软件工程习题解答

目录

习题 1

1、简述什么是软件危机？

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。这些问题绝不仅仅是“不能正常运行的”软件才具有的，实际上几乎所有软件都不同程度地存在这些问题。主要表现，如：对软件开发成本和进度估计不准确、软件产品的质量靠不住、用户对“已完成的”软件系统不满意、软件开发速度跟不上、软件不可维护以及没有适当的文档资料等等。

2、简述软件危机爆发的原因。

答：(1)软件开发过程的进展情况较难衡量，很难检验开发的正确性且软件开发的质量也较难评价，因此，控制软件开发过程相当困难；软件维护常常意味着修改原来的设计，维护的费用十分惊人，使得软件

较难维护。

(2) 软件开发的过程是多人分工合作，相当多的软件开发人员对软件的开发和维护存在不少错误的观念，或多或少采用了一些错误的方法和技术，这是造成软件危机的主要原因。

(3) 开发和管理人员只重视开发而轻视问题的定义，使软件产品无法满足用户的要求。

(4) 软件管理技术不能满足现代软件开发的需要，没有统一的软件质量管理规范。

(5) 在软件的开发和维护关系问题上存在错误的观念。软件维护工作通常是在软件完成之后进行的，因此是极端艰巨复杂的工作。

3、列举软件工程的七条基本原理。

答：（1）用分阶段的生命周期计划严格管理

（2）坚持进行阶段评审

（3）实行严格的产品控制

（4）采用现代程序设计技术

（5）结果应能清楚地审查

（6）开发小组的人员应该少而精

（7）承认不断改进软件工程实践的必要性

4、简述什么是软件工程。

答：软件工程是指导计算机软件开发和维护的一门工程学科。采用工程的概念、原理、技术和方法来开发和维护软件，把经过时间考验而证明正确的管理技术和当前能够得到的最好技术结合起来，以经济地开发出高质量的软件并有效地维护它，这就是软件工程。

5、软件生存期分哪几个时期？

答：软件生命周期由软件定义、软件开发和软件维护三个时期组成。

6、软件定义时期通常划分成哪三个阶段？

答：软件定义时期通常划分成三个阶段，即问题定义、可行性研究和需求分析。

7、软件开发时期通常由哪四个阶段组成？

答：开发时期通常由四个阶段组成：总体设计、详细设计、编码、单元测试和综合测试。

8、目前使用最广泛的软件工程方法学为哪两种？

答：结构化方法和面向对象方法。

9、简述结构化开发方法的基本思想。

答：用系统工程的思想和工程化的方法，按用户至上的原则，结构化、模块化、自顶向下地对系统进行分析与设计。具体来说，就是先将整个信息系统开发过程划分出若干个相对独立的阶段，如系统规划、系统分析、系统设计、系统实施等。在前三个阶段坚持自顶向下地对系统进行结构化划分，在系统调查或理顺管理业务时，应从最顶层的管理业务入手，逐层深入至最底层。在系统分析、提出新系统万案和系统设计时，应从宏观整体人手，先考虑系统整体的优化，然后再考虑局部的优化问题。在系统实施阶段，则应坚持自底向上地逐步实施。也就是说，组织力量从最底层模块做起(编程)，然后按照系统设计的结构，将模块一个个拼接到一起进行调试，自底向上、逐渐地构成整体系统。

10、简述什么是面向对象开发方法。

答：面向对象方法主张从客观世界固有的事物出发来构造系统，提倡用人类在现实生活中常用的思维方法来认识、理解和描述客观事物，强调最终建立的系统能够映射问题域，也就是说，系统中的对象以及对象之间的关系能够如实地反映问题域中固有事物及其关系。

11、列举几个具有代表性的需求分析和设计工具。

答：IBM Rational Requirement Composer，Rational Rose，IBM Rational Software Architect (RSA)，PowerDesigner，Microsoft Office Visio，Enterprise Architect，等等。

12、简述瀑布模型的核心思想和特点。

答：瀑布模型核心思想是按工序将问题化简，将功能的实现与设计分开，便于分工协作，即采用结构化的分析与设计方法将逻辑实现与物理实现分开。

瀑布模型的特点如下：

（1）阶段间具有顺序性和依赖性。即必须等前一阶段的工作完成之后，才能开始后一阶段的工作；前一阶段的输出文档就是后一阶段的输入文档。

（2）推迟实现。瀑布模型在编码之前设置了系统分析和系统设计的各个阶段，在这两个阶段主要考虑目标系统的逻辑模型，不涉及软件的物理实现。所以，清楚地区分逻辑设计与物理设计，尽可能推迟程序的物理实现，是按照瀑布模型开发软件的一条重要的指导思想。

（3）质量保证。每个阶段都必须完成规定的文档，没有交出合格的文档就是没有完成该阶段的任务。每个阶段结束前都要对所完成的文档进行评审，以便尽早发现问题，改正错误。

13、简述瀑布模型的优缺点。

答：瀑布模型具有下列优点：

（1）可强迫开发人员采用规范化的方法。

（2）严格地规定了每个阶段必须提交的文档。

（3）要求每个阶段交出的所有产品都必须是经过验证的。

瀑布模型的缺点：

（1）由于瀑布模型几乎完全依赖于书面的规格说明，很可能导致最终开发出的软件产品不能真正满足用户的需要。

（2）瀑布模型只适用于项目开始时需求已确定的情况。

14、什么是增量模型？有哪些优点？

答：增量模型也称为渐增模型，增量式开发该方法使得描述活动、开发活动和有效性验证括动交织在一起。系统的开发是建立一系列的版本（增量），每个版本添加部分功能到先前的版本中。

增量模型具有以下优点：

(1)能在较短时间内向用户提交可完成一些有用的工作产品，即从第 1 个增量交付之日起，用户就能做一些有用的工作。

(2)逐步增加产品的功能可以使用户有较充裕的时间学习和适应新产品，从而减少一个全新的软件可能给用户组织带来的冲击。

(3)项目失败的风险较低，虽然在某些增量构件中可能遇到一些问题，但其他增量构件将能够成功地交付给客户。

(4)优先级最高的服务首先交付，然后再将其他增量逐次集成进来。因此，最重要的系统服务将接受最多的测试。

15、什么是快速原型模型？有哪些优点？

答：快速原型模型又称原型模型，是增量模型的另一种形式。它是在开发真实系统之前，构造一个原型，即快速建立起来可以在计算机上运行的程序，它所能完成的功能往往是最终产品能完成的功能的一个子集。

然后，在该原型的基础上，逐渐完成整个系统的开发工作。

快速原型模型具有如下优点：

(1)有助于满足用户的真实需求。

(2)原型系统已经通过与用户的交互而得到验证，据此产生的规格说明文档能够正确地描述用户需求。

(3)软件产品的开发基本上是按线性顺序进行。

(4)因为规格说明文档正确地描述了用户需求，因此，在开发过程的后续阶段不会因为发现规格说明文档的错误而进行较大的返工。

(5)开发人员通过建立原型系统已经学到了许多东西，因此，在设计和编码阶段发生错误的可能性也比较小，这自然减少了在后续阶段需要改正前面阶段所犯错误的可能性。

(6) 快速原型的突出特点是“快速”。开发人员应该尽可能快地建造出原型系统，以加速软件开发过程，节约软件开发成本。

16、简述喷泉模型的特点。

答：喷泉模型是一种以用户需求为动力，以对象为驱动的模型，主要用于描述面向对象的软件开发过程。“喷泉”一词体现了迭代和无间隙特性，该模型认为软件开发过程自下而上周期的各阶段是相互迭代和无间隙的特性。软件的某个部分常常被重复工作多次，相关对象在每次迭代中随之加入渐进的软件成分。无间隙指在各项活动之间无明显边界，如分析和设计活动之间没有明显的界限，由于对象概念的引入，表达分析、设计、实现等活动只用对象类和关系，从而可以较为容易地实现活动的迭代和无间隙，使其开发自然地包括复用。

17、简述喷泉模型的优缺点。

答：喷泉模型具有以下优点：

喷泉模型不像瀑布模型那样，需要分析活动结束后才开始设计活动，设计活动结束后才开始编码活动。该模型的各个阶段没有明显的界限，开发人员可以同步进行开发。其优点是可以提高软件项目开发效率，节省开发时间，适应于面向对象的软件开发过程。

喷泉模型也存在以下缺点：

由于喷泉模型在各个开发阶段是重叠的，因此在开发过程中需要大量的开发人员，因此不利于项目的管理。此外这种模型要求严格管理文档，使得审核的难度加大，尤其是面对可能随时加入各种信息、需求与资料的情况。

18、什么是螺旋模型？

答：螺旋模型是一种演化软件开发过程模型，它兼顾了快速原型的迭代的特征以及瀑布模型的系统化与严格监控，其最大的特点在于引入了其他模型不具备的风险分析，使软件在无法排除重大风险时有机会停止，以减小损失。同时，在每个迭代阶段构建原型是螺旋模型用以减小风险的途径。

19、简述螺旋模型的限制条件。

答：(1)螺旋模型强调风险分析，但要求许多客户接受和相信这种分析，并做出相关反应是不容易的，因此，这种模型往往适应于内部的大规模软件开发。

(2)如果执行风险分析将大大影响项目的利润，那么进行风险分析毫无意义，因此，螺旋模型只适合于大规模软件项目。

(3)软件开发人员应该擅长寻找可能的风险，准确地分析风险，否则将会带来更大的风险。

20、简述螺旋模型的优缺点。

答：螺旋模型具有以下优点：

(1)设计上的灵活性，可以在项目的各个阶段进行变更。

(2)以小的分段来构建大型系统，使成本计算变得简单容易。

(3)客户始终参与每个阶段的开发，保证了项目不偏离正确方向以及项目的可控性。

(4)随着项目推进，客户始终掌握项目的最新信息，从而使得客户能够和管理层有效地交互。

(5)对可选方案和约束条件的强调有利于已有软件的重用，也有助于把软件质量作为软件开发的一个重要目标。

螺旋模型也存在以下缺点：

螺旋模型是风险驱动的，因此要求软件开发人员必须具有丰富的风险评估经验和这方面的专门知识，否则将出现真正的风险：当项目实际上正在走向灾难时，开发人员可能还以为一切正常。所以，很难让用户确信这种演化方法的结果是可以控制的。

21、统一过程模型包括哪 6 个核心工作流？

答：业务建模、需求、分析设计、实现、测试、部署。

22、统一过程模型有哪 4 个阶段？

答：初始阶段、细化阶段、构造阶段和移交阶段。

23、简述什么是敏捷开发。

答：敏捷开发以用户的需求进化为核心，采用迭代、循序渐进的方法进行软件开发。在敏捷开发中，软件项目在构建初期被切分成多个子项目，各个子项目的成果都经过测试，具备可视、可集成和可运行使用的特征。

24、简述敏捷开发的原则。

答：（1）快速迭代

（2）让测试人员和开发者参与需求讨论

（3）编写可测试的需求文档

（4）多沟通，尽量减少文档

（5）做好产品原型

（6）及早考虑测试

习题 2

1、问题定义要回答的关键问题是？

答：问题定义要回答的关键问题是：“要解决的问题是什么？”。

2、简述问题定义的规范化要求。

答：问题定义的规范化要求如下：

（1）重视问题定义，不能把其当作一件小事

（2）客观、全面的定义，不能避重就轻、偷工减料

（3）清楚问题定义的工作内容，不能把问题定义当作解决方法

（4）深入分析，抓住问题的本质

（5）严格评审

3、可行性研究要回答的关键问题是？

答：可行性研究要回答的关键问题是：“对于上一个阶段所确定的问题有行得通的解决办法吗？”

4、一般从哪三个方面对系统进行可行性分析？

答：1．技术可行性 2．经济可行性 3．法律可行性

5、需求获取要解决的问题主要包括哪些？

答：(1) 发现和分析问题，并分析问题的因果关系。

(2) 与用户进行各种方式的交流，并使用调查研究方法收集信息。

(3) 按照三个成分观察问题的不同侧面：即数据、过程和接口。

(4) 将获取的需求文档化，形式有用例、决策表、数据流图、数据字典等。

6、简述需求获取的原则和步骤。

答：需求获取应遵循以下原则：

(1) 深入浅出的原则。就是说，需求获取要尽可能全面、细致，获取的需求是个全集，目标系统真正实现的是个子集。

(2) 以流程为主线的原则。在与用户交流的过程中，应该用流程将所有的内容串起来，如信息、组织结构、处理规则等，这样便于交流沟通。

步骤：

（1）深入了解应用领域，开发高层的业务模型

（2）定义项目范围和高层需求

（3）识别用户类型和用户代表

（4）获取具体的需求

（5）确定目标系统的业务工作流

（6）需求整理与总结

7、结构化分析的框架主要包括哪些内容？

答：结构化分析的框架主要包括功能建模（数据流图）、数据建模（实体-关系模型）、行为建模（状态转换图）以及框架核心（数据字典）。

8、简述功能建模的思想。

答：功能建模的思想就是用抽象模型的概念，按照软件内部数据传递、变换的关系，自顶向下逐层分解，直到找到满足功能要求的所有可实现的软件为止。

9、数据流图的基本图形符号包括哪些？

答：（1）外部实体：数据输入源或数据输出汇点，不是目标系统的一部分，只是外围环境中的实体部分，包括人员、组织、部门或其他相关的软件系统。

（2）数据流：数据在系统内传播的路径，数据沿箭头方向流动。数据流可以在加工和加工之间也可以在数据存储和加工之间传送，数据流在数据存储和加工之间传送时含义明确，数据存储就足以说明数据流，所以不必命名。同一数据流图上不能有同名数据流。

（3）加工：又称数据处理，是对数据对象进行某些处理或变换，其名称简要的描述完成什么加工。流入加工的可以是多个数据流，流出加工的也可以是多个数据流。

（4）数据存储：又称数据文件，可以是数据库文件或任何形式的数据组织。流入数据存储的数据流表示写入数据，流出数据存储的数据流表示读出数据。

10、简述什么是实体-关系模型及其三种组成元素。

答：实体-关系模型表示为可视化的实体-关系图，也称为 E-R 图。 E-R 图中仅包含 3 种相互关联的元素：数据对象、描述数据对象的属性及数据对象彼此间相互连接的关系。

数据对象，也称为实体，是目标系统所需要的复合信息的表示，所谓复合信息是具有若干不同属性的信息。属性定义数据对象的特征。实体间的联系是错综复杂的，但就两个实体型的联系来说，主要有以下 3 种情况：(1) 一对一（1:1）联系(2) 一对多（1:m）联系(3) 多对多（m:n）联系

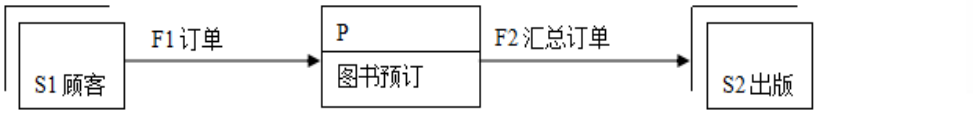
11、什么是行为建模？

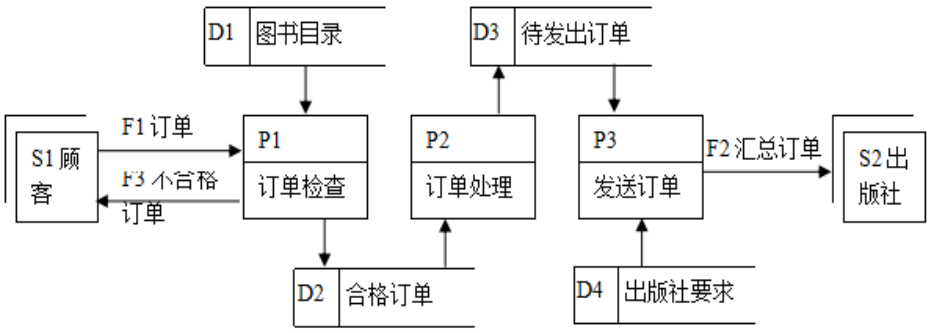
答：行为建模即绘制系统的状态转换图（简称状态图），通过描绘系统的状态及引起系统状态转换的事件来表示系统的行为。

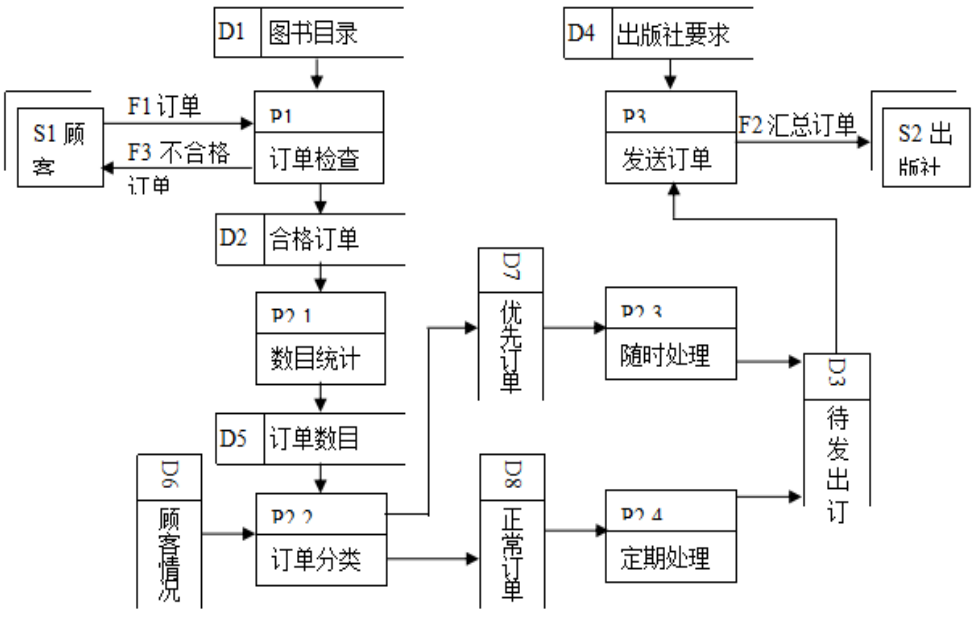
12、什么是数据字典？

答：数据字典以词条方式定义在数据模型、功能模型和行为模型中出现的数据对象及控制信息的特性，给出它们的准确定义。主要是数据流图上所有的成分的定义和解释的文字集合，包括数据流、加工、数据存储、数据元素，以及数据源点和汇点等。

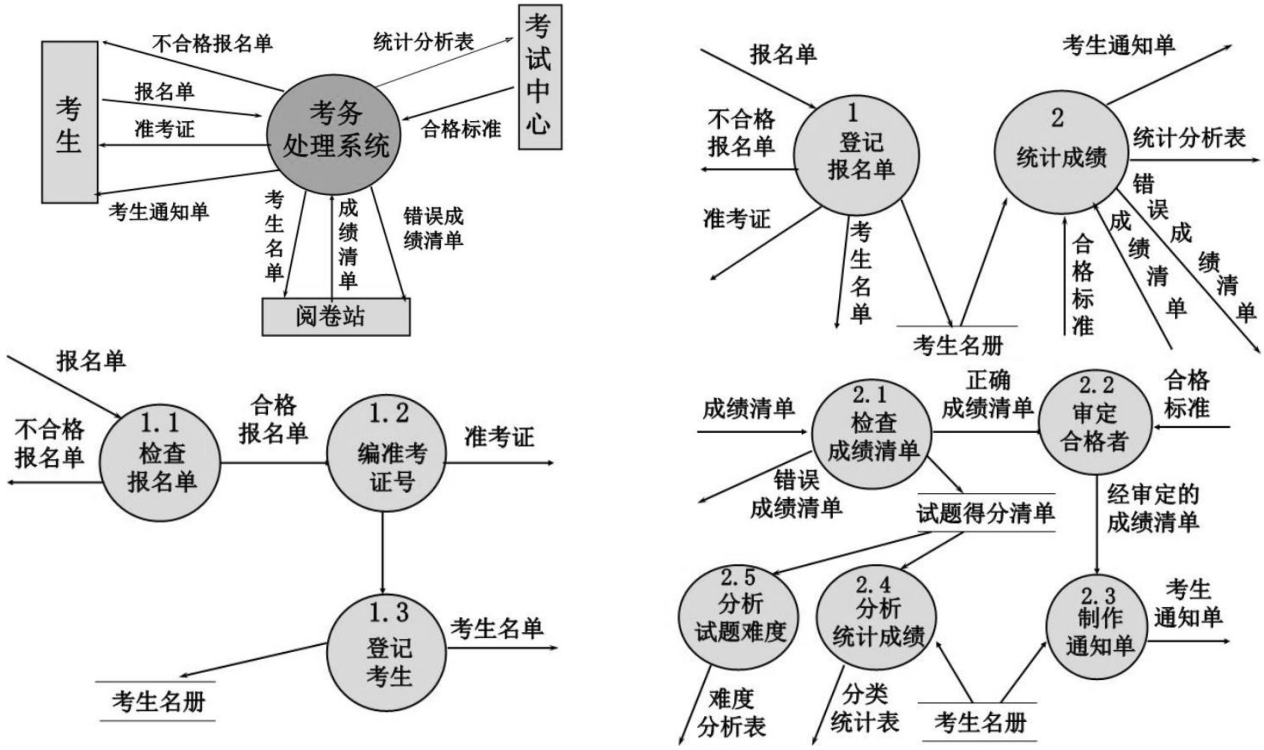
13、要设计一个图书预订系统，画出此系统的分层数据流图。



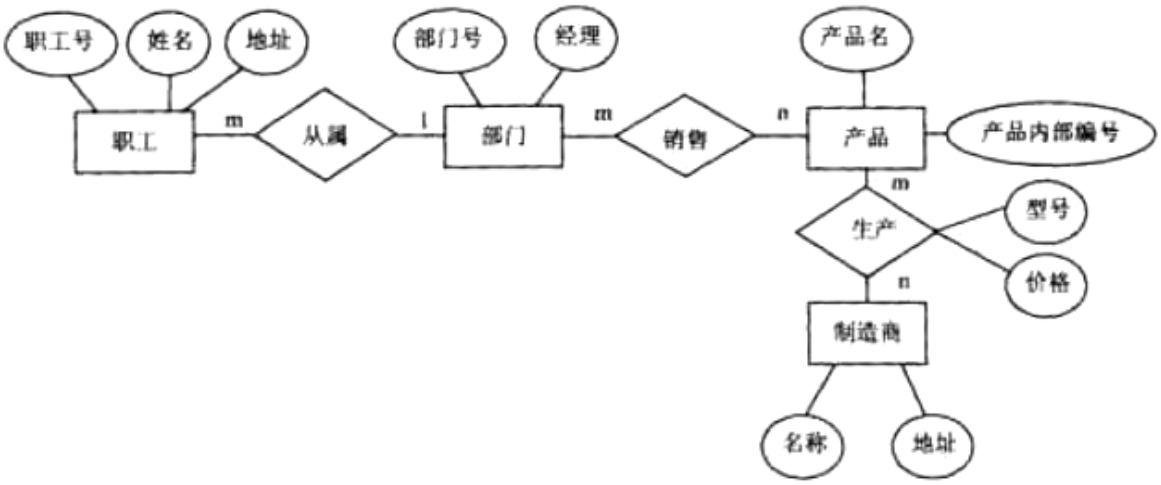




14、简单的考务处理系统，画出此系统的分层数据流图。



15、某销售部门管理系统，试画出这个数据库的 E-R 图。

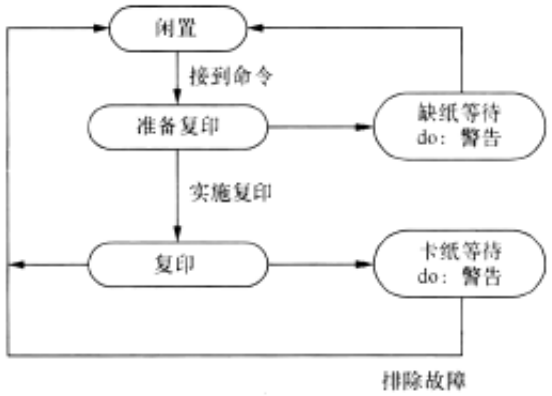


16、用状态转换图描述复印机行为。

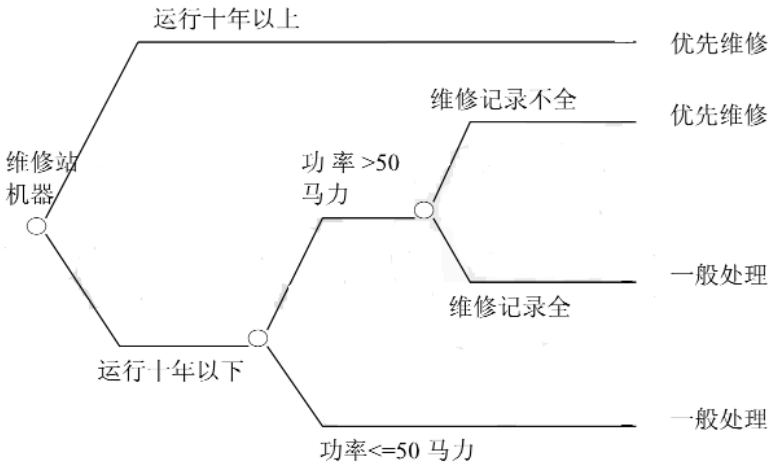
17、某维修站对“功率大于 50 马力”的机器且“维修记录不全”或“已运行十年以上”的机器应给予优先维修，否则做一般处理。请绘制判定表和判定树。

答：

		1	2	3	4
	运行时间	>10 年	<=10 年	<=10 年	<=10 年
	功率	-	>50 马力	>50 马力	<=50 马力-



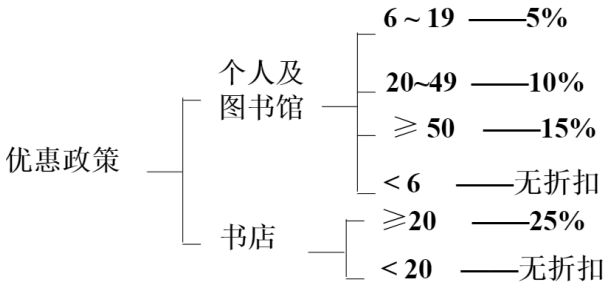
	维修记录	-	不全	全	-
动	优先维修	√	√		
作	一般处理			√	√



18、某图书发行公司采用下列政策优惠用户。书店订购 20 册以上优惠 25%，否则不优惠；图书馆和个人订购 6 册以下则不优惠；6~19 册则优惠 5%，20~49 册则优惠 10%，50 册以上则优惠 15%。请绘制判定表和判定树。

答：

		1	2	3	4	5	6
条件	用户身份	书店	书店	个人	个人	个人	个人
	订购数量	>=20	<20	<6	6~19	20~49	>=50
动作	25%	√					
	15%						√
	10%					√	
	5%				√		
	无折扣		√	√			



19、略

习题 3

1、结构化设计通常划分成哪两个步骤？

答：结构化总体设计和结构化详细设计。

2、什么是过程设计？

答：过程设计的主要工作是确定软件各个模块内的算法及内部数据结构，并选定某种过程的表达形式来描述各种算法，其设计依据为结构化需求分析阶段的数据流图、状态转换图及数据字典。

3、简述结构化设计的原则。

答：（1）模块化（2）高内聚、低耦合（3）抽象（4）信息隐藏（5）一致性

4、什么是模块独立性？它的两个定性标准度量是什么？

答：模块独立是指每个模块只完成一个相对独立的特定子功能，并且和其他模块之间的关系很简单，和其他模块之间没有过多的相互作用。

模块独立性由两个定性标准度量：内聚和耦合，提倡模块遵循高内聚、低耦合的原则，保证模块具有较好的独立性。

5、什么是耦合性？一般分为哪 7 种类型？

答：耦合性是程序结构中各个模块之间相互关联的度量，它取决于各个模块接口的复杂程度、模块的调用方式以及通过接口的信息。

7 种类型：（1）非直接耦合（2）数据耦合（3）标记耦合（4）控制耦合（5）外部耦合（6）公共耦合（7）内容耦合

6、什么是模块的内聚？一般分为哪 7 种类型？

答：模块的内聚是指模块内部各元素彼此结合的紧密程度 ，一个内聚程度高的模块在理想状况下应只做一件事。

7 种类型：（1）功能内聚（2）信息内聚（3）通信内聚（4）过程内聚（5）时间内聚（6）逻辑内聚（7）巧合内聚

7、解释数据耦合和功能内聚。

答：数据耦合：一个模块访问另一个模块时，彼此之间是通过简单数据参数 (不是控制参数、公共数据结构或外部变量)来交换输入、输出信息的。

功能内聚：一个模块中各个部分都是完成某一具体功能必不可少的组成部分，或者说该模块中所有部分都是为了完成一项具体功能而协同工作，紧密联系，不可分割的，则称该模块为功能内聚模块。

8、简述体系结构的设计应遵循启发式设计原则。

答:(1)提高模块独立性(2)模块规模适中(3)结构图的深度和宽度适中(4)结构图中扇入和扇出适当(5)模块的作用域应在控制域之内(6)模块功能的完善化(7)消除重复功能，改善软件结构

9、简述体系结构的设计中面向数据流方法的设计过程。

答：首先对需求分析阶段得到的数据流图进行复查，必要时进行修改和精华；接着仔细分析系统数据流图，确定数据流图的类型，并按照相应的设计步骤将数据流图转化为软件结构；最后根据体系结构的启发式设计原则对得到的软件结构进行精化。

10、简述变换型数据流向变换型结构映射的一般步骤。

答：(1)整合数据流图，划分边界(2)进行一级分解，设计系统的上层模块(3)进行二级分解，设计系统的中、下层模块

11、事务型数据流的特征是什么？

答：事务型数据流的特征通常是接受一项事务，根据事务处理的特点和性质，选择分派一个适当的处理单元，然后给出结果。

12、简单解释顺序文件、随机文件和索引文件。

答：顺序文件是最常用的文件组织形式，是记录按其在文件中的逻辑顺序依次进入存储介质而建立的，即顺序文件中物理记录的顺序和逻辑记录的顺序是一致的。

随机文件直接组织记录的存储与存储顺序无关，它与顺序组织完全不同，其中每一笔记录都存储在直接存取存储装置中的某一个特定的地址上。

索引文件由数据文件组成，它是带索引的顺序文件，由索引表和主文件两部分构成。索引表本身非常小，是一张指示逻辑记录和物理记录之间对应关系的表。

13、将 E-R 图映射成关系数据库中的关系模型时，数据对象实体如何映射？

答：一个数据对象（实体）可以映射为一个表。根据用户的数据操作需求，也可以分解为多个表，一般采用横切和竖切的方法进行分解。

14、将 E-R 图映射成关系数据库中的关系模型时，两个实体间一对一联系如何映射？

答：可以在两个表中都引入对方实体的主键作为外键，进行双向导航；也可以将两个实体的联系单独构成一个表，此表包含两个实体的主键，加上两个实体映射成的表，总共映射成三个表；还可以将两个实体合并成一个表。

15、将 E-R 图映射成关系数据库中的关系模型时，两个实体间一对多联系如何映射？

答：可以将联系中的“一”端毫无变化地映射到一张表，将联系中“多”端上的实体映射到带有外键的另一

张表，外键即为“一”端实体的主键，满足关系的参照完整性；也可以将两个实体的联系单独构成一个表，此表包含两个实体的主键，加上两个实体映射成的表，总共映射成三个表。

16、将 E-R 图映射成关系数据库中的关系模型时，两个实体间多对多联系如何映射？

答：为了表示多对多联系，关系模型必须将两个实体的联系单独构成一个表，即将两个实体的主键作为外键的第三个表，这样，将两个数据实体之间的多对多联系就转换成了两个一对多联系。

17、简述视图的优点。

答：(1)为用户集中数据，使得分散在多个表中的数据，通过视图定义在一起，屏蔽了数据库的复杂性。(2)保证数据的逻辑独立性。 (3)提高了数据的安全性。

18、简述过程设计阶段的目标和任务。

答：过程设计阶段的目标是确定具体的实现所要求的系统，从而在软件实现阶段可以把这个系统设计直接翻译成某种程序设计语言编码实现，所以过程设计最重要的是尽可能地将各模块的处理过程描述得简明易懂。

过程设计的任务包括：(1)算法设计(2)数据结构细节和数据操作的设计 (3)输入/输出格式设计 (4)编写过程设计说明书

19、程序流程图包括哪五种基本控制结构？

答：1)顺序结构 2)选择结构 3)先判断（WHILE）型循环结构 4)后判断（UNTIL）型循环 5)多分支（CASE）型选择结构

20、简述 N-S 图的特点。

答：1)除 CASE 型结构中表示条件取值的矩形框外，图中每个矩形框都是明确定义了的一个特定控制结构的作用域。

2)N-S 图不能进行随意的控制转移，可以严格遵守结构化程序设计的要求。

3)结构清晰，很容易确定变量的作用域，很容易表现嵌套关系，也可以表示模块的层次结构。

21、简述 PAD 的优点。

答：1)使用 PAD 符号所设计出来的程序不能进行随意的控制转移，必然是结构化程序。

2)PAD 图描绘程序结构清晰，图中竖线的总条数就是程序的层次数，所以用 PAD 图表现程序逻辑易读、易懂、易记。

3)容易将 PAD 图自动转换为高级语言源程序。

4)PAD 图既可以表示程序逻辑，也可用于描绘数据结构。

5)PAD 图的符号支持自顶向下、逐步求精方法的使用。

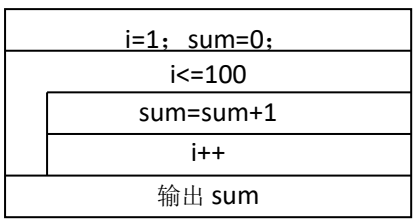
(4)图形工具程序设计举例

22、分析程序流程图的功能，并画出相应的 N-S 图和 PAD 图。

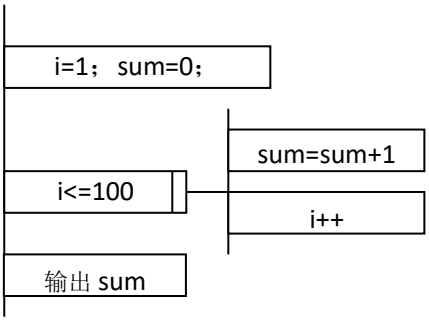
答：

功能：求 1 到 100 的和。

N-S 图：



PAD 图：

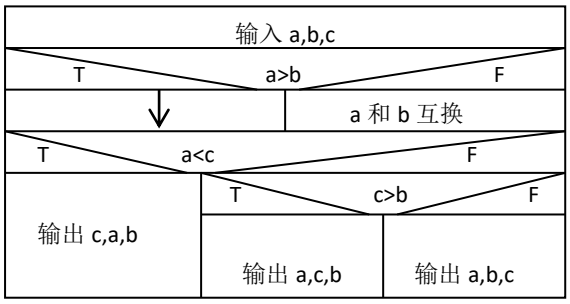


23、分析程序流程图的功能，并画出相应的 N-S 图和 PAD 图。

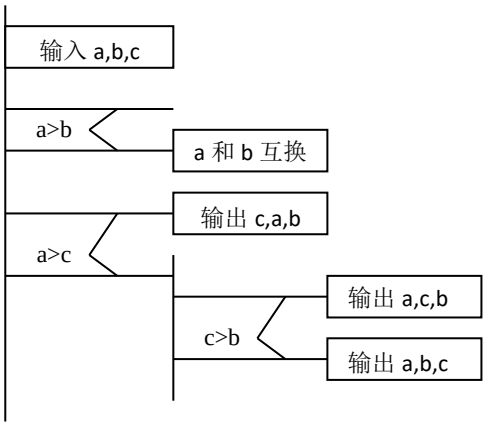
答：

功能：将三个数从大到小输出。

N-S 图：



PAD 图：



24、简述 PDL 语言的特点和优点。

答：特点：

(1)结构化程序设计语言的关键字提供了结构化控制结构、数据说明和模块化的特点。

(2)自然语言的自由语法是它描述处理的特点。

(3)数据说明的手段应该既包括简单的数据结构（例如单变量和数组），又包括复杂的数据结构（例如，链表或层次的数据结构）。

(4)模块定义和调用的技术，应该提供各种接口描述模式。

优点：

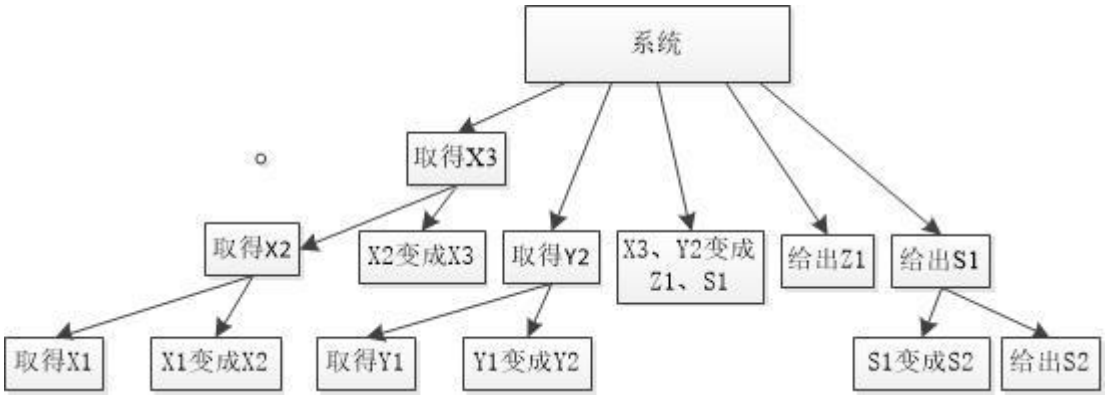
(1)可以作为注释直接插在源程序中间。

(2)可以使用普通的文字处理系统，很方便地完成 PDL 的编辑。

(3)可以自动由 PDL 生成程序代码。

25、有一个系统的变换型数据流图如图 3-58 所示，其中用虚线划分出了逻辑输入、逻辑输出和变换中心。请画出此系统的体系结构图。

答：



26、略。

习题 4

1、简述低级语言和高级语言的区别。

答：低级语言包括机器语言和汇编语言，优点是执行速度快，但代码编写难度较大，可读性较差；另外，低级语言编写的程序与具体的机器有关，想要运行在不同的机器上，必须重写，但是在实现与硬件系统的接口部分时，易于实现、实现效率高。

高级语言的特点是在一定程度上与具体机器无关，具有可移植性，仅需稍作修改甚至不用修改，就可将一段代码运行在不同类型的计算机上；它更接近于人的思维，易于编程，易于阅读，易于修改。但是，运行高级语言程序时，需要先将其翻译成机器语言，运行效率相对较低；对硬件的可控性相对于低级语言较弱。

2、在选择程序设计语言时，可以考虑哪几方面的因素？

答：（1）应用领域的不同决定选择的语言。（2）系统用户的要求。（3）程序员的经验和知识。（4）开发和维护成本。（5）软件可移植性要求。

3、为便于理解，程序设计中如何进行标识符的命名。

答：标识符包括模块名、变量名、常量名、标号名、子程序名以及数据区名、缓冲区名等。这些名字应能反映它所代表的实际东西，使其能够见名知意，有助于对程序功能的理解。应当选择精练的意义明确的名字，才能简化程序语句，易于对程序功能的理解。另外标识符名不能与关键词同名。

4、简述程序设计中使用注释的好处。

答：正确的注释能够帮助读者理解程序，为测试和维护阶段提供明确的指导。注释的原则是有助于对程序的阅读理解，所以注释语言必须准确、易懂、简洁，注释不宜太多也不能太少，注释的内容要清楚、明了、含义准确，防止注释二义性，该加的地方一定要加，但不必要的地方一定不要加。好的程序编码，注释行的数量应占到整个源程序的 1/3 到 1/2。

5、简述程序设计中如何进行程序代码的布局以增强程序的可读性。

答：首先，恰当地利用空格，可以增加大型表达式的清晰度。其次，自然的程序段之间可用空行隔开，使得模块之间的界限分明。最后，缩进也叫做向右缩格，对于选择语句和循环语句，把其中的程序段语句向右做成阶梯形式，可使程序的逻辑结构更加清晰，层次更加分明。

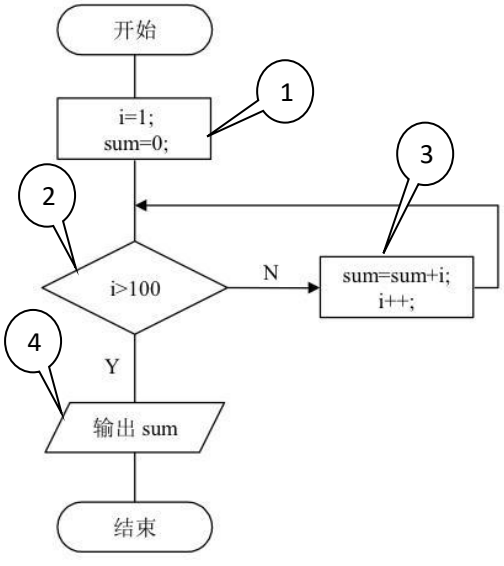
6、什么是 McCabe 度量法？

答：McCabe 度量法以图论为工具，先画出流图，然后用该图的环路数作为程序复杂性的度量值。也就是说，把程序流程图的每一个处理符号都退化成一个结点，原来连接不同处理符号的流线变成连接不同结点的有向弧，这样得到的有向图就叫做流图。

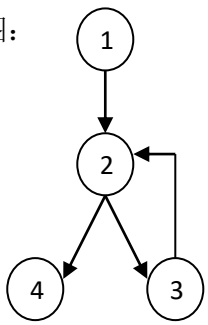
7、采用 McCabe 度量法进行程序效率分析，首先画出习题 3 第 22 题程序流程图的流图，然后计算其环形复杂度。

答：

标号：



流图：

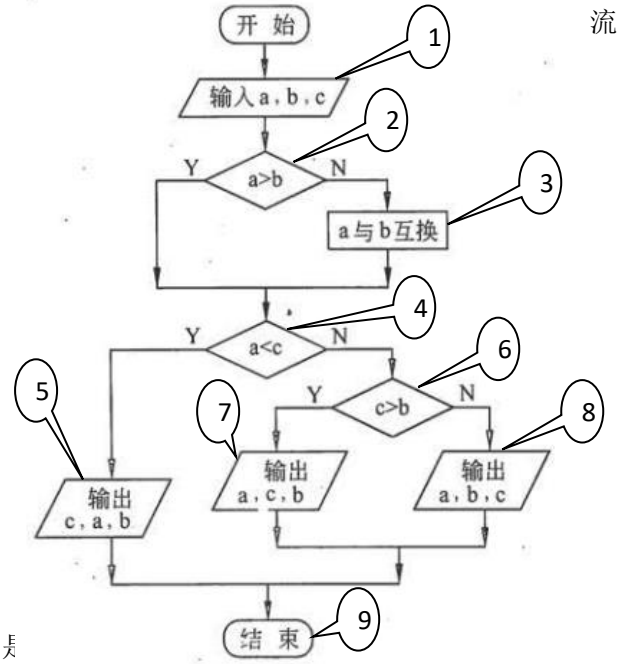


环形复杂度为 2。

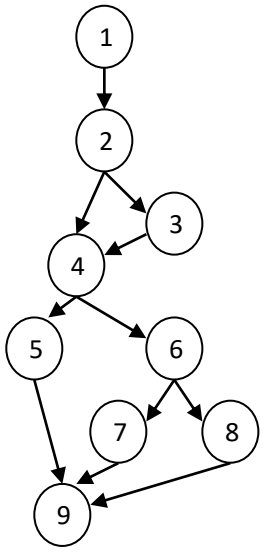
8、采用 McCabe 度量法进行程序效率分析，首先画出习题 3 第 23 题程序流程图的控制流图，然后计算其环形复杂度。

答：

标号：



流图：



环形复杂度为 4。

9、什么是

答：软件测试是为了发现错误而执行程序的过程。或者说，软件测试是根据软件开发各阶段的规格说明和程序的内部结构而精心设计一批测试用例（即输入数据及其预期的输出结果），并利用这些测试用例去运行程序，以发现程序错误的过程。

10、简述软件测试目的。

答：（1）测试是程序的执行过程，目的在于发现错误；

（2）一个好的测试用例在于能发现至今未发现的错误；

（3）一个成功的测试是发现了至今未发现的错误的测试。

11、简述软件测试的原则。

答：（1）应当把“尽早地和不断地进行软件测试”作为软件开发者的座右铭。

（2）测试用例应由测试输入数据和与之对应的预期输出结果这两部分组成。

（3）应由第三方人员从事测试工作。

（4）在设计测试用例时，应当包括合理的输入条件和不合理的输入条件。

（5）注意测试中的错误群集现象。

（6）严格执行测试计划，排除测试的随意性。

（7）妥善保存测试计划、测试用例、出错统计和最终分析报告。

12、什么是白盒测试？白盒测试的目的是什么？

答：白盒测试是一种测试方法，盒子指的是被测试的软件，白盒指的是盒子内部是可见的，即清楚盒子内部的东西以及里面是如何运作的。在系统中出现一个缺陷往往不是由一个原因导致的，就需要通过白盒测试，提前把每个功能模块都测试一次。

白盒测试的目的一般包括：

（1）保证程序中所有关键路径都被测试到，防止系统投入使用后用户发现系统问题。

（2）便于衡量测试的完整性，即有没有把某个功能点的所有可能情况都测试到。

（3）可以测试到程序中所有的真分支、假分支。

（4）检查局部数据结构的有效性。

（5）检查程序的异常处理能力。

（6）检查代码是否遵循编码规范。

13、逻辑覆盖技术一般分为哪几种？

答：根据覆盖目标的不同，逻辑覆盖技术可分为语句覆盖、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，一般来说，发现错误的能力成由弱至强的变化。

14、什么是语句覆盖、判定覆盖、条件覆盖、判定-条件覆盖、条件组合覆盖、路径覆盖？

答：语句覆盖就是设计若干个测试用例，运行被测程序，使得每一可执行语句至少执行一次。

判定覆盖就是设计若干个测试用例，运行被测程序，使得程序中每个判断的取真分支和取假分支至少经历一次。判定覆盖又称为分支覆盖。

条件覆盖就是设计若干个测试用例，运行被测程序，使得程序中每个判断的每个条件的可能取值至少执行一次。

所谓判定-条件覆盖就是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执行一次，同时每个判断本身的所有可能判断结果至少执行一次。

条件组合覆盖就是设计足够的测试用例，运行被测程序，使得每个判断的所有可能的条件取值组合至少执行一次。

路径覆盖是设计足够的测试用例，覆盖程序中所有可能的路径。

15、什么是基本路径测试?

答：基本路径测试是在程序控制流图的基础上，通过分析控制结构的环路复杂性，导出基本可执行路径集合，从而设计测试用例的方法。设计出的测试用例要保证在测试中程序的每一个可执行语句至少执行一次。

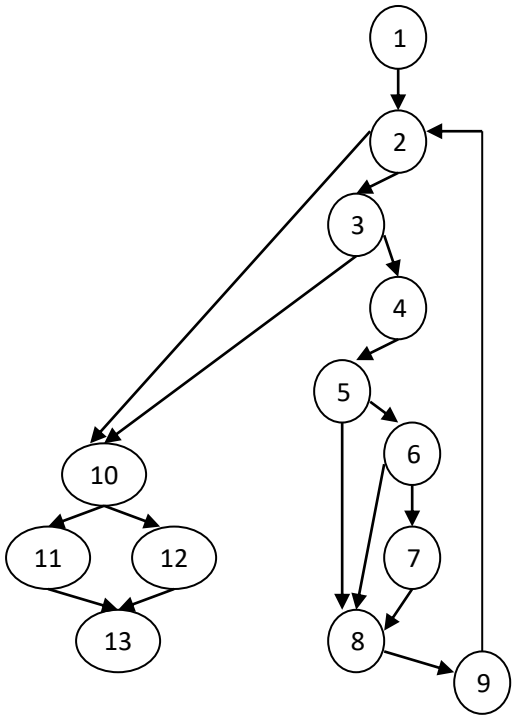
16、为程序流程图如图 4-36 设计满足判定-条件覆盖的测试用例。

答：条件有 4 个，取值分别设为（x<4：T1F1），（y>0：T2F2），（x>=5：T3F3）（y>1：T2F4）；3 个判定的编号分别设为（x<4 or y>0：1 号，x>=5：2 号，y>1：3 号）。

测试用例：（3,1）：(3,1)，覆盖 T1T2F4，1 号判定为真，3 号判定为假；（4,2）：(4,3)，覆盖 T1T2T4，1 号判定为真，3 号判定为真；（5,0）：(5,0)，覆盖 F1F2T3，1 号判定为假，2 号判定为真；（4,0）：(4,0)，覆盖 F1F2F3，1 号判定为假，2 号判定为假。

17、以下 PDL 程序段中已经对将要映射为对应控制流图中一个结点的语句或语句组加上了数字标号。请画出对应控制流图，完成基本路径测试。

答：



基本路径共 6 条。第一条：1，2，10，11，13
第二条：1，2，10，12，13
第三条：1，2，3，10，11，13
第四条：1，2，3，4，5，8，9，2，……
第五条：1，2，3，4，5，6，8，9，2，……
第六条：1，2，3，4，5，6，7，8，9，2，……
对应的 6 组测试用例：设 minimum=10，maximum=100。
第一条路经不可达，因为这样的独立路径并不完全孤立，是正常控制流的一部分，所以可以做为下面的某个路径测的一部分；
第二条：value[1]= -999，total.input=0，averagy= -999；
第三条路经不可达，测试方法同第一条；
第四条：value[1]= 9，total.input=0，averagy= -999；
第五条：value[1]= 101，total.input=0，averagy= -999；
第六条测试用例 1：value[1]= 100，total.input=0，value[2]= 10，value[3]= -999，averagy=55，同时也测试到了第一条路经。
第六条测试用例 2：value[1]到 value[100]都不等于-999，averagy 为所有有效值的平均值，同时也测试到了第三条路经。
18、程序流程图如图所示，已经对将要映射为对应控制流图中一个结点的语句或语句组加上了数字标号。试完成基本路径测试。
答：此流程图完成的程序功能与 17 题程序段基本相同，控制流图和路径也基本相同，所以可参照 17 题答案。
19、什么是黑盒测试?
答：黑盒测试也称功能测试，它是通过测试来检测每个功能是否都能正常使用。它把测试对象看做一个黑盒子，测试人员完全不考虑程序内部的逻辑结构和内部特性，只依据程序的需求规格说明书，检查程序的功能是否符合它的功能说明。
20、解释等价类划分法、边界值分析法、错误推测法、因果图法。
答：等价类划分法是一种典型的黑盒测试方法，是把程序的输入分为若干类（子集），然后每一类中选取少数代表性数据作为测试用例，每一类的代表性数据在测试中的作用等价于这一类中的其他值。
边界是指，相对于输入等价类和输出等价类而言，稍高于其边界值及稍低于其边界值的一些特定情况，所以边界值分析是通过选择等价类边界的测试用例进行黑盒测试的。边界值分析法不仅重视输入条件边界，而且也必须考虑输出域边界。
错误推测法是基于经验和直觉推测程序中所有可能存在的各种缺陷和错误，从而有针对性的设计测试用例的方法。
因果图是一种逻辑模型，非常适合描述各种输入条件的组合产生的输出结果，清晰明了，容易理解。因果图法的黑盒测试最终生成的就是判定表，它着重分析输入条件的各种组合，每种组合条件就是“因”，它必然有一个输出的结果，这就是“果”。
21、假设某网站用户注册时要求输入的年龄是 18~60 之间的整数，试采用等价类划分和边界值分析的混合方法进行黑盒测试，并设计相应的测试用例。
答：等价类划分：

输入数据	有效等价类	无效等价类

年龄输入的范围	18~60 之间的整数（1）	小于 18 的整数（2）大于 60 的整数（3）
		18~60 之间的实数（4）

测试用例：

测试用例	覆盖的等价类
18	（1）
60	（1）
17	（2）
61	（3）
18.5	（4）

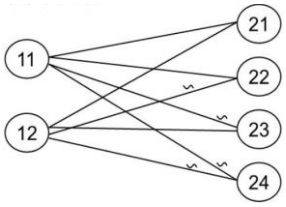
22、某软件的一个模块的需求规格说明书中描述：（1）年薪制员工：严重过失，扣年终风险金的 4%；过失，扣年终风险金的 2%；（2）非年薪制员工：严重过失，扣当月薪资的 8%；过失，扣当月薪资的 2%。请绘制出因果图和判定表，并给出相应的测试用例。

答：分析需求规格说明书：

原因：11、是否年薪制员工 12、是否严重过失

结果：21、扣年终风险金的 4% 22、扣年终风险金的 2% 23、扣当月薪资的 8% 24、扣当月薪资的 2%

画因果图：



判定表：

	原因		最终结果			
编号	11	12	21	22	23	24
1	1	1	T	F	F	F
2	1	0	F	T	F	F
3	0	1	F	F	T	F
4	0	0	F	F	F	T

测试用例：

编号	输入数据	预期结果
1	年薪制员工，严重过失	扣年终风险金的 4%
2	年薪制员工，过失	扣年终风险金的 2%
3	非年薪制员工，严重过失	扣当月薪资的 8%
4	非年薪制员工，过失	扣当月薪资的 2%

23、略。

24、什么是单元测试？其内容包括哪些？

答：单元测试又称为模块测试，是针对软件设计的最小单位—程序模块，进行正确性检验的测试工作，其目的在于发现各模块内部可能存在的各种差错。它需要从程序内部结构出发设计测试用例，多个模块可以平行地独立进行。

在单元测试中需要在 5 个方面对被测模块进行检查。

1、模块接口测试 2、局部数据结构测试 3、重要路径测试 4、错误处理测试 5、边界测试

25、什么是驱动模块？什么是桩模块？

答：（1）驱动模块：相当于被测模块的主程序，它接收测试数据，并把这些数据传送给被测模块，最后再输出实测结果。

（2）桩模块：也叫做存根模块，用以代替被测模块调用的子模块。桩模块可以做少量的数据操作，不需要把子模块所有功能都带进来，但不允许什么事情也不做。

26、什么是集成测试？什么是系统测试？什么是验收测试？

答：集成测试是单元测试的逻辑扩展，也叫做组装测试，是测试和组装软件的系统化技术。集成测试采用的方法是测试软件单元的组合能否正常工作，以及与其他组的模块能否集成起来工作。最后，还要测试构成系统的所有模块组合能否正常工作。

系统测试是将已经集成好的软件系统，作为整个基于计算机系统的一个元素，与计算机硬件、外设、某些支持软件、数据和人员等其他系统元素结合在一起，在实际运行（使用）的环境下，对计算机系统进行一系列的组装测试和确认测试。

验收测试，就是相关的用户和独立测试人员根据测试计划和结果对系统进行测试和接收，它让系统用户决定是否接收系统，它是一项确定产品是否能够满足合同或用户所规定需求的测试，属于管理性和防御性控制。

27、什么是 Alpha 测试？有哪些优缺点？

答：Alpha 测试，即 α 测试，是由某个用户在开发环境下进行的测试，也可以是公司内部的用户在模拟实际操作环境下进行的测试。软件在一个自然设置状态下使用，开发者坐在用户旁边，随时记下错误情况和使用中的问题。α测试的目的是评价软件产品的功能、局域化、可使用性、可靠性、性能和支持，尤其注重产品的界面和特色。

优点：

（1）要测试的功能和特性都是已知的。

（2）可以对测试过程进行评测和监测。

（3）可接受性标准是已知的。

（4）与正式验收测试相比，可以发现更多由于主观原因造成的缺陷。

缺点：

（1）要求资源和计划。

（2）无法控制所使用的测试用例。

（3）最终用户可能沿用系统工作的方式，并可能无法发现缺陷。

（4）最终用户可能专注于比较新系统与遗留系统，而不是专注于查找缺陷。

（5）用于验收测试的资源不受项目的控制，并且可能受到压缩。

28、什么是 Beta 测试？有哪些优缺点？

答：Beta 测试，即β测试，是由软件的多个用户在一个或多个用户的实际使用环境下进行的测试。这些用户是与公司签定了支持产品预发行合同的外部客户。与α测试不同的是，开发者通常不在测试现场，由用户记下遇到的所有问题。

优点：

（1）测试由最终用户实施。

（2）大量的潜在测试资源。

（3）提高客户对参与人员的满意程度。

（4）与正式或非正式验收测试相比，可以发现更多由于主观原因造成的缺陷。

缺点：

（1）未对所有功能和/或特性进行测试。

（2）测试流程难以评测。

（3）最终用户可能沿用系统工作的方式，并可能没有发现缺陷。

（4）最终用户可能专注于比较新系统与遗留系统，而不是专注于查找缺陷。

（5）用于验收测试的资源不受项目的控制，并且可能受到压缩。

（6）可接受性标准是未知的。

（7）您需要更多辅助性资源来管理β测试员。

习题 5

1、名词解释：对象、类、继承性、封装性、多态性、面向对象、UML。

答：对象是要研究的任何事物。

类是对象的模板，即类是对一组有相同数据和相同操作的对象的定义，一个类所包含的数据和方法描述一组对象的共同属性和行为。

继承性是子类自动共享父类中数据和方法的机制，它由类的派生功能体现。一个类直接继承其父类类的全部描述，同时可修改和扩充。

封装使数据和加工该数据的方法封装为一个整体，以实现独立性很强的模块，使得用户只能见到对象的外特性，而对象的内特性对用户是隐蔽的。

多态性是指相同的操作可作用于多种类型的对象上并获得不同的结果。不同的对象收到同一消息可以产生不同的结果，这种现象称为多态性。

所谓面向对象(Object–Oriented，OO)就是基于对象概念，以对象为中心，以类和继承为构造机制，来认识、理解、刻画客观世界和设计、构建相应的软件系统。

UML 是一个支持模型化和软件系统开发的图形化语言，为软件开发的所有阶段提供模型化和可视化支持，包括由需求分析到规格到构造和配置。

2、简述 UML 的特点。

答：(1)UML 支持面向对象技术的主要概念，提供了一批基本的模型元素的表示图形和方法，能简洁明了地表达面向对象的各种概念，而且容易掌握和使用。

(2)UML 统一了各种方法对不同类型的系统、不同开发阶段以及不同内部概念的不同观点，从而有效的消除了各种建模语言之间不必要的差异。

(3)UML 建模能力比其它面向对象建模方法更强。

(4)UML 是一种建模语言，而不是一个开发过程。

(5)用 UML 建立的软件系统模型可以用 Java、VC++、dephi 等任何一种面向对象的程序设计来实现。

3、简述 UML 的基本组成。

答：1）UML 基本构造块，有 3 种：事物、关系和图。

2)UML 规则，不能简单地把 UML 的构造块随机地放在一起。像任何语言一样，UML 有一套规则，这些规则描述了一个结构良好的模型看起来应该像什么。

3)UML 公共机制，有 4 种贯穿整个语言且一致应用的公共机制。

4、面向对象的分析模型一般包括哪 3 大类?

答：面向对象的分析模型一般有 3 大类：用例模型、对象模型和交互模型。

5、什么是用例图?

答：用例图是从用户的观点描述系统的功能，它由一组用例、参与者以及它们之间关系所组成。参与

者（Actor）是与系统交互的外部实体，它既可以是使用该系统的用户，也可以是系统交互的其他外部系统、硬件设备或组织机构。用例（Use Case）是从用户角度描述系统的行为，它将系统的一个功能描述成一系列事件，这些事件最终对参与者产生有价值的可观测结果。

6、简述用例图中《communicate》关系、《include》关系、《extend》关系和泛化关系。

答：参与者和用例之间通过无方向的直线相连，并通过《communicate》关系进行通信。

用例和用例之间可能存在包含关系，表示一个用例所执行的功能中总是包括被包含用例的功能，通过有向直线将具有包含关系的两用例相连，箭头指向被包含用例，并在直线上标明《include》关系。

用例和用例之间还可能存在扩展关系，表示一个用例的执行可能需要有其他用例的功能来扩展，但基本用例的功能并不依赖于扩展用例，通过有向直线将具有扩展关系的两用例相连，箭头指向基本用例，并在直线上标明《extend》关系。

参与者和参与者以及用例和用例之间可能存在泛化关系（也称为继承关系），泛化关系用带空心箭头的直线表示，箭头指向被继承方。

7、什么是对象模型？

答：对象模型是模型的静态结构，用于表示软件要处理的数据，在 UML 中表示为类图。类图中包括类、类的内部结构以及类与类之间的关系。

8、什么是聚合关系？什么是关联关系？

答：聚合关系即类与类中对象之间的组合关系，如果发现一个类中对象都是由另一个类中多个对象组合而成，那么这两个类就具有聚合关系。

类之间的关系绝大部分都是关联关系，表示类中具体对象之间的联系，具有多重性。

9、简单解释交互模型中的顺序图、状态图和活动图。

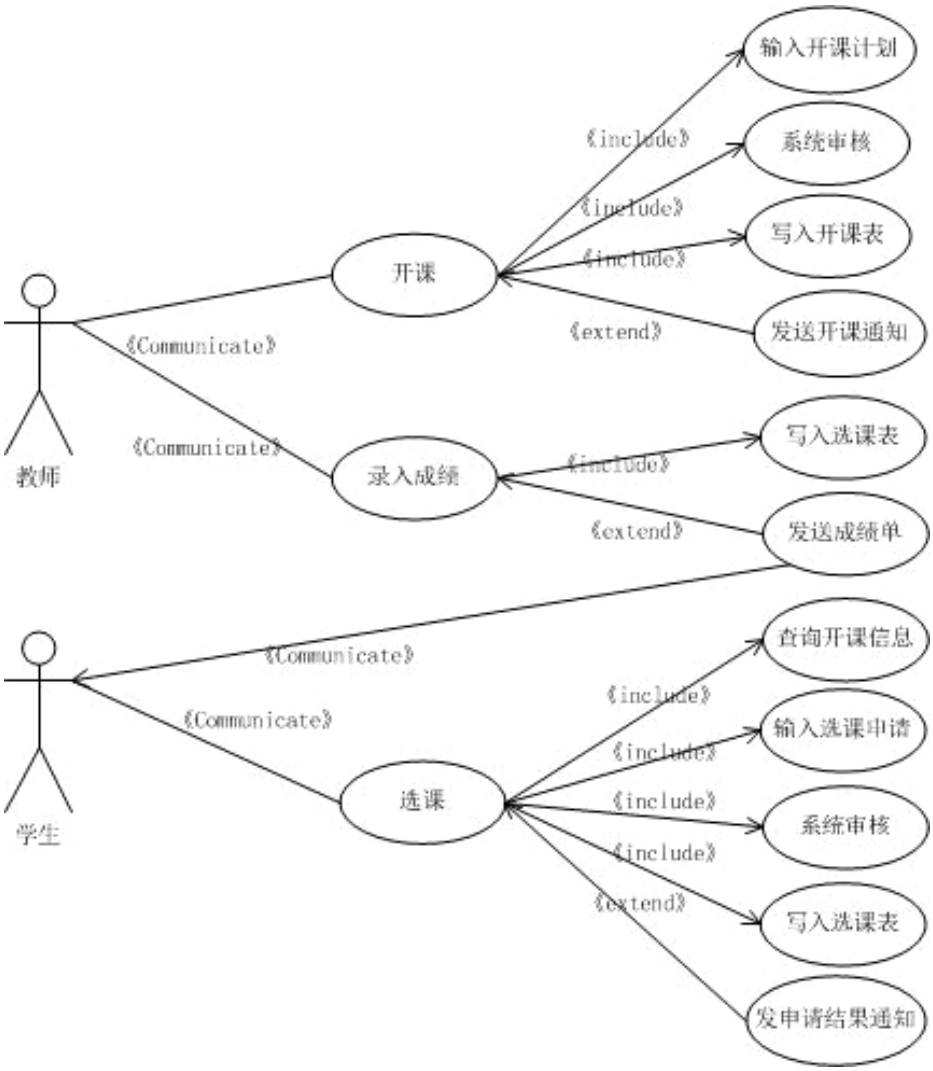
答：顺序图可以以图形的形式描绘这种场景，其中既具有交互的时间顺序又包含了参与交互的类的对象以及完成功能细节时对象之间要交互的信息，另外这些类的对象还包含用户的某些操作界面，其实例为表单对象（Form），以及系统服务器对象（server）。

状态图描述的是每一类对象的动态行为。状态图描绘的是事件与对象状态的关系，当对象接收到一个事件后，其状态会发生改变。

活动图是一个流图，描述了从活动到活动的流。它是另一种描述交互的方式，它描述采取何种动作，动作的结果是什么(动作状态改变)，何时发生(动作序列)，以及在何处发生(泳道)。

10、某学生选课系统的功能要求：教师提出开课计划，系统批准并写入开课表后给教师下发开课通知；学生可通过系统查询开课信息并提出选课申清，系统批准并写入选课表后给学生下发选课申请结果通知；课程结束后，教师可以录入学生成绩，同时把成绩单发送给学生。根据以上功能要求画出参与者学生和教师的用例图。

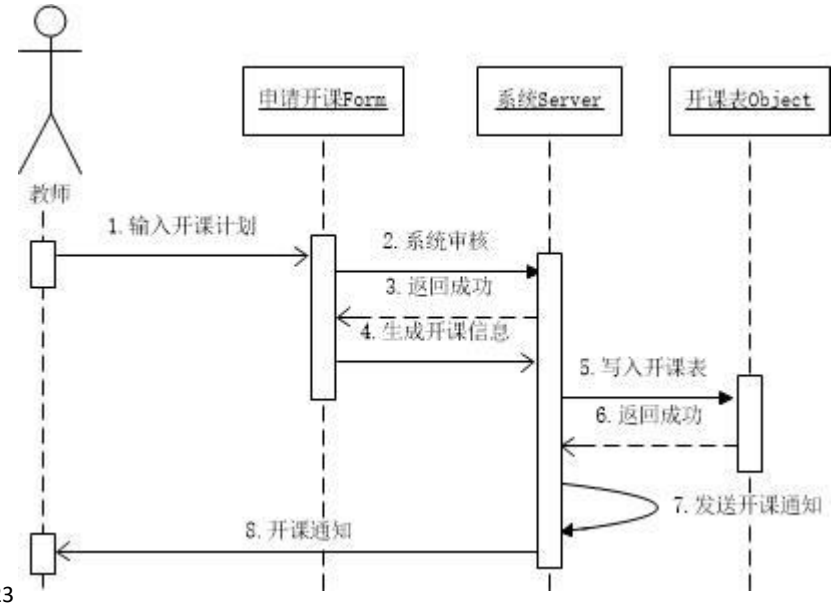
答：



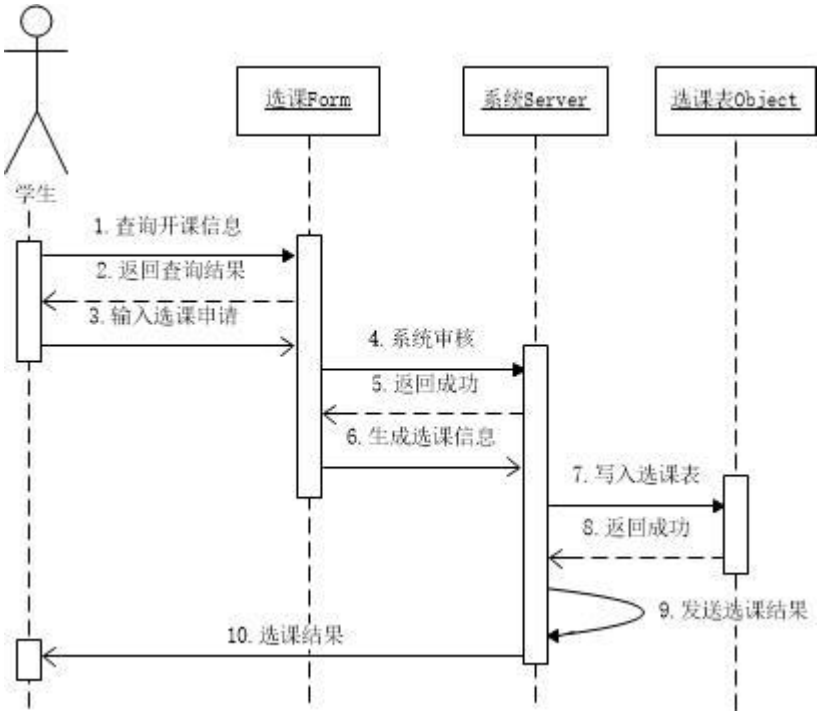
11、分别画出第 10 题教师开课申请、学生选课申请用例的顺序图和活动图。

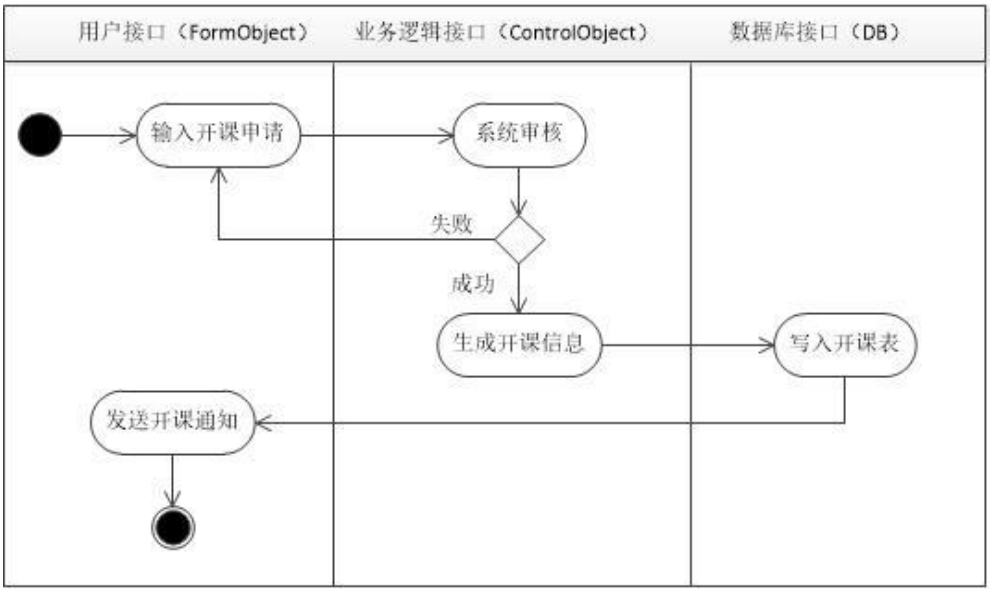
答：

顺序图：（1）教师开课申请



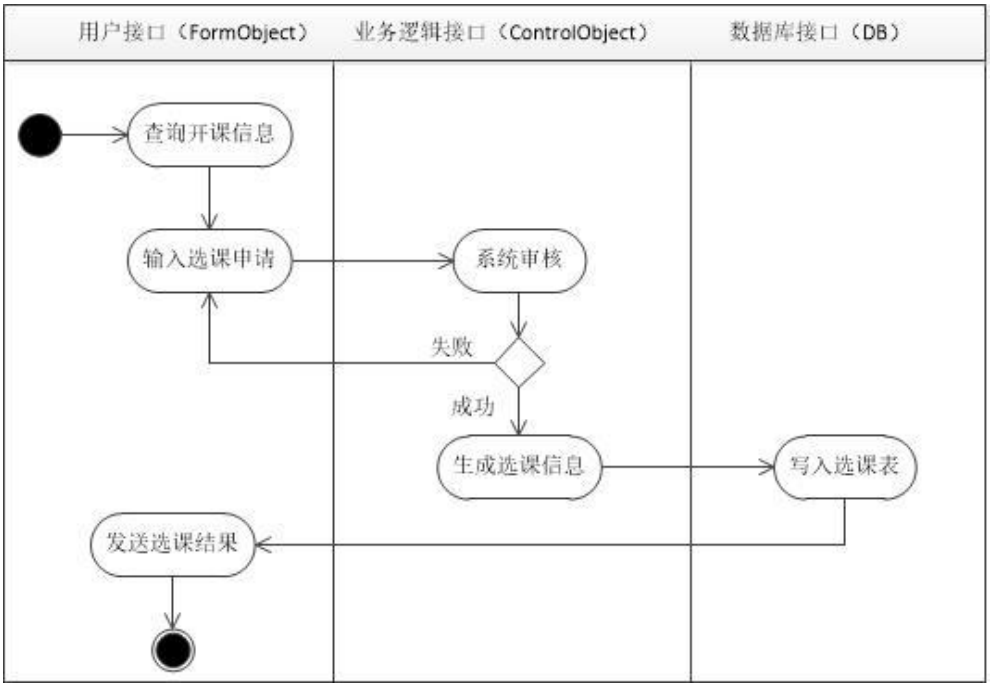
（2）学生选课





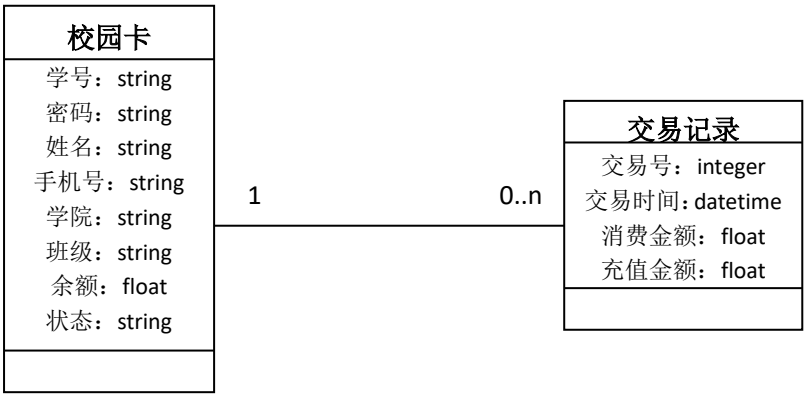
活动图：（1）教师开课申请

（2）学生选课



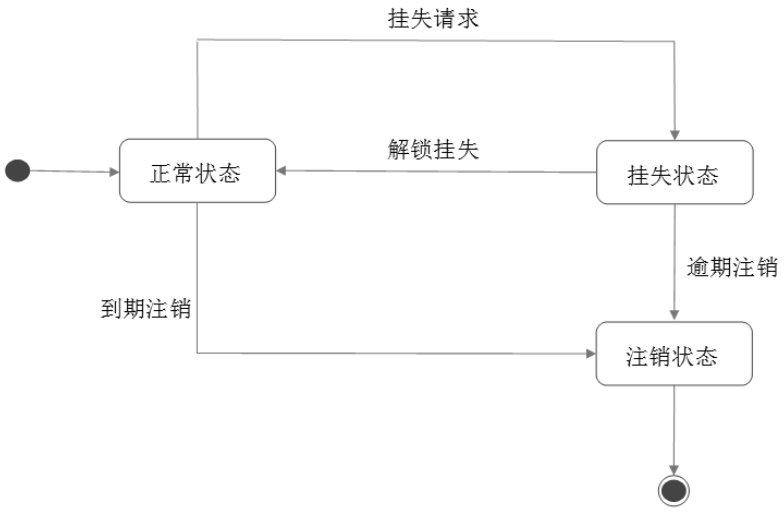
12、高校学生一卡通管理系统可以完成充值、消费、挂失、注销等功能，其中校园卡对象的属性包括学号、密码、姓名、手机号、学院、班级、余额和卡状态（正常、挂失冻结、注销），交易记录对象的属性包括交易号、交易时间、消费金额、充值金额，试画出此系统的类图。

答：



13、画出第 12 题校园卡对象的状态图。

答：



14、略。

习题 6

1、简述面向对象的设计准则。

答：（1）模块化（2）抽象（3）信息隐藏（4）低耦合（5）强内聚（6）可重用性

2、什么是信息隐藏？

答：对于类对象的用户来说，类对象中属性的表示方法和操作的实现算法是隐藏的，他们只能通过类对象的接口访问其属性和操作。

3、简述面向对象的设计步骤。

答：（1）设计系统的环境模型（2）设计系统的体系结构（3）设计各子系统（4）对象的设计

4、简述管道与过滤器风格的优缺点。

答：优点：使得构件具有良好的隐蔽性和高内聚、低耦合的特点；允许设计师将整个系统的输入/输出行为看成是多个过滤器的行为的简单合成；支持软件重用；系统维护和增强系统性能简单；允许对一些如吞吐量、死锁等属性的分析；支持并行执行。

缺点：通常导致进程成为批处理结构；不适合处理交互的应用；因为在数据传输上没有通用的标准，每个过滤器都增加了解析和合成数据的工作，这样就导致了系统性能下降，并增加了编写过滤器的复杂性。

5、简述数据抽象与面向对象风格的优缺点。

6、黑板系统主要由哪三部分组成？

答：(1)知识源，包含独立的、与应用程序相关的知识，知识源之间不直接进行通讯，它们之间的交互只通过黑板来完成。(2)黑板数据结构，按照与应用程序相关的层次来组织的解决问题的数据，知识源通过不断地改变黑板数据来解决问题。(3)控制，完全由黑板的状态驱动，黑板状态的改变决定使用的特定知识。

7、简述黑板系统的优缺点。

答：优点：控制算法和中心存储库严格分离，具有可更改性和可维护性；知识源具有可复用性；容忍噪声和不确定性结论，具有容错性。

缺点：算法具有不确定性，所以测试困难；成本高、效率低；不能保证有好的解决方案；难以建立好的控制策略；不支持并行机制。

8、简述什么是三层 C/S 结构。

答：应用系统一般划分为三层：（1）数据管理层，实现数据库安全性的要求；数据库访问并发性的控制；数据库前端的客户应用程序的全局数据完整性规则；数据库的备份与恢复。（2）应用逻辑层，实现与业务相关的处理逻辑。（3）表示层，提供用户与数据库交互的界面，向数据库服务器提交用户请求并接收来自数据库服务器的信息。

三层 C/S 结构中，服务器端实现数据管理层，应用服务器实现应用逻辑层，客户机应用程序实现表示层。

9、简述 B/S 结构的优缺点。

答：优点：B/S 结构的“零客户端”方式，使组织的供应商和客户的计算机方便地成为管理信息系统的客户端，进而在限定的功能范围内查询组织相关信息，完成与组织的各种业务往来的数据交换和处理工作，扩大了组织计算机应用系统的功能覆盖范围，可以更加充分利用网络上的各种资源，同时应用程序维护的工作量也大大减少。

缺点：B/S 体系结构缺乏对动态页面的支持能力，没有集成有效的数据库处理功能；B/S 体系结构的系统扩展能力差，安全性难以控制；采用 B/S 体系结构的应用系统，在数据查询等响应速度上，要远远低于 C/S 体系结构；B/S 体系结构的数据提交一般以页面为单位，数据的动态交互性不强，不利于在线事务处理。

10、设计任务管理子系统时，常见的任务有哪些？

答：常见的任务有事件驱动型任务、时钟驱动型任务、优先任务、关键任务和协调任务等。

11、简述关系数据库管理系统的优缺点。

答：优点：(1) 提供了各种最基本的数据管理功能。

(2) 为多种应用提供了一致的接口。

(3) 标准化的语言(大多数商品化关系数据库管理系统都使用 SQL 语言)。

缺点：(1) 运行开销大。

(2) 不能满足高级应用的需求。

(3) 与程序设计语言的连接不自然。

12、面向对象数据库管理系统有哪两种设计途径？

答：(1)扩展的关系数据库管理系统

(2) 扩展的面向对象程序设计语言

13、一般在什么情况下一个普通的类可以采用横切的方法映射为多个表？

答：横切常常用于记录与时间相关的对象，如成绩记录、运行记录等。由于一段时间后，这些对象很少被查看，所以往往在主表中只记录最近的对象，而将以前的记录转到对应的历史表中。

14、一般在什么情况下一个普通的类可以采用竖切的方法映射为多个表？

答：竖切常用于实例较少而属性很多的对象，一般是现实中的事物，将不同分类的属性映射成不同的表。通常将经常使用的属性放在主表中，而将其他一些次要的属性放到其他表中。

15、简述一对一关联的映射方法。

答：一对一关联的映射：对于一对一关联，可以在两个表中都引入对方的主键作为外键，这样两个表之间可以进行双向导航。也可以只在其中一个表中引入另一方的主键作为外键，进行单向导航。如果两个类中属性不多，而且经常组合使用，也可以将两个类组合成一张表。根据程序功能需求，还可以把关联映射为一张独立的表，表中包含两个类的主键。

16、简述一对多关联的映射方法。

答：一对多关联的映射：可以在关联中“多”端上的类映射的表加入一端的主键作为外键。也可以根据程序功能需求，把关联映射为一张独立的表，表中包含两个类的主键。

17、简述多对多关联的映射方法。

答：多对多关联的映射：由于记录的一个外键最多只能引用另一条记录的一个主键值，因此关系数据库模型不能在表之间直接维护一个多对多联系。为了表示多对多关联，关系模型必须引入一个关联表，将两个类之间的多对多关联转换成表上的两个一对多关联。

18、通常，继承关系的映射有哪三种方法？

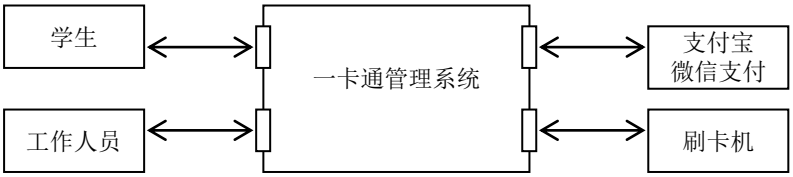
答：（1）将基类和所有子类映射到一张表，表中包含基类和子类的所有属性。这种方法适用于子类的个数很少（一般只有两个），子类和基类的属性都比较少的情况。

（2）将每个子类映射到一张表，没有基类表。在每个子类的表中包括基类的所有属性。这种方法适用于子类的个数不多，基类属性比较少的情况。

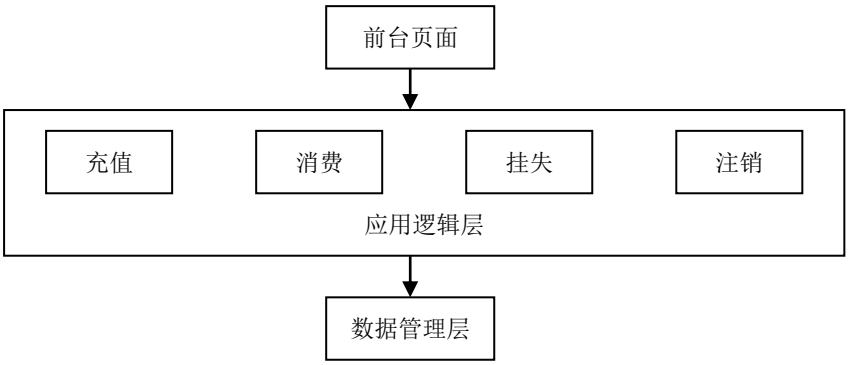
（3）将基类映射到一张表，每个子类映射到一张表。在基类对应的表中定义主键，而在子类定义的表中定义外键。这种方法适用于子类的属性和基类的属性都比较多的情况。

19、设计习题 5 中第 12 题的环境模型和体系结构。

答：环境模型：



体系结构：B/S 或 C/S 结构



20、设计习题 5 中第 12 题的关系数据模型。

答：校园卡表：T_Card

字段	含义	数据类型	键值	取值范围	允许空否
SNO	学号	Char(10)	主键	数字	否
SName	性名	Nvarchar(10)			否
SPassword	密码	char(6)		数字	否
STelephone	手机号	Char(11)		数字	否
SDepartment	学院	Varchar(30)	索引		否
SClass	班级	Varchar(30)	索引		否
Balance	余额	Float			否
CState	卡状态	Nvarchar(10)			否

此表可以根据时间横切为两个表。

交易记录表：T_Record

字段	含义	数据类型	键值	取值范围	允许空否
TNO	交易号	Int	主键	数字	否
TTime	交易时间	Datetime			否
C_Amount	消费金额	Float			是
R_ Amount	充值金额	Float			是
SNO	学号	Char(10)	外键	数字	否

此表也可以根据时间横切为两个表。

21、略。

习题 7

1、简述面向对象语言的技术特点。

答：1）. 具有支持类和对象概念的定义与实现机制

2）. 具有实现继承的语言机制

3）. 具有实现属性和服务的机制

4）. 具有参数化类

5）. 提供类型检查

6）. 提供类库

7）. 提供持久对象的保存

8）. 提供封装

9）. 提供可视化开发环境

2、简述弱类型语言和强类型语言的区别。

答：弱类型语言，仅要求每个变量或属性隶属于一个对象；强类型语言，要求每个变量或属性必须准确地属于某个特定的类。通常使用强类型编译型语言开发软件产品，弱类型用于解释型语言快速开发原型。

3、强类型语言有哪些优点？

答：两个优点：一是有利于在编译时发现程序错误，有助于提高软件的可靠性和运行效率；二是增加了优化的可能性。

4、简述面向对象设计选择编程语言的关键因素。

答：选择编程语言的关键因素是语言的一致表达能力、可重用性和可维护性。

（1）一致的表示方法：在软件开发从分析到实现的全过程中始终保持表示方法稳定不变，既有利于在软件开发过程中使用统一的概念，又有利于维护人员理解软件的各种配置成分。

（2）可重用性：在软件开发从分析到实现的过程中始终显示地表示问题域语义，这样既可能重用面向对象分析结果，也可能重用相应的面向对象设计结果和程序设计结果。

（3）可维护性：维护人员面对的往往主要是源程序，如果程序能显示地表达问题域语义，对维护人员理解待维护的软件有很大帮助。

5、面向对象设计有哪两种代码重用？

答：代码重用有两种：本项目内的代码重用和新项目重用旧项目的代码。内部重用一般使用继承机制，外部重用即一个项目重用另一个项目的代码，要考虑的面比较广，需要有长远的眼光、反复考虑、精心设计。

6、简述提高代码重用性程序设计准则。

答：（1）提高方法的内聚

（2）减小方法的规模

（3）保持方法的一致

（4）把策略与实现分开

（5）全面覆盖

（6）尽量不使用全局信息

（7）利用继承机制

（8）使用委托机制

7、什么是策略类方法？什么是实现类方法？

答：策略方法负责做出决策、提供变元和管理全局资源，这种方法应该检查系统运行状态，并处理出错情况，它们并不直接完成计算或实现复杂的算法。这种方法紧密依赖于具体应用，比较容易编写、易于理解。

实现方法负责完成具体操作，并不决定是否执行此操作，也不知道为什么执行此操作，即不制定决策也不管理全局资源。此方法仅仅针对具体数据完成特定处理，通常用于实现复杂的算法。在执行过程中如果发现错误，它们只返回执行状态而不对错误采取行动。这种方法相对独立于具体应用，在其他系统中也可以重用。

8、简述继承和委托机制的区别。

答：通过下面的例子来理解委托和继承的区别：水果需要工厂进行榨汁，而水果本身不具有榨汁的功能，将榨汁的行为写给水果也不合适，于是我们把水果交给工厂（类）中的方法进行处理，这是委托（delegation）；水果具有下落的行为，苹果也有下落的行为，在水果（类）中写下落的行为，然后在苹果（类）中通过 extends 获得这个方法的复用，这是继承（inheritance）。

9、简述提高面向对象程序设计的可扩充性的准则。

答：（1）封装类的实现策略

（2）减少一个方法访问对象的数量

（3）不使用多分支语句

（4）精心选择和定义公有方法

10、简述提高面向对象程序设计的健壮性的准则。

答：　（1）能预期用户的错误操作

（2）能检查参数是否合法

（3）动态分配数据容量

（4）根据测试结果进行优化

11、简述传统的测试和面向对象的测试的异同点。

答：传统的计算机软件测试的策略是从小型测试开始，逐步走向大型测试。即我们从单元测试开始，然后逐步进入集成测试，最后是有效性和系统测试。

面向对象程序的结构不再是传统的功能模块结构，作为一个整体，原有集成测试所要求的逐步将开发的模块搭建在一起进行测试的方法已成为不可能。而且，面向对象软件抛弃了传统的开发模式，对每个开发阶段都有不同以往的要求和结果，已经不可能用功能细化的观点来检测面向对象分析和设计的结果。

12、面向对象的软件测试分为哪几种？

答：面向对象的软件测试分为：面向对象分析的测试、面向对象设计的测试、面向对象编程的测试、面向对象单元测试、面向对象集成测试和面向对象系统测试。

13、对 OOA 的测试，应从哪几个方面进行？

答：对认定的对象的测试、对认定的结构的测试、对认定的主题的测试、对定义的属性和实例关联的测试、对定义的服务和消息关联的测试。

14、对 OOD 的测试中对认定的类的测试，应从哪几个方面考虑？

答：对认定的类的测试、对构造的类层次结构的测试、对类库的支持的测试。

15、对 OOD 的测试中对构造的类层次结构的测试，应测试哪几个方面？

答：（1）类层次结构是否含盖了所有定义的类。

（2）是否能体现 OOA 中所定义的实例关联。

（3）是否能实现 OOA 中所定义的消息关联。

（4）子类是否具有父类没有的新特性。

（5）子类间的共同特性是否完全在父类中得以体现。

16、对 OOP 的测试中，如何检查数据成员是否满足数据封装的要求？

答：检查数据成员是否满足数据封装的要求，基本原则是数据成员是否被外界（数据成员所属的类或子类以外的调用）直接调用。更直观的说，当改变数据成员的结构时，是否影响了类的对外接口，是否会导致相应外界必须改动。值得注意，有时强制的类型转换会破坏数据的封装特性。

17、对 OOP 的测试中，如何测试类是否实现了要求的功能？

答：类所实现的功能，都是通过类的成员函数执行。在测试类的功能实现时，应该首先保证类成员函数的正确性。类函数成员的正确行为只是类能够实现要求的功能的基础，类成员函数间的作用和类之间的服务调用是单元测试无法确定的。因此，需要进行面向对象的集成测试。

18、测试类的成员函数，在设计测试用例选择输入数据时，可以基于哪两个假设？

答：（1）如果类成员函数对某一类输入中的一个数据正确执行，对同类中的其他输入也能正确执行。

（2）如果类成员函数对某一复杂度的输入正确执行，对更高复杂度的输入也能正确执行。

19、面向对象的单元测试中，方法层次的测试，常用的测试技术有？

答：常用的测试技术：

（1）等价类划分测试：根据输入参数把取值域分成若干个等价类。

（2）组合功能测试：针对那些依据输入参数和成员变量的不同取值组合而选择不同动作的方法。

20、面向对象的单元测试中，类层次的测试，常用的测试技术有？

答：常用的测试技术：

（1）不变式边界测试。类层次测试的一个主要困难是成员变量的某些状态可能不会出现，这就是所谓的类不变式。（2）模态类测试：模态类是指对该类所接受的成员方法的调用序列设置一定的限制。（3）非模态类测试：该类所接受的成员方法的调用序列没有任何限制。

21、面向对象的单元测试中，类树层次的测试，常用的测试技术有？

答：常用的测试技术：

（1）多态服务测试：多态服务测试是为了测试子类中的多态方法的实现是否保持了父类对该方法的规格说明。

（2）展平测试：将子类自身定义的成员方法和成员变量以及从父类和祖先类继承来的成员方法和成员变量全部放在一起组成一个新类，并对其进行测试。

22、面向对象程序集成测试策略主要有哪几种？

答：（1）协作集成（2）基干集成（3）高频集成（4）基于事件（消息）的集成（5）基于使用的集成（6）客户机/服务器的集成（7）分布式集成

23、什么是类关联的多重性测试？什么是受控异常测试？什么是往返场景测试？

答：类关联的多重性测试：在面向对象中，类间的关联关系存在多重性方面的限制，对多重性的测试是针对类间连接测试的重要方面。此时，测试关注的重点是与连接关系有关的增删改操作。

受控异常测试：由于使用异常处理，异常的抛出和异常的接收可以被放在不同的类中，这实际上是类间隐含的控制依赖关系。测试时，需要尽可能地覆盖这些隐式的依赖关系。

往返场景测试：面向对象中，许多功能是通过多个类相互协作完成的，往返场景测试就是针对类间协作的一种测试技术。

24、简述传统的测试方法和面向对象的测试方法都应该遵循的原则。

答：（1）应当把“尽早和不断地测试”作为开发者的座右铭。

（2）程序员应该避免检查自己的程序，测试工作应该由独立的专业的软件测试机构来完成。

（3）设计测试用例时，应该考虑到合法的输入和不合法的输入，以及各种边界条件，特殊情况下要制造极端状态和意外状态，比如网络异常中断、电源断电等情况。

（4）一定要注意测试中的错误集中发生现象，这和程序员的编程水平和习惯有很大的关系。

（5）对测试错误结果一定要有一个确认的过程。一般由 A 测试出来的错误，一定要有一个 B 来确认，严重的错误可以召开评审会进行讨论和分析。

（6）制定严格的测试计划，并把测试时间安排得尽量宽松，不要希望在极短的时间内完成一个高水平的测试。

（7）回归测试的关联性一定要引起充分的注意，修改一个错误而引起更多错误出现的现象并不少见。

（8）妥善保存一切测试过程文档，意义是不言而喻的，测试的重现性往往要靠测试文档。

习题 8

1、什么是软件维护？一般有哪四种维护活动？

答：软件维护就是在软件已经交付使用之后，为了改正错误、提高性能或满足新的需要而修改软件的过程。4 种维护活动包括改正性维护、适应性维护、完善性维护和预防性维护。

2、名词解释：改正性维护，适应性维护，完善性维护，预防性维护。

答：改正性维护是为了识别和纠正软件错误、改正软件性能上的缺陷、排除实施中的误使用，应进行的诊断和改正错误的过程。

适应性维护，是为了适应环境的变化而修改软件的活动，是既必要又经常的维护活动。

为了满足新的功能与性能要求，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叫做完善性维护。

为了改进未来的可维护性或可靠性，为了适应未来的软硬件环境的变化，或为了给未来的改进奠定更好的基础而修改软件，这种维护活动通常称为预防性维护。

3、对于适应性维护，可以采用哪策略加以控制？

答：（1）在配置管理时，把硬件、操作系统和其他相关环境因素的可能变化考虑在内，可以减少某些适应性维护的工作量。

（2）把与硬件、操作系统以及其他外围设备有关的程序归到特定的程序模块中。可把因环境变化而必须修改的程序局部于某些程序模块之中。

（3）使用内部程序列表、外部文件以及处理的例行程序包，可为维护时修改程序提供方便。

（4）使用面向对象技术，增强软件系统的稳定性，易于修改和移植。

4、简述非结构化和结构化维护的区别。

答： 非结构化：如果软件配置的唯一成分是程序代码，那么维护活动从艰苦地评价程序代码开始，常常由于程序内部文档不足而使评价更困难，对于软件结构、全程数据结构、系统接口、性能和(或)设计约束

等经常会产生误解，而且对程序代码所做的改动的后果也是难于估量的。

结构化：如果有一个完整的软件配置存在，那么维护工作从评价设计文档开始，确定软件重要的结构、性能以及接口等特点；估量要求的改动将带来的影响，并且计划实施途径。然后首先修改设计并且对所做的修改进行仔细复查，然后编写相应的源程序代码，接下来使用在测试说明书中包含的信息进行回归测试，最后把修改后的软件再次交付使用。

5、软件维护存在哪些问题？

答：（1）理解别人写的程序通常非常困难，而且困难程度随着软件配置成分的减少而迅速增加。如果仅有程序代码没有说明文档，则会出现严重的问题。

（2）需要维护的软件往往没有合格的文档，或者文档资料显著不足。认识到软件必须有文档仅仅是第一步，容易理解的并且和程序代码完全一致的文档才真正有价值。

（3）当要求对软件进行维护时，不能指望由开发人员给人们仔细说明软件。由于维护阶段持续的时间很长，因此，当需要解释软件时，往往原来写程序的人已经不在附近了。

（4）绝大多数软件在设计时没有考虑将来的修改。除非使用强调模块独立原理的设计方法学，否则修改软件既困难又容易发生差错。

（5）软件维护不是一项吸引人的工作。形成这种观念很大程度上是因为维护工作经常遭受挫折。

6、简述软件维护过程。

答：首先必须建立一个维护组织；然后必须确定维护报告的内容，并为每个维护要求规定一个标准化的事件序列，以及建立一个适用于维护活动的维护记录保管过程并确定需要保存的记录内容；最后必须确定对软件维护的评价内容和评价过程。

7、维护组织包括哪些成员？他们是怎么配合工作的？

答：包括维护管理员、熟悉该产品的系统管理员、程序技术人员和软件变化的授权人。

每个维护要求都通过维护管理员转交给熟悉该产品的系统管理员去评价，系统管理员对维护任务做出评价之后，由软件变化的授权人决定应该进行的活动，由系统管理员指定一些程序技术人员作为维护人员进行维护工作。

8、软件维护报告包含哪些信息？

答：(1)满足维护要求表中提出的要求所需要的工作量。

(2)维护要求的性质。

(3)这项要求的优先次序。

(4)与修改有关的事后数据。

9、对于改正性维护的申请，简述其维护事件流程。

答：对于改正性维护的申请，首先要评价错误的严重性程度。如果发现严重错误，则必须马上开始分析问题，寻找错误发生的原因，并安排人员进行紧急维护；如果是不严重的错误，则制定错误改正计划，按错误的轻重缓急进行排队，并编写和存储错误改正目录，统一安排时间完成。

10、对于适应性和完善性维护的申请，简述其维护事件流程。

答：对于适应性和完善性维护的申请，首先要确定每个申请的优先级。高优先级的申请，马上开始分析问题，并安排人员进行维护；低优先级的申请，写入开发目录，和其他工作一起排队，统一安排时间完成。

11、如何定义软件的可维护性？

答：软件的可维护性可以定义为：维护人员理解、改正、改动或改进这个软件的难易程度。

12、软件可维护性的决定因素主要有哪些？

答：可理解性、可测试性、可靠性、可修改性、可移植性和可重用性。

13、哪些方法有助于提高软件的可理解性？

答： 以下方法都有助于提高软件的可理解性：（1）模块化（2）详细的设计文档（3）结构化设计方法（4）程序内部的文档（5）良好的高级程序设计语言

14、简述与软件的可测试性相关的几个方面。

答：（1）首先，软件的可测试性取决于软件容易理解的程度。

（2）良好的文档和软件结构、可用的测试工具和调试工具、以前设计的测试过程对软件的可测试性也是非常关键的。

（3）软件维护人员需要得到开发阶段用过的测试方案，以便维护人员进行回归测试。

（4）在设计阶段就应该考虑到软件的可测试性，尽力把软件设计成容易测试和容易诊断的。

（5）对于程序模块来说，可以用程序复杂度来度量它的可测试性。

15、如何度量软件的可靠性？

答：关于可靠性的度量，一般有以下两种方法：

（1）根据程序错误统计数预测软件可靠性。

（2）根据程序复杂性预测软件可靠性。

16、一般认为什么样的程序具有可修改性？

答：一个可修改的程序应该是可理解的、通用的（适用于各种功能变化，而无需修改）、简单的。

17、什么是软件的可移植性？

答：软件的可移植性是指把程序从一种计算环境（硬件配置和操作系统）转移到另一种计算环境的难易程度。

18、简述大量使用可重用的软件构件来开发软件，为什么可以提高软件的可维护性？

答：(1)通常，可重用的软件构件在开发时都经过很严格的测试，可靠性比较高，且在每次重用过程中都会发现并清除一些错误，随着时间推移，这样的构件将变成实质上无错误的构件。

(2)很容易修改可重用的软件构件使之再次应用在新环境中。

19、为什么说文档是软件可维护性的主要影响因素？

答：文档是软件可维护性的主要影响因素，对于维护人员理解软件起着至关重要的作用。即使是十分简单的软件系统，为了高效地维护，编制文档来描述其设计目的和内容也是非常必要的；而对于长期使用的大型软件系统而言，在使用过程中必然会经受多次修改，所以文档比程序代码更重要。

20、为什么要进行可维护性复审？

答：不能准确反映软件当前状态的设计文档可能比完全没有文档更坏。

如果对软件的可执行部分的修改没有及时反映在用户文档中，则必然会使用户因为受挫折而产生不满。

在软件再次交付使用之前，对软件配置进行严格的复审，则可大大减少文档的问题。

在需求分析阶段的复审过程中，应该对将来要改进的部分和可能会修改的部分加以注意并指明；应该讨论软件的可移植性问题，并且考虑可能影响软件维护的系统界面。

21、什么是软件逆向工程？

答：软件逆向工程又称软件反向工程，是指从可运行的程序系统出发，运用解密、反汇编、系统分析、程序理解等多种计算机技术，对软件的结构、流程、算法、代码等进行逆向拆解和分析，推导出软件产品的源代码、设计原理、结构、算法、处理过程、运行方法及相关文档等。

22、软件逆向工程一般应用于哪些方面？

答：（1）利用和改造现有软件形成新的软件

（2）将软件系统转化为易演化系统

（3）研究和学习优秀软件

（4）作为开放源代码的前期工程

23、软件逆向工程存在哪些问题？

答：（1）缺乏统一的逆向工程概念和术语，缺乏统一的逆向工程机制的分类框架，导致研究人员之间交流的困难，限制了逆向工程工具和技术研究的进展。

（2）逆向工程工具还不能与其他开发工具有效集成，而且缺乏对现有逆向工程工具统一的评估标准和验证工具，限制了逆向工程的应用和逆向工程技术的发展。

（3）由于受软件知识产权保护及相关法律法规的限制，软件逆向工程并不能像其它软件技术那样公开、透明地为大家所熟知、了解和广泛交流与应用。

（4）软件逆向工程所涉及到的技术很多，它不仅要求逆向工程人员必须熟悉如操作系统、汇编语言、加解密等相关知识，同时还要具有丰富的多种高级语言的编程经验，熟悉多种编译器的编译原理，较强的程序理解和逆向分析能力等，这些也都限制了软件逆向工程的发展。

24、什么是软件再工程？

答：软件再工程是指通过对目标系统的检查和改造，其中包括设计恢复(库存目录分析)、再文档、逆向工程、程序代码和数据重构以及正向工程等一系列活动，旨在将逆向工程、重构和正向工程组合起来，将现存系统重新构造为新的形式，以开发出质量更高、维护性更好的软件。

25、想要对一个老程序进行文档重构，一般需如何处理？

答：(1)因为建立文档非常耗费时间，不可能为数百个程序都重新建立文档。

(2) 为了便于今后的维护，必须建立文档，但是由于资源有限，应采用“使用时建文档”的方法。

(3) 如果某软件是完成业务工作的关键，而且必须重构全部文档，则仍然应该设法把文档工作减少到必需的最小量。

26、简述代码重构和数据重构的区别。

答：某些老程序具有比较完整、合理的体系结构，但是，个体模块的编码方式却是难以理解、测试和维护的。在这种情况下，可以重构可疑模块的代码。 与代码重构不同，数据重构发生在相当低的抽象层次上，它是一种全范围的再工程活动，即对数据的修改必然会导致体系结构或代码层的改变。

习题 9

1、简述设计模式的 4 个基本要素。

答：设计模式一般有 4 个基本要素：模式名称（pattern name），问题（problem），解决方案（solution），效果（consequences）。

2、简述简单工厂模式中的 3 种角色。

答：（1）工厂(Creator)角色：简单工厂模式的核心，它负责实现创建所有实例的内部逻辑。工厂类的创建产品类的方法可以被外界直接调用，创建所需的产品对象。

（2）抽象产品(Product)角色：简单工厂模式所创建的所有对象的父类，它负责描述所有实例所共有的公共接口。

（3）具体产品(Concrete Product)角色：是简单工厂模式的创建目标，所有创建的对象都是充当这个角色的某个具体类的实例。

3、简述工厂方法的 4 种角色。

答：（1）抽象工厂(Creator)角色：是工厂方法模式的核心，与应用程序无关。任何在模式中创建的对象

的工厂类必须实现这个接口。

（2）具体工厂(Concrete Creator)角色：这是实现抽象工厂接口的具体工厂类，包含与应用程序密切相关的逻辑，并且受到应用程序调用以创建产品对象。

（3）抽象产品(Product)角色：工厂方法模式所创建的对象的超类型，也就是产品对象的共同父类或共同拥有的接口。

（4）具体产品(Concrete Product)角色：这个角色实现了抽象产品角色所定义的接口。某具体产品有专门的具体工厂创建，它们之间往往一一对应。

4、简述工厂方法与简单工厂模式的区别。

答：工厂方法定义一个用于创建对象的接口，让子类决定实例化哪一个类，使一个类的实例化延迟到其子类。工厂方法去掉了简单工厂模式中工厂方法的静态属性，使得它可以被子类继承，这样在简单工厂模式里集中在工厂方法上的压力可以由工厂方法里不同的工厂子类来分担。

5、简述抽象工厂模式与工厂方法的区别。

答：当有多个不同的等级结构的产品时，如果使用工厂方法就势必要使用多个独立的工厂等级结构来对付这些产品的等级结构。如果这些产品等级结构是平行的，会导致多个平行的工厂等级结构。抽象工厂模式使用同一个工厂等级结构负责这些不同产品等级结构产品对象的创建。对于每一个产品族，都有一个具体工厂。而每一个具体工厂创建属于同一个产品族，但是分属于不同等级结构的产品。

6、简述抽象工厂模式的优缺点。

答：抽象工厂模式的优点是当一个产品族中的多个对象被设计成一起工作时，它能保证客户端始终只使用同一个产品族中的对象。其缺点是产品族扩展非常困难，要增加一个系列的某一产品，既要在抽象的 Creator 里加代码，又要在具体的里面加代码。

7、什么是单例模式？

答：这种模式涉及到一个单一的类，该类负责创建自己的对象，同时确保只有单个对象被创建。这个类提供了一种访问其唯一的对象的方式，可以直接访问，不需要实例化该类的对象。

单例模式的意图就是保证一个类仅有一个实例，并提供一个访问它的全局访问点，主要解决一个全局使用的类频繁地创建与销毁，可以在想控制实例数目、节省系统资源的时候使用。

8、简述单例模式的要点。

答：单例模式的要点：一是某个类只能有一个实例；二是它必须自行创建这个实例；三是它必须自行向整个系统提供这个实例。从具体实现角度来说，就是以下三点：一是单例模式的类只提供私有的构造函数；二是类定义中含有一个该类的静态私有对象；三是该类提供了一个静态的公有的方法，用于创建或获取它本身的静态私有对象。

9、简述单例模式的优缺点。

答：单例模式的优点：（1）在内存里只有一个实例，减少了内存的开销，尤其是频繁的创建和销毁实例。 （2）提供了对唯一实例的受控访问，避免对资源的多重占用。

单例模式的缺点：（1）没有接口，不能继承，因此单例类的扩展有很大的困难。 （2） 单例类的职责过重，在一定程度上违背了"单一职责原则"。

10、什么是适配器模式？

答：适配器模式就是将一个类的接口转换成客户希望的另外一个接口，使得原本由于接口不兼容而不能一起工作的那些类可以在一起工作，所以适配器模式的主要目的就是实现兼容性。

11、简述适配器模式中的角色。

答：适配器模式中的角色包括：

（1）目标接口（Target）：客户所期待的接口，目标可以是具体的或抽象的类，也可以是接口。

（2）需要适配的类（Adaptee）：需要适配的类或适配者类。

（3）适配器（Adapter）：通过包装一个需要适配的对象，把原接口转换成目标接口。

12、简述适配器模式模式适合的场景。

答：适配器模式适合下列场景：

（1）系统需要使用现有的类，而这些类的接口不符合系统的需要。

（2） 想要建立一个可以重复使用的类，用于彼此之间没有太大关联的一些类（包括一些可能在将来引进的类）一起工作。

（3） 需要一个统一的输出接口，而输入端的类型不可预知。

13、简述适配器模式的三种类型。

答：适配器模式主要分为三类：类适配器模式、对象适配器模式和缺省适配器模式。类适配器模式是通过继承来实现适配器功能。对象适配器是通过组合来实现适配器功能的。缺省适配器模式通过抽象类来实现适配，是适配器模式的一种变体。

14、什么是外观模式？

答：外观模式通过引入一个新的外观类(Facade)来实现该功能，外观类充当了软件系统中的“服务员”，它为多个业务类的调用提供了一个统一的入口，简化了类与类之间的交互。在外观模式中，那些需要交互的业务类被称为子系统(Subsystem)。

15、简述外观模式中的角色。

答：（1）门面角色：外观模式的核心。它被客户角色调用，它熟悉子系统的功能。内部根据客户角色的需求预定了几种功能的组合。

（2）子系统角色：实现了子系统的功能。它对客户角色和 Façade 是未知的。它内部可以有系统内的相互交互，也可以由供外界调用的接口。

（3）客户角色：通过调用 Facede 来完成要实现的功能。

16、简述外观模式的适用范围。

答：以下情况下可以考虑使用外观模式：

（1）设计初期阶段，应该有意识的将不同层分离，层与层之间建立外观模式。

（2）开发阶段，子系统越来越复杂，增加外观模式提供一个简单的调用接口。

（3）维护一个大型遗留系统的时候，可能这个系统已经非常难以维护和扩展，但又包含非常重要的功能，为其开发一个外观类，以便新系统与其交互。

17、简述外观模式的优点。

答：由于在外观类中维持了对子系统对象的引用，客户端可以通过外观类来间接调用子系统对象的业务方法，而无须与子系统对象直接交互，所以引入外观类后，客户端代码变得非常简单。

18、什么是责任链模式？

答：责任链模式将请求的处理对象像一条长链一般组合起来，形成一条对象链。请求并不知道具体执行请求的对象是哪一个，这样就实现了请求与处理对象之间的解耦。在这种模式中，通常每个接收者都包含对另一个接收者的引用。如果一个接收者不能处理该请求，那么它会把相同的请求传给下一个接收者，

依此类推。

19、简述责任链模式涉及的角色。

答：（1）抽象处理者(Handler)角色：定义出一个处理请求的接口。如果需要，接口可以定义出一个方法以设定和返回对下家的引用。这个角色通常由一个抽象类或者接口实现。

（2）具体处理者(ConcreteHandler)角色：具体处理者接到请求后，可以选择将请求处理掉，或者将请求传给下家。

20、什么是观察者模式？

答：观察者模式，又叫发布-订阅模式，是软件设计模式中行为型模式的一种。在这种设计模式中，一个目标物件管理所有相依于它的观察者物件，并且在它本身的状态改变时主动发出通知，这通常通过呼叫各观察者所提供的方法来实现。

21、简述观察者模式中的角色。

答：（1）抽象主题（Subject，抽象目标，抽象被观察者）：把所有观察者对象保存在一个集合里，每个主题都可以有任意数量的观察者；它提供了一个用于保存观察者对象的聚集类和增加、删除观察者对象的方法，以及通知所有观察者的抽象方法。

（2）具体主题（ConcreteSubject，具体目标，具体被观察者）：实现抽象主题中的通知方法，在具体主题的内部状态发生改变时，给所有注册过的观察者发送通知。

（3）抽象观察者（Observer）：它是一个抽象类或接口，它包含了一个更新自己的抽象方法，当接到具体主题的更改通知时被调用。

（4）具体观察者（ConcrereObserver）：实现抽象观察者中定义的抽象方法，以便在得到目标的更改通知时更新自身的状态。

22、简述实现观察者模式时需要注意的问题。

答：实现观察者模式的时候要注意，观察者和被观察对象之间的互动关系不能体现成类之间的直接调用，否则就将使观察者和被观察对象之间紧密的耦合起来，从根本上违反面向对象的设计原则。无论是观察者"观察"观察对象，还是被观察者将自己的改变"通知"观察者，都不应该直接调用。

习题 10

1、软件项目管理的内容主要包括哪几个方面？

答：软件项目管理的内容主要包括如下几个方面：人员的组织与管理，软件度量，软件项目计划，风险管理，软件质量保证，软件过程能力评估，软件配置管理等。

2、项目管理小组的主要职责有哪些？

答：（1）草拟项目管理的各项制度；

（2）组织项目阶段评审；

（3）保存项目过程中的相关文件和数据；

（4）为优化项目管理提出建议。

3、项目评审小组的主要职责有哪些？

答：（1）对项目可行性报告进行评审；

（2）对市场计划和阶段报告进行评审；

（3）对开发计划和阶段报告进行评审；

（4）项目结束时，对项目总结报告进行评审。

4、一般从哪几个方面对人员风险进行控制？

答：（1）保证开发组中全职人员的比例，且项目核心部分的工作应该尽量由全职人员来担任， 以减少兼职人员对项目组人员不稳定性的影响。

（2）建立良好的文档管理机制，包括项目组进度文档、个人进度文档、版本控制文档、整体技术文档、个人技术文档、源代码管理等。一旦出现人员的变动，比如某个组员因病退出，替补的组员能够根据完整的文档尽早接手工作。

（3）加强项目组内技术交流，比如定期开技术交流会，或根据组内分工建立项目组内部的开发小组，使开发小组内的成员能够相互熟悉对方的工作和进度，能够在必要的时候替对方工作。

（4）对于项目经理，可以从一开始就指派一个副经理在项目中协同项目经理管理项目开发工作，如果项目经理退出开发组，副经理可以很快接手。但是只建议在项目经理这样的高度重要的岗位采用这种冗余复制的策略来预防人员风险，否则将大大增加项目成本。

（5）为项目开发提供尽可能好的开发环境，包括工作环境、待遇、工作进度安排等等，同时一个优秀的项目经理应该能够在项目组内营造一种良好的人际关系和工作氛围。

5、什么是软件度量？

答：软件度量是对软件开发项目、过程及其产品进行数据定义、收集以及分析的持续性定量化过程，目的在于对此加以理解、预测、评估、控制和改善。

6、软件度量的目标可大致概括为哪两类？

答：（1）使用度量来进行估计，这使得我们可以同步地跟踪一个特定的软件项目。

（2）应用度量来预测项目的一些重要的特性。但需要注意，我们不能过分夸大这些预测，因为它们并不是完全正确的，软件度量得到也仅仅是预测而已。

7、什么是软件开发项目规模度量？

答：软件开发项目规模度量(size measurement)是估算软件项目工作量、编制成本预算、策划合理项目进度的基础。要点在于：由开发现场的项目成员进行估算；灵活运用实际开发作业数据；杜绝盲目迎合顾客需求的“交期逆推法”。

8、软件开发成本度量包括哪几个估算法？

答： 软件开发成本度量主要指软件开发项目所需的财务性成本的估算。主要方法：

类比估算法是通过比较已完成的类似项目系统来估算成本，适合评估一些与历史项目在应用领域、环境和复杂度方面相似的项目。

细分估算法是将整个项目系统分解成若干个小系统，逐个估算成本，然后合计起来作为整个项目的估算成本。

周期估算法是按软件开发周期进行划分，估算各个阶段的成本，然后进行汇总合计。

9、简述软件的顾客满意度度量要素。

答：

顾客满意度项目	顾客满意度度量要素
软件产品	功能性、可靠性、易用性、效率性、可维护性、可移植性
开发文档	文档的构成、质量、外观、图表以及索引、用语

项目进度以及交期	交期的根据、进度迟延情况下的应对、进展报告
技术水平	项目组的技术水平、项目组的提案能力、项目组的问题解决能力
沟通能力	事件记录、式样确认、Q&A
运用维护	支持、问题发生时的应对速度、问题解决能力

10、简述 FCM 3 层模型。

答：FCM 3 层模型：（1）第一层：质量要素。描述和评价软件质量的一组特性。

（2）第二层：衡量标准。衡量标准的组合反映某一软件质量要素。（3）第三层：量度标准。量度标准可由各使用单位自定义，根据软件的需求分析、概要设计、详细设计、编码、测试、确认、维护与使用等阶段，针对每一个阶段制定问卷表，以此实现软件开发过程的质量度量。

11、简述 Boehm 模型。

答：与 FCM 模型类似，它的质量模型结构包括 3 层：高层属性、中层属性和原始属性。

高层属性主要关注 3 个问题：通用实用（Genreal utility）、实用性（As-is utility）和可维护性(Maintainability)。

中层属性包含 7 个质量要素：可靠性(Reliability)、效率(Efficiency)、人类工程(Human Engineering)、可测试性(Testability)、可理解性(Understandability)、灵活性(Flexibility)和可移植性(Portability)。

原始层属性包含 15 个质量特性。

12、ISO9126 软件质量模型。

答：它由 6 个特性和 27 个子特性组成，这个模型是软件质量标准的核心，对于大部分的软件，都可以考虑从这几个方面着手进行测评。

（1）功能性：软件所实现的功能达到它的设计规范和满足用户需求的程度。

（2）可靠性：在满足一定条件的应用环境中，软件能够正常维持其工作的能力。

（3）可用性：对于一个软件，用户在学习、操作和理解过程中所做努力的程度。

（4）效率：在规定条件下，用软件实现某种功能所需的计算机资源(包括时间)的有效程度。

（5）维护性：当环境改变或软件运行发生故障时，为使其恢复正常运行所做努力的程度。

（6）可移植性：为使一个软件从现有运行平台向另一个运行平台过度所做努力的程度。

13、什么是软件的过程度量？

答：过程度量是对软件开发过程的各个方面进行度量，目的在于预测过程的未来性能，减少过程结果的偏差，对软件过程的行为进行目标管理，为过程控制、过程评价、持续改善提供定量性基础。

14、软件的软件过程管理包括那些步骤和内容？

答：软件过程管理包括定义过程、计划度量、执行软件过程、应用度量、控制过程和改善过程，其中计划度量和应用度量是软件过程管理中的重要步骤，也是软件过程度量的核心内容。

15、软件的软件过程度量主要包括哪三大方面？

答：软件过程度量主要包括三大方面的内容，一是成熟度度量(maturity metrics)，主要包括组织度量、资源度量、培训度量、文档标准化度量、数据管理与分析度量、过程质量度量等等；二是管理度量(management metrics)，主要包括项目管理度量、质量管理度量、配置管理度量；三是生命周期度量(life cycle metrics)，主要包括问题定义度量、需求分析度量、设计度量、制造度量、维护度量等。

16、简述什么是软件项目计划？

答：软件项目计划是一个软件项目进入系统实施的启动阶段，主要进行的工作包括：确定详细的项目实施范围、定义递交的工作成果、评估实施过程中主要的风险、制定项目实施的时间计划、成本和预算计划、人力资源计划等。

17、常用的制定进度计划的工具主要哪两种？

答：常用的制定进度计划的工具主要有甘特图和网络图两种。

18、简述甘特图的特点和作用。

答：甘特图的特点是突出了生产管理中最重要的时间因素，它的作用表现在三个方面：（1）计划产量与计划时间的对应关系。（2）每日的实际产量与预定计划产量的对比关系。（3）一定时间内实际累计产量与同时期计划累计产量的对比关系。

19、甘特图有哪些优点？

答：甘特图具有如下优点：直观、简单、通用、容易制作，便于理解，能很清晰地标识出直到每一项任务的起始与结束时间；有专业软件支持，无须担心复杂计算和分析。

20、简述什么是单代号网络图，其特点有哪些？

答：单代号网络图，也称为节点图，构成的基本要素是节点，节点表示活动(任务)，箭头线表示各活动(任务)之间的逻辑关系。单代号网络图的特点：单代号网络图用节点及其编号表示工作，而箭头线仅表示工作间的逻辑关系；单代号网络图作图简便，图面简洁，由于没有虚箭头线，产生逻辑错误的可能较小；单代号网络图用节点表示工作：没有长度概念，不够形象，不便于绘制时标网络图；单代号网络图更适合用计算机进行绘制、计算、优化和调整。

21、简述单代号网络图绘图规则。

答：首先，单代号网络图必须正确表达已定的逻辑关系；

第二，单代号网络图中，严禁出现循环回路，所谓循环回路是指从网络图中某一个节点出发，顺着箭头线方向又回到了原来出发点的线路；

第三，单代号网络图中，严禁出现双箭头或无箭头的连线；单代号网络图中，严禁出现没有箭尾节点的箭头线和没有箭头节点的箭头线；

第四，绘制网络图时，箭头线不宜交叉，当交叉不可避免时，可采用过桥法或指向法绘制。

最后，单代号网络图中应有一个起点节点和一个终点节点。

22、什么是双代号网络图？

答：双代号网络图，也称为箭线图。其中用箭头线表示活动(任务)，节点(事件)表示前一道工序的结束，同时，也表示后一道工序的开始。

23、绘制双代号网络图有哪两个基本规则。

答：绘制双代号网络图有两个基本规则：每个节点有唯一编号，即图中不会有相同的节点号；每个活动必须由唯一的紧前事件号和紧后事件号组成。

24、什么是软件项目风险？

答：软件项目风险是指在软件开发过程中遇到的预算和进度等方面的问题以及这些问题对软件项目的影响。软件项目风险会影响项目计划的实现，如果项目风险变成现实，就有可能影响项目的进度，增加项目的成本，甚至使软件项目不能实现。

25、软件项目开发中常见的风险有哪几类？

答：软件项目开发中常见的风险有如下几类：1.需求风险 2.计划编制风险 3.组织和管理风险 4.人员风

险 5.开发环境风险 6.客户风险 7.产品风险 8.设计和实现风险 9.过程风险

26、风险估算一般从几个方面评估风险清单中的每一个风险？

答：(1)建立一个尺度，以反映风险发生的可能性；

(2)描述风险的后果；

(3)估算风险对项目及产品的影响；

(4)标注风险预测的整体精确度，以免产生误解。

27、简述在风险评估过程中可以采取的步骤。

答：(1)定义项目的风险参考水平值。

(2)建立每一组（风险、风险发生的概率、风险产生的影响）与每一个参考水平值的关系。

(3)预测一组临界点以定义项目终止区域，该区域由一条曲线或不确定区域界定。

(4)预测什么样的风险组合会影响参考水平值。

28、风险驾驭包括哪些内容？

答：风险驾驭包括对策制定、风险缓解、风险监控、风险跟踪等内容。

29、简述软件质量保证的目标。

答：（1）事前预防工作，例如，着重于缺陷预防而不是缺陷检查。

（2）尽量在刚刚引入缺陷时即将其捕获，而不是让缺陷扩散到下一个阶段。

（3）作用于过程而不是最终产品，因此它有可能会带来广泛的影响与巨大的收益。

（4）贯穿于所有的活动之中，而不是只集中于一点。

30、简述软件质量保证的主要任务。

答：（1）SQA 审计与评审。SQA 审计包括对软件工作产品、软件工具和设备的审计，评价这几项内容是否符合组织规定的标准。SQA 评审的主要任务是保证软件工作组的活动与预定的软件过程一致，确保软件过程在软件产品的生产中得到遵循。

（2）SQA 报告。SQA 人员应记录工作的结果，并写入到报告之中，发布给相关的人员。

（3）处理不符合问题。这是 SQA 的一个重要的任务，SQA 人员要对工作过程中发现的问题进行处理及时向有关人员及高级管理者反映。

31、软件质量保证主要包括哪些措施？

答：（1）应用好的技术方法

（2）测试软件

（3）进行正式的技术评审

（4）标准的实施

（5）控制变更

（6）程序正确性证明

（7）记录、保存和报告软件过程信息

32、简述软件质量保证的实施一般包括哪几个步骤？

答：（1）目标。以用户需求和开发任务为依据，对质量需求准则、质量设计准则的质量特性设定质量目标进行评价。

（2）计划。设定适合于待开发软件的评测检查项目，一般设定 20-30 个。

（3）执行。在开发标准和质量评价准则的指导下，制作高质量的规格说明书和程序。

（4）检查。以计划阶段设定的质量评价准则进行评价，算出得分，以图形的形式表示出来，比较评价结果的质量得分和质量目标，确定是否合格。

（5）改进。对评价发现的问题进行改进活动，重复计划到改进的过程直到开发项目完成。

33、简述 CMM 的软件过程成熟度等级。

答：(1)初始级。工作无序，项目进行过程中常放弃当初的计划。管理无章法，缺乏健全的管理制度。开发项目成效不稳定，项目成功主要依靠项目负责人的经验和能力，他一但离去，工作秩序面目全非。

(2)可重复级。管理制度化，建立了基本的管理制度和规程，管理工作有章可循。 初步实现标准化，开发工作比较好地按标准实施。 变更依法进行，做到基线化，稳定可跟踪，新项目的计划和管理基于过去的实践经验，具有重复以前成功项目的环境和条件。

(3)已定义级。开发过程，包括技术工作和管理工作，均已实现标准化、文档化。建立了完善的培训制度和专家评审制度，全部技术活动和管理活动均可控制，对项目进行中的过程、岗位和职责均有共同的理解 。

(4)已管理级。产品和过程已建立了定量的质量目标。开发活动中的生产率和质量是可量度的。已建立过程数据库。已实现项目产品和过程的控制。可预测过程和产品质量趋势，如预测偏差，实现及时纠正。

(5)优化级。可集中精力改进过程，采用新技术、新方法。拥有防止出现缺陷、识别薄弱环节以及加以改进的手段。可取得过程有效性的统计数据，并可据进行分析，从而得出最佳方法。

34、简述 CMM 的能力评估。

答：（1）软件过程评估：用于确定一个组织当前的软件工程过程状态及组织所面临的软件过程的优先改善问题，为组织领导层提供报告以获得组织对软件过程改善的支持。软件过程评估集中关注组织自身的软件过程，在一种合作的、开放的环境中进行。评估的成功取决于管理者和专业人员对组织软件过程改善的支持。

（2）软件能力评价：用于识别合格的软件承包商或者监控软件承包商开发软件的过程状态。软件能力评价集中关注识别在预算和进度要求范围内完成制造出高质量的软件产品的软件合同及相关风险。评价在一种审核的环境中进行，重点在于揭示组织实际执行软件过程的文档化的审核记录。

35、简述 CMM 的过程改善。

答：软件过程改善是一个持续的、全员参与的过程。CMM 建立了一组有效地描述成熟软件组织特征的准则，该准则清晰地描述了软件过程的关键元素，并包括软件工程和管理方面的优秀实践。企业可以有选择地引用这些关键实践指导软件过程的开发和维护，以不断地改善组织软件过程，实现成本、进度、功能和产品质量等目标。

36、CMM 评估过程主要分成哪三个阶段？

答：准备阶段、评估阶段和报告阶段。

37、简述 CMM 评估过程详细的实施步骤。

答：第一步：建立一个评估评价组，该组的成员应具有丰富的软件工程和管理知识的专业人员，并接受过 CMM 模型基本概念和评估及评价方法方面的有关培训。

第二步：填写提问单，完成问卷调查和取样工作。

第三步：进行响应分析。评估和评价组对提问单响应进行统计分析，定义必须作进一步探查的区域。

第四步：进行现场访问。评估小组开始深入被评估的单位，以分析结果为依据，组织会谈和评审有关文档，以便更好地理解软件过程的情况。

第五步：提出调查发现清单。在现场工作阶段结束时，评估或评价组必须提供出评估单位软件过程的优缺点及强项和弱项清单。

第六步：制作关键过程域（KPA）剖面图。评估和评价组依据关键过程的基本情况列出评估提纲，指出被评估单位已经满足的软件过程域目标和尚未满足的软件过程域目标。

38、什么是软件配置？什么是软件配置管理？

答：软件配置由一组相互关联的对象组成，这些对象也称为软件配置项，它们是作为某些软件工程活动的结果而产生的。除了文档、程序和数据这些软件配置项之外，用于开发软件的开发环境也可置于配置控制之下。

软件配置管理是一种标识、组织和控制修改的技术，它应用于整个软件生存期，是指通过执行版本控制、变更控制的规程，以及使用合适的配置管理软件，来保证所有配置项的完整性和可跟踪性。

39、软件配置管理过程中主要涉及哪些角色？

答：1.项目经理 2.配置控制委员会 3.配置管理员 4.系统集成员 5.开发人员

40、简述在项目计划阶段软件配置管理计划的制定过程的主要流程。

答：CCB 根据项目的开发计划确定各个里程碑和开发策略；

CMO 根据 CCB 的规划，制定详细的配置管理计划，交 CCB 审核；

CCB 通过配置管理计划后交项目经理批准，发布实施。

41、简述在项目开发和维护阶段软件配置管理活动的三个主要层面。

答：（1）主要由 CMO 完成的管理和维护工作；（2）由 SIO 和 DEV 具体执行软件配置管理策略；（3）变更流程。这三个层面是彼此之间既独立又互相联系的有机的整体。

42、简述在项目开发和维护阶段软件配置管理过程的核心流程。

答：（1）CCB 设定研发活动的初始基线；（2）CMO 根据软件配置管理规划设立配置库和工作空间，为执行软件配置管理就阿做好准备；（3）开发人员按照统一的软件配置管理策略，根据获得的授权的资源进行项目的研发工作；（4）SIO 按照项目的进度集成组内开发人员的工作成果，并构建系统，推进版本的演进；（5）CCB 根据项目的进展情况，审核各种变更请求，并适时的划定新的基线，保证开发和维护工作有序的进行。

43、软件配置管理的关键活动有哪些？

答：1.配置项识别 2.工作空间管理 3.版本控制 4.变更控制 5.状态报告 6.配置审计