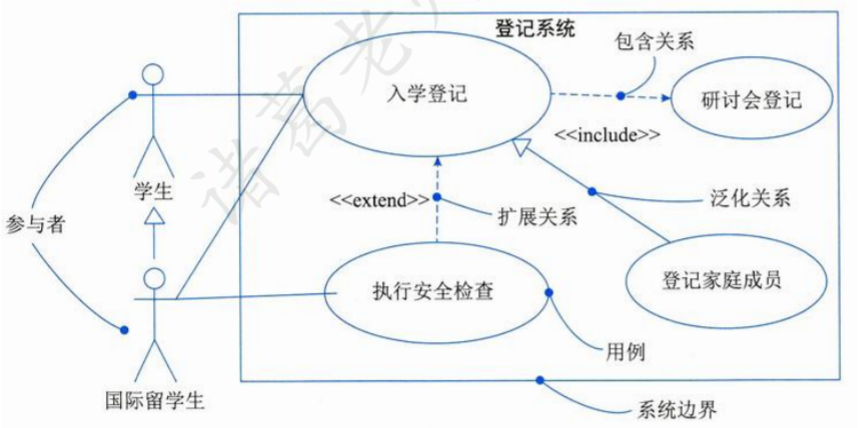
B. 包含

C. 继承

D. 扩展

答案：C

解析：用例图描述了一组**用例、参与者以及它们之间的关系**。这里没考查用例之间的关系，而是考查了参与者之间的关系，但我们上课时给大家讲了学生登记的例子，其中讲了参与者->国际留学生与参与者->学生之间是**继承**（泛化）的关系。



使用哪两个图可以描述用户界面元素和界面跳转_____。

A.用例图、活动图

B.用例图、顺序图

C.类图、活动图

D.类图、顺序图

答案：D

解析：类图描述**类、类的特性以及类之间的关系**。

顺序图又称**序列图**，描述**对象之间的交互（消息的发送与接收）**，重点在于**强调顺序（时间顺序）**，反映对象间消息的发送与接收。

用例图描述了**一组用例、参与者以及它们之间的关系**。(也就是描述系统与外部系统及用户的交互，用图形描述谁将使用系统，用户期望用什么方式与系统交互。常用于系统语境建模、系统需求建模)

活动图描述**过程行为和并行行为**。它是一种特殊的状态图，展现了**在系统内从一个活动到另一个活动的流程**，对于系统的功能建模特别重要，并强调对象间的控制流程。

以下关于设计模式描述正确的是_____。

A. 装饰器模式属于行为型模式

B. 原型模式属于创建型模式

C. 解释器模式和代理模式属于同一类型模式

D. 观察者模式属于结构型模式

答案：B

解析：装饰器模式是结构型模式；**原型模式属于创建型模式**；解释器模式属于行为型模式，代理模式属于结构型模式，两者不属于同一类型模式；观察者模式属于行为型模式；综上可得选项 B 是正确的。

设计模式分类

	创建型	**结构型**	**行为型**
类	工厂方法模式	适配器模式（类）	解释器模式 模板方法模式
对象	抽象工厂模式 生成器模式 原型模式 单例模式	适配器模式（对象） 桥接模式 组合模式 装饰模式 外观模式	责任链模式 命令模式 迭代器模式 中介者模式 备忘录模式

		享元模式 代理模式	观察者模式 状态模式 策略模式 访问者模式

信息安全基础知识

下面关于 MD5 的说法，错误的是_____。

A.任意长度生成 128 位，不可逆

B.通过数据碰撞都无法进行解密还原

C.MD5 算法可以用来校验数据的完整性

D.MD5 算法可以用来进行加密

答案：D

解析：MD5 产生 128 位的散列值，且不可逆，A 正确。

“数据碰撞”指的是找到两个不同的输入产生相同的哈希值。**任何不同的输入数据，都会产生不同的信息摘要**。MD5 存在碰撞漏洞（不同输入可能生成相同哈希值），但碰撞与“解密”无关。哈希算法本身不涉及加密密钥，因此无法通过碰撞实现数据还原，B 正确。

MD5 常用于校验数据完整性，通过对比哈希值判断数据是否被篡改。即使微小改动也会导致哈希值显著变化，C 正确。

加密（如 AES、RSA）需要可逆性，即加密后的数据能通过密钥解密还原。而哈希是单向过程，无解密机制。MD5 生成的哈希值无法恢复原始数据，因此不能用于加密，D 错误。

安全审计 4 要素包括_____。

A.控制目标、安全漏洞、控制措施和控制测试

B.控制流程、安全漏洞、控制措施和控制测试

C.控制目标、安全漏洞、控制开发和控制测试

D.控制目标、安全漏洞、控制脚本和控制测试

答案：A

解析：超纲题。安全审计 4 要素包括**控制目标、安全漏洞、控制措施和控制测试**。

安全审计（补充）

1. 安全审计的定义

安全审计是通过系统的、独立的检查，评估信息系统的安全性，识别潜在的安全风险，并提出改进建议的过程。其目的是确保系统的机密性、完整性和可用性（CIA 三元组）。

2. 安全审计的 4 要素

安全审计通常包含以下 4 个核心要素：

（1）控制目标

①明确审计的目标，例如确保系统符合安全策略、法律法规或行业标准。

②目标可以是防止未授权访问、检测异常行为或确保数据完整性。

（2）安全漏洞

①识别系统中可能存在的安全漏洞或潜在风险。

②包括技术漏洞（如软件缺陷、配置错误）和管理漏洞（如权限分配不当）。

（3）控制措施

①制定和实施安全控制措施，以防范或减少安全风险。

②例如访问控制、加密、日志记录、入侵检测系统（IDS）等。

（4）控制测试

①通过测试验证控制措施的有效性。

②包括渗透测试、漏洞扫描、日志分析等方法。

3. 安全审计的主要内容

安全审计通常涵盖以下方面：

- **访问控制审计**：检查用户权限分配是否合理，是否存在未授权访问。
- **日志审计**：分析系统日志，检测异常行为或安全事件。
- **配置审计**：检查系统配置是否符合安全基线要求。
- **数据完整性审计**：确保数据在传输和存储过程中未被篡改。
- **合规性审计**：评估系统是否符合相关法律法规或行业标准（如 GDPR、等级保护）

4. 安全审计的流程

安全审计通常遵循以下流程：

（1）规划：确定审计范围、目标和资源。

（2）数据收集：收集系统日志、配置信息、访问记录等。

（3）分析：识别安全漏洞和潜在风险。

（4）报告：记录审计结果，提出改进建议。

（5）整改：跟踪并验证整改措施的实施情况。

5. 安全审计的工具

常用的安全审计工具包括：

- 漏洞扫描工具：如 Nessus、OpenVAS。
- 日志分析工具：如 Splunk、ELK Stack。
- 渗透测试工具：如 Metasploit、Burp Suite。
- 配置审计工具：如 OpenSCAP、Lynis。

6. 安全审计的意义

- 风险识别：帮助发现系统中的安全漏洞和潜在风险。
- 合规性：确保系统符合法律法规和行业标准。
- 持续改进：通过审计结果优化安全策略和控制措施。
- 事件响应：为安全事件的调查和响应提供依据。

数据分类分级属于数据安全治理体系的_____。

A. 基础安全层

B. 权限控制层

C. 战略安全层

D. 数据全生命周期安全

答案：A

解析：超纲题。数据分类分级属于数据安全治理体系的**基础安全层**。出自《数据安全治理能力通用评估方法》（YD/T 4558-2023）。

信息系统基础知识

信息化需求包含三个层次，即_____。

A. 战略需求、运作需求和信息资源开发

B. 战略需求、运作需求和技术需求

C. 战略需求、信息资源开发和技术需求

D. 信息资源开发、运作需求和技术需求

答案：B

解析：超纲题。信息化需求包含 3 个层次，即：**战略需求、运作需求和技术需求**。

战略需求：组织信息化的战略需求的目标是提升组织的竞争能力、为组织的可持续发展提供一个支持环境。从某种意义上来说，信息化对组织不仅仅是服务的手段和实现现有战略的辅助工具，信息化可以把组织战略提升到一个新的水平，为组织带来新的发展契机。特别是对于企业，信息化战略是企业竞争的基础。

运作需求：组织信息化的运作需求是组织信息化需求非常重要且关键的一环，它包含三方面的内容：一是实现信息化战略目标的需要；二是运营策略的需要；三是人才培养的需要。

技术需求：由于系统开发时间过长等问题在信息技术层面上对系统的完善、升级、集成和整合提出了需求。也有的组织，原来基本上没有大型的信息系统项目，有的也只是一些单机应用，这样的组织的信息化需求，一般是从头开发新的系统。

专家系统能够学习的机制有_____。

A.知识库、推理机

B.注意力机制、推理机

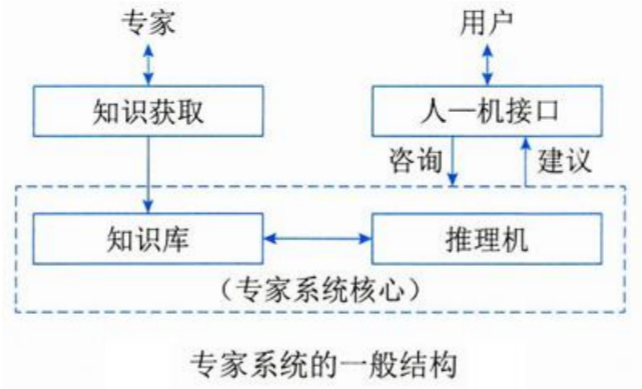
C.知识库、注意力机制,

D.注意力机制、综合知识库

答案：A

解析：基于知识的专家系统简称为**专家系统**（Expert System，**ES**），它是**人工智能的一个重要分支**。专家系统的能力来自于它所拥有的专家知识，知识的表示及推理的方法则提供了应用的机理。因此这种基于知识的系统设计是**以知识库和推理机为中心**而展开的。

知识+推理=系统



专家系统的一般结构

软件架构设计

在体系结构描述中，典型的 4+1 视图模型包括_____。

A. 逻辑视图、进程视图、开发视图、物理视图和场景

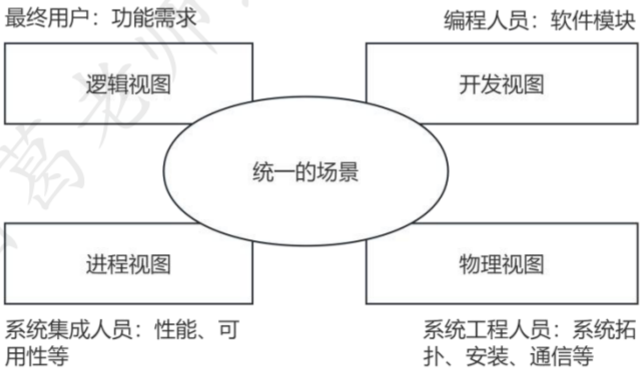
B. 逻辑视图、进程视图、开发视图、部署视图和场景

C. 逻辑视图、进程视图、开发视图、物理视图和类视图

D. 逻辑视图、进程视图、开发视图、物理视图和部署视图

答案：A

解析：教材 7.1.2 软件架构设计与生命周期，P249。软件架构设计的 4+1 视图：



逻辑视图：也称设计视图，主要描述系统的功能需求。

进程视图：也称过程视图，主要关注一些非功能性的需求，例如系统的性能和可用性。进程视图强调并发性、分布性、系统集成性和容错能力，以及逻辑视图中的主要抽象的进程结构。

开发视图：也称实现视图，侧重于软件模块的组织和管理。

物理视图：主要描述如何把软件映射到硬件上，通常要考虑到解决系统拓扑结构、系统安装、通信等问题。

统一的场景：可以看作是那些重要系统活动的抽象，它使 **4 个视图有机地联系起来**，从某种意义上说，**场景是最重要的需求抽象**。在开发架构时，它可以帮助设计者找到架构的构件以及它们之间的作用关系。**逻辑视图和开发视图描述系统的静态结构，而进程视图和物理视图描述系统的动态结构**。对于不同的软件系统来说，侧重的角度也有所不同。例如，对于管理信息系统来说，比较侧重于从逻辑视图和开发视图来描述系统，而对于实时控制系统来说，则比较注重于从进程视图和物理视图来描述系统。

ABSD 体系结构需求来自_____。

A.客户的需求目标、系统的商业目标，系统开发人员的商业目标

B.系统的功能需求、系统的商业目标，系统开发人员的开发任务

C.系统的质量目标、系统的质量要求，系统开发人员的商业目标

D.系统的质量目标、系统的商业目标，系统开发人员的商业目标

答案：D

解析：**基于架构的软件设计**（Architecture-Based Software Design，**ABSD**）方法是**由架构驱动**，即由构成架构的**商业、质量和功能需求**的组合驱动架构设计。**系统开发人员的商业目标即功能需求**。

在仓库体系结构风格中，_____用于说明当前数据的状态。

A. 黑板

B. 中央数据结构

C. 独立构件

D. 知识源

答案：B

解析：教材 7.3.4 以数据为中心的体系结构风格，P262。

1. 仓库体系结构风格

仓库（Repository）是存储和维护数据的中心场所。仓库架构风格由**中央数据结构（说明当前数据状态）和一组独立构件（对中央数据进行操作）**组成。连接件是仓库与独立构件之间的交互。

_____不属于调用/返回体系结构风格的子风格。

A. 层次型体系结构风格

B. 主程序/子过程体系结构风格

C. 黑板体系结构风格

D. 面向对象体系结构风格

答案：C

解析：教材 7.3.3 调用/返回体系结构风格，P261。**调用/返回风格**是指在系统中采用了**调用与返回机制**。其主要思想是将一个复杂的大系统分解为若干子系统，以便降低复杂度，并且增加可修改性。构件之间**存在互相调用的关系，一般是显式的调用**，主要包括**主程序/子程序风格、面向对象风格、层次型风格、客户端/服务器风格以及浏览器/服务器体系结构风格**。

以数据为中心的体系结构以**数据**为中心，所有的操作都是围绕建立的数据中心进行的，主要包括**仓库体系结构风格和黑板体系结构风格**。

关于软件架构风格的描述，以下说法正确的是_____。

A.批处理是并行的

B.管道-过滤器可以是并行的

C.黑板和管道-过滤器都是以数据为中心架构风格的子类型

D.在数据流架构风格中，批处理风格本身主要强调的是顺序执行

答案：B

解析：数据流体系结构风格，面向数据流，按照一定的顺序从前向后执行程序，主要包括**批处理风格和管道-过滤器风格**。

1. 批处理体系结构风格

构件为一系列固定顺序的计算单元，构件之间**只通过数据传递交互**。每个处理步骤是一个**独立的程序，每一步必须在其前一步结束后才能开始**，数据必须是完整的，以**整体的方式**传递。构件是独立的应用程序，连接件是某种类型的媒介。

2. 管道-过滤器体系结构风格

每个过滤器（处理步骤）都有一组输入和输出，过滤器从管道（数据传输）中读取输入的数据流，经过内部处理，产生输出数据流并写入管道中。**前一个过滤器的输出作为后一个过滤器的输入**，前后数据流关联。构件就是过滤器，连接件就是管道。早期编译器就是采用的这种架构，要一步一步处理的，均可考虑此架构风格。

二者区别在于**批处理前后构件不一定有关联，并且是作为整体传递**，即必须前一个执行完才

能执行下一个，因此批处理是串行的，A 错误。而管道-过滤器是**前一个输出作为后一个输入，前面执行到部分可以开始下一个的执行**，因此可以使串行的，也可以是并行的，B 正确。

黑板架构风格是以数据为中心的架构风格子类型，管道-过滤器是数据流架构风格的子类型，C 错误。

批处理架构风格是一种针对大规模数据处理、批量任务执行而设计的软件架构风格，其核心特点是将数据处理任务分解为一系列独立的、顺序执行的步骤（或称阶段、任务、作业），因此本身强调的是并行的，D 错误。此选项题目可能不对。

进程通信体系结构风格的连接件是_____。

A.消息传递

B.消息队列

C.消息中间件

D.独立的进程

答案：A

解析：教材 7.3.6 独立构件体系结构风格，P264。**进程通信体系结构风格：构件是独立的进程，连接件是消息传递**。构件通常是命名过程，消息传递的方式可以是**点对点、异步或同步方式，以及远程过程（方法）调用**等。

独立构件事件驱动架构，是消息发送给_____模块。

A.独立、耦合

B.独立、非耦合

C.非独立、耦合

D.非独立、非耦合

答案：B

解析：独立构件体系结构风格主要强调构件之间是互相**独立**的，不存在显式的调用关系，而是**通过某个事件触发、异步的方式来执行**，以降低耦合度，提升灵活性。主要包括**进程通信和事件系统风格（隐式调用）**。

Web 服务描述语言 WSDL 描述了 Web 服务的三个基本属性，包括_____。

A. 谁要访问服务、如何访问服务和服务位于何处

B. 服务做什么、谁要访问服务和服务位于何处

C. 服务做什么、如何访问服务和谁要访问服务

D. 服务做什么、如何访问服务和服务位于何处

答案：D

解析：教材 15.4.2 WSDL 规范，P522。WSDL(Web 服务描述语言)，是一个用来描述 Web 服务和说明如何与 Web 服务通信的 XML 语言。可描述三个基本属性：服务做些什么、如何访问服务、服务位于何处。

（1）服务做些什么——服务所提供的操作（方法）。

（2）如何访问服务——和服务交互的数据格式以及必要协议。

（3）服务位于何处——协议相关的地址，如 URL。

关于 SOAP(Simple Object Access Protocol，简单对象访问协议)，下面描述错误的是_____。

A.提供什么服务，如何使用，谁可以使用

B.信封和 XML 编码定义在相同命名空间

C.SOAP 封装，定义了一个描述消息中的内容是什么，是谁发送的，谁应当接收并处理它以及如何处理它们的框架

D.SOAP RPC 表示是远程过程调用和应答的协定

答案：B

解析：教材 15.4.3 SOAP 协议，P523。SOAP 是在分散或分布式的环境中交换信息的简单的协议，是一个基于 XML 的协议。它包括 4 个部分：SOAP 封装，定义了一个描述消息中的内容是什么，是谁发送的，谁应当接收并处理它以及如何处理它们的框架；SOAP 编码规则，用于表示应用程序需要使用的数据类型的实例；SOAP RPC 表示是远程过程调用和应答的协定；SOAP 绑定是使用底层协议交换信息。

虽然这 4 个部分都作为 SOAP 的一部分，作为一个整体定义的，但它们在功能上是相交的、彼此独立的。特别地，**信封和编码规则是被定义在不同的 XML 命名空间（Namespace）中**，这样使得定义更加简单。

系统质量属性与架构评估

基于软件系统的生命周期，可以将软件系统的质量属性分为开发期质量属性和运行期质量属

性两个部分，其中，__（1）__关注软件因适应新需求或需求变化而增加新功能的能力；__（2）__是关注软件系统同时兼顾向合法用户提供服务，以及阻止非授权使用的能力。

（1）A. 安全性

B. 可扩展性

C. 性能

D. 可重用性

（2）A. 可测试性

B. 安全性

C. 可移植性

D. 可用性

答案：B B

解析：P271。基于软件系统的生命周期，可以将软件系统的质量属性分为**开发期质量属性**和**运行期质量属性** 2 个部分。

1. 开发期质量属性

开发期质量属性主要指在软件开发阶段所关注的质量属性，主要包含 6 个方面。

（1）**易理解性：**指设计被开发人员理解的难易程度。

（2）**可扩展性：软件因适应新需求或需求变化而增加新功能的能力，也称为灵活性。**

（3）**可重用性：**指重用软件系统或某一部分的难易程度。

（4）**可测试性：**对软件测试以证明其满足需求规范的难易程度。

（5）**可维护性：**当需要修改缺陷、增加功能、提高质量属性时，识别修改点并实施修改的难易程度。

（6）**可移植性：**将软件系统从一个运行环境转移到另一个不同的运行环境的难易程度。

2. 运行期质量属性

运行期质量属性主要指在软件运行阶段所关注的质量属性，主要包含 7 个方面。

（1）**性能：**性能是指软件系统及时提供相应服务的能力，如速度、吞吐量和容量等的要求。

（2）**安全性：指软件系统同时兼顾向合法用户提供服务，以及阻止非授权使用的能力。**

（3）**可伸缩性：**指当用户数和数据量增加时，软件系统维持高服务质量的能力。例如，通过增加服务器来提高能力。

（4）**互操作性：**指本软件系统与其他系统交换数据和相互调用服务的难易程度。

（5）**可靠性：**软件系统在一定的时间内持续无故障运行的能力。

（6）**可用性**：指系统在一定时间内正常工作的时间所占的比例。可用性会受到系统错误，恶意攻击，高负载等问题的影响。

（7）**鲁棒性**：是指软件系统在非正常情况（如用户进行了非法操作、相关的软硬件系统发生了故障等）下仍能够正常运行的能力，也称健壮性或容错性。

安全性可划分为多种特性。其中__（1）__保证信息不泄露给未授权的用户、实体或过程；__（2）__保证信息的完整和准确，防止信息被非法修改。

（1）A. 不可否认性

B. 机密性

C. 可控性

D. 完整性

（2）A. 可控性

B. 不可否认性

C. 机密性

D. 完整性

答案：B D

解析：本题考查面向架构评估的质量属性，P273。**安全性**：是指系统在向合法用户提供服务的同时能够阻止非授权用户使用的企图或拒绝服务的能力。可分为**机密性、完整性、不可否认性、可控性**。**机密性**保证信息不泄露给未授权的用户、实体或过程；**完整性**保证信息的完整和准确，防止信息被非法修改；不可否认性是指信息交换的双方不能否认其在交换过程中发送信息或接收信息的行为；可控性保证对信息的传播及内容具有控制的能力，防止为非法者所用。设计策略：入侵检测、用户认证、用户授权、追踪审计。

以下_____不属于可用性的选项。

A. 可用时间

B. 可用时间间隔

C. 数据延迟时间

D. 故障间隔

答案：C

解析：**可用性**：是系统**能够正常运行的时间比例**，经常用两次故障之间的时间长度或在出现故障

时系统能够恢复正常的速度来表示。如**故障间隔时间**。设计策略：**冗余、心跳、选举、Ping/Echo**。

可用性质量属性场景所关注的方面包括系统故障发生的频率、出现故障时会发生什么情况、允许系统有多长时间是非正常运行、什么时候可以安全地出现故障、如何防止故障的发生以及发生故障时要求进行哪种通知。

这里的可用时间、可用时间间隔明显属于可用性，而**数据延迟时间属于性能**。

可用时间：直接关联到系统的可用性，指系统能够正常工作的时间，是衡量可用性的重要指标之一。

可用时间间隔：可以理解为系统在两次故障之间的可用时间段，与可用性密切相关，它反映了系统在正常运行情况下能够持续工作的时间长度。

故障间隔：通常指系统两次故障之间的时间间隔，是评估系统可用性的关键指标之一。故障间隔越长，说明系统的稳定性越高，可用性也就越高。

而**数据延迟时间**是指数据从一个点传输到另一个点所需的时间，这更多地与**性能**相关，而不是直接与可用性挂钩。虽然低延迟对于很多应用的用户体验很重要，但它并不直接衡量系统是否可用。

以下_____不属于系统易用性关注的指标。

A. 服务器修复能力

B. 用户满意度

C. 系统学习曲线

D. 操作效率

答案：A

解析：**易用性**质量属性场景主要关注用户在使用系统时的容易程度，包括**系统的学习曲线、完成操作的效率、对系统使用过程的满意程度**等。

服务器修复能力更多地与系统的可靠性和维护性相关，而不是易用性，因此，选项 A 不属于系统易用性关注的指标。

易用性通常关注用户与系统交互的便捷性、直观性和效率，以下是易用性通常关注的指标：

用户满意度：用户对系统易用性的直接反馈，是衡量系统易用性的重要指标。

系统学习曲线：指用户从开始使用系统到熟练操作所需的时间和努力，与易用性密切相关。

操作效率：涉及用户完成任务的速度和效率，是易用性的关键组成部分。

除以上外，易用性关注的指标还有界面一致性、错误处理、文档和帮助等。

时间要求和安全性是一对矛盾点，某项目对时间有明确要求，这时安全性就成了_____。

A.敏感点

B.权衡点

C.风险点

D.非风险点

答案：B

解析：敏感点和权衡点。 敏感点和权衡点是关键的架构决策。**敏感点：**是指**为了实现某种特定的质量属性**，一个或多个构件所具有的特性。**权衡点：**是**影响多个质量属性的特性**，是多个质量属性的敏感点。改变加密级别可能会对安全性和性能产生非常重要的影响。提高加密级别可以提高安全性，但可能要耗费更多的处理时间，影响系统性能。如果某个机密消息的处理有严格的时间延迟要求，则加密级别可能就会成为一个**权衡点**。

风险点与非风险点：不是以标准专业术语形式出现的，只是一个常规概念，即可能引起风险的因素，可称为风险点。某个做法如果有隐患，有可能导致一些问题，则为风险点；而如果某件事是可行的可接受的，则为非风险点。

质量属性场景的_____是激励到达后所采取的行动。

A.刺激源

B.响应

C.制品

D.环境

答案：B

解析：质量属性场景描述，为了精确描述软件系统的质量属性，通常采用**质量属性场景**（Quality Attribute Scenario）作为描述质量属性的手段。质量属性场景是一种**面向特定质量属性的需求**。它由 6 部分组成：

- 刺激源（Source）：这是某个**生成该刺激的实体**（人、计算机系统或者任何其他刺激器）。
- 刺激（Stimulus）：该刺激是**当刺激到达系统时需要考虑的条件**。
- 环境（Environment）：该刺激**在某些条件内发生**。当激励发生时，系统可能处于过载、运行或者其他情况。
- 制品（Artifact）：某个制品**被激励**。这可能是整个系统，也可能是系统的一部分。

• 响应（Response）：该响应是在激励到达后所采取的行动。

• 响应度量（Measurement）：当响应发生时，**应当能够以某种方式对其进行度量**，以对需求进行测试。

基于度量的评估方法，首先需要建立质量属性和度量之间的映射原则，然后从软件架构文档中获取度量信息，最后根据映射原则分析推导出系统的_____。

A. 质量属性

B. 功能性

C. 架构元素

D. 架构决策

答案：A

解析：教材 8.2 系统架构评估，P277。**基于度量的评估方法：**制定一些**定量指标**来度量架构，如代码行数等。要制定质量属性和度量结果之间的映射，要求评估人员对架构熟悉。它是建立在软件架构度量的基础上的，涉及 3 个基本活动，首先需要建立质量属性和度量之间的映射原则，然后从软件架构文档中获取度量信息，最后根据映射原则分析推导出系统的**质量属性**。

在软件架构评估中，__（1）__方法采用效用树这一工具来对质量属性进行分类和优先级排序。效用树的结构包括：__（2）__。

（1）A. SAEM

B. ATAM

C. SAAM

D. CBAM

（2）A. 树根--质量属性--属性分类--质量属性场景(叶子节点)

B. 树根--属性分类--属性描述--质量属性场景(叶子节点)

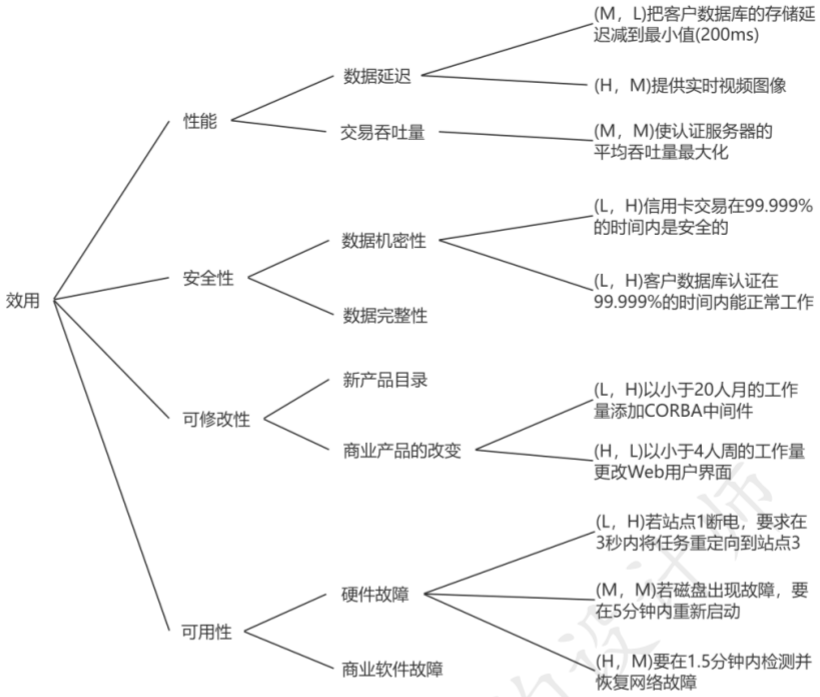
C. 树根--质量属性--属性描述--质量属性场景(叶子节点)

D. 树根--功能需求--需求描述--质量属性场录(叶子节点)

答案：B A

解析：教材 8.2.2 系统架构评估方法，P279。ATAM 方法采用**效用树**（Utility tree）这一工具来对质量属性进行分类和优先级排序。**效用树的结构包括：树根--质量属性--属性分类--质量属性场景（叶子节点）**。需要注意的是，ATAM 主要关注 4 类质量属性：**性能、安全性、可修改性和可用**

性，这是因为这 4 个质量属性是利益相关者最为关心的。



利用质量属性效用树进行评估的是_____架构。

A.CBAM

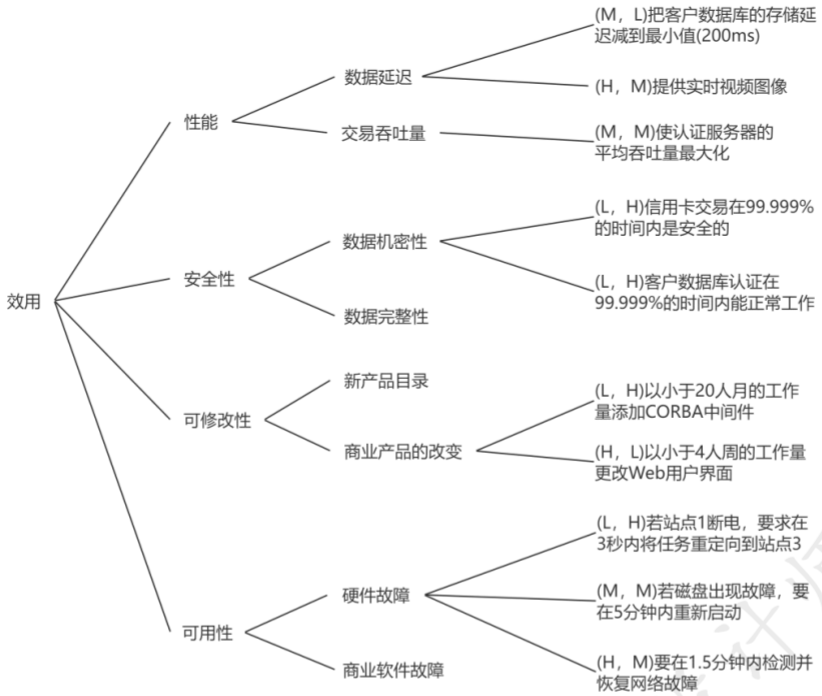
B.SAAM

C.ATAM

D.ABSD

答案：C

解析：教材 8.2.2 系统架构评估方法，P279。ATAM 方法采用**效用树**（Utility tree）这一工具来对质量属性进行分类和优先级排序。**效用树的结构包括：树根--质量属性--属性分类--质量属性场景（叶子节点）**。需要注意的是，ATAM 主要关注 4 类质量属性：**性能、安全性、可修改性和可用性**，这是因为这 4 个质量属性是利益相关者最为关心的。



ATAM 使用头脑风暴的三种场景不包括_____。

A.用例场景

B.增长情景

C.探索性场景

D.环境场景

答案：D

解析：ATAM 使用头脑风暴的三种场景：

- **用例场景**：在这种情况下，利益相关者就是最终用户。
- **增长情景**：代表了架构发展的方式。
- **探索性场景**：代表架构中极端的增长形式。

软件可靠性基础知识

N 版本设计较传统多了哪三个步骤_____。

A.相同成分规范评审、相异性确认、面对面测试

B.相同成分规范评审、相异性确认、背对背测试

C.相异成分规范评审、相异性确认、背对背测试

D.相异成分规范评审、相同性确认、背对背测试

答案：C

解析：超纲题。出自系统分析师教材第十九章。

N 版本设计较传统设计多了三个步骤：**相异成分规范评审、相异性确认和背对背测试。**

相异成分规范评审：在这一阶段，工作组与项目管理人员通过问题单形式交换 SRS（软件需求规格说明书）相关疑问，确保每个版本在需求理解上的一致性。

相异性确认：在详细设计后进行，目的是评估不同版本之间的差异，确保每个版本在设计和实现上的独特性。

背对背测试：使用相同的测试数据对 N 个版本的程序进行测试，并比较结果以发现软件故障。这种方法可以有效地发现和屏蔽潜在的错误。

未来信息综合技术

基于内容的推荐是信息过滤技术的延续与发展，不属于该方法优点的是_____。

A.能推荐新的或不是非常流行的项目，没有新项目问题

B.能为具有特殊兴趣爱好的用户进行推荐

C.不需要其它用户的数据，与他人的行为无关

D.能够为新用户产生推荐

答案：D

解析：教材上没有，超纲题。基于内容的推荐系统（Content-based Recommendation System）是一种个性化推荐技术，其核心原理是利用物品的内容特征和用户的历史行为来预测用户可能感兴趣的物品。

基于内容的推荐方法的优点主要包括：

（1）**不依赖其他用户的数据：**基于内容的推荐不依赖于其他用户的评分或行为数据，因此不存在冷启动问题和稀疏问题。

（2）**个性化推荐：**能够为具有特殊兴趣爱好的用户进行推荐，因为推荐是基于用户对特定内容特征的兴趣。

（3）**推荐新项目：**能推荐新的或不是很流行的项目，没有新项目问题。

（4）**透明度高：**由于推荐理由通常是基于用户已知的兴趣点，因此用户更容易理解为什么这些项目被推荐给他们。

综上，可以得出选项 ABC 都是基于内容推荐方法的优点，而选项 D 不是，因为新用户没有历史数据基础，基于内容的推荐方法无法产生推荐。

知识产权与标准化

执行本单位的任务所完成的职务发明创造，包括退休、调离原单位后或者劳动、人事关系终止后_____内作出的，与其在原单位承担的本职工作或者原单位分配的任务有关的发明创造。

A. 3 个月

B. 1 年

C. 6 个月

D. 18 个月

答案：B

解析：《中华人民共和国专利法实施细则》第十三条 专利法第六条所称执行本单位的任务所完成的职务发明创造，是指:

(一)在本职工作中作出的发明创造；

(二)履行本单位交付的本职工作之外的任务所作出的发明创造；

(三)**退休、调离原单位后或者劳动、人事关系终止后 1 年内作出的，与其在原单位承担的本职工作或者原单位分配的任务有关的发明创造**。

专利法第六条所称本单位，包括临时工作单位；专利法第六条所称本单位的物质技术条件，是指本单位的资金、设备、零部件、原材料或者不对外公开的技术信息和资料等。

一项外观设计专利里面相似设计最多有_____个。

A. 10

B. 6

C. 8

D. 5

答案：A

解析：一件外观设计专利申请中的相似外观设计不得超过 10 项。如果超过 10 项，审查员会发出审查意见通知书，申请人修改后未克服缺陷的，会驳回该专利申请。

在外观设计专利申请中，相似设计指的是与基本设计具有相同或相似的设计特征，且二者之间的区别点在于局部细微变化、该类产品的惯常设计、设计单元重复排列或者仅色彩要素的变化等情形。这些相似设计可以作为一件专利申请提出，但数量上有限制。根据《中华人民共和国专利法》第三十一条第二款规定，同一产品两项以上的相似外观设计，或者用于同一类别并且成套出售或者使用的产品的两项以上外观设计，可以作为一件申请提出。同时，《专利审查指南》也指出，一件外观设计专利申请中的相似外观设计不得超过 10 项。

基于对软件工作原理和结构进行研究，需要对软件进行安装、部署、运行，以下说法正确的是_____。

A. 需要告知软件著作人，经作者同意后向其付费

B. 需要告知软件著作人，经作者同意后可以不用付费

C. 不需要告知软件著作人，也不需要支付费用

D. 不需要告知软件著作人，需要支付费用

答案：C

解析：考查知识产权的侵权判定。

侵权判定：

①未经著作权人同意而发表或登记其软件作品。

②将他人软件当作自己的作品发表或登记。

③未经合作者同意将共同开发的软件当作自己的作品发表或登记。

④在他人开发的软件上署名或更改他人署名。

⑤未经著作权人或其合法受让者许可，修改、翻译、复制或部分复制、向公众发行或出租、办理权利许可或转让或通过网络传播其作品。

非侵权判定：

为了学习和研究软件内含的设计思想和原理为由，通过安装、显示、传输或存储软件的方式使用软件时，可以不经许可，不支付报酬。

数学与经济管理

0-1000 的数字里，恰好只有一个 5 的数的个数为_____。

A. 242

B. 243

C. 244

D. 245

答案：B

解析：概率题。

个位数为 5，十位和百位数不为 5 的概率为 1/10*9/10*9/10=81/1000，同理十位为 5，个数和百数不为 5 的概率也是 81/1000，百位数为 5，个位和十位不为 5 的概率也是 81/1000，三者相加为：243/1000，即 1000 个数里有 243 个只有一个数字为 5。

即只有一位数是 5，即在个位、十位或百位上有一个 5，其他位数不能是 5。

• 个位为 5，数字的形式是 _ _ 5，百位和十位除了 5 以外各有[012346789]这 9 个数字选择，共有 9*9=81 个数；

• 十位为 5，数字的形式是 _ 5 _，百位和个位除了 5 以外各有[012346789]这 9 个数字选择，共有 9*9=81 个数；

• 百位为 5，数字的形式是 5 _ _，十位和个位除了 5 以外各有[012346789]这 9 个数字选择，共有 9*9=81 个数；

三者加起来 81+81+81=243，所以选 B。

有 100 个人，其中 45 个人会打篮球，53 个人会打乒乓球，55 个人会踢足球，其中有 28 个人同时会打篮球和乒乓球，有 35 个人同时会打乒乓球和足球，有 32 个人同时会打篮球和足球，有 20 个人三种球都会，那么三种球都不会的有_____人?

A.22

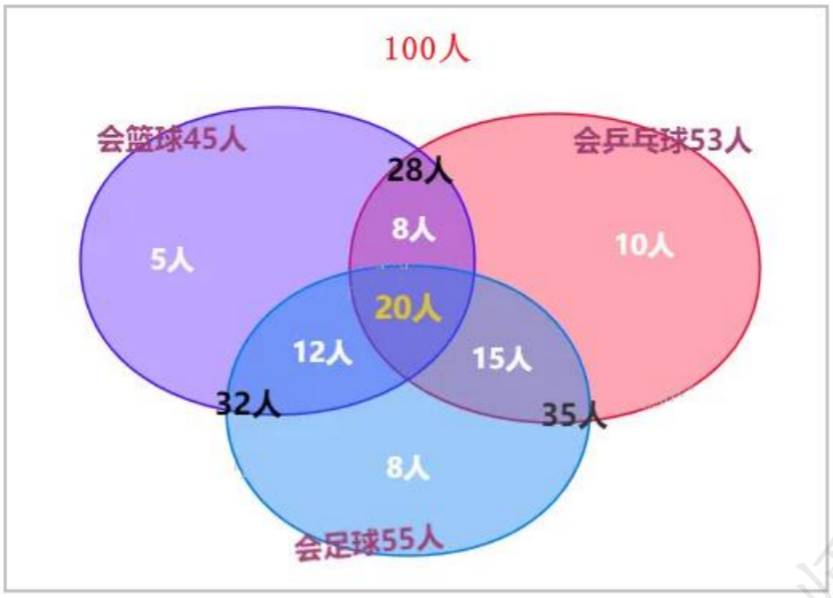
B.23

C.21

D.20

答案：A

解析：解法一：根据题目画 Venn 图如下，由下图可知，三者都不会的有 100 -（20+8+12+15+5+10+8）=22 人。



解法二：3 个集合容斥原理的表示形式为：

|AUBUC|=|A|+|B|+|C|-|A∩B|-|A∩C|-|B∩C|+|A∩B∩C|

即 A ∪ B ∪ C = 45 + 53 + 55 − 28 − 32 − 35 + 20 = 78，100 − 78 = 22。

即，三个集合的并集的元素数量，等于每个集合元素的数量的和，减去每两个集合交集的元素数量，加上三个集合交集的元素数量。

设会打篮球的人数是|A|=45

设会打乒乓球的人数是|B|=53

设会踢足球的人数是|C|=55

同时会打篮球和打乒乓球的人数表示为|A∩B|=28

同时会打篮球和踢足球的人数表示为|A∩C|=32

同时会打乒乓球和踢足球的人数表示为|B∩C|=35

同时会三项运动的人数表示为|A∩B∩C|=20

总共参与调查的人数为 100

会一种球的人数为|A|+|B|+|C|-|A∩B|-|A∩C|-|B∩C|+|A∩B∩C|=45+53+55-28-32-35+20=78

三种球都不会的人数为：100-78=22。

英语

Blackboard architecture, also known as the blackboard system, is a problem-solving approach that utlizes a modular and ___(1)___ framework. It effectively solves complex problems that ___(2)___ a

well-defined algorithm or a pre-determined architecture. Blackboard architecture is inspired by human experts colaborating and solving difficult problems by __(3)__ information and contributing their expertise.The architecture is based on how people work together around a __(4)__ - each person would sit around the board and a problem would be written on it. When a person can solve the problem, they would go to the board and add the a __(5)__ solution they know how to do. This process is repeated until a collective solution is found.

(1)A.conceptualized

B.centerlized

C.decenterlized

D. controlled

(2)A.create

B.improve

C.lack

D. use

(3)A.sharing information

B.solving problem

C.getting information

D.disscussing problem

(4)A.expertise

B.architecture

C.question

D.blackboard

(5)A.possible

B. partial

C. difficult

D.whole

答案：C C A D B

解析：原文出自：https://www.geeksforgeeks.org/blackboard-architecture/

翻译：黑板架构，也称为黑板系统，是一种利用模块化和分散式框架解决问题的方法。它有效地解决了缺乏明确算法或预定架构的复杂问题，黑板架构的灵感来自人类专家通过共享信息和

贡献专业知识来协作和解决难题。该架构基于人们如何围绕黑板一起工作每个人都会坐在黑板周围，并在黑板上写下问题，当一个人可以解决问题时，他们会走到黑板前并添加他们知道如何解决的部分解决方案。这个过程会重复，直到找到一个集体解决方案。

综合知识考点汇总

章节	考分	考点
操作系统	5	死锁预防、进程三态、进程执行时空图、用户态使用特权指令是什么中断、分段存储
数据库设计	5	数据库约束、自然连接结果、关系运算操作对象、求候选键、三级模式
计算机网络	3	传输层协议、路由器属于网络层、如何提高传输速率
软件工程	16	RUP 用例驱动、进度成本压缩计算、白盒测试、测试覆盖范围、四种维护类型判断、设计模式归类、设计模式判断、UML 图描述用户界面、过程内聚、V 模型测试阶段对应、UML 用例参与者关系、基于数据驱动的测试、不同领域元素重用、配置管理、螺旋模型、数据流图判断
信息安全基础	7	机密性、完整性、数据安全治理、数据分级、SQL 注入、安全审计 4 要素、MD5 判断
信息系统基础	4	企业应用集成、数据集成包含什么、信息化需求 3 个层次、专家系统学习机制
系统架构设计	23	架构风格、易用性指标、可用性指标、SOAP 特点、架构 4+1 视图、ATAM 质量属性效用树、效用树的结构、架构风格判断、权衡点、质量属性六要素、ABSD 架构需求三方面、架构复用步骤、进程通信风格连接件、WSDL 三个基本属性、效用树正误判断、调用返回风格、事件驱动风格、ATAM 头脑风暴 3 种场景、基于度量的评估方法、架构演化 6 个步骤、DSSA 建立过程正确性、可扩展性、安全性
软件可靠性	1	N 版本程序设计多的 3 个步骤
未来信息综合	1	信息过滤技术

知识产权	3	著作权期限、外观设计专利、侵权判定
数学与经济	2	排列组合、逻辑推导
专业英语	5	黑板架构

案例分析

案例一 质量属性

关于人工智能系统的需求分析，给出十几个需求。

a.系统发生业务故障时，3 秒内启动 XXX，属于可靠性

b.系统中的数据进行导出，要求在 3 秒内完成，属于可用性

c.质量属性描述，XXX，属于____

d.质量属性描述，XXX，属于可用性

e.质量属性描述，XXX，属于____

f.系统需要增加新的模块，需要 3 天完成，属于____

g.系统分为三个不同的国家语言，完成 XXXX 功能，属于____

h.系统使用了云服务器，要求 1 分钟以内检测错误和故障，并且 1 个小时以内恢复

i.系统使用 XX 检测云服务器故障，需要在 2 秒切换

【问答题】

1.从质量属性的角度分析填空。（14 分）

字母	属性
a	功能性
b	可靠性
c	（1）
d	（2）
e	（3）
f	-
g	-
h	可用性

2.针对质量属性可以使用 ping/echo 和心跳模式实现，分别简述 ping/echo 和心跳模式的实现原理，

张工认为从资源利用率的角度来看采取心跳模式策略比较合适，简述为什么。(11 分)

答案：

实现原理：

ping/echo：通过 ICMP 协议发送一个“Echo Request”消息到目标主机，等待其回复“Echo Reply”。这是一种网络层的连通性测试方法。echo：更广泛地指任何类型的请求-响应模式，用于验证远程服务或组件是否可达和正常工作。这可以是基于 HTTP、TCP 等协议的简单请求。

心跳模式：客户端定期向服务器发送心跳信号（如 HTTP 请求），服务器接收到后立即回应。如果连续几次心跳未得到响应，则认为服务器可能出现故障。心跳信号通常包含少量数据，比如时间戳或简单的健康检查信息，以确保不仅网络连接正常，而且应用程序本身也在正确运行。

采取心跳模式的原因：

针对性强，减少不必要的通信开销；

按需调整频率和内容，优化资源使用，比较灵活；

结合健康检查，提高诊断精度；

支持复杂环境下的高效运作。

案例二 数据库 cache-aside 架构

关于 web 架构的题目，主要是涉及到数据库和缓存。

（Cache-aside 架构，也称为旁路缓存模式，是一种常见的缓存使用策略。

基本概念：在此架构中，缓存和数据库是两个独立的存储系统，它们之间没有直接的交互。应用程序代码负责管理缓存和数据库之间的数据一致性。

读操作流程:

①应用程序首先尝试从缓存中读取数据；

②如果数据存在于缓存中(缓存命中)，则直接将缓存的数据返回给应用程序；

③如果数据不存在于缓存中(缓存未命中)则应用程序从数据库中检索数据，将数据写入缓存，并返回给用户。

写操作流程：

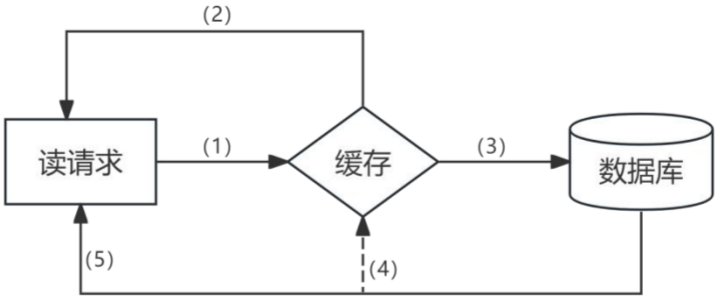
①应用程序更新数据库中的数据；

②应用程序可以选择删除缓存中的旧数据(缓存失效)或更新缓存中的数据；

③为了确保数据一致性，通常推荐在更新数据库后删除或更新缓存中的相关数据。）

【问答题】

1.填空流程图说明缓存读写的过程。（10 分）



答：

（1）向缓存请求读取该商品信息

（2）若命中则返回该商品信息

（3）若未命中则访问数据库查询该商品信息

（4）将查询到的数据库数据更新到缓存

（5）将查询到的数据库目标数据返回

2.填空流程图说明写缓存的过程。（6 分）

答：

（1）更新数据库中的目标商品信息。

（2）将数据库中更新的商品信息写入到缓存中，确保数据一致性。

3.王工使用了多线程技术进行缓存处理，线程 1 负责写入，线程 2 负责读取，可能存在数据一致性问题，请解释其原因，并给出 3 个以上的解决办法。（9 分）

答案：**原因：**

竞态条件：如果没有适当的同步机制，两个或多个线程可能同时访问和修改共享资源，导致最终结果取决于线程调度顺序。例如，线程 1 正在更新缓存中的某个条目，但线程 2 在同一时间点读取该条目，它可能会读取到部分更新或旧版本的数据。

可见性问题：在多线程环境中，一个线程对共享变量所做的更改不一定立刻被其他线程看到。这是因为现代 CPU 架构通常会使用寄存器和缓存来加速性能，这可能导致某些更新在一段时间

内只对本地线程可见。

原子性破坏：写入操作可能不是一个原子操作，意味着它可以被中断或分段执行。如果线程 1 在完成整个写入之前就被切换出去，而线程 2 此时尝试读取，那么它将得到一个不完整的或者错误的数据状态。

解决方法：延时双删、同步删除、加互斥锁（分布式锁）、消息队列、基于缓存更新策略。

案例三 嵌入式

机器人操作系统 ROS。

【问题 1】（13 分）

（1）ROS 定义和特点。

（2）ROS2 与 ROS1 相比哪些地方做了改进？

答案：

（1）ROS（Robot Operating System）是一个用于编写机器人软件的框架。它提供了一系列的工具和库，帮助开发者创建复杂的、可以在多种操作系统上运行的机器人应用程序。

ROS 的主要特点包括：

• 分布式计算能力：ROS 提供了一种方式让多个计算机或设备协同工作，通过网络进行通信

• 消息传递：ROS 使用了一种进程间通信(IPC)机制，可以在不同的节点之间传递数据。

• 包管理：ROS 提供了一个包管理系统，允许用户分享和安装软件组件。

• 工具和库：ROS 提供了一系列工具和库，例如用于 3D 渲染的 RViz，用于模拟机器人的 Gazebo，以及用于路径规划的 navigation stack 等。

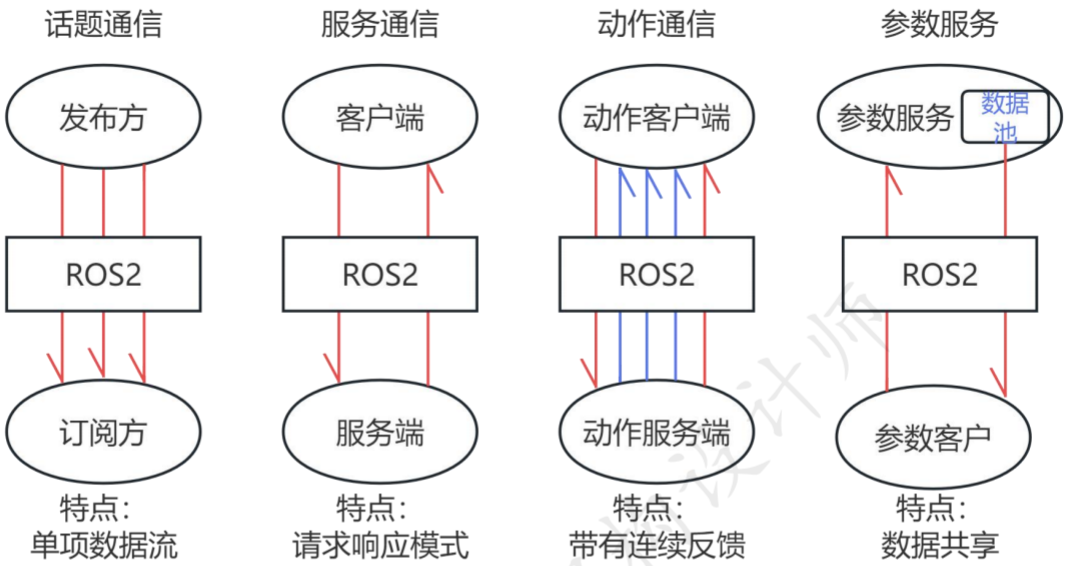
（2）ROS2 与 ROS1 的对比，如下表所示：

ROS1	ROS2
所有节点需 Master 节点管理	所有节点平行分布，基于 DDS 的 Discover
rosbuid、catkin 管理项目	ament、colcon 管理项目
使用 C++03 和 Python2	使用 C++11 和 Python3
CMake 构建系统	支持其他构件系统的选项
基于 2009 设计的 API	全新的用户 API
TCPROS/UDPROS 通信机制	DDS 数据分发服务

Linux 系统平台	Linux/Windows/Mac/RTOS 系统平台

【问题 2】（8 分）

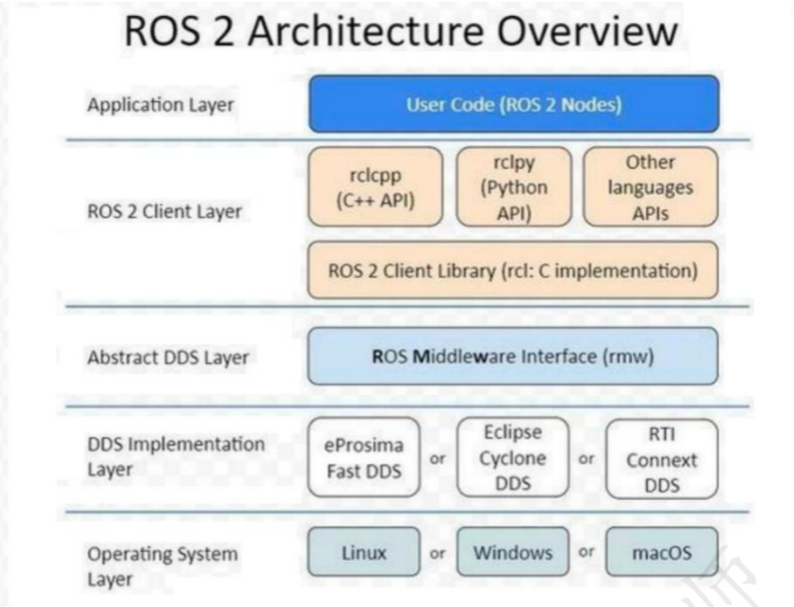
3.四种通信服务，选词填空。



答案：1.话题通信：是一种单向通信模型，在通信双方中，发布方发布数据，订阅方订阅数据，数据流单向的由发布方传输到订阅方,。2.服务通信：是一种基于请求响应的通信模型，在通信双方中，客户端发送请求数据到服务端，服务端响应结果给客户端 3.动作通信：是一种带有连续反馈的通信模型，在通信双方中，客户端发送请求数据到服务端，服务端响应结果给客户端，但是在服务端接收到请求到产生最终响应的过程中，会发送中间连续的反馈(进度)信息到客户端。4.参数服务：是一种基于共享的通信模型，在通信双方中，服务端可以设置数据，而客户端可以连接服务端并操作服务端数据。

【问题 3】（7 分）

根据下图，请解释 ROS2 架构每一层含义。



案例四 web 应用

基于 Elasticsearch 分词的商品推荐系统（微信小程序接入）。

1. 基于 Elasticsearch 分词，解释下 Standard, Simple, Whitespace, Keyword 几种分词器的分词原理。(6 分)

答案：Standard：standard 分词器通常用于处理多种语言的文本，它会识别并拆分单词、数字、电子邮件地址、网址等，并且能够处理一些标点符号。对于中文、日文、韩文等不使用空格的语言，Standard 分词器可能不是最佳选择，因为它不能很好地理解这些语言的语法规则。例如：“Hello, World!” 分词结果为[“Hello”, “,”, “World”, “!”]。

Simple：simple 分词器非常基础，它只是简单地按照非字母字符进行分割，即将所有的非字母字符视为分隔符。因此，所有连续的字母序列都会被视为一个词汇单元，而任何非字母字符（如空格、标点符号、数字）都会被丢弃或用作分隔标志。例如：“Hello, World! How are you?”分词结果为[“Hello,”, “World!”, “How”, “are”, “you?”]

Whitespace：whitespace 分词器仅仅根据空白字符（如空格、制表符、换行符）来分割文本，不会去除任何字符，也不会考虑标点符号。这意味着标点符号会被当作独立的词汇单元处理。例如：“Hello, World! How are you?”分词结果为[“Hello,”, " ", “World!”, " ", “How”, " ", “are”, " ", “you?”]

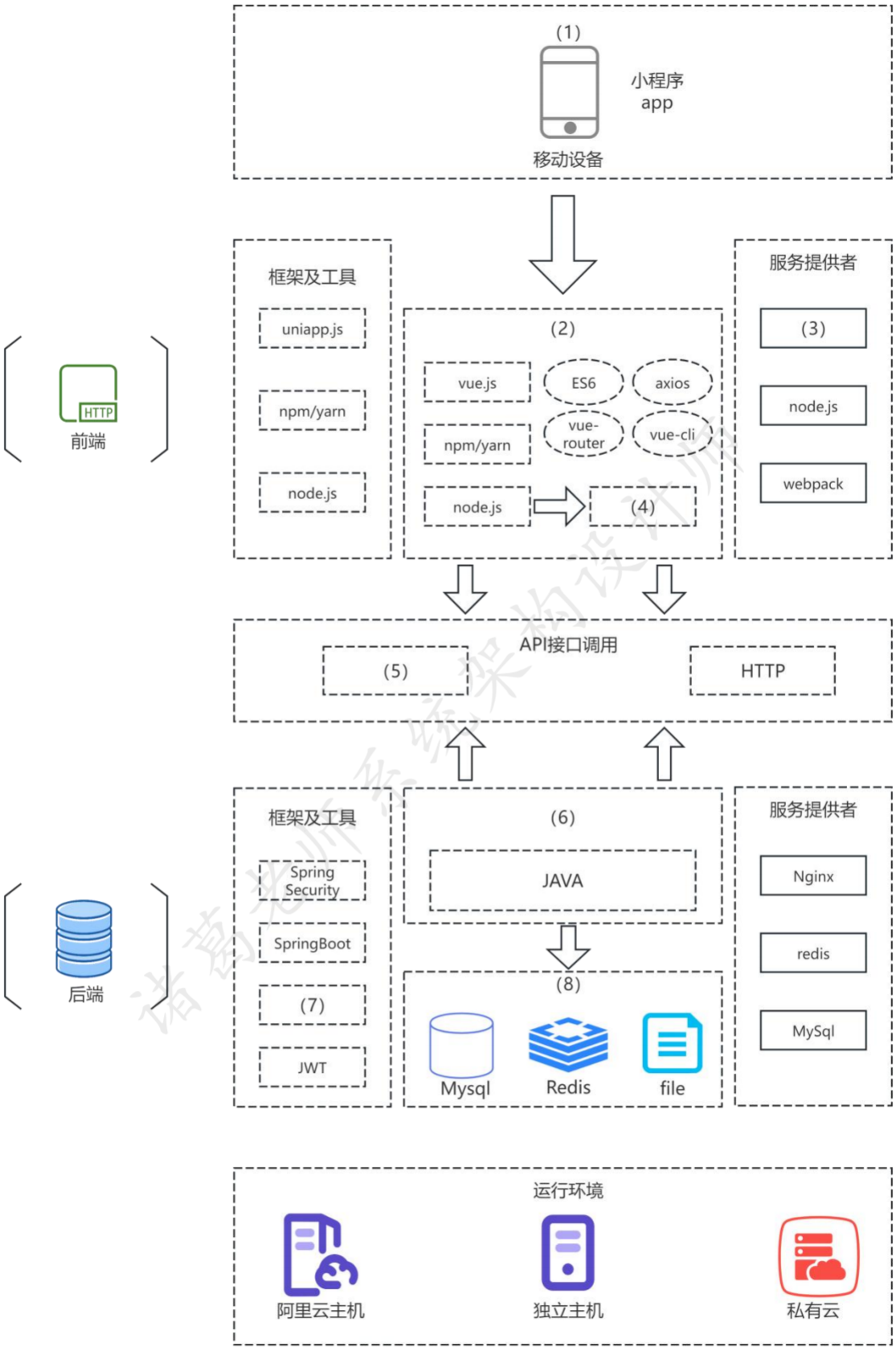
Keyword：keyword 分词器并不真正执行分词操作。相反，它会将整个输入文本作为一个单

独的词汇单元输出。这意味着输入文本中的所有内容都被认为是一个不可分割的整体。例如：“Hello, World!” 分词结果为 [“Hello, World!”]。

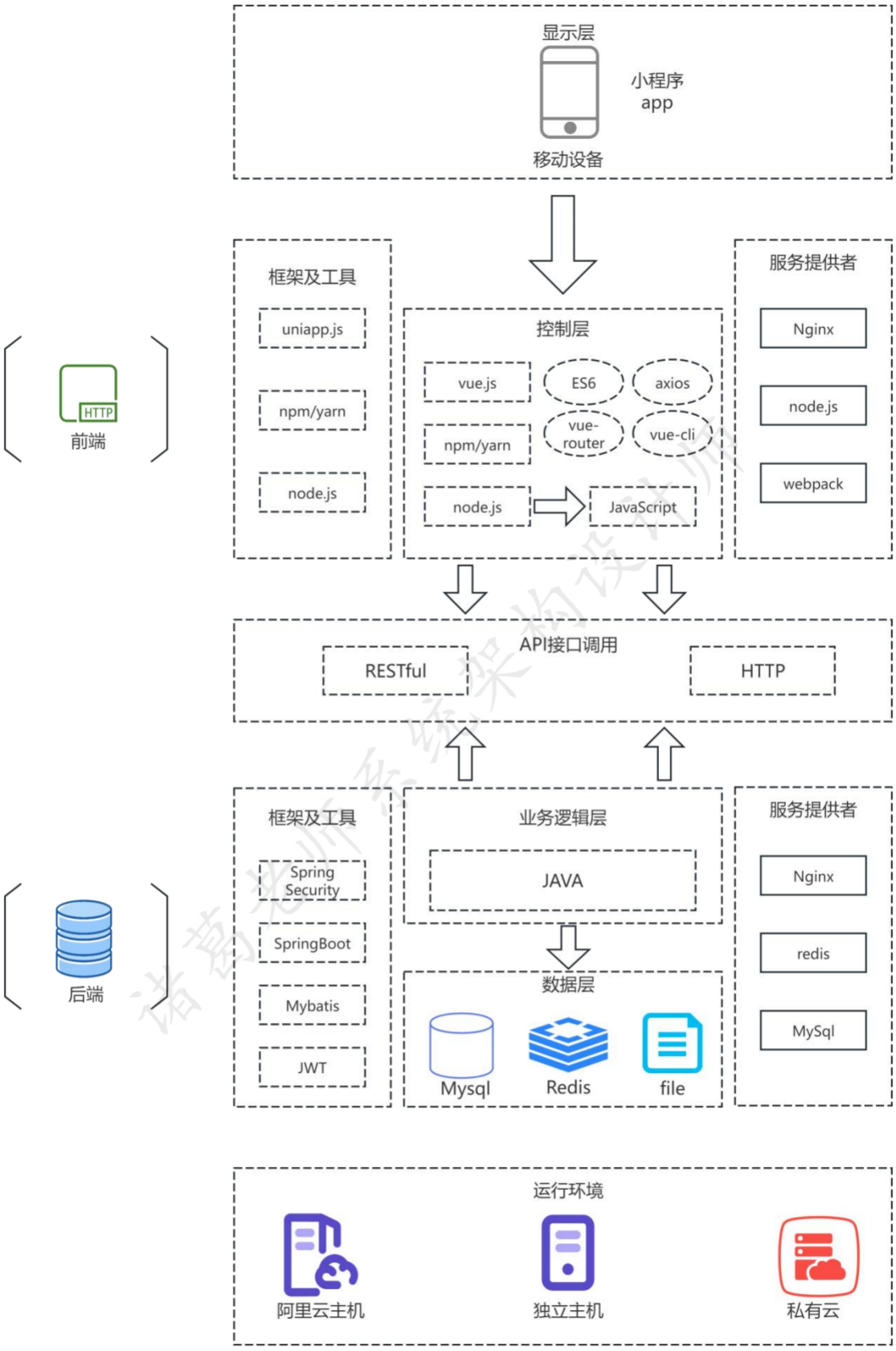
2. 系统架构图填空，从给出的选项中选出对应的选项填入对的位置，8 个空。（12 分）

分层：接入层、显示层、网络层、应用层、业务逻辑层、控制层、数据层。

技术：Mybatis、Nginx、Flink、Javascript、Node.js、RESTful、Elasticsearch、Kafka


(1)
小程序
app
移动设备
框架及工具
uniapp.js
npm/yarn
node.js
(2)
vue.js
ES6
axios
npm/yarn
vue-router
vue-cli
node.js
(4)
服务提供者
(3)
node.js
webpack
前端
HTTP
API接口调用
(5)
HTTP
框架及工具
Spring Security
SpringBoot
(7)
JWT
(6)
JAVA
(8)
Mysql
Redis
file
服务提供者
Nginx
redis
MySql
后端
运行环境
阿里云主机
独立主机
私有云

答：



3. RESTful 架构有什么特点，是如何实现前后端分离的。（7 分）

答案：

无状态性：每个请求从客户端到服务器都必须包含理解该请求所需的所有信息。服务器不会存储任何客户端的状态或会话数据，这使得前端和后端可以独立扩展，并且任意一个都可以在不影响另一个的情况下进行更新。

统一接口：Restful API 提供了一套标准的操作集（GET、POST、PUT、DELETE 等），这些操作与资源（通常是名词形式的数据实体，如用户、订单等）相关联。这种一致性简化了前端与后端之间的交互，使前端开发者可以更容易地理解和使用 API。

资源导向：在 RESTful 设计中，所有的操作都是围绕着资源展开的。资源通过 URL 标识，并通过 HTTP 方法来表示对资源的不同操作。这种方式让前端能够以直观的方式与后端资源进行交互。

分层系统：RESTful 架构支持分层系统，这意味着客户端不必直接连接到最终的服务端；相反，它可以经过多个中间层（如负载均衡器、缓存代理等）。这增加了系统的灵活性和安全性，同时也便于前后端各自独立演化。

基于 HTTP 协议：RESTful 服务通常基于 HTTP 协议构建，利用 HTTP 的方法和状态码来传递语义信息。这使得任何能够发起 HTTP 请求的技术栈都可以作为前端与 RESTful 后端通信，从而促进了技术选型的多样性。

JSON 或 XML 数据格式：RESTful API 常常使用 JSON 或 XML 格式交换数据。这两种格式都是轻量级的、易于解析的文本格式，非常适合跨平台的数据传输。前端可以通过 JavaScript 轻松处理 JSON 对象，而后端可以用多种语言和技术生成和解析它们。

前后端分离部署：前端代码（HTML、CSS、JavaScript）可以托管在一个静态文件服务器上，甚至可以直接嵌入到单页应用（SPA）中，而不需要依赖于后端逻辑。后端则专注于提供业务逻辑和服务，两者之间仅通过 API 进行通信。

案例五 软件设计

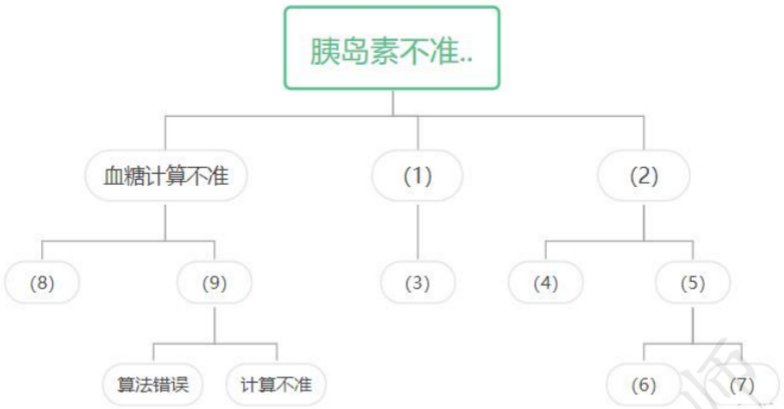
医用血糖监测系统：题目大体是说有一个打胰岛素泵的系统，首先通过血糖仪测量的数据，然后自动计算出要使用的胰岛素剂量，再通过这个泵系统自动去打胰岛素给病人。

1.危险驱动的安全分析的 4 个步骤，并简要说明。（10 分）

确定系统、识别危险、评估风险、确定风险控制措施、监控和复查。

2.胰岛素系统，填空题（9 分）

好像有以下选项，但是不全，填空的地方有些直接用文字表达了①血糖传感器错误 ②传感系统异常 ③血糖计算不准 ④传感器错误 ⑤泵信号失效 ⑥错误时间推送预定的量 ⑦定时器失效 ⑧胰岛素计算错误。



答：

（1）②传感系统异常

（2）⑥错误时间推送预定的量

（3）⑤泵信号失效

（4）⑦定时器失效

（5）⑧胰岛素计算错误

（6）算法错误

（7）计算错误

（8）①血糖传感器错误

（9）③血糖计算不准

3.形式化开发和软件测试技术的特点。（6 分）

答案：

形式化开发特点：

数学基础：形式化开发方法基于数学原理，使用形式化语言来描述系统需求和设计，使得开发过程更加严谨和精确。

严格性：形式化方法要求对系统行为的每一个细节都进行明确的定义，减少歧义性。

可验证性：由于使用了数学化的描述，形式化开发的结果可以进行逻辑上的证明，确保系统满足预定的规格说明。

抽象层次：形式化方法通常涉及不同层次的抽象，从高级的需求规范到低级的实现细节。

自动化支持：形式化方法可以更容易地与自动化工具结合，例如模型检查器和定理证明器。

高成本：形式化开发通常需要高度专业化的知识和技能，因此成本较高，且不适合所有项目。

软件测试技术特点：

验证和确认：软件测试旨在验证软件是否满足规定的需求，并确认软件是否按照预期工作。

多样性：测试技术多样，包括单元测试、集成测试、系统测试、验收测试等，每种测试针对不同层次的问题。

迭代过程：软件测试是一个迭代的过程，随着软件的开发和需求的变更，测试也需要不断进行和调整。

人为因素：测试很大程度上依赖于测试人员的经验、知识和创造力，因此具有一定的主观性。

成本效益：测试可以发现缺陷并指导修复，从而减少软件发布后的问题，具有较好的成本效益。

无法完全保证质量：由于测试的局限性，如测试用例的覆盖率问题，无法保证测试能够发现所有的缺陷。

测试工具：现代测试技术往往依赖于自动化测试工具，这些工具可以提高测试效率和准确性。

持续集成：在敏捷开发实践中，软件测试通常与持续集成相结合，确保代码的持续质量和稳定性。

总结：形式化开发强调的是开发过程的严谨性和可证明性，而软件测试技术则更侧重于通过实际的运行来发现和修正软件中的缺陷。两者可以互补，共同提高软件的质量。

论文写作

试题一 论软件维护及其应用

请围绕“论软件维护及其应用”论题，依次从以下三个方面进行论述。

1.概要叙述你参与分析设计的软件项目以及你在其中所承担的主要工作。

2.请介绍软件维护的内容有哪些，以及常见提高可维护性的技术或方法。

3.在软件维护中，你遇到什么问题，你是用什么技术手段处理，以及处理后的效果如何。

答案：

论文素材参考：

软件维护的类型：

(1)**改正性维护**：改正性维护旨在修复软件中在开发阶段遗留的错误。这些错误可能是由于需

求分析不全面、设计缺陷、编码错误等原因导致的。例如，在一个金融交易系统中，如果在计算利息时出现错误，导致用户账户金额不准确，这就需要进行改正性维护。这种类型的维护通常是在软件运行过程中出现故障或异常时触发，需要开发人员根据错误信息和相关的代码逻辑进行问题定位和修复。

(2)**适应性维护**：随着软件运行环境的变化，适应性维护变得必不可少。环境变化包括硬件设备的更新、操作系统的升级、数据库管理系统的变更等。例如，当企业将服务器硬件升级后，作系统从 Windows Server2016 升级到 Windows Sever2019，运行在其上的业务软件可能会出现兼容性问题，如某些功能无法正常使用或性能下降。此时，就需要对软件进行适应性维护，调整软件与新环境的适配关系，确保软件能够正常运行。

(3)**完善性维护**：为了满足用户不断增长的需求和提高软件的性能，完善性维护发挥着重要作用。它主要涉及对软件功能的扩充、性能的优化以及用户体验的提升。比如，在一款电商购物软件中，根据用户反馈和市场趋势，增加商品推荐功能、优化搜索算法以提高搜索结果的准确性和相关性、改进界面设计以提高用户操作的便捷性等都属于完善性维护的范法。这种维护可以增强软件的竞争力，提高用户满意度，

(4)**预防性维护**：预防性维护是一种前瞻性的维护策略，它是在软件还未出现明显问题时，为了预防潜在的故障和提高软件的可维护性而进行的操作。例如，对软件代码进行定期审查，发现并修复可能存在的逻辑漏洞、代码异味(如复杂度过高、耦合度过大等问题)。通过对软件架构的优化，如将紧密耦合的模块进行解耦，增加软件的可扩展性和灵活性，以便更好地应对未来可能的变化。

软件维护的方法：

(1)**软件配需管理**：软件配置管理是软件维护的基础。它通过对软件项目中的各种文档(如雲求文档、设计文档、测试文档等)、代码数据等进行标识、版本控制、变更管理和配置审计等操作，确保软件在维护过程中的完整性和一致性。例如，使用 Git 等版本控制工具开发人员可以方便地记录软件每次的变更内容、时间和人员，方便回溯和管理不同版本的软件。在进行维护操作时，可以基于版本控制系统创建分支，在不影响主版本的情况下进行修改和测试，完成后再合并回主版本。

(2)**软件再工程**：软件再工程包括对现有软件进行读向工程、重构和正向工程等步骢，逆向工程是从已有的软件代码和文档中提取系统设计和需求信息，帮助维护人员理解软件的结构和功能。重构则是在不改变软件外部行为的前提下，对软件的内部结构进行改进，提高代码的可读性、可维护性和可扩展性。例如，将一个大型的、复杂的函数拆分成多个功能单一、结构清晰的小函数。

正向工程是基于重构后的软件结构，重新生成新的软件系统，可能会使用更先进的技术和设计模式，进一步提升软件质量。

(3)**软件测试**：在软件维护过程中，测试是保证软件质量的关键环节。包括回归测试、功能测试、性能测试等多种类型。回归测试用于检查软件在修改后是否引入新的问题，确保原有功能不受影响。例如，当对一个软件模块进行功能扩充后，需要运行之前的测试用例来验证没有破坏原有的功能罗辑。功能测试则是针对新添加或修改的功能进行的测试，确保其满足预期的功能需求。性能测试用于评估软件在维护后的性能指标，如响应时间、吞叶是等是否满足要求，特别是在对软件进行性能优化的维护操作后，需要通过性能测试来验证优化效果。

(4)**用户反馈收集与分析**：用户是软件的直接使用者，他们在使用过程中遇到的问题和提出的建议是软件维护的重要依据。通过建立多样化的用户反馈渠道，如在线客服系统、用户论坛、问卷调查等，收集用户反馈信息。对这些反馈进行深入分析，可以发现软件中存在的问题和需要改进的方向。例如，如果大量用户反馈某个功能操作复杂或容易出错，维护人员就需要对该功能进行针对性的检查和优化。

软件维护应用案例分析：

(1)**案例背景：某企业资源规划(ERP)软件维护**

某制造企业使用的 ERP 软件已经运行了数年，在使用过程中出现了一些问题，同时企业业务发展也对软件提出了新的需求。

(2)**维护过程中的方法应用**

改正性维护：企业财务人员反映在生成财务报表时，有时会出现数据不一致的问题。维护团队通过对软件的日志文件分析和代码调试，发现是在财务数据汇总模块中存在一个编码错误，导致部分数据重复计算。通过修改代码中的错误逻辑，并经过严格的单元测试和回归测试，成功解决了数据不一致的问题，保证了财务报表的准确性。

适应性维护：企业决定将数据库从 Oracle 11g 升级到 Oracle 19c，同时对服务器硬件讲行了升级。ERP 软件在新的数据库环境下出现了一些兼容性问题，如某些查询功能运行缓慢。维护团队通过分析数据库的性能指标和软件与数据库的交互代码，对相关的 SQL 查询语句进行优化，调整了软件与数据库的连接参数，并对部分依赖于旧数据库特性的代码进行了修改。同时，针对新的硬件环境，对软件的配置参数进行了调整，以充分利用硬件资源，提高软件的性能，确保 ERP 软件在新环境下能够稳定、高效地运行。

完善性维护：根据企业生产部门和销售部门的反馈，希望在 ERP 软件中增加生产计划的可视化功能和销售数据分析功能。维护团队首先进行了详细的需求分析，然后对软件架构进行了适

当调整，在不影响原有功能的基础上，添加了新的模块。在开发过程中，采用了软件再工程的方法，对部分相关代码进行了重构，提高代码的可读性和可维护性。新功能开发完成后，通过功能测试、性能测试和用户试用，确保新功能满足业务需求且不会对原有功能产生负面影响。

预防性维护：为了提高 ERP 软件的可维护性和应对未来可能的变化，维护团队定期对软件代码进行审查。利用代码分析工具发现了一些代码存在的潜在问题，如部分模块之间的耦合度过高、代码的重复率较高等。针对这些问题，对软件进行了重构，将高耦合的模块进行解耦，提取公共代码形成独立的函数或类，降低了代码的复杂度。同时，对软件的文档进行了更新和完善，包括对软件架构、模块功能、接口设计等方面的详细描述，方便后续维护人员理解和操作。

(3)效果分析

通过上述软件维护工作，该 ERP 软件在企业中的运行状况得到了显著改善。软件的稳定性和可靠性提高，减少了因软件问题导致的业务中断风险。新功能的添加满足了企业业务发展的需求，提高了企业的运营效率和管理水平。软件的可维护性增强，降低了后续维护的难度和成本，为软件的长期使用和持续改进奠定了良好的基础。

软件维护面临的挑战与应对策略：

(1)挑战

软件复杂性增加：随着软件系统功能的不断扩充和技术的发展，软件的复杂性呈指数级增长。大型软件系统可能包含大量的模块、复杂的业务逻辑和众多的技术组件，这使得维护人员很难全面掌握软件的结构和功能，增加了维护的难度。例如，一些复杂的企业级软件系统可能涉及多种不同的技术领域，如数据库技术、网络技术、人工智能技术等，维护人员需要具备广泛的知识和技能。

技术更新换代快：信息技术领域的发展日新月异，新的技术、框架和工具不断涌现。软件维护人员需要不断学习和适应这些新技术，以便在维护过程中能够运用最新的技术手段解决问题。例如，当软件需要从传统的架构向 微服务架构转型时，维护人员需要掌握微服务相关的技术，如容器化技术(Docker、Kuberetesa 等)、服务治理技术等，这对维护人员的学习能力和知识更新速度提出了很高的要求。

维护成本控制：软件维护需要投入大量的人力、物力和财力资源，包括维护人员的薪酬、硬件设备的更新、软件工具的采购等。随着软件规模的扩大和维护周期的延长，维护成本可能会不断增加，企业需要在保证软件质量的同时，合理控制维护成本，避免维护成本过高影响企业的经济效益。

(2)应对策略

加强知识管理与人员培训：建立企业内部的软件知识管理体系，对软件的开发文档、维护记录、技术资料等进行有效的管理和共享，方便维护人员学习和查询。同时，定期为维护人员提供技术培训，包括新技术、新方法的培训以及针对特定软件系统的业务培训，提高维护人员的专业素养和业务能力。鼓励维护人员之间的交流和经验分享，形成良好的学习氛围。

采用先进的维护工具和技术：利用现代化的软件维护工具，如自动化测试工具(Selenium、JUnit 等)、代码分析工具(SonarQube 等)、配置管理工具(Git、SVN 等)，提高维护工作的效率和质量。在技术更新时，采用渐讲式的汗移策略，例如在将传统软件向微服务架构转型时，可以先将部分功能模块进行微服务化改造，逐步积累经验，降低技术更新的风险。同时，关注行业内的最佳实践案例，积极引进和应用先进的维护技术和方法。

成本效益分析与优化：在进行软件维护决策时，进行详细的成本效益分析。根据软件对企业业务的重要性、维护成本、潜在收益等因素制定合理的维护计划。对于一些对业务影响较小且维护成本高昂的功能，可以考虑进行替代或优化。例如，如果某个软件模块的维护成本过高且其功能可以通过其他方式实现，可以评估是否停用该模块或采用第三方软件替代。同时，优化维护流程，减少不必要的工作环节提高维护资源的利用效率。

试题二 论面向服务的架构设计

请围绕“论面向服务的架构设计”，依次从以下三个方面进行论述。

1.概要叙述你参与分析设计的软件项目以及你在其中所承担的主要工作。

2.论面向服务的架构设计基于 WebService 的面向服务架构实现过程，SOA 具有哪些特征，支撑软件功能重用。

3.具体阐述你参与的软件项目是如何以面向服务的架构为指导实施的，在实施过程中遇到哪些问题，是如何解决的。

答案：

解题思路：

我参与分析和开发的项目是一个大型的电商平台，该平台拥有数亿用户和海量数据。为了应对业务快速发展和需求变更，我们采用了面向服务架构(SOA)来设计和开发系统。我主要负责系统架构只设计和技术方案制定。

SOA 的主要技术和标准包括：Web 服务：一种基于 XML 的标准，用于在互联网上进行数据交换。WSDL：Web 服务描述语言，用于描述 Web 服务的功能和接口。SOAP：简单对象访问协议，用于在 Web 服务之间进行消息传递。UDDI：通用描述、发现和集成，用于注册和查找

Web 服务。ESB：企业服务总线，用于提供服务路由、消息转换等功能。

我们在构建 SOA 架构时遇到了以下问题:服务粒度划分:如何划分服务粒度是 SOA 架构设计中的关键问题，服务粒度划分过大，会导致服务过于复杂，难以维护;服务粒度划分过小，会导致服务数量过多，增加系统复杂度。服务接口设计:服务接口设计需要考虑服务的易用性、可扩展性等因素。服务安全性：SOA 架构中，服务之间通过网络进行通信，因此需要考虑服务安全问题。服务治理：SOA 架构中，需要对服务进行有效地管理，包括服务注册、发现、调用、监控等。

通过采用 SOA 架构，我们有效地提高了系统的灵活性、可扩展性和可维护性。具体实施效果如下:提高了系统的灵活性：SOA 架构使得我们可以快速地添加、修改和删除服务，以满足业务需求的变化。提高了系统的可扩展性：SOA 架构使得我们可以很容易地扩展系统以满足业务发展需求。提高了系统的可维护性：SOA 架构使得我们可以更容易地维护系统，降低了维护成本。

面向服务架构是一种有效的软件架构设计方法，可以有效地提高系统的灵活性、可扩展性和可维护性。在实际应用中，需要根据具体情况选择合适的 SOA 技术和标准，并解决好 SOA 架构设计和实施过程中遇到的问题。在未来的工作中，我们将继续研究和实践面向服务架构，不断提高 SOA 架构设计和实施水平。

论文素材参考：

面向服务的架构(SOA)概念和特征

概念：面向服务的架构是一种分布式计算架构，它将企业的业务功能抽象为一系列相互独立、可复用的服务。这些服务通过明确定义的接口进行通信，并可以在不同的应用程序和系统中被调用。SOA 的核心思想是将业务逻辑从具体的技术实现中分离出来，以服务的形式对外提供，实现业务功能的灵活组合和复用。

特点：

松耦合性：服务之间通过接口进行交互，彼此之间的依赖关系较弱。这种松耦合的特性使得服务可以独立开发、部署和更新，不会因为某个服务的变化而对其他服务产生重大影响。例如，在一个电商企业中，订单服务和库存服务是松耦合的，当订单服务升级以适应新的促销策略时，库存服务可以保持不变。

可复用性：服务是独立的业务功能单元，可以在多个不同的业务流程和应用场景中被重复使用。比如，用户认证服务可以在企业的多个系统(如内部办公系统、客户服务系统等)中使用，减少了重复开发的工作量，提高了开发效率。

互操作性：SOA 支持不同平台、不同编程语言和不同技术架构的系统之间的交互。通过使用标准的通信协议(如 HTTP、SOAP 等)和数据格式(如 XML、JSON 等)，服务可以被各种类型的

客户端调用，促进了企业内部和企业间的系统集成。

灵活性和适应性：企业业务经常面临变化，SOA 架构可以轻松地对业务流程进行重新组合和调整。通过增加、修改或删除服务，可以快速响应市场变化和业务需求的调整。例如，当企业推出新的业务模式时，可以通过组合现有的服务或开发新的服务来满足新的业务流程。

面向服务的架构关键技术

(1)**服务描述语言：**常用的服务描述语言有 WSDL (Web Services Description Language)等。WSDL 用于描述服务的功能、接口、输入输出参数以及调用方式等信息，使服务使用者能够清楚地了解服务的内容和使用方法。它是实现服务互操作性的重要基础，为服务的发布和调用提供了标准化的描述。

(2)**服务注册与发现：**服务注册中心是 SOA 的关键组件之一，例如 UDDI(Universal Description,Discovery,and Integration)。服务提供者将服务的描述信息注册到服务注册中心，服务使用者可以在注册中心查找所需的服务。这种机制方便了服务的管理和查找，提高了服务的利用率。

(3)**消息传递机制：**在 SOA 中，服务之间的诵信通常通过消息传诺来实现、可以使用 SOAP(Simple ObiectAccess Protocl)或 RES(Representational State Transfer)等方式。SOAP 是一种基于 XML 的协议，它提供了一种标准化的方法在不同的系统之间交换信息，REST 则是一种轻量级的架构风格，利用 HTTP 协议的方法(如 GET、POST、PUT、DELETE)来实现资源的操作和信息传递，更适合于简单的 Web 应用场景。

面向服务的架构设计原则

(1)**业务驱动原则：**SOA 设计应以企业业务需求为出发点，将业务流程分解为一系列可管理的服务。首先要深入理解企业的业务模型和业务流程，识别出核心业务功能和业务规则，然后将这些业务功能抽象为服务。例如，在金融企业中，贷款审批流程可以分解为客户信息查询、信用评估、风险分析等服务。

(2)**服务粒度适中原则：**服务粒度的选择至关重要。如果服务粒度过细，会导致系统过于复杂，增加服务管理和调用的成本;如果粒度过粗，则会降低服务的复用性。需要根据业务功能的性质和使用频率来确定合适的服务粒度。例如，在一个物流企业中，货物跟踪服务可以是一个相对独立且合适粒度的服务，而不是将货物的所有信息查询和操作都合并在一个大的服务中。

(3)**分层架构原则：**采用分层的架构设计可以提高系统的可维护性和可扩展性。一般可以分为表现层、业务逻辑层和数据访问层等。表现层负责与用户的交豆，业务逻辑层包含了各种业务服务，数据访问层负责与底层数据库的交互。每个层次都有明确的职责，通过接口进行通信。例如，

在一个企业资源规划(ERP)系统中，用户在表现层提交订单，业务逻辑层的订单服务处理订单逻辑，数据访问层负责将订单数据存储到数据库中。

面向服务的架构设计案例分析：

(1)案例背景

某大型制造企业拥有多个生产基地和销售部门，其原有的信息系统是基于传统的单体架构构建的，随着业务的拓展和企业规模的扩大，系统面临着升级困难、功能扩展不便、与外部合作伙伴系统集成复杂等问题。因此，企业决定采用面向服务的架构对信息系统进行重构。

(2)SOA 设计过程

业务分析与服务识别：通过与企业各部门的深入沟通和对业务流程的详细梳理，识别出了一系列核心服务，包括生产计划服务、原材料采购服务、订单处理服务、产品库存服务、客户管理服务等。例如，生产计划服务根据销售订单和库存情况制定生产计划，涉及到多个部门的业务数据和流程。

服务设计与接口定义：针对每个服务，设计其功能和接口。以订单处理服务为例，其接口定义了订单创建、订单查询、订单修改和订单删除等操作，使用 XML 格式来描述订单数据结构，并采用 REST 风格的接口实现服务的调用。同时，考虑到服务的复用性和互操作性遵循了行业标准和企业内部的规范。

服务实现与部署：根据设计好的服务和接口，采用合适的技术实现服务。例如，订单处理服务可以使用 Java 语言和 Soring 框架来实现将服务部署在企业的服务器集群上。在部署过程中，要考虑服务的性能、可靠性和安全性等因素，配置相应的服务器资源和安全策略。

服务注册与调用：建立企业级的服务注册中心，使用开源的 UDDI 实现感自定义的注册机制，服务提供者将服务信息注册到注册中心后其他应用程序和系统可以在注册中心查找并调用所需的服务。例如，销售部门的订单管理系统可以通过注册中心查找并调用订单处理服务来处理客户订单，

(3)效果分析

通过实施面向服务的架构设计，该制造企业的信息系统得到了显著的改善。首先，系统的灵活性大大提高，当企业推出新的产品或调整业务流程时，可以快速地通过组合或修改现有服务来适应变化。其次，服务的复用性减少了重复开发的工作量，提高了开发效率。例如，原材料采购服务可以在多个不同的生产项目中使用。此外，与外部合作伙伴系统的集成变得更加容易，通过标准的接口和协议，可以与供应商、物流企业等合作伙伴的系统进行无缝对接，提升了企业的整体运营效率。

SOA 设计面临的挑战与应对策略：

(1)**挑战**

服务治理问题：随着服务数量的增加，服务的管理和协调变得复杂。包括服务的版本控制、服务的依赖关系管理、服务的性能监控等如果服务治理不当，可能会导致服务之间的冲突、系统性能下降等问题

安全问题：SOA 环境下，服务的分布式和开放性特点增加了安全风险。服务之间的通信可能会受到攻击，企业的核心数据可能会因服务的漏洞而泄露。需要考虑服务的身份认证、授权、数据加密等安全措施。

性能问题：服务之间的通信开销、服务的响应时间等可能会影响整个系统的性能。特别是在高并发的业务场景下，如何保证服务的高效运行是一个挑战。

(2)**应对策略**

建立完善的服务治理框架：通过建立服务治理框架，对服务进行统一管理。包括使用版本管理工具来管理服务的版本，通过配置管理数据库(CMDB)来记录服务的依赖关系，利用性能监控工具实时监测服务的性能指标，并根据监控结果及时调整服务。

加强安全设计：在 SOA 架构设计的各个环节融入安全设计。采用安全的通信协议(如 HTTPS)，实施服务的身份认证机制(如 OAuthJWT 等)，对服务的访问进行严格的授权控制，对敏感数据进行加密处理。同时，定期进行安全审计和漏洞扫描，确保系统的安全。

性能优化措施：从服务的设计、实现和部署等方面进行性能优化。在设计阶段，合理规划服务粒度和服务接口，减少不必要的通信开销；在实现阶段，优化服务代码，提高代码质是和执行效率:在部署阶段，根据服务的负载情况合理分配服务器资源，采用缓存技术、负载均衡技术等提高服务的响应速度。

试题三 论多源异构数据集成方法

请围绕“论多源异构数据集成方法”论题，依次从以下三个方面进行论述。

1.概要叙述你参与分析设计的软件项目以及你在其中所承担的主要工作。

2.多源异构数据集成的主要内容，以及实现异构数据源集成的技术路线。

3.具体阐述你参与的软件项目是如何做到多源异构数据集成，过程中遇到哪些问题，是如何解决的，以及处理后的效果如何。

答案：

论文素材参考：

在当今数字化时代，企业和组织内部的数据来源日益多样化，包括关系数据库、文件系统、XML 文档、Web 服务等多种形式，这些数据在结构、语义和存储方式上存在显著差异，即多源异构数据。有效地集成这些数据对于企业的数据分析、决策支持和业务流程优化至关重要。然而，多源异构数据集成面临着诸多复杂的问题，需要合适的方法来解决。

多源异构数据集成的重要性与挑战：

(1)**重要性**

支持全面的决策分析：企业的决策需要综合考虑来自各个业务部门和不同系统的数据。例如，在企业资源规划 ERP)中，需要整合财务数据、生产数据、销售数据等多源异构数据，才能准确分析企业的运营状况，制定合理的战略决策，如优化生产计划、调整销售策略等。

提升业务流程效率：通过集成多源异构数据，可以打破不同业务系统之间的信息孤岛，实现业务流程的自动化和优化。比如在供应链管理中，集成供应商数据、物流数据和库存数据，可以使采购、运输和存储等环节更加协同，减少库存积压和缺货现象，提高供应链的整体效率。

挖掘数据潜在价值：不同来源的数据可能蕴含着互补的信息。将客户关系管理(CRM)系统中的客户行为数据与企业的产品数据集成，可以通过数据挖掘技术发现客户需求与产品特征之间的关联，为产品创新和个性化营销提供依据，从而挖掘出数据的潜在价值。

(2)**挑战**

数据结构差异：多源异构数据在结构上存在巨大差异，包括关系型数据的表格结构、XML 的层次结构、JSON 的键值对结构以及文本文件的无结构形式等。例如，将结构化的数据库数据与半结构化的 XML 文档数据集成时，需要解决如何统一表示和处理这些不同结构数据的问题。

语义异构性：即使数据结构相似，不同数据源中的数据可能具有不同的语义。相同的术语在不同的业务领域或系统中可能有不同的含义或者不同的术语可能表示相同的概念。例如，在一个企业中，“客户”在销售系统和售后服务系统中的定义和包含的信息可能不完全相同这就需要进行语义的统一和映射。

数据质量问题：不同数据源的教据质量参差不齐，可能存在数据缺失、错误、重复等问题，这些问题在集成讨程中会相互影响，增加了数据处理的复杂性。例如，从多个传感器采集的数据可能由于设备故障或环境干扰而存在误差，在与其他业务数据集成时需要进行数据清洗和质量提升。

数据源的动态变化：数据源可能会随着时间发生变化，包括数据内容的更新、数据模式的修改、新数据源的加入和旧数据源的删除等。数据集成系统需要具备足够的灵活性和适应性来应对这些变化，确保集成数据的准确性和及时性。

多源异构数据集成方法：

(1)数据仓库方法

原理：数据仓库方法是将各个数据源的数据抽取、转换和加载(ETL)到一个集中的数据仓库中。在 ETL 过程中，对数据进行清洗、转换和集成，使其符合数据仓库的统一数据模型。数据仓库通常采用多维数据模型，如星型模型或雪花型模型，以便于进行数据分析和查询。

优点：数据在存储前经过了预处理，数据质量较高，有利于提高查询和分析效率；数据仓库提供了统一的数据视图，便于用户使用；可以使用成熟的商业智能工具进行数据分析和挖掘。

缺点：ETL 过程复杂且耗时，特别是当数据源频繁变化时，需要重新设计和执行 ETL 流程；数据仓库的数据更新存在一定延迟，不能实时反映数据源的变化;需要大量的存储空间来存储预处理后的集成数据。

(2)中间件方法

原理：中间件方法是在数据源和应用程序之间插入一个中间件层。中间件通过包装器(Wrapper)对不同数据源的数据进行访问和处理将数据源的本地数据格式和操作转换为统一的接口形式。应用程序通过这个统一接口与中间件交互，实现对多源异构数据的访问和集成,

优点：对数据源的影响较小，不需要对数据源进行大量修改；具有较好的灵活性和适应性，可以快速集成新的数据源；能够实时或近实时地访问数据源数据，适用于对数据时效性要求较高的应用。

缺点：中间件的设计和实现复杂，需要处理不同数据源的多种协议和数据格式；由于需要实时访问数据源，可能会对数据源的性能产生一定影响；在数据一致性和完整性方面可能存在一定挑战，因为数据没有经过集中的预处理。

(3)联邦数据库方法

原理：联邦数据库方法是将多个异构数据库系统联合在一起，形成一个虚拟的、统一的数据库系统。在联邦数据库中，各个参与的数据库系统仍然保持其自治性，同时通过全局模式和映射机制实现对多个数据库的统一访问。全局模式定义了联邦数据库的逻辑结构，映射机制将全局模式与各个数据源的本地模式相连接。

优点：保留了数据源的自治性，各数据源可以独立管理和更新;对已有数据库系统的改动较小，适用于集成现有数据库系统;可以在一定程度上实现数据的共享和互操作，提高数据的利用率。

缺点：全局模式的设计和维护复杂，需要处理不同数据库之间的语义差异和结构差异;查询处理可能较为复杂，因为需要在多个数据源之间进行协调和数据整合;数据一致性维护困难，尤其是当不同数据源之间存在并发更新时。

多源异构数据集成方法应用案例分析：

(1)案例背景：某大型制造企业的数据集成项目

某大型制造企业拥有多个生产基地和业务部门，其数据来源包括生产管理系统(关系数据库)、设备监控系统(实时数据采集系统)、质量检测系统(XML 文件和关系数据库混合)等，企业需要整合这些数据用于生产决策、质量控制和供应链优化。

(2)方法选择与应用过程

数据仓库方法应用于生产决策:对于生产决策分析，采用数据仓库方法。从各个生产基地的生产管理系统中抽取生产计划、产量、工时等数据，从设备监控系统中抽取设备运行参数和故障信息，从质量检测系统中抽取产品质量数据。通过 ETL 过程对这些数据进行清洗、转换和集成，构建了以生产订单为核心的星型数据模型的数据仓库。企业管理人员可以使用商业智能工具从数据仓库中快速获取不同维度(如时间、生产基地、产品类型等)的生产数据报表和分析结果，为生产计划调整和资源分配提供决策依据。

中间件方法应用于实时质是监控:在质是控制方面，为了实时监控产品质量，采用中间件方法。中间件通过包装器与质量检测系统中的 XML 文件和关系数据库建立连接，将质量数据的访问接口统一。质量监控应用程序通过中间件实时获取最新的质量检测数据，包括原材料检验数据、生产过程中的半成品检验数据和成品检验数据。当检测到质量异常时，可以及时发出警报并采取相应的措施，确保产品质量的稳定性。

联邦数据库方法应用于供应链优化:对于供应链优化，采用联邦数据库方法。将企业内部的采购数据库、库存数据库和物流合作伙伴的数据库联合起来。通过设计全局模式，将采购订单、库存信息和物流运输信息进行统一表示。在这个虚拟的联邦数据库中，企业可以方便地查询和分析整个供应链上的货物流动情况，实现采购、库存和物流的协同优化，减少库存成本和运输时间。

(3)效果分析

通过综合应用不同的数据集成方法，该制造企业成功实现了多源异构数据的有效集成。在生产决策方面，数据仓库提供了高质量、全面的数据分析支持，提高了决策的准确性和效率，中间件方法在实时质量监控中保证了数据的时效性，及时发现和处理质量问题，降低了产品次品率，联邦数据库方法优化了供应链管理，提高了供应链的协同性和灵活性，降低了运营成本。

多源异构数据集成实施的关键因素和应对策略：

(1)关键因素

数据模型设计：合理的数据模型是数据集成的基础。无论是数据仓库的多维数据模型、中间件的统一接口模型还是联邦数据库的全局模式，都需要准确地反映数据源的数据结构和语义关系，

同时考虑到数据的扩展性和易用性。

数据质量保障：在集成过程中，要重视数据质量问题。包括数据清洗、去重、补全和数据一致性检查等措施。需要建立数据质量评估标准和监控机制，确保集成后的数据能够满足业务需求。

性能优化：数据集成系统的性能直接影响到用户体验和业务应用的效果。需要考虑数据抽取、转换、加载的效率，中间件的访问性能以及联邦数据库的查询性能等。优化数据存储结构、查询算法和网络通信等方面，提高系统的整体性能。

安全与隐私保护：在集成多源异构数据时，可能涉及企业的核心数据和敏感信息。需要建立严格的安全机制，包括数据访问控制、加密传输和存储等措施，保护数据的安全和隐私。

(2)应对策略

选代式开发与模型改进：由于数据源的复杂性和业务需求的动态变化，数据集成项目通常难以一次性完成。采用迭代式开发方法，根据业务需求的优先级逐步集成数据源，并不断改进数据模型，以适应新的数据和业务变化。

自动化与智能化的数据处理：利用数据清洗、数据质量评估和 性能优化 Q 的自动化工具，提高数据处理的效率和质量。同时，可以探索智能化技术，如机器学习 四算法在数据清洗、数据四配和语义映射中的应用，进一步优化数据集成过程。

建立安全管理体系：制定全面的安全策略和管理制度，包括用户身份认证、授权管理、数据加密标准和安全审计等内容，定期对数据集成系统进行安全检查和漏洞扫描，确保数据的安全和隐私不受侵犯。

(3)总结

多源导构数据集成是企业和组织在数字化发展过程中面临的重要挑战，数据仓库、中间件和联邦数据库等方法各有优缺点，在不同的应用场景中发挥着独特的作用。在实施数据集成时，需要综合考虑数据模型设计、数据质量保障、性能优化和安全与隐私保护等关键因素，并采取相应的应对策略。通过合理选择和应用数据集成方法，企业可以有效地整合多源异构数据，挖掘数据的潜在价值，提升业务决策水平和运营效率，从而在激烈的市场竞争中获得优势。

试题四 论分布式事务及其解决方案

请围绕“论分布式事务及其解决方案”论题，依次从以下三个方面进行论述。

1.概要叙述你参与分析设计的软件项目以及你在其中所承担的主要工作。

2.请介绍 4 种分布式事务的解决方案及简单说明。

3.具体阐述你参与的软件项目是如何做到分布式事务的，过程中遇到哪些问题，是如何解决

的。

答案：

论文素材参考：

分布式事务背景和面临的挑战：

(1)概念

分布式事务是指在分布式系统中，涉及多个数据源(如不同的数据库、消息队列等)或多个服务的操作，这些操作需要作为一个整体来执行，要么全部成功，要么全部失败，以保证数据的一致性。例如，在电商系统中，下单操作可能涉及库存系统扣减库存、订单系统创建订单、支付系统处理支付等多个子系统的操作，这些操作构成了一个分布式事务。

(2)产生背景

数据分散存储：随着业务的发展，数据往往被存储在多个不同的数据库或存储系统中，以满足不同的功能需求和性能只要求。例如，一个大型企业可能有多个分公司，每个分公司都有自己的本地数据库，而总公司需要对这些数据进行统一管理和业务操作，这就导致了分布式事务的需求。

微服务架构的兴起：微服务架构 Q 将一个大型应用拆分成多个独立的小服务，每个服务都有自己的数据库。当一个业务流程需要多个微服务协同完成时，就会产生分布式事务。比如，在一个基于微服务的金融系统中，贷款审批服务可能需要调用客户信息服务、信用评估服务和风险控制服务等，这些服务之间的操作需要保证事务的一致性。

(3)面临的挑战

网络通信问题：分布式系统中的节点通过网络进行通信，网络的不可靠性(如延迟、丢包、中断等)可能导致事务消息的丢失或延迟，从而影响事务的正常执行。例如，在两阶段提交过程中，协调者与参与者之间的通信故障可能导致参与者无法及时收到决策信息，陷入等待状态。

数据一致性问题：在分布式事务中，要保证多个数据源的数据在任何时候都保持一致是非常困难的。不同节点的数据更新可能由于各种原因(如部分节点故障、网络分区等)不能同步进行，从而导致数据不一致，例如，在一个跨数据库的转账事务中，如果一个数据库更新成功，另一个数据库更新失败，就会出现数据不一致的情况。

性能问题：分布式事务的处理通常需要额外的协调和通信开销，这可能会对系统的性能产生较大影响。特别是在高并发场景下，频繁的事务协调可能导致系统响应时间延长，吞叶是降低。例如，两阶段提交协议需要多次网络交互，在大量事务同时执行时，会占用大量的网络资源和计算资源。

分布式事务解决方案：

(1)两阶段提交(2PC)

原理：2PC 协议将分布式事务的提交过程分为两个阶段:准备阶段和提交阶段。在准备阶段，协调者向所有参与者发送准备请求，参与者执行本地事务操作，但不提交，然后向协调者反馈准备结果。如果所有参与者都准备成功，协调者在提交阶段向所有参与者发送提交请求，参与者提交本地事务;否则，协调者发送回滚请求，参与者回本地事务。

优点：实现原理相对简单，能够保证事务的强一致性，适用于对数据一致性要求极高的场景，如银行转账等核心金融业务。

缺点：存在单点故障问题，协调者故障可能导致整个事务阻塞;性能较差，由于多次网络交互和等待，在高并发场景下可能成为系统性能瓶颈;可能出现数据不一致问题，如在提交阶段协调者发送的提交请求部分参与者未收到时，这些参与者可能会自行决定回滚或等待，导致数据不一致。

(2)三阶段提交(3PC)

原理：3PC 在 2PC 的基础上增加了一个预提交阶段。在预提交阶段，协调者询问参与者是否可以提交事务，参与者进行本地事务的预处理并反馈结果。如果大多数参与者反馈可以提交，协调者进入准备阶段，后续流程与 2PC 类似。这个预提交阶段可以减少参与者在等待协调者决策时的阻赛时间。

优点：相比 2PC，在一定程度上降低了参与者的阻塞范围和时间，减少了协调者单点故障对事务的影响，提高了系统的可用性

缺点：实现复杂度增加，性能仍然受到多次网络交互的影响，目仍然不能完全避免数据不一致的情况，只是降低了发生的概率

(3)补偿事务(TCC)

原理：TCC 将分布式事务拆分为三个阶段：Try、Confimm 和 Cancel。Try 阶段主要是对业务资源的检查和预留，如冻结库存、预留资金等，但不进行实际的业务操作。Confrm 阶段在所有参与者都 Try 成功的情况下，执行真正的业务操作，如扣减库存、完成转账等。Cancel 阶段则在事务需要回滚时，对 Try 阶段预留的资源进行释放，恢复到事务前的状态。

优点：具有较好的灵活性和可扩展性，对业务的侵入性相对较小，可以根据不同的业务逻辑定制 Try、Confrm 和 Cance! 操作。适用于长事务和对性能要求较高的场景。

缺点：业务实现复杂度高，需要开发人员手动编写大量的补偿逻辑，对开发人员的要求较高;如果补偿逻辑编写不当，可能会导致数据不致或业务异常。

(4)本地消息表

原理：在本地消息表方案中，每个参与事务的服务都有一个本地消息表。当一个服务执行本地事务时，同时将需要其他服务执行的操作以消息的形式插入本地消息表。然后通过一个后台任务不断扫描本地消息表，将消息发送给其他服务。其他服务收到消息后执行相应操作。并反馈执行结果。如果执行成功，删除本地消息表中的消息;如果执行失败，可以进行重试或人工干预。

优点：避免了分布式事务协调器的单点故障问题，提高了系统的可靠性;对业务代码的侵入性相对较小，实现相对简单;适用于对最终致性要求较高的场景。

缺点：消息可能会堆积，需要合理设计消息处理机制和重试策略;可能存在消息丢失的风险，需要增加额外的机制(如消息持久化和确认机制)来保证消息的可靠性。

(5)消息队列的最终一致性

原理：基于消息队列实现分布式事务时，业务操作首先向消息队列发送一条消息，消息队列保证消息的可靠存储和传说。其他服务从消息队列中获取消息并执行相应的业务操作。通过消息的重试机制和补偿机制来保证即使在出现部分失败的情况下，系统最终能够达到一致性状态。这种方案不要求所有操作在同一时间点完成，允许一定的时间延迟来实现最终一致性。

优点：具有很高的性能和可扩展性，适合高并发的分布式系统;通过消息队列的异步处理方式，可以降低系统的耦合度，提高系统的灵活性;可以根据业务需求灵活调整消息的处理策略和重试次数。

缺点：实现最终一致性可能需要较长的时间，在这个过程中系统可能处于一种不一致的中间状态;需要处理消息的重复消费问题，防止因消息重复执行导致业务异常。

分布式事务解决方案应用案例分析：

(1)案例背景:某电商系统的分布式事务处理

某电商系统采用微服务架构，包括订单服务、库存服务、支付服务等多个微服务，每个微服务都有自己的数据库。在用户下单、支付等业务流程中涉及分布式事务问题。

(2)解决方案选择与应用

订单创建与库存扣减：对于订单创建和库存扣减这两个操作，由于对数据一致性要求较高，采用了 TCC 方案。在 Try 阶段，库存服务冻结用户购买商品的数量，订单服务创建一个临时订单(状态为待支付)。如果用户成功支付(Confrm 阶段)，库存服务扣减冻结的库存，订单服务将订单状态更新为已支付;如果支付失败(Cancel 阶段)，库存服务释放冻结的库存，订单服务删除临时订单。

支付外理与议单状态思新：支付外理洗及支付服务与认单服务之间的不口，这里买用了不地消县美方案。当支付成功时，支付服条将支付成功的消息插入本地消息表，同时更新自己的支付

记录。后台有一个消息处理任务，不断扫描本地消息表，将支付成功的消息发送给订单服务。订单服务收到消息后更新订单状态。如果消息发送失败，会进行重试。

物流信息更新与订单状态同步：物流信息更新与订单状态同步对实时性要求不是特别高，采用了消息队列的最终一致性方案。物流服务在处理物流信息更新时向消息队列发送消息，订单服务从消息队列中获取消息后更新订单的物流状态。通过消息的重试机制和去重机制，保证即使在网络不稳定或物流服务短暂故障的情况下，最终订单的物流状态能够与实际物流情况一致。

(3)效果分析

通过合理选择和应用不同的分布式事务解决方案，该电商系统在保证业务数据一致性的前提下，提高了系统的性能和可扩展性。TCC 方案保证了关键业务操作(如订单创建和库存扣减)的强一致性，本地消息表方案有效地协调了支付和订单状态更新的操作，消息队列的最终一致性方案满足了物流信息更新的灵活性和高并发处理需求。同时，系统在面对网络故障、部分服务故障等情况时，能够通过相应的机制保证业务的正常运行和数据的最终一致性。

选择和实施分布式事务解决方案的考虑因素：

(1)业务需求

根据业务对一致性的要求来选择解决方案。如果业务对数据一致性要求极高，如金融核心业务，可能需要选择强一致性的方案(如 2PC、3PC)；如果对性能和灵活性要求较高，且可以接受一定的不一致时间窗口，则可以选择最终一致性方案(如本地消息表、消息队列最终一致性等)。

(2)系统性能

考虑不同解决方案对系统性能的影响，特别是在高并发场景下。例如，2PC 和 3PC 在高并发时可能会导致性能瓶颈，而基于消息队列的方案通常具有更好的性能表现，但可能需要处理消息的重复消费和消息堆积等问题。

(3)开发难度和成本

不同的解决方案对开发人员的要求和开发成本不同。TCC 方案雲要开发人员编写复杂的补偿罗辑，开发难度较大:而本地消息表和消息队列方案相对简单，但需要考虑更多的异常处理和消息管理问题。需要综合评估企业的技术实力和开发资源来选择合适的方案。

(4)系统的可靠性和可用性

考虑方案在面对网络故障、节点故障等异常情况时的应对能力。例如，2PC 存在协调者单点故障问题，而本地消息表和消息队列方案可以通过几余和重试机制提高系统的可靠性和可用性。

分布式事务是分布式系统中一个复杂而关键的问题，其解决方案多种多样，各有优缺点和适用场景。在设计分布式系统架构时，需要根据业务需求、系统性能、开发难度和系统可靠性等多

方面因素综合考虑，选择合适的分布式事务解决方案。通过合理的解决方案，可以在保证数据一致性的同时，提高分布式系统的性能、可扩展性和可用性，确保分布式系统能够稳定、高效地运行，满足企业业务发展的需求。