**1、简述统一建模语言(UML)：**

答：统一建模语言(UML)是一种绘制软件蓝图的标准语言。可以用UML 对软件密集型系统的制品进行可视化详述和文档化。UML 是一种定义良好、易于表达、功能强大且普遍适用的可视化建模语言。它融入了软件工程领域的新思想、新方法和新技术。它的作用域不限于支持面向对象的分析与设计，还支持从需求分析开始的软件开发的全过程。UML 的作用就是用很多图从静态和动态方面来全面描述我们将要开发的系统。

**2、类图在UML 中有何重要作用？**

答： 1)为开发人员提供这种模仿现实世界的表达方式。2)让分析员使用客户所采用的术语和客户交流，促使客户说出所要解决的问题的重要细节。

**3、阐述用例对于系统开发人员来说的价值。**

答：是用来从用户的观察角度收集系统需求的一项技术，便于分析员与客户和用户交流，使系统更符合用户的需求。

**4、简述如何在实际工作中发现类。**

答：在与客户的交谈中，要注意客户用来描述业务实体的名词术语。这些名词可作为领域模型中的类。

还要注意你听到的动词，因为这些动词可能会构成这些类中的操作。当得到一组类的核心列表后，应当向客户询问在业务过程中每个类的作用。他们的回答将告诉你这些类的职责。

**5、简述怎样发现类之间的继承关系。**

答： 作为候选的类有可能和它的父类、子类在谈话中同时被发现。系统分析员意识到某个类的属性和操作也许能被运用到其他多个类当中去。另一种可能的情况是系统分析员注意到两个或者多个类可能具有相同的属性和操作数。

**7、UML 中的交互图有两种，分别是顺序图和协作图，请分析一下两者之间的主要差别和各自的优缺点。**

答：协作图可视化地表示了对象之间随时间发生的交互，它除了展示对象之间的关联，还显示出对象之间的消息传递。

与顺序图一样，协作图也展示对象之间的交互关系。顺序图强调的是交互的时间顺序，而协作图强调的是交互的语境和参与交互的对象的整体组织。顺序图按照时间顺序布图，而协作图按照空间组织布图。

顺序图可以清晰地表示消息之间的顺序和时间关系，但需要较多的水平方向的空间。协作图在增加对象时比较容易，而且分支也比较少，但如果消息比较多时难以表示消息之间的顺序。

**8、顺序图和协作图中，消息有哪三种？各自的意义和表示法什么？**

答：消息可以是简单的(simple)、同步的(synchronous)或异步的(asynchronous)。简单消息是从—个对象到另一个对象的控制流的转移。如果一个对象发送了—个同步消息，那么它要等待对方对消息的应答，收到应答后才能继续自己的操作。

而发送异步消息的对象不需要等待对方的应答便可以继续自己的操作。在顺序图中，简单消息是—个简单箭头，同步消息是实心箭头。异步消息是—个半边箭头。

**10、统一过程中有哪四个阶段，各阶段需要完成的主要工作有哪些？**

答：使用RUP 过程开发软件系统，软件的生命周期分四个阶段：初始阶段（Inception）、细化阶段（Elaboration）、构造阶段（Construction）、提交阶段（Transition）。各阶段需要完成的主要工作如下：

1）初始阶段：编制简要的愿景文档、业务案例、确定范围、粗略评估成本。

2）细化阶段：细化愿景文档、迭代地实现核心构架、解决高风险的问题、定义大多数的需求和范围、进一步评估成本。

3）构造阶段：迭代地实现系统的其余部分、准备部署4）提交阶段：beta 测试、部署

**11. 缩写词UML 的全称是什么？**

答案：UML 的中文意思是统一建模语言(Unified Modeling Language)，它是一种通用的可视化建模语言。

**12. UML 只适用于软件工程领域吗？**

答案：UML 同样也可以用来描述非软件领域的系统，如机械系统、企业机构或业务过程，以及处理复杂数据的信息系统、具有实时要求的工业系统或工业过程等。

**13. UML 包括哪些组成部分？**

答案：UML 包括语义概念、表示法和指导规范，提供了静态、动态、系统环境及组织结构的模型。UML 体系包括三个部分：UML 基本构造块、UML 规则和UML 公共机制。

**14. UML 事物（基本元素）有哪些？**

答案：UML 事物有：结构事物、行为事物、分组事物、注释事物。

**15. UML 关系有哪些？**

答案：UML 中关系包括四种：依赖、关联、泛化和实现。

**16. UML 图有哪些？其中哪些是静态图？哪些是动态图？**

答案：在UML 中包含9 类图：①类图；②对象图；③用例图；④顺序图； ⑤协作图；⑥状态图；⑦活动图；⑧组件图；⑨部署图。

静态图有：类图、对象图、用例图、组件图、部署图。

动态图有：顺序图、协作图、状态图、活动图。

**17. UML 虽然统一了软件建模符号，但仍然保有弹性可以与时俱进，这是采用了什么方法？**

答案：扩展机制

**18. UML 中也哪些扩展机制？**

答案：构造型、标记值、约束

**19. 什么是参与者？如何确定系统的参与者？**

答案：参与者(actor ，有些书翻译成“角色”)是一种特殊的类，是系统外部的一个实体，这个实体可以是任何的人或物，它以某种方式参与了用例的执行过程。

在获取用例前首先要确定系统的参与者，可以根据下面的一些问题来寻找系统的参与者：①谁使用系统？②谁安装系统、维护系统？③谁启动系统、关闭系统？④谁从系统中获取信息，谁提供信息给系统？⑤在系统交互中，谁扮演了什么角色？⑥系统会与哪些其他系统相关联？

**20. 什么是用例？如何确定系统的用例？**

答案：用例是对一组序列动作的描述，系统执行这些动作将对用例的参与者产生可以观察的结果。可以根据下面的一些问题来识别用例：

①参与者希望系统提供什么功能；

②系统是否存储和检索信息；

③当系统改变状态时，是否通知参与者；

④是否存在影响系统的外部事件，是哪个参与者通知系统这些外部事件。

**21. 什么是顺序图，顺序图是由那些部分组成？**

答案：顺序图是强调消息时间顺序的交互图。是描述系统中类和类之间的交互，它将这些交互建模成消息交换，也就是说，顺序图描述了类相互协作的完成预期行为的动态过程。顺序图是由：类角色、生命线、激活期和消息组成。

**22. 什么是协作图，协作图是由那些部分组成？**

答案：协作图作为另一种交互图而言，强调的是参加交互的对象的组织。协作图是由：类角色、关联角色和消息流组成。

**23. 试论述类与用例的区别。**

答案：类是对一组具有相同属性、操作、关系和语义的对象的描述。类是对事物的抽象。而用例是对一组序列动作的描述，系统执行这些动作将对用例的参与者产生可以观察的结果。

**24. 试比较边界类与实体类的异同。**

答案：实体类是对系统中需要存储的信息和其信息的行为建立模型。实体类具有永久的特性，这类似于数据库中的表一样用于保存系统的业务信息。

边界类位于系统与外界的交接处，它在一个或多个角色和系统之间建立相互作用的模型。

**25. 什么是依赖？他与关联有什么区别？**

答案：依赖是一种使用关系，它说明了一个事物声明说明的变化可能影响到使用它的另一个事物，但反之未必。也就是说，服务的使用者以某种方式依赖于服务的提供者。而关联是一种结构关系，它详述了一个事物的对象与另一个事物的对象相互联系。

**26. 什么是泛化？泛化是否就是类的继承，如果不是请说明理由。**

答案：泛化是一般事物（称为父类或超类）和较特殊事物（称为子类或孩子类）之间的关系。泛化不是类的继承，类的继承是泛化的一种。

**27. 试论述聚合和组合的异同。**

答案： 聚合描述了整体对象拥有部分对象的关系。组合是聚合的一种形式，它具有强的拥有关系，而且整体与部分的生命周期是一致的。

**28. 什么是动作状态？什么是活动状态？它们有什么区别？**

答案：动作状态是活动图中最小单位的构造块，表示原子动作。活动状态表示的是可以分割的动作。可以将对象的活动状态理解为一个组合，它的控制流由其他活动状态或动作状态组成。

**29. 活动图与传统的流程图有什么区别？**

答案：流程图一般限于顺序过程，而活动图则可以对付并行过程。流程图明确指出了每个活动的先后顺序，而活动图仅描述了活动和必要的工作顺序。

**30. 什么是对象图，请简述对象的作用？**

答案：对象图是描述在某一时刻，一组对象以及它们之间关系的图形。

对象图是描述系统交互的静态图形，它由协作的对象组成，但不包含在对象之间传递的任何消息。

**31. 包的访问可见性有几种？请描述各访问可见性的区别。**

答案：可见性可以分成3 种：公有访问（public）、保护访问（protected）、私有访问（private）。公有访问（public）：包内的模型元素可以被任何引入此包的其他包的内含元素访问。保护访问（protected）：表示此元素只能被当前包的子包访问。私有访问（private）：表示此元素只能被当前包内的模型元素访问。

**32. 什么是对象图，请简述对象的作用？**

答案：对象图（Object Diagram）是描述在某一时刻，一组对象以及它们之间关系的图形。

对象图是描述系统交互的静态图形，它由协作的对象组成，但不包含在对象之间传递的任何消息。

**33. 包的访问可见性有几种？请描述各访问可见性的区别。**

答案：可见性可以分成3 种：公有访问（public）、保护访问（protected）、私有访问（private）。

公有访问：包内的模型元素可以被任何引入此包的其他包的内含元素访问。

保护访问：表示此元素只能被当前包的子包访问。

私有访问：表示此元素只能被当前包内的模型元素访问。

**34. 请描述组件图和部署图的关系？**

答案：组件图用于描述系统中软件的构成，但没有描述系统中与硬件有关的构成情况。部署图则用于描述系统硬件的物理拓扑结构以及在此结构上运行的软件。

**35. 请叙述类，组件和节点的关系？**

答案：组件是系统中遵从一组接口且提供实现的一个物理部件，通常指开发和运行时类的物理实现。

节点和组件的关系：

①组件是参与系统执行的事物，而节点是执行组件的事物。简单的说就是组件是被节点执行的事物，如假设节点是一台服务器，则组件就是其上运行的软件。

②组件表示逻辑元素的物理模块，而节点表示组件的物理部署。这表明一个组件是逻辑单元（如类）的物理实现，而一个节点则是组件被部署的地点。一个类可以被一个或多个组件实现，而一个组件也可以部署在一个或多个节点上。

**36、简述UML 的9 种图。**

答案：（1）用例图：用例图展现了一组用况、参与者以及它们间的关系。可以用用例图描述系统的静态使用情况。在对系统行为组织和建模方面，用例图的是相当重要的。

（2）类图：

（3）对象图：对象图展示了一组对象及它们间的关系。用对象图说明类图中所反应的事物实例的数据结构和静态快照。对象图表达了系统的静态设计视图或静态过程视图，除了现实和原型的方面的因素外，它与类图作用是相同的。

（4）交互图：交互图展现了按一定的目的进行的一种交互，它由在一个上下文中的一组对象及它们间交互的信息组成。

交互图也可用于描述一个用况的行为。顺序图和协作图都是交互图，顺序图和协作图可以相互转换。

顺序图：展现了一组对象和由这组对象收发的消息，用于按时间顺序对控制流建模。用顺序图说明系统的动态视图。

协作图：展现了一组对象，这组对象间的连接以及这组对象收发的消息。它强调收发消息的对象的结构组织，按组织结构对控制流建模。

（5）状态图：展示了一个特定对象的所有可能状态以及由于各种事件的发生而引起的状态间的转移。一个状态图描述了一个状态机，用状态图说明系统的动态视图。它对于接口、类或协作的行为建模尤为重要，可用它描述用况实例的生命周期。

（6）活动图：活动图是一种特殊的状态图，描述需要做的活动、执行这些活动的顺序（多为并行的）以及工作流（完成工作所需要的步骤）。它对于系统的功能建模特别重要，强调对象间的控制流程。

高层活动图用于表示需要完成的一些任务，即用于分析用例，理解涉及多个用例的工作流、多线程及并行，显示相互联系的行为整体，还可用于对企业过程建模，对系统的功能建模。低层活动图用于表示类的方法。但活动图不适用于描述动作与对象间的关系，显示对象间的合作以及显示对象在生命周期内的运转情况。

（7）构件图（组件图）：构件图展现了一组构件之间的组织和依赖，用于对原代码、可执行的发布、物理数据库和可调整的系统建模。

（8）部署图：部署图展现了对运行时处理节点以及其中构件的配署。它描述系统硬件的物理拓扑结构(包括网络布局和构件在网络上的位置)，以及在此结构上执行的软件（即运行时软构件在节点中的分布情况）。用部署图说明系统结构的静态部署视图，即说明分布、交付和安装的物理系统。

**37. 请简要解释主动类的概念，并说明它建模的意义，并通过1-2 个实例来说明，主动类可以用来表示哪些设计元素。**

主动类的实例称为主动对象，一个主动对象拥有一个控制线程并且能够发起控制活动；它不在别的线程、堆栈或状态机内运行，具有独立的控制期。从某种意义上说，它就是一个线程。例如Socket 服务器的监听线程就可以表示为主动类。

**38. 在绘制类图时，第一步就是发现类，最常用的方法是什么？请简要说明它的使用方法。**

发现类的方法有很多种，其中最广泛应用的莫过于“名词动词法”，其主要规则是从名词与名词短语中提取对象与属性；从动词与动词短语中提取操作与关联；而所有格短短语通常表明名词应该是属性而不是对象。

**39. 请说明对象图的适用场景以及它的优缺点。**

对象图适用于论证类模型的设计以及对源代码进行分析和说明。其优点是能够直观理解出系统运行时的实时状态，缺点是比较复杂，工作量大。

**40. 交互概述图有哪两种形式，它们分别适用于什么场景？相对而言，哪一种会更加常用一些？**

一是活动图的变体，它将活动节点进行细化，用一些小的顺序图来表示活动节点内部的对象控制流；二是顺序图的变体，它用活动图来补充顺序图。但相对而言后者是十分不常见的，因此UML 的定义更倾向于将其称为活动图的变体。

**41、请简要说明交互概述图在建模时的优点和缺点。并简要描述交互概述图的绘制方法。**

优点：充分结合交互图和活动图的长处，在表达了对象到对象的控制流的同时，充分展示了从活动到活动的控制流。

缺点：对于规模稍大的场景，它并不是一个很好的选择，它将使模型的可读性大大降低绘制方法：理解活动控制流，并用活动图表示出来；分析活动节点：用交互图表述细节。

对象：对象是一个实体，这个实体具有明显定义的边界和标识，并且封装了状态和行为。类：类是一系列对象的抽象描述，它将相似的实体抽象成相同的概念，这种抽象过程强调相关特性而忽略其他特征。抽象：揭示事物区别于其他事物的本质的的过程，这个过程取决于使用者的目的，而忽略掉其他不相关的部分。封装：是指对象对其客户隐藏具体的实现，它是软件模块化的体现。通过封装实现信息隐藏和数据抽象。泛化：是类与类之间一种非常重要的关系，通过这种关系一个类可以共享另一个或多个类的结构和行为。多态：是在同一外表下表现多种行为的能力，它是对象技术的根本特征，是将对象技术称为面向对象的原因所在。

通用机制：达到对象建模目的的4种策略，在UML的不同语境中反复运用：（1）规格说明：文本维度的模型描述。（2）修饰：描述建模元素的细节信息。（3）通用划分：建模时对事物的划分方法。（4）扩展机制：用于扩展UML建模元素，包括构造性、约束、标记值三种机制。

业务用例模型：是说明预期功能的模型；是业务建模阶段的核心模型，用于确定组织的各个角色和可交付工件。业务用例模型由业务用例和业务参与者构成，主要目的是说明客户和合作伙伴是如何展开业务的：（1）识别业务参与者：为了充分理解业务目的，必须了解业务与谁进行交互，即业务为谁提供服务，即业务活动的服务对象。（2）识别业务用例：是对组织内部业务流程的说明，它定义一组业务用例实例，其中每个实例都是业务执行的一个操作序列，对于特定的业务参与者来说，这些操作序列所产生的结果是可见的。（3）描述业务用例：作为业务流程的封装体，业务用例是一个抽象的表示，业务建模过程还需要详细描述业务用例的内部流程，并将它作为软件建模的方法表示出

分析的基本原则：（1）分析模型应使用业务语言，分析模型中的类主要应该是业务领域的术语。（2）分析模型中类的细节和关系等应该业务中明确存在的，不要刻意去细化或封装这些细节。（3）分析活动是对需求模型的重新表述，是一种以理想化的方式来实现用例所描述的行为，并不考虑具体的技术实现。（4）分析侧重于系统的主要部分，关注核心的业务场景。对于那些支撑性行为，非功能需求等内容一般不做深入分析。（5）所有的分析类应该是为项目涉众产生价值的。

识别分析类(分析类主要用来表现系统应该提供哪些对象来满足需求)：（1）边界类：处于系统的最上层，它从那些系统和外界进行交互的对象中归纳和抽象出来，代表了系统与外界参与者交互的边界。存在两类边界类：用户界面类和系统/设备接口类。（2）控制类：处于三层构架的中间层，它封装控制系统上层的边界类和下层的实体类之间的交互行为，是整个用例行为的协调器。控制类能够有效的将边界对象和实体对象分开，让系统更能够适应其边界对象内发生的变更；并且还可以将用例所特有的行为与实体对象分开，使实体对象在用例和系统中具有更高的复用性。（3）实体类：代表了系统的核心概念，来自于对业务中的实体对象的归纳和抽象，用于记录系统所需要维护的数据和对这些数据的处理行为。

顺序图中的交互片段：交互片段将顺序图中的若干消息和对象封装为一个片段，针对这个片段可以实施不同的操作，从而来表示这个片段是以选择、循环还是并行等各种非顺序方式执行。（1）可选片段：操作符opt，表示该片段只有在守卫条件成立时才能够执行，否则跳过片段往后执行。（2）选择片段：操作符为alt，该片段的主体用水平虚线分割成几个分区。每个分区都有一个守卫条件，表示当守卫条件为真时执行该分区，且每次只能执行一个。（3）循环片段：操作符为loop，该片段在守卫条件为“真”的情况下循环执行，一旦为“假”，则跳过该片段往后执行。（4）并行片段：操作符为par，该片段主体也被水平虚线分成几个分区，当进入该片段后，这几个分区要并发执行

构造用例实现：是整个用例分析最核心的工作，目标是获得实现用例行为所必需的分析类，并利用这些分析类来描述其实现逻辑。（1）完善用例文档（2）识别分析类，包括三类分析类：边界类、控制类和实体类（重点）（3）分析交互。利用识别的分析类，利用交互图来分析用例实现的交互过程。（4）完成参与类类图。根据交互图中的消息和实体类内在的关系来绘制参与当前用例实现的类的类图。

Liskov替换原则：子类型必须能够替换它们的基类型。

开放-封闭原则：任何一个模块分解技术应该满足该原则，模块应该既是开放的又是封闭的。

设计模式：是在对象设计阶段，通过定义类或特定对象之间的结构和行为，从而解决某类设计问题的通用解决方案。作用：（1）可以有效地利用前人的经验来设计系统，而不用“重复劳动”。在提高质量的同时，通过复用可进一步加快开发效率。（2）作为一种“设计语言”，便于设计者之间相互交流而不产生误解。（3）是培养优秀设计师的捷径。

通用职责分配，创造者模式：指导分配与创建对象相关职责，其基本目的是找到一个在任何情况下都与创建者对象相关联的创建者，以支持低耦合。这些情况包括了整体聚合了部分、容器包含了对象，记录器记录了被记录的数据，通过在类图中定义的这些关系来选择创建者。

信息专家模式：是职责分配中使用最广泛的模式，它表达了一种“直觉”，对象处理自己拥有的相关信息的事物；或者说对象所能处理的职责依赖于其所拥有的数据。

构架：是一个系统的组织结构，包括系统分解成的各个部分、它们的连接性、交互机制和指导系统设计的相关原则。具有合理架构的系统，将使得对系统的理解、测试、维护和扩展都很容易。

包：一种将模型元素分组的机制，用来包含其他的ULM元素。同时，还为其内部元素提供了名字空间，外界需要通过包的名字来访问其内部的元素。包还作为一个配置管理单元，以用于管理软件的开发和发布。

依赖关系：包之间的关系称为依赖关系，用带箭头的虚线表示，箭头的方向标明了依赖的方向。（包合并关系；包导入关系[共有，私有]）

子系统设计基础：~的输入是具有接口定义的子系统，而输出模型则包括子系统的内部设计模型以及更新后的接口定义文档。（1）将子系统行为分配给子系统元素：一个子系统对外提供的行为完全由其接口进行描述，因此接口操作的集合代表了子系统的职责。子系统设计的第一步就是针对接口所描述的每一个操作进行设计，通过交互图将操作的职责分配给子系统内部的设计元素。（2）描述子系统内部的设计元素：在交互图的基础上，定义每个设计元素的结构和关系，完成子系统内部设计模型（3）定义子系统间的依赖关系：分析子系统与外部设计元素之间的依赖关系，明确子系统之间的耦合，以便于子系统的复用。

使用聚合和组合关系：存在整体和部分含义的关联关系，为聚合关系；具有很强的归属关系和一致的生命周期的整体和部分关系表示为组合关系。（组合关系是一种特殊的聚合关系，在整体拥有部分同时，部分不能脱离整体而存在；当整体不存在时，部分也没有存在的意义。从现实角度来说，聚合表示一种引用关联，即整体保存部分的引用，部分本身可以相互独立的存在；而组合则表示一种关联，整体直接拥有的值，并负责部分的创建和删除）

正向工程：指按照软件开发的基本过程，将抽象层次较高的模型转换成相对具体的模型的过程。将设计模型转换成实现模型就是一种典型的正向工程。

逆向工程：是正向工程的逆操作，即根据已有的源代码获得其设计模型。主要有两种使用场合：（1）在编码时，可能会存在和设计模型不一致的地方，可以通过你逆向工程更新原有的设计模型，从而需要保持设计模型的有效性。（2）针对已有的系统,在缺少或丢失了设计文档时，可以通过逆向工程重新获得系统的设计模型,理解程序和完善文档。

Uml2面向对象分析与设计

P27 图2-7

P35用例图，主要元素

P37活动图（借书）

P62旅店建模实例

P76 表4-1

P84 旅游用例图、旅游业务申请

P90 表4-7，用例参考

P96 用例文档实践

P114 包含扩展关系

P119 表4-24 旅游申请系统，分析方案

P135 典型的构架模式 表5-1 表5-2

P163 财务系统分析图

P167 VOPC图，模型

P188