软件工程章节题库

第一章 软件工程简介

一、单项选择

1、软件是计算机系统中与硬件相互依存的另一部分，它是包括(1 B)、(2 A)及(3 D)的完整集合。其中，(1 B)是按事先设计的功能和性能要求执行的指令序列。(2 A)是使程序能够正确操纵信息的数据结构。(3 D)是与程序开发、维护和使用有关的图文材料。

1 .A. 数据　　B. 程序　　C. 用户使用手册　　D. 图表

2 A. 数据　　B. 文档　　C. 代码　　D. 安装说明

3 .A. 程序　　B. 数据　　C. 外设　　D. 文档

2、有人将软件的发展过程划分为 4 个阶段：

第一阶段（1950～1950 年代末）称为“程序设计的原始时期”，这时既没有(A ①)，也没有(B ④)，程序员只能用机器指令编写程序。

第二阶段（1950 年代末～1960 年代末）称为“基本软件期”。出现了(A ①)，并逐渐普及。随着(B ④)的发展，编译技术也有较大的发展。

第三阶段（1960 年代末～1970 年代中期）称为“程序设计方法时代”。这一时期，与硬件费用下降相反，软件开发费急剧上升。人们提出了(C ⑤)和(D ⑧)等程序设计方法，设法降低软件的开发费用。

第四阶段（1970 年代中期～现在）称为“软件工程时期”。软件开发技术不再仅仅是程序设计技术，而是包括了与软件开发的各个阶段，如(E ⑤)、(F ④)、编码、单元测试、综合测试、(G ①)及其整体有关的各种管理技术。

A--D：① 汇编语言　② 操作系统　③ 虚拟存储器概念　④ 高级语言
⑤ 结构式程序设计　⑥ 数据库概念 ⑦ 固件　⑧ 模块化程序设计

E--G：① 使用和维护　② 兼容性的确认　③ 完整性的确认　④ 设计
⑤ 需求定义　⑥ 图象处理

3、软件工程的最终目的是以较少的投资获得可维护的、可靠的、高效率的和可理解的软件产品。软件工程技术应遵循(A ⑦)、(B ⑧)、(C ⑥)、(D ③)、一致性、确定性、完备性、可验证性。

A--D：① 有效性　② 合理性　③ 局部化　④ 协同性 ⑤ 实用性
⑥ 模块化　⑦ 抽象　⑧ 信息隐蔽

4、软件产品的生产主要是 __C__。

A. 制造　　B. 复制　　**C. 开发**　　D. 研制

5、个体手工劳动是__B__时代的软件生产方式。

A. 程序系统　　**B. 程序设计**　　C. 软件工程　　D. 程序编码

6、软件工程是一门__C__学科。

A. 理论性　　B. 原理性　　**C. 工程性**　　D. 心理性

二、填空题

1、软件由计算机程序、数据和（ **文档** ）组成。

2、软件是一种（**逻辑**）产品，它与物质产品有很大的区别。

3、计算机系统由硬件、软件、使用计算机的人、数据库、（**文档**）和执行过

程组成。

4、软件的发展，到现在为止，经历了三个阶段：（**程序设计**）、（**程序系统**）和软件工程。

5、程序设计时代的生产方式是（**个体手工艺者**），程序系统时代的生产方式是（**手工作坊**），软件工程时代的生产方式是（**工程化**）。

6、软件工程是一门综合性的交叉学科，它涉及计算机学科、（**工程**）学科、管理学科和（**数学**）学科。

7、计算机科学中的成果都可用于软件工程，但计算机科学着重于（理论和原理），软件工程着重于（**建造软件系统**）。

8、软件工程研究的主要内容是（**方法**）、（**过程**）和（**工具**）等三个方面。

9、软件开发各阶段任务的划分应尽可能（**相对独立**），同一阶段任务的性质应尽可能（**相同**）。

三、判断题

1. A system is a collection of objects and activities.（F）

2. An abstraction is a description of the problem at some level of generalization that allows us to concentrate on the key aspects of the problem without getting mired in the details.（T）

四、问答题

1．什么是软件？它的特点是什么？

软件是计算机系统中与硬件相互依存的另一部分，它是包括程序，数据及其相关文档的完整集合。其中，程序是按事先设计的功能和性能要求执行的指令序列；数据是使程序能正常操纵信息的数据结构；文档是与程序开发，维护和使用有关的图文材料。

软件的特点是：

（1）依赖性：软件的开发和运行常常受到计算机系统的限制，对计算机系统有着不同程度的依赖性。软件不能完全摆脱硬件单独活动。在开发和运行中必须以硬件提供的条件为依据。有的软件依赖于某个操作系统。

（2）可移植性：为了解除这种依赖性，在软件开发中提出了软件移植的问题，并且把软件的可移植性做为衡量软件质量的因素之一。

（3）复用性：软件的开发至今尚未完全摆脱手工艺的开发方式。由于传统的手工艺开发方式仍然占据统治地位，开发的效率自然受到很大的限制。为此，人们在软件技术方面做了许多卓有成效的工作，提出了许多新的开发方法，例如充分利用现成软件的复用技术、自动生成技术，也研制了一些有效的软件开发工具或软件开发环境。

（4）复杂性：软件本身是复杂的。软件的复杂性可能来自它所反映的实际问题的复杂性，也可能来自程序逻辑结构的复杂性。软件开发，特别是应用软件的开发常常涉及到其它领域的专门知识，这对软件人员提出了很高的要求。

（5）昂贵性：软件成本相当昂贵。软件的研制工作需要投入大量的、复杂的、高强度的脑力劳动，它的成本是比较高的。然而，也并非在所有软件开发上的花费都能获得成果。

（6）社会性：相当多的软件工作涉及到社会因素。许多软件的开发和运行涉及机构、体制及管理方式等问题，甚至涉及到人的观念和人们的心理。它直接影响到项目的成败。

2．试比较软件发展的三个时期的特点，从软件所指、软件工作范围、软件开发组织、决定质量的因素、开发技术和手段等几个方面说明它们的差别。

时期 / 特点	程序设计	程序系统	软件工程
软件所指	程序	程序及说明书	程序，文档，数据
软件工作范围	程序编写	包括设计和测试	软件生存期
软件开发组织	个人	开发小组	开发小组及大中型软件开发机构
决定质量的因素	个人编程技术	小组技术水平	技术水平及管理水平
开发技术和手段	子程序和程序库	结构化程序设计	数据库，开发工具，开发环境，工程化开发方法，标准和规范，网络及分布式开发，面向对象技术及软件复用

3．软件工程是开发、运行、维护和修复软件的系统化方法，它包含哪些要素？试说明之。

软件工程包括三个要素：方法、工具和过程。

软件工程方法为软件开发提供了“如何做”的技术。它包括了多方面的任务，如项目计划与估算、软件系统需求分析、数据结构、系统总体结构的设计、算法过程的设计、编码、测试以及维护等。

软件工具是指为了支援软件人员的开发和维护活动而使用的软件。例如项目估算工具、需求分析工具、设计工具、编程和调试工具、测试工具和维护工具等。使用了软件工具后可以大大提高软件的生产率和质量。

软件工程的过程则将软件工程的方法和工具综合起来以达到合理、及时地进行计算机软件开发的目的。过程定义了方法使用的顺序、要求交付的文档资料、为保证质量和协调变化所需要的管理、及软件开发各个阶段完成的里程碑。

4．软件工程过程有哪几个基本过程活动？试说明之。

软件工程过程通常包含四种基本的过程活动：

P (Plan)： 软件规格说明。规定软件的功能及其运行的限制；

D (Do)： 软件开发。产生满足规格说明的软件；

C (Check)：软件确认。确认软件能够完成客户提出的要求；

A (Action)：软件演进。为满足客户的变更要求，软件必须在使用的过程中演进。

5.软件产品的质量一直是用户高度重视的问题，简述有哪些评论质量的观点。

用户的观点：质量是恰好达到目的

制造的观点：质量是与需求说明的一致

产品的观点：质量是与产品的内在特性相联系的

基于价值的观点：质量取决于顾客愿意支付的金额

超越的观点：质量是可以认识而不能定义的

6.什么是软件质量？如何度量软件质量？

ANSI/IEEE Std 729-1983 定义软件质量为"与软件产品满足规定的和隐含的需求的能力有关的特征或特性的全体"。M.J. Fisher 定义软件质量为"所有描述计算机软件优秀程度的特性的组合"。也就是说，为满足软件的各项精确定义的功能、性能需求，符合文档化的开发标准，需要相应地给出或设计一些质量特性及其组合，作为在软件开发与维护中的重要考虑因素。如果这些质量特性及其组合都能在产品中得到满足，则这个软件产品质量就是高的。

1991 年 ISO 发布的 ISO/IEC9126 质量特性国际标准定义了 6 个质量特性，即功能性、可靠性、可维护性、效率、可使用性、可移植性；并推荐了 21 个子特性，如适合性、准确性、互用性、依从性、安全性、成熟性、容错性、可恢复性、可理解性、易学习性、操作性、时间特性、资源特性、可分析性、可变更性、稳定性、可测试性、适应性、可安装性、一致性、可替换性，但不做为标准。

7.软件产品质量评价金三角“产品运行、产品修改、产品变迁”中的“产品变迁”包含哪些质量要素和与之对应的评价标准。

产品变迁包含的质量要素和与之对应的评价标准为：

可移植性：简单性、软件系统独立性、硬件独立性

可复用性：简单性、通用性、模块化、软件系统独立性、硬件独立性

互用性：模块化、通信通用性、数据通用性

8.什么是 CMM-SEI 能力成熟度模型？其五级成熟度水平是什么？

CMM-SEI 能力成熟度模型(Capability Manurity Model for Software,CMM)是软件工程协会 SEI(Software Engineering Institution)在卡内基.梅隆大学开发完成的对一个组织软件开发能力进行评价的标准,它侧重于对软件开发过程和开发方法论的考察。

CMM 五级成熟水平：

初始级

可重复级：有纪律的过程

已确定级：标准一致的过程

已管理级：可预测的过程

优化级：不断改进的过程

9.解释系统（system）的概念。

A system （系统）is a collection of things: a set of entities, a set of activities, a description of the relationships among entities and activities, and definition of the boundary of the system.。系统是一组事务的集合：实体的集合、活动的集合、实体和活动之间关系的描述以及系统边界的定义。

10. 软件工程要达到的基本目标是什么？

软件工程需要达到的基本目标是：付出较低的开发成本、达到要求的软件功能、取得较好的软件性能、开发的软件易于移植、需要较低的维护费用、能按时完成开发工作及时交付使用。

11. 软件工程的基本原则有哪些？试说明之。

在软件开发过程中必须遵循下列软件工程原则。

抽象 ： 采用分层次抽象，自顶向下、逐层细化的办法进行功能分解和过程分解，可以由抽象到具体、由复杂到简单，逐步得到问题的解。

信息隐蔽 ： 遵循信息封装，使用与实现分离的原则，将模块设计成“黑

箱”，可以将实现的细节隐藏在模块内部，使用者只能通过模块接口访问模块中封装的数据。

模块化 ： 按模块划分系统的体系结构，使得各模块间有良好的接口。这样有助于信息隐蔽和抽象，有助于表示复杂的系统。

局部化 ： 按抽象数据类型思想及问题域中的概念来建立模块，确保模块之间低耦合，模块内部高内聚。这有助于控制解的复杂性。

确定性 ： 软件开发过程中所有概念的表达应是确定的、无歧义性的、规范的。这有助于人们之间的沟通，保证整个开发工作协调一致。

一致性 ： 强调软件开发过程的标准化、统一化。包括文档格式的一致，工作流程的一致，内、外部接口的一致，系统规格说明与系统行为的一致等。

完备性 ： 软件系统不丢失任何重要成分，可以完全实现系统所要求功能。

可验证性 ： 开发大型的软件系统需要对系统自顶向下、逐层分解。系统分解应遵循系统易于检查、测试、评审的原则，以确保系统的正确性。

12．B. W. Boehm 有七条准则是确保软件产品质量和开发效率的原理的最小集合。简述 B. W. Boehm 的软件工程基本准则。

用分阶段的生命周期计划严格管理；

坚持进行阶段评审；

实行严格的产品控制；

采用现代程序设计技术；

结果应能清楚地审查；

开发小组的成员应该少而精；

承认不断改进软件工程实践的必要性。

第二章 过程和生命周期的建模

一、单项选择

1、开发软件时对提高软件开发人员工作效率至关重要的是(　A　①　)。软件工程中描述生存周期的瀑布模型一般包括计划、(　B　①　)、设计、编码、测试、维护等几个阶段，其中设计阶段在管理上又可以依次分成(　C　③　)和(　D　⑥　)两步。

A. ① 程序开发环境　② 操作系统的资源管理功能
③ 程序人员数量　④ 计算机的并行处理能力

B. ① 需求分析　② 需求调查　③ 可行性分析　④ 问题定义

C、D. ① 方案设计　② 代码设计　③ 概要设计　④ 数据设计
⑤ 运行设计　⑥ 详细设计　⑦ 故障处理设计　⑧ 软件体系结构设计

2、软件开发费用只占软件生存期全部费用的_B___。

A. 1/2　**B. 1/3**　C. 1/4　D. 2/3

3、在软件开发过程中大约要花费__C__%的工作量进行测试和调试。

A. 20　B. 30　**C. 40**　D. 50

4、准确地解决“软件系统必须做什么”是__B__阶段的任务。

A. 可行性研究　**B. 需求分析**　C. 软件设计　D. 程序编码

5、软件生存期中时间最长的是_D__ 阶段。

A. 需求分析　B. 软件设计　C. 软件测试　**D. 软件运行/维护**

6、在软件生存期的模型中，_D__适合于大型软件的开发，它吸收了软件工程中“演化”的概念。

A .喷泉模型　B. 基于知识的模型　C. 瀑布模型　**D. 螺旋模型**

7、在软件生存期中，用户的参与主要在_A___。

A.软件定义阶段 B.软件开发阶段 C.软件维护阶段 D.整个软件生存期过程中

8、在软件开发过程中的每个阶段都要进行严格的__D___，以尽早发现在软件开发过程中产生的错误。

A. 检验　B. 验证　C. 度量　**D. 评审**

9、在软件开发和维护过程中需要变更需求时，为了保持软件各个配置成分的一致性，必须实施严格的__B___。

A. 产品检验　**B. 产品控制**　C. 产品标准化　D. 开发规范

10、实践表明，采用先进的开发技术可提高软件开发的生产率，还可提高软件的　__D__　　。

A. 可靠性　B. 可使用性　C. 安全性　**D. 可维护性**

11、为了提高软件开发过程的___A_，有效地进行管理，应当根据软件开发项目的总目标及完成期限，规定开发组织的责任和产品标准。

A. 可见性　B. 生产率　C. 安全性　D. 有效性

12、随着开发小组人数的___A__，因交流开发进展情况和讨论遇到的问题而造成的通信开销也急剧增加。

A. 增加　B. 降低　C. 稳定　D. 不稳定

13、为保证软件开发的过程能够跟上技术的进步，必须不断地灵活地改进软件工程__C__。

A. 原则　　B. 工具　　**C. 过程**　　D. 方法

二、填空题

10、瀑布模型是将各个活动规定为依（**软件生存期**）连接的若干阶段的模型。它规定了各阶段的活动由前至后，相互衔接的固定次序，如瀑布流水，逐级下落。

11、螺旋模型将开发过程分为几个螺旋周期。在每个螺旋周期内分为四个工作步骤：（**制定计划**）、（**风险分析**）、**开发实施**、（**用户评估**）。

12、喷泉模型是一种以（**用户要求**）为动力，以（**对象**）为驱动的模型。它使开发过程具有迭代性和无间隙性，适用于（**面向对象**）开发方法。

三、判断题

1.We can think of a set of ordered tasks as a process: a series of steps involving activities constraints and recourses that produce an intended output of some kind. （T）

2.The software development process is sometimes called the software life cycle. (T)

四、问答题

1、the meaning of process　过程的含义

A process defines who is doing what, when and how, in order to reach a certain goal.　过程定义了谁在作什么，什么时间怎样作。以便完成一个确定的目标。

2、 What is Process?

A Series of steps involving activities, constraints, and resources that produce an intended output of some kind.一系列涉及到活动、约束和资源的步骤，他们产生某种类型的有目的的输出。

3、Process Characteristics? 过程的特征

The process prescribes all of the major process activities 过程规定了所有主要过程活动

Process uses resources, subject to a set of constraints (such as schedule),and produces intermediate and final products　过程使用资源、服从于一组约束(比如进度约束)，产生中间结果和最终产品。

The process may be composed of subprocesses that are linked in some way. The process may be defined as a hierarchy of processes, organized so that each subprocess has its own process model 可由子过程组成，这些子过程用某种方式链接起来。过程可以定义为分层的过程等级结构，以便每个子过程具有自己的过程模型。

Each process activity has entry and exit criteria , so that we know when the activity begins and ends. 每个过程活动具有有入口和出口标准，这样可以知道活动何时开始及何时结束。

The activities are organized in a sequence, so that it is clear when one activity is performed relative to the other activities. 活动以一定

顺序组织，因此，一个活动相对于其他活动何时完成是很清楚的。

Every process has a set of guiding principles that explain the goals of each activity 每个过程具有一系列的指导原则，以解释每个活动的目标

Constraints or controls may apply to an activity, resource or product 约束与控制可以应用到任何活动、资源或产品中。

4、Why software process modeling?

Writes down a description of development process, forms a common understanding of the activities, resources, and constraints involved in software development. 形成对软件开发中涉及到的活动、资源和约束的共同理解。

Helps the development team find inconsistencies, redundancies, and omissions in the process and in its constituent parts. 有助于开发小组发现过程及其组织成分中的不一致、冗余和遗漏。

The model reflects the goals of development, such as building high-quality software finding faults early in development, and meeting required budget and schedule constraints. 反映开发的目标(如构建高质量软件、早期发现错误、满足预算和开发进度)。

Every process should be tailored for the special situation in which it will be used. 根据每个过程将被使用的特殊情况对其进行裁剪。

5．试说明“软件生存期”的概念。

软件与任何一个事物一样，有它的孕育、诞生、成长、成熟、衰亡的生存过程。这就是软件的生存期。它分为 6 个阶段：

(1) 软件项目计划 ：在这一步要确定软件工作范围，进行软件风险分析，预计软件开发所需要的资源，建立成本与进度的估算。根据有关成本与进度的限制分析项目的可行性。

(2) 软件需求分析和定义 ：在这一步详细定义分配给软件的系统元素。可以用以下两种方式中的一种对需求进行分析和定义。一种是正式的信息域分析，可用于建立信息流和信息结构的模型，然后逐渐扩充这些模型成为软件的规格说明。另一种是软件原型化方法，即建立软件原型，并由用户进行评价，从而确定软件需求。

(3) 软件设计 ：软件的设计过程分两步走。第一步进行概要设计，以结构设计和数据设计开始，建立程序的模块结构，定义接口并建立数据结构。第二步做详细设计，考虑设计每一个模块部件的过程描述。经过评审后，把每一个加细的过程性描述加到设计规格说明中去。

(4) 程序编码 ：在设计完成之后，用一种适当的程序设计语言或 CASE 工具生成源程序。应当就风格及清晰性对代码进行评审，而且反过来应能直接追溯到详细设计描述。

(5) 软件测试 ：单元测试检查每一单独的模块部件的功能和性能。组装测试提供了构造软件模块结构的手段，同时测试其功能和接口。确认测试检查所有的需求是否都得到满足。在每一个测试步骤之后，要进行调试，以诊断和纠正软件的故障。

(6) 软件维护 ：为改正错误，适应环境变化及功能增强而进行的一系列修改活动。与软件维护相关联的那些任务依赖于所要实施的维护的类型。

6、 List the stages of Waterfall model, and state the advantages and shortage of the Waterfall model.

Include stages of requirements analysis, system design, program design, coding, unit & integration testing, acceptance testing, operation & maintenance.

Merits of Waterfall model

Has been used to prescribe software development activities in a variety of contexts. 已被用于在各种情况下规定软件开发活动。

Is very useful in helping developers lay out what they need to do. 帮助开发人员明确需要做什么

Easy to explain to customers who are not familiar with software development 易于向不熟悉开发的顾客作出解释

It makes explicit which intermediate products are necessary in order to begin the next stage. 易于向不熟悉开发的顾客作出解释

More complex models are really just embellishments of the waterfall. 更复杂的模型是它的修改。其他模型的基础

Shortage of Waterfall model

Does not reflect the way code is really developed.它不能反映实际的代码开发方式。

The model imposes a project management structure on system development. 这个模型给系统开发强加了一种项目管理结构

Fail to treat software as a problem-solving process. present a manufacturing view.没能把软件看成是一个问题解决的过程，仅表达了一种制造观点。

The model tells us nothing about the typical back-and-forth activities that lead to creating a final product. 模型没告诉我们开发最终产品所需的典型的不断改进的活动。

7．试说明螺旋模型软件开发方法的基本过程，比较它的优点和缺点。

螺旋模型是一种风险驱动的模型。在软件开发中存在各种风险。项目越复杂，设计方案、资源、成本、进度等因素的不确定性越大，项目开发的风险也就越大。及时对风险进行识别、分析，采取对策，可消除或减少风险的损害。螺旋模型将开发过程分为几个螺旋周期，每个螺旋周期大致和瀑布模型相吻合。在每个螺旋周期内按四个象限，分为四个工作步。第一，制定计划：确定软件目标，选定实施方案，明确项目开发的限制条件；第二，风险分析：分析所选方案，识别风险，通过原型消除风险；第三，开发实施：实施软件开发；第四，客户评估：评价开发工作，提出修正建议，建立下一个周期的计划。

螺旋模型的优点：

实质上相当于在瀑布模型的每个阶段开始前引入风险分析，并由客户对阶段性产品做出评审，这对保证软件产品质量十分有利；

由于引入风险分析等活动，测试活动的确定性增强了；

螺旋模型最外层代表维护，开发与维护采用同样方式，使维护得到与开发同样的重视。

螺旋模型的缺点：

主要适合内部开发，否则风险分析必须在签订合同前完成，或者争取客户的最大理解；

只适合大型软件项目的开发，否则，每个阶段的风险分析将占用很大一部分

资源，增加成本；

对开发人员的风险分析能力是极大的考验，否则，模型将退化到瀑布模型，甚至更糟。

8、 List at least 6 typical process models

Waterfall model 瀑布模型

Prototyping 原型化模型

V-model V-模型

Operational specification 操作说明模型

Transformational model 变换模型

Phased development: increments and iteration 阶段化开发：增量和迭代模型

Spiral model 螺旋模型

9、列举出 5 个以上的经典过程模型，详细阐述增量和迭代模型的原理、用途和开发难点。

瀑布模型、原型化模型、V-模型、操作说明模型、变换模型、螺旋模型

增量开发：定义发布时首先是定义一个小的、具有一定功能的子系统，然后在每一个新的发布中增加新的功能。

迭代开发：是在一开始就移交一个完整的系统，然后在每一个新的发布版本中改变每一个子系统的功能。

用途：

1.培训可以在早期的版本中开始；

2.可以为那些以前从未实现的功能提前开拓市场 ；

3.当在使用的系统中有未预料的问题报告时，在新版本中开发人员可以全面快速修正这些问题；

4.开发小组可以把不同的发布版本针对不同的领域。

难点：

1.在把每个新的增量构件集成到现有软件体系结构中时，必须不破坏原来已经 开发出的产品；

2.必须把软件的体系结构设计得便于按这种方式进行扩充，向现有产品中加入新构件的过程必须简单、方便，也就是说，软件体系结构必须是开放的。

10、列举出 5 个以上的经典过程模型，详细阐述 v 模型原理、优点和缺点。

瀑布模型、原型化模型、V-模型、操作说明模型、变换模型、螺旋模型

v 模型是瀑布模型的变种，增加了测试活动与分析和设计的关系

强调测试活动与分析和设计之间的关联：

用单元测试和集成测试来校验程序设计；

用系统测试来校验（verify）系统设计；

用验收测试来确认（validate）需求；

与瀑布模型关注文档和工作产品不同，V 模型的关注点是软件开发各阶段的活动以及正确性，因此 V 模型是以活动驱动的。

优点与缺点：

本质是把瀑布模型中一些隐含的迭代过程明确出来，使开发活动和验证活动的相关性更加明显；

V 模型使抽象等级的概念也更明显：所有从需求到实现部分的活动关注的是建立更多的系统详细表述，而所有从实现到交付运行的活动关注的是对系统的验

证和确认。

和瀑布模型一样，都是对软件开发过程过份简单、理想化的抽象，对需求变化的适应性差。

11、List the steps of software development, and roles of developers.

Include stages of requirements analysis, system design, program design, coding, unit & integration testing, acceptance testing, operation & maintenance.

Roles include analyzer, designer, programmer, tester, trainer.

第三章 项目的计划和管理

一、判断题

1、A milestone is the completion of an activity -- a particular point in time. （T）

2、The loss associated with a risk is called the risk impact.（T）

3、Risk is an unwanted event that has negative consequences.（T）

4、A precursor is an event or set of events that must occur before the activity can begin; it describes the set of conditions that allows the activity to begin. (T)

二、填空题

1、要成功地完成软件开发工作的一个主要的决定性因素是（**软件项目管理**）。

2、软件过程是软件生存期中的一系列相关（**软件工程活动**）集合。

3、所有的软件开发都可以看成是一个问题循环解决过程，其中包括 4 个截然不同的阶段：（**状态捕获**）、**问题定义**、**技术开发**和（**方案综合**）。

4、在制定软件项目计划之前，必须先明确项目的（**目标**）和（**范围**）。项目的（**目标**）标明了软件项目的目的但不涉及如何去达到这些目的。

5、对软件进行度量，是为了表明软件产品的（**质量**），弄清软件开发人员的（**生产率**），建立项目估算（**基线**），帮助调整对新的工具和附加培训的要求。

6、软件质量的事后度量包括（**正确性**）、**可维护性**、（**完整性**）和**可使用性**。其中，（**完整性**）包括危险性和安全性。

7、软件范围包括功能、性能、限制、（**接口**）和可靠性。

8、软件项目计划的第二个任务是对完成该软件项目所需的（**资源**）进行估算。（**资源**）包括人与工具。

9、对于一个大型的软件项目，要进行一系列的估算处理。主要靠（**分解**）和类推的手段进行。

10、基本 COCOMO 模型是一个（**静态单变量**）模型，它用一个已估算出的源代码行数（**LOC**）为自变量的（**经验**）函数来计算软件开发工作量。

11、成本－效益分析的目的，是从经济角度评价开发一个新的软件项目是否（**可行**）。

12、风险估计从两个方面估价风险。一是估计一个风险发生的（**可能性**）。一是估价与风险相关的问题出现后将会产生的（**结果**）。

13、一个软件任务由一个人单独开发，生产率（**最高**）。

14、在与软件成本相关的影响因素中，（**人员的能力**）是最大影响因素。

15、软件开发所需的人力随开发的进展逐渐增加，在（**编码与单元测试**）阶段达到高峰，以后又逐渐减少。

16、在建立项目组织时应注意的原则有三：①尽早（**落实责任**），指定专人负责；② 减少（**接口**），要有合理的人员分工、好的组织结构、有效的通信，

减少不必要的生产率的损失；③（**责权**）均衡。

17、风险出现概率可以使用从过去项目、直觉或其它信息收集来的度量数据进行（**统计分析**）估算出来。

18、用各种不同的方法对风险进行分类是可能的。从宏观上来看，可将风险分为项目风险、技术风险和（**商业风险**）。

三、选择题

1、所有的软件开发都可以看成是一个问题__ B __过程。

A. 顺序解决 **B. 循环解决** C. 分类解决 D. 分组解决

2、软件项目管理所涉及的范围覆盖了整个软件__D__。

A. 开发过程 B. 运行与维护过程 C. 定义过程 **D. 生存期**

3、为使软件开发获得成功，一个关键问题是必须对软件范围、风险、资源、任务、里程碑、成本，进度等做到心中有数，而__C__可以提供这些信息。

A. 计算机辅助工程 B. 软件开发工具 **C. 软件项目管理** D. 软件估

4、软件范围标明了软件要实现的基本功能，并尽量以_A__的方式界定这些功能。

A. 定量 B. 规范 C. 统一 D. 定性

5、只要事先建立特定的度量规程，很容易做到__B__开发软件所需要的成本和工作量、产生的代码行数等。

A. 间接度量 **B. 直接度量** C. 间接估算 D. 直接估算

6、为了计算特征点，可以像计算功能点那样，对信息域值进行计数和加权。此外，需要对一个新的软件特征__A__进行计数。

A. 算法 B. 计算误差 C. 程序复杂性 D. 效率

7、对于软件的__D__，有一种简单的面向时间的度量，叫做平均变更等待时间 MTTC（Mean Time To Change)。 这个时间包括开始分析变更要求、设计合适的修改、实现变更并测试它、以及把这种变更发送给所有的用户。

A. 可靠性 B. 可修改性 C. 可测试性 **D. 可维护性**

8、软件的完整性是度量一个系统抗拒对它的__C__攻击(事故的和人为的)的能力。

A. 可靠性 B. 正确性 **C. 安全性** D. 容错性

9、对每一种软件资源，应说明 4 个特性：资源的描述，资源的有效性说明，资源在何时开始需要，使用资源的持续时间。最后两个特性统称为__A__。

A. 时间窗口 B. 时间安排 C. 日程安排 D. 资源定义

10、业务系统计划工具借助特定的__C__建立一个组织的战略信息需求的模型，导出特定的信息系统。

A.过程性语言 B. 形式化语言 **C. 元语言** D. 伪码

11、软件开发成本主要是指软件开发过程中所花费的__B___及相应的代价。

A. 劳动力 **B. 工作量** C. 资源 D. 持续时间

12、自顶向下估算软件成本的方法主要是从项目的整体出发进行__B__，即根据已完成项目的总成本(或总工作量)，来推算待开发软件的总成本(或总工作量)，然后按比例将它分配到各开发任务单元中去。

A. 分解 **B. 类推** C. 推导 D. 评估

13、自底向上估算软件成本的方法主要是把待开发软件__A__，直到每一个子任务都已经明确所需要的开发工作量，然后把它们加起来，得到软件开发的总

工作量。

A. 分解 B. 类推 C. 推导 D. 评估

14、Putnam 提出的模型，是一种__D__模型。它是假定在软件开发的整个生存期中工作量有特定的分布。

A. 模块化成本 B. 结构化成本 C. 动态单变量成本 **D. 动态多变量成本**

15、Boehm 提出的___B__估算模型是一种精确、易于使用的成本估算方法。

A. 模块化成本 **B. 结构化成本** C. 动态单变量成本 D. 动态多变量成本

16、系统的经济效益__B___因使用新系统而增加的收入加上使用新系统可以节省的运行费用。

A. 大于 **B. 等于** C. 小于 D. 不等于

17、项目复杂性、规模和结构的不确定性构成__C___(估算)风险因素。

A. 技术 B. 经济 **C. 项目** D. 商业

18、识别风险的一种最好的方法就是利用一组__A__来帮助人们了解在项目和技术方面有哪些风险。因此，Boehm 建议使用一个“风险项目检查表”。

A. 提问 B.项目 C.脚本 D.场景

19、在做风险评价时常采用的一个非常有效的方法就是定义__D__。

A. 风险评价标准 B. 风险影响因子 C. 风险调整因素 **D. 风险参照水准**

20、Pareto 的 80 / 20 规则用到软件风险上表明，所有可能导致项目失败的 80% 的潜在因素能够通过__B__的已识别风险来说明。

A. 30% **B. 20%** C. 50% D. 25%

21、当几个人共同承担软件开发项目中的某一任务时，人与人之间必须通过交流来解决各自承担任务之间的_A_问题，即所谓通信问题。

A. 接口 B. 衔接 C. 调用 D. 控制

22、假设一个人单独开发软件，生产率是 5000 行 / 人年。若 4 个人组成一个小组共同开发这个软件，在每条通信路径上耗费的工作量是 250 行 / 人年。则组中每人的生产率降低为__C__

A. 4200 B. 4350 **C. 4675** D. 4375

23、PERT 技术叫做 __B__技术，它是采用网络图来描述一个项目的任务网络，安排开发进度，制定软件开发计划的最常用的方法。

A. 日程安排 **B. 计划评审** C. 关键路径 D. 因果图

24、在进度压力下赶任务，其成果往往是以__B__产品的质量为代价的。

A. 浪费 **B. 牺牲** C. 抵押 D. 维持

四、选择填空题

1、软件过程是软件(A ④)中的一系列相关软件工程(B ③)的集合。每一个软件过程又是由一组(C ⑥)、项目(D ⑧)、软件工程产品和交付物以及质量保证(SQA)点等组成。一个软件过程可以用右图的形式来表示。首先建立一个(E ②)过程框架，其中定义了少量可适用于所有软件项目的框架(B ③)，再给出各个框架(B ③)的任务集合，最后是保护伞活动，如软件质量保证、软件配置管理以及测量等。软件过程模型的选择基于项目和应用的特点、采用的(F ⑤)和工具、要求的控制和需交付的产品。

A～F. ① 工程 ② 公共 ③ 活动 ④ 生存期
⑤ 方法 ⑥ 工作任务 ⑦ 功能 ⑧ 里程碑

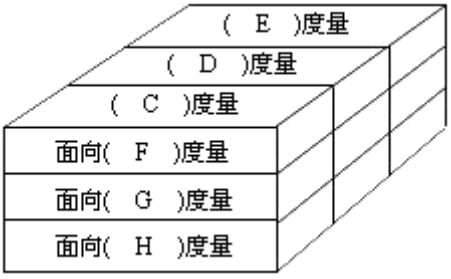
2、由于软件工程有如下的特点，使软件管理比其它工程的管理更为困难。

软件产品(　A　②　)。(　B　⑥　)标准的过程。大型软件项目往往是(　C　③)项目。(　D　②)的作用是为有效地定量地进行管理，把握软件工程过程的实际情况和它所产生的产品质量。在制定计划时，应当对人力、项目持续时间、成本作出(　E　④　)；(　H　⑤　)实际上就是贯穿于软件工程过程中一系列风险管理步骤。最后，每一个软件项目都要制定一个(　F　①)，一旦(　G　⑥)制定出来，就可以开始着手(　H　⑤)。

A～C.　① 可见的　② 不可见的　③ “一次性”　④ “多次”
⑤ 存在　⑥ 不存在

D～H.　① 进度安排　② 度量　③ 风险分析　④ 估算
⑤ 追踪和控制　⑥ 开发计划

3、软件的度量包括(　A　①)和(　B　④)。软件产品的(　A ①　)包括产生的代码行数、执行速度等。软件产品的(　B　④)则包括若干质量特性。我们还可进一步将软件度量如图所示那样分类。软件(　C　④)度量主要关注软件工程过程的结果；(　D　①　)度量则指明了软件适应明确和不明确的用户要求到什么程度；(　E　②)度量主要关注软件的一些特性而不是软件开发的全过程。从图中还可看到另一种分类方法：面向(　F　⑤)的度量用于收集与直接度量有关软件工程输出的信息和质量信息。面向(　G　⑦)的度量提供直接度量的尺度。面向(　H　③)的度量则收集有关人们开发软件所用方式的信息和人们理解有关工具和方法的效率的信息。



A～B.　直接度量　② 尺度度量　③ 二元度量　④ 间接度量

C～E.　① 质量　② 技术　③ 成本　④ 生产率

F～H.　① 过程　② 对象　③ 人　④ 存取　⑤ 规模　⑥ 进程
⑦ 功能　⑧ 数据

4、估算资源、成本和进度时需要经验、有用的历史信息、足够的定量数据和作定量度量的勇气。通常估算本身带有(　A　③　)。项目的复杂性越高，规模越大，开发工作量(　B　②　)，估算的(　A　③　)就(　C　⑦　)。项目的结构化程度提高，进行精确估算的能力就能(　D　①)，而风险将(　E　③)。有用的历史信息(　F　②　)，总的风险会减少。

A.　① 风范（范型）　② 风格　③ 风险　④ 度量

B～F.　① 增加　② 越多　③ 降低　④ 不变
⑤ 越少　⑥ 越高　⑦ 越大

5、在软件项目估算时，将代码行 LOC 和功能点 FP 数据在两个方面使用：一是作为一个估算变量，度量软件每一个(　A　③)的大小；一是联合使用从过去的项目中收集到的(　B　⑦　)和其它估算变量，进行成本和(　C　⑤　)估

算。LOC 和 FP 是两种不同的估算技术，但两者有许多共同的特征，只是 LOC 和 FP 技术对于分解所需要的(　D　①)不同。当用(　E　②)作为估算变量时，功能分解是绝对必要且应达到很详细的程度，而用(　F　①)作为估算变量时，分解程度可以不很详细。(　E　②)是直接估算，(　F　①)是间接估算。若计划人员对每个功能分别按最佳的、可能的、悲观的三种情况给出 LOC 或 FP 估计值，记作 a，m，b，则 LOC 或 FP 的期望值 E 的公式为(　G　②)，m 是加权的最可能的估计值，遵循(　H　③)。

A～C.　① 模块　② 软件项目　③ 分量　④ 持续时间
⑤ 工作量　⑥ 进度　⑦ 基线数据　⑧ 改进数据

D.　① 详细程度　② 分解要求　③ 改进过程　④ 使用方法

E,F.　① FP　② LOC

G.　① E = (a+m+b)/3　② E = (a+4m+b)/6
③ E = (2a+3m+4b)/3　④ $E=\sqrt[3]{a\cdot 4m\cdot b}$

H.　① x 概率　② γ 概率　③ β 概率　④ 泊松

6、定义一个人参加劳动时间的长短为(　A　④)，其度量单位为 PM(人月)或 PY(人年)。而定义完成一个软件项目(或软件任务)所需的(　A　④)为(　B　②)，其度量单位是人月／项目(任务)，记作 PM(人月)。进一步地，定义单位(　A　④)所能完成的软件(　C　④)的数量为软件(　D　①)，其度量单位为 LOC／PM。它表明一般指(　E　①)的一个平均值。例如，一个软件的开发工作量如下表所示。该软件共有源代码 2900 行，其中，500 行用于测试，2400 行是执行(　F　②)的源代码。则劳动生产率是(　G　④) (LOC／PM)。

表 软件开发所需工作量例

阶　段	软件计划	需求分析	设 计	编 码	测 试	总计
需要工作量(人月)	1.0	1.5	3.0	1.0	3.5	10.0

A，B，D.　① 生产率　② 工作量　③ 成本　④ 劳动量

E.　① 开发全过程　② 某开发阶段　③ 软件生存期　④ 某开发任务

F，C.　① 软件　② 程序　③ 进程　④ 产品

G.　① 520　② 120　③ 320　④ 240

7、对于一个大型的软件项目，由于项目的复杂性，需要进行一系列的估算处理。主要按(　A　③)和(　B　①)手段进行。估算的方法分为三类：从项目的整体出发，进行(　B　①)的方法称为(　C　②)估算法。把待开发的软件细分，直到每一个子任务都已经明确所需要的开发工作量，然后把它们加起来，得到软件开发总工作量的方法称为(　D　③)估算法。而把待开发的软件项目与过去已完成的软件项目做类比，区分出类似部分和不同部分分别处理的方法称为(　E　①)估算法。(　F　④)是由多位专家进行成本估算的方法。

A,B.　① 类推　② 类比　③ 分解　④ 综合

C～F.　① 差别　② 自顶向下　③ 自底向上　④ 专家判定技术
⑤ 循序渐进　⑥ 比较

8、假设开发某个计算机应用系统的投资额为3000元，该计算机应用系统投入使用后，每年可以节约1000元，5年内可能节约5000元。3000元是现在投资的钱，5000元是5年内节省的钱，两者不能简单地比较。

假定年利率为12%，利用计算货币现在价值的公式，可以算出该计算机应用系统投入使用后每年预计节省的金额的现在价值。

年	节省(元)	利率$(1+0.12)^n$	现在价值(元)	累计现在价值(元)
1	1000	1.12	892.86	892.86
2	1000	1.25	800.00	1692.86
3	1000	1.40	714.29	2407.15
4	1000	1.57	636.94	3044.09
5	1000	1.76	568.18	3612.27

则该系统的纯收入是(A ④)，投资回收期是(B ②)，投资回收率为(C ③)。

A. ① 512.3元 ② 729.28元 ③ 602.4元 ④ 612.27元

B. ① 2. 4年 ② 3.93年 ③ 4.25年 ④ 2.78元

C. ① 25% ② 30% ③ 20% ④ 15%

该计算机应用系统在5年中的纯收入为：3612.27 - 3000 = 612.27（元）。

投资回收期约为：3 +(3000-2407.15)/(3044.09-2407.15) ≈ 3.93（年）。

投资回收率设为r，由下列方程式：

$$3000=1000/(1+r)+1000/(1+r)^2+1000/(1+r)^3+\cdots\cdots+1000/(1+r)^5$$

解得r = 20%。

纯收入就是在整个生存期之内系统的累计经济效益(折合成现在值)与投资之差。投资回收期就是使累计的经济效益等于最初的投资所需的时间。投资回收率是投入资金所获得的利率。

9、在特定情况下，是否必须进行风险分析，是对项目开发的形势进行(A ②)后确定的。(A ②)可以按如下步骤进行：明确项目的目标、总策略、具体策略和为完成所标识的目标而使用的方法和资源；保证该目标是(B ①)，项目成功的标准也是(B ①)；考虑采用某些条目作为项目成功的(C ②)；根据估计的结果来确定是否要进行风险分析。

一般来说，风险分析的方法要依赖于特定问题的需求和有关部门所关心的方面。具体分3步进行。第一步识别潜在的风险项，首先进行(D ②)过程；第二步估计每个风险的大小及其出现的可能性，选择一种(E ③)，它可以估计各种风险项的值；第三步进行风险评估。风险评估也有三个步骤：确定(F ④)，确定(G ③)，把风险与“参照风险”做比较。

A. ① 风险管理 ② 风险估计 ③ 风险评价 ④ 风险测试

B. ① 可度量的 ② 不可度量的 ③ 准确的 ④ 不确定的

C. ① 规范 ② 标准 ③ 过程模型 ④ 设计要求

D,E. ① 信息分类 ② 信息收集 ③ 度量尺度 ④ 标准 ⑤ 度量工具 ⑥ 信息获取

F,G. ① 风险的范围 ② 风险的特性 ③ 风险的级别 ④ 风险的评价标准 ⑤ 风险的排除策略

10、任何软件项目都必须做好项目管理工作，最常使用的进度管理工具是(A ④)，当某一开发项目的进度有可能拖延时，应该(B ②)。对于一个典型的软件开发项目，各开发阶段需投入的工作量的百分比大致是(C ③)。各阶段所需不同层次的技术人员大致是(D ③)，而管理人员在各阶段所需数量也不同，相对而言大致是(E ①)。

A. ① 数据流图 ② 程序结构图 ③ 因果图 ④ PERT 图

B. ① 增加新的开发人员 ② 分析拖期原因加以补救
③ 从别的小组抽调人员临时帮忙 ④ 推迟预定完成时间

C.

		需求分析	设 计	编 码	测 试
投入工作量	①	25	25	25	25
	②	10	20	30	40
	③	15	30	15	40
	④	5	10	65	30

D.

技术人员水平	①	初级	高级	高级	高级
	②	中级	中级	高级	中级
	③	高级	中高级	初级	中高级
	④	中级	中高级	中级	初级

E.

管理人员数量	①	多	中	少	中
	②	中	中	中	中
	③	多	少	多	多
	④	少	多	少	多

11、对于一个小型的软件开发项目，一个人就可以完成需求分析、设计、编码和测试工作。但随着软件项目规模增大，需要有多人共同参与同一软件项目的工作。当几个人共同承担软件开发项目中的某一任务时，人与人之间必须通过交流来解决各自承担任务之间的(A ③)问题，即通信问题。通信需花费时间和代价，会引起软件错误(B ②)，(C ①)软件生产率。如果一个软件开发小组有 n 个人，每两人之间都需要通信，则共有(D ②)条通信路径。假设一个人单独开发软件，生产率是 5000 行 / 人年，且在每条通信路径上耗费的工作量是 250 行 / 人年。若 4 个人组成一个小组共同开发这个软件，则小组中每个人的软件生产率为(E ③)。若小组有 6 名成员，则小组中每个成员的软件生产率为(F ②)。因此，有人提出，软件开发小组的规模不能太大，人数不能太多，一般在(G ④)人左右为宜。

A. ① 分配 ② 管理 ③ 接口 ④ 协作

B,C. ① 降低 ② 增加 ③ 不变

D. ① n(n+1)/2 ② n(n-1)/2 ③ n(n-1)(n-2)/6 ④ n2/2

E,F. ① 4875 ② 4375 ③ 4625 ④ 5735

G. ① 8～15 ② 1～2 ③ 2～5 ④ 2～8

12、软件项目的进度管理有许多方法，但(A ②)不是常用的进度控制图示方法。在几种进度控制图示方法中，(B ①)难以表达多个子任务之间的逻辑关系，使用(C ③)不仅能表达子任务之间的逻辑关系，而且可以找出关键子任务。在(C ③)中，用带箭头的边表示(D ⑥)，用圆圈结点表示(E ③)，它标明(D ⑥)的(F ⑤)。

A～C. ① 甘特图 ② IPO ③ PERT ④ 时标网状图

D～F. ① 数据流 ② 控制流 ③ 事件 ④ 处理 ⑤ 起点或终点 ⑥ 任务

13、软件项目管理的主要职能包括：(A ②)，建立组织，配备人员，(B ④)和(C ⑥)。由于软件项目的特有性质，使得项目管理存在一定困难。第一、(D ②)，软件工程过程充满了大量高强度的脑力劳动；第二、(E ③)，在特定机型上，利用特定的硬件配置，由特定的系统软件和支撑软件支持，形成了特定的开发环境；第三、(F ⑤)，软件项目经历的各个阶段都深透了大量的手工劳动，远未达到自动化的程度；第四、(G ④)，用户要经过专门的培训，才能掌握操作步骤，且需要配备专职维护人员进行售后服务；第五、(H ①)，为高质量地完成软件项目，充分发掘人员的智力才能和创造精神。

在总结和分析足够数量失误的软件项目之后可知，造成软件失误的原因大多与(I ④)工作有关。在软件项目开始执行时，执行的过程中及项目进行的最后阶段都会遇到种种问题。

A～C. ① 编码 ② 制定计划 ③ 开发 ④ 指导 ⑤ 测试 ⑥ 检验

D～H.①软件工作渗透了人的因素 ②智力密集,可见性差 ③单件生产 ④ 使用方法繁琐，维护困难 ⑤ 劳动密集，自动化程度低

I.① 设计 ② 维护 ③ 测试 ④ 管理 ⑤ 实践 ⑥指导 ⑦审核 ⑧分析

14、软件项目组织的原则是(A ②)、(B ③)和(C ⑥)。一般有(D ③)、(E ④)、(F ①)三种组织结构的模式。(F ①)实际上是(D ③)和(E ④)两种模式的复合。(E ④)这种模式在小组之间的联系形成的接口较多，但便于软件人员熟悉小组的工作，进而成为这方面的专家。

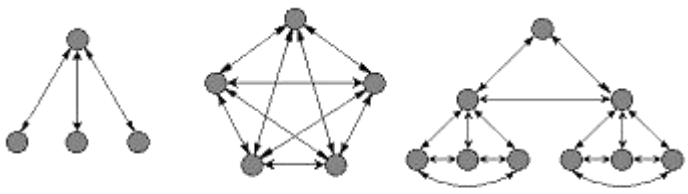
A～C. ① 推迟责任的落实 ② 尽早落实责任 ③ 减少接口 ④ 增加联系 ⑤ 责权分离 ⑥ 责权均衡

D～F. ① 矩阵形模式 ② 主程序员小组模式 ③ 按课题划分的模式 ④ 按职能划分的模式 ⑤ 民主制小组模式

15、软件开发小组的目的是发挥集体的力量进行软件研制。因此，小组从培养(A ②)的观点出发进行程序设计消除软件的(B ④)的性质。通常，程序设计小组的组织形式有三种，如下图所示的 a 属于(C ③)，b 属于(D ②)，c 属于(E ①)。

A, B. ① “局部” ② “全局” ③ “集体” ④ “个人”

C～E. ① 层次式小组 ② 民主制小组 ③ 主程序员制小组



（a）主程序员制小组 （b）民主制小组 （c）层次式小组

五、解释概念

1、 risk impact.

the loss associated with the risk 与该风险有关的损失。

2、 risk leverage

difference in risk exposure divided by cost of reducing the risk

3、milestone

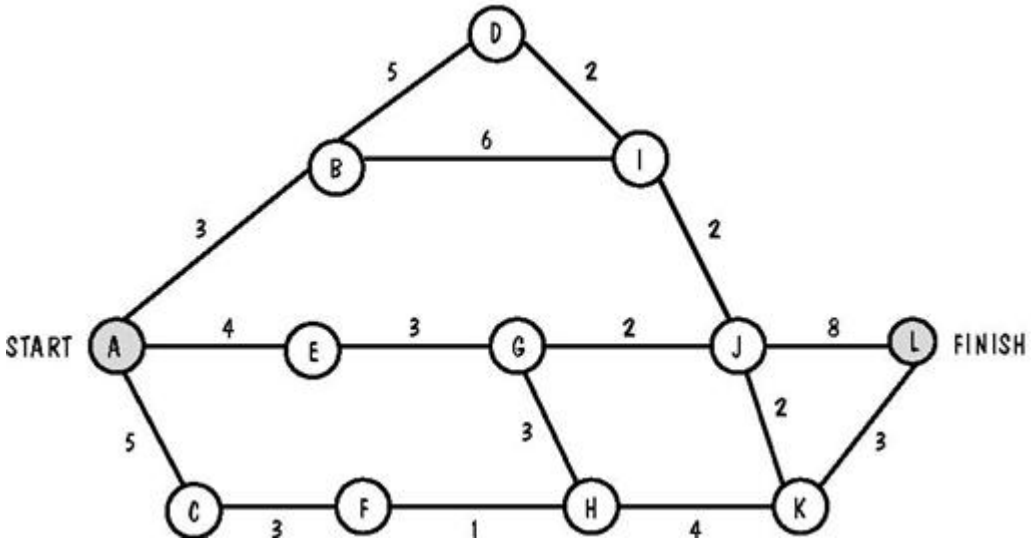
A milestone is the completion of an activity -- a particular point in time.

4、Precursor

event or set of events that must occur in order for an activity to start 在活动开始之前必须发生的一个或一组事件，它描述了活动开始的一组条件。

六、计算题

1、This is an activity graph for a software development project. The number corresponding to each edge of the graph indicates the number of days required to complete the activity represented by that branch. For example, it will take 3 days to complete the activity that ends in milestone B. For each activity, list its precursors, and compute the earliest start time, the latest start time, and the slack. Then, identify the critical path.



Earliest start Time: ES latest Start Time: LS

AB: Precursor{A} , ES: 1, LS:1 Slacktime:0;

BD: Precursor{B} , ES: 4, LS:4 Slacktime:0;

DI: Precursor{D} , ES: 9, LS:9 Slacktime:0;

BI: Precursor{B} , ES: 4, LS:5 Slacktime:1;

AE: Precursor{A} , ES: 1, LS:5 Slacktime:3;

EG: Precursor{E} , ES: 5, LS:8 Slacktime:3;

GJ: Precursor{G} , ES: 8, LS:11 Slacktime:3;

GH: Precursor{G} , ES: 8, LS:11 Slacktime:3;

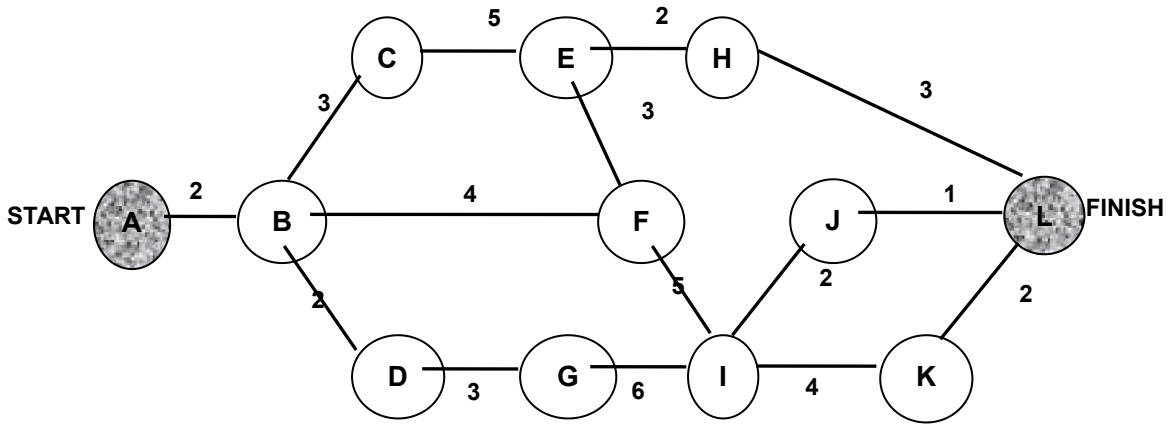
IJ: Precursor{B,D} , ES: 11, LS:11 Slacktime:0;

AC: Precursor{A} , ES: 1, LS:4 Slacktime:3;

CF: Precursor{A} , ES: 6, LS:9 Slacktime:3;

FH: Precursor{F} , ES: 9, LS:12 Slacktime:3;
HK: Precursor{H} , ES: 11, LS:14 Slacktime:3;
JK: Precursor{J} , ES: 13, LS:16 Slacktime:3;
JL: Precursor{J} , ES: 13, LS:13 Slacktime:0;
KL: Precursor{K} , ES: 15, LS:18 Slacktime:3;
The critical path: A-B-D-I-J-L

2、This is an activity graph for a software development project. The number corresponding to each edge of the graph indicates the number of days required to complete the activity represented by that branch. For example, it will take 2 days to complete the activity that ends in milestone B. For each activity, list its precursors, and compute the earliest start time, the latest start time, and the slack. Then, identify the critical path.



Activity graph

Earliest start Time: ES, latest Start Time:LS
AB: Precursor{A} , ES:1 , LS:1 Slacktime:0;
BC: Precursor{B} , ES:3 , LS:3 Slacktime:0;
CE: Precursor{C} , ES:6 , LS:6 Slacktime:0;
EF: Precursor{E} , ES:11 , LS:11 Slacktime:0;
FI: Precursor{F} , ES:14 , LS:14 Slacktime:0;
IK: Precursor{I} , ES:19 , LS:19 Slacktime:0;
KL: Precursor{K} , ES:23 , LS:23 Slacktime:0;
EH: Precursor{E} , ES:11 , LS:20 Slacktime:9;
HL: Precursor{H} , ES:13 , LS:22 Slacktime:9;
IJ: Precursor{I} , ES:19 , LS:23 Slacktime:3;
JL: Precursor{J} , ES:21 , LS:24 Slacktime:3;
BF: Precursor{B} , ES:3 , LS:10 Slacktime:7;
BD: Precursor{B} , ES:3 , LS:8 Slacktime:5;
DG: Precursor{D} , ES:5 , LS:10 Slacktime:5;
GI: Precursor{G} , ES:8 , LS:13 Slacktime:5;

Critical path: A-B-C-E-F-I-K-L

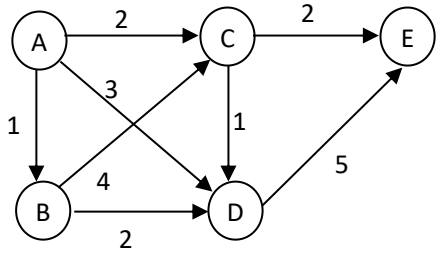
七、简答

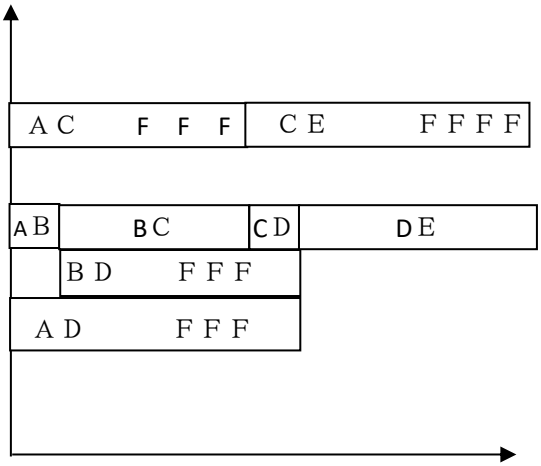
1、如果我们可以用“人力换时间”的办法缩短一个软件的开发时间，那么软件项目的开发时间最多可以减少到正常开发时间的多少？如果要求一个软件系统的开发时间过短，则开发成功的概率是多少？

假设个人最高生产率为500LOC/月,每条通信路径可导致生产率下降10%，如果每个组员都必须与组内所有其他组员通信,则项目组规模为多少人时与生产率的关系最佳，计算说明。

软件项目的开发时间最多可以减少到正常开发时间的７５％？如果要求一个软件系统的开发时间过短，则开发成功的概率０。

2、项目计划与管理的核心是如何制定开发进度的时间表，画出该项目的Gannt图





3. Three strategies for risk reduction

avoiding the risk（避免风险）： change requirements for performance or functionality 通过改变性能或功能需求。

transferring the risk（转移风险）:by allocating risks to other system or buy insurance to cover any financial loss should the become a reality. 通过把风险分配到其他系统中,或者购买保险以便在风险成为事实时能弥补经济上的损失

assuming the risk（假设风险会发生）: by accepting it and controlling it with the project’s resources 用项目资源承受和控制风险

4. Project plan contents

1.project scope 项目范围
2.project schedule 项目进度
3.project team organization 项目小组组织结构
4.technical description of the proposed system 项目的技术描述
5.project standards, procedures and proposed techniques and tools 项目标准、过程和建议的技术及工具
6.quality assurance plan 质量保证计划
7.configuration management plan 配置管理计划
8.documentation plan 文档计划
9.data management plan 数据管理计划

10. resource management plan 资源管理计划
11. test plan 测试计划
12. training plan 培训计划
13. security plan 安全计划
14. risk management plan 风险管理计划
15. maintenance plan 维护计划

第四章 需求获取

一、填空题

1、在软件需求分析阶段，分析人员要确定对软件的综合要求，其中最重要的是(**功能要求**) 。

2、需求分析阶段产生的最主要的文档是(**需求规格说明书**)。

3、解决一个复杂的问题，往往采取的策略是(**分解**)。

4、可行性研究的目的是用（**最小**）的代价，在尽可能（**短**）的时间内，确定该软件项目是否能够（**开发**)。

5、可行性研究实质上是进行一项（**简化**)、压缩了的需求分析、(**设计**）过程。

6、可以从 4 个方面研究可行性，即 （**技术经济**）可行性、(经济）可行性、（**社会**）可行性、(**解决方案**）可行性。

7、结构化分析方法从三个方面建模：（**数据**）建模、(**功能**）建模、(**行为**）建模。

8、实体－关系图用于（**数据**）建模，它最初用于（**数据库**）设计。

9、数据流图中的每一个加工至少有（一）个输入数据流和（一）个输出数据流。

10、状态－迁移图用于（**行为**）建模，状态中包含（**活动**)，状态因（**事件**）发生转移。

11、数据词典中有四类条目，分别为（**数据流**)、(**加工**)、(**数据存储**)、(**外部实体**)。

二、单项选择题

1、软件需求分析阶段的工作可以划分以下四个方面：对问题的识别、分析与综合、制定需求规格说明和__C__。

A.总结 B. 阶段性报告 **C. 需求分析评审** D. 以上答案都不正确

2、各种需求分析方法都有它们共同适用的__D___。

A. 说明方法 B. 描述方法 C. 准则 **D. 基本原则**

3、软件需求分析应从问题的信息域和功能域出发。信息域应包括信息流、信息内容和 __C__。

A. 信息项 B. 数据结构 **C. 信息结构** D. 信息内容

4、需求分析产生的文档是 __C__。

A.项目开发计划 B.可行性分析报告 **C.需求规格说明书** D.软件设计说明书

5、需求分析中，分析人员要从用户那里解决的最重要的问题是__A__。

A. 要让软件做什么 B. 要给该软件提供什么信息

C. 要求软件工作效率如何 D. 要让该软件具有何种结构

6、可行性研究的目的是__B__。

A.开发项目 **B.项目值得开发否** C.规划项目 D. 维护项目

7、技术可行性要解决_D__。

A. 存在侵权否 B. 成本效益问题 C.运行方式可行 **D. 技术风险问题**

8、研究开发资源的有效性属于 _A__ 可行性的一部分。

A. 技术 B. 经济 C. 社会 D. 操作

9、在可行性研究过程中，对每一个合理的候选方案，分析人员都应准备如下资料 __D__。

A. 系统流程 B. 组成系统的物理元素清单、成本－效益分析

C. 实现该系统的进度计划 **D. 以上全部**

10、软件需求分析的任务不应包括__C__。

A. 问题分析 B.信息域分析 **C.结构化程序设计** D.确定逻辑模型

11、结构化语言、判定表和判定树属于__A__规格说明的描述工具。

A. 加工 B. 控制 C. 数据描述 D. 脚本

12、分层数据流图是一种比较严格又易于理解的描述方式，它的顶层数据流图描述了系统的__B__。

A. 细节 **B. 输入与输出** C. 软件的作者 D. 绘制的时间

13、对于分层的数据流图，父图与子图的平衡是指子图的输入、输出数据流同父图的输入、输出数据流__A__。

A. 必须一致 B. 数目必须相等 C.名字必须相同 D. 数目必须不等

14、在数据流图的基本图形符号中，加工是以信息结构或__B__作为加工对象的。

A. 数据结构 **B. 信息内容** C. 信息流 D. 数据内容

15、一个局部数据存储当它作为__D__时就把它画出来。

A.某些加工的数据接口 B.某个加工的特定输入 C.某个加工的特定输出

D.某些加工的数据接口或某个加工的特定输入/输出

16、软件需求规格说明书的内容不应包括对__B__的描述。

A.主要功能 **B.算法的详细过程** C.用户界面及运行环境 D.软件的性能

17、需求规格说明书的作用不应包括__B___。

A. 软件设计的依据 **B. 软件可行性研究的依据**

C. 软件验收的依据 D. 用户和开发人员对软件要做什么的共同理解

28、快速原型化思想是在研究_D__阶段的方法技术中产生的。

A. 可行性研究 B. 软件设计 C. 程序编码 **D. 需求分析**

19、用于整个开发阶段，及早提供一个原型系统的是 _D___原型。

A. 实验型 B. 探索型 C. 提交型 **D. 演化型**

20、用于软件设计阶段，考察实现方案是否可行的是__C___原型。

A. 探索型 B. 演化型 **C. 实验型** D. 增量型

三、选择填空

1、软件需求分析的任务不应包括(A，③)。进行需求分析可使用多种工具，但(B，③)是不适用的。在需求分析中，分析员要从用户那里解决的最重要的问题是(C,①)。需求规格说明书的内容不应当包括(D ,②)。该文档在软件开发中具有重要的作用，但其作用不应当包括(E ,④)。

A. ① 问题分析 ② 信息域分析 ③ 结构化程序设计 ④ 确定逻辑模型

B. ① 数据流图 ② 判定表 ③ PAD 图 ④ 数据词典

C. ① 要让软件做什么 ② 要给该软件提供哪些信息
③ 要求软件工作效率如何 ④ 要让软件具有什么样的结构

D. ① 对重要功能的描述 ② 对算法的详细过程性描述

③ 软件确认准则　　④ 软件的性能

E.① 软件设计的依据 ② 用户和开发人员对软件要“做什么”的共同理解

③ 软件验收的依据 ④ 软件可行性分析的依据

2、当前系统的（　　A，②）模型描述现行系统的实际业务处理过程，反映了现行系统具体（　B,　①）的现实。当前系统的（　C　，　⑤）模型描述现行系统的功能结构、数据组织以及动态行为，反映了现行系统（　D　，　③）的本质。

目标系统是指待开发的新系统。根据计算机系统的特点，分析、比较目标系统和当前系统逻辑上的差别，确定目标系统的软件工作范围，采用自顶向下逐步分解的分析策略，确定目标系统的功能结构、数据组织以及动态行为，从而建立起目标系统的（　E　，　⑤　）模型。

A, C, E：① 对象　② 物理　③ 服务　④ 过程　⑤ 逻辑

B, D：　① 怎么做　② 何时做　③ 做什么　④ 为何做　⑤ 谁来做

3、结构化分析模型从多视角来描述系统。在分析模型的核心是（　A　，②），它描述了所有在目标系统中使用和生成的数据对象。围绕着这个核心有三种图：（　B　，　⑧）、（　C,　④　）和（　D，　③）。（B　，　⑧）描述数据对象及其关系，用于建立数据模型；（C,　④　）描述数据在系统中如何被传递和变换，用于建立功能模型，同时还需要给出加工规格说明；（　D，　③　）描述系统对外部事件如何响应，用于建立行为模型，同时还需要给出控制规格说明。

Petri 网主要用于描述相互独立，协同操作的处理系统，即（ E,　① ）的处理系统。

A～D：①对象图　②数据词典　③状态迁移图　④数据流程图　⑤时序图

⑥事件追踪图　⑦控制流程图　⑧实体关系图　⑨仿真图　⑩行为图

E：　①并发执行　②事件驱动　③时钟驱动　④随机执行　⑤ 顺序执行

4、软件需求分析方法必须能够理解和表达问题领域的信息域和功能域。信息域包括（　A　，　⑦）、（　B,　②）和（C　，　⑥）。

（　A　，　⑦））表示数据和控制在系统中传递时的变化方式。输入对象首先被变换成数据和控制的（　D　，　③）信息，然后再变换成输出结果信息。

（B,　②　）表示信息在计算机中的组织形式。各种数据和控制对象按什么逻辑关系组织在一起，又按什么物理关系存储在计算机中，必须靠（B,　②）分析来解决。

（　C　，　⑥　）可以利用数据词典明确地表示，也可以通过数据或数据对象的层次结构隐含地表示。

对数据进行变换就是程序所表现的功能。两个功能之间的数据传递确定了功能之间的（　E　，　②　）。

A～C：①信息属性　②信息结构　③信息服务　④信息通信　⑤信息抽象

⑥信息内容　⑦信息流　⑧信息层次　⑨信息项　⑩信息行为

D～E：①连接　②接口　③中间　④通讯　⑤ 联系

5、原型化方法是用户和软件开发人员之间进行的一种交互过程，适用于（　A，①　　）系统。它从用户界面的开发入手，首先形成（　B　，③　　），用户（　C，　④　　），并就（　D，　①　）提出意见，它是一种（　E，①）型的设计过程。

A.　① 需求不确定性高的　② 需求确定的　③ 管理信息　④ 决策支持

B.　① 用户界面使用手册　② 用户界面需求分析说明书

③ 系统界面原型　　④ 完善的用户界面

C. ① 改进用户界面的设计　　② 阅读文档资料
③ 模拟用户界面的运行　　④ 运行用户界面原型

D. ① 同意什么和不同意什么　② 使用和不使用哪一种编程语言
③ 程序的结构　　④ 执行速度是否满足要求

E. ① 自外向内　② 自顶向下　③ 自内向外　④ 自底向上

四、问答题

1、在软件需求分析时，首先建立当前系统的物理模型，再根据物理模型建立当前系统的逻辑模型。试问：什么是当前系统？当前系统的物理模型与逻辑模型有什么差别？

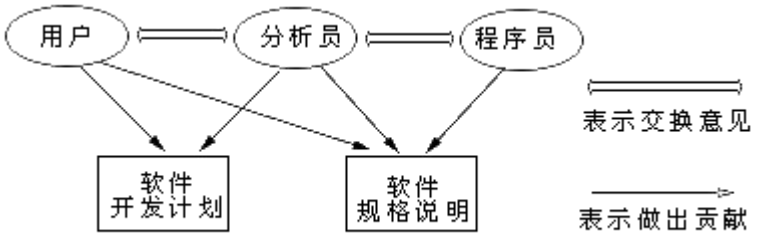
所谓当前系统可能是需要改进的某个已在计算机上运行的数据处理系统，也可能是一个人工的数据处理过程。

当前系统的物理模型客观地反映当前系统实际的工作情况。但在物理模型中有许多物理的因素，随着分析工作的深入，有些非本质的物理因素就成为不必要的负担，因而需要对物理模型进行分析，区分出本质的和非本质的因素，去掉那些非本质的因素即可获得反映系统本质的逻辑模型。所以当前系统的逻辑模型是从当前系统的物理模型抽象出来的。

2、软件需求分析是软件工程过程中交换意见最频繁的步骤。为什么交换意见的途径会经常阻塞？

软件需求分析过程中，由于最初分析员对要解决的问题了解很少，用户对问题的描述、对目标软件的要求也很凌乱、模糊，再加上分析员和用户共同的知识领域不多，导致相互间通信的需求。但是由于分析员和用户之间需要通信的内容相当多，业务知识上的不足，表达方式的不足，可能对某些需求存在错误解释或误解的可能性，造成需求的模糊性。另外，用户和分析员之间经常存在无意识的“我们和他们”的界限，不是按工作需要组成统一的精干的队伍，而是各自定义自己的“版图”，并通过一系列备忘录、正式的意见书、文档，以及提问和回答来相互通信。历史已经证明，这样会产生大量误解。忽略重要信息，无法建立成功的工作关系。

3、你认为一个系统分析员的理想训练和基础知识是什么？请说明理由。系统分析员处在用户和高级程序员之间，负责沟通用户和开发人员的认识和见解，起着桥梁的作用。一方面要协助用户对所开发的软件阐明要求，另一方面还要与高级程序员交换意见，探讨用户所提要求的合理性以及实现的可能性。最后还要负责编写软件需求规格说明和初步的用户手册。



为能胜任上述任务，分析员应当具备如下的素质:

(1) 能够熟练地掌握计算机硬、软件的专业知识，具有一定的系统开发经验。

(2) 善于进行抽象的思维和创造性的思维，善于把握抽象的概念，并把它们重新整理成为各种逻辑成分，并给出简明、清晰的描述。

(3) 善于从相互冲突或混淆的原始资料中抽出恰当的条目来。

(4) 善于进行调查研究，能够很快学习用户的专业领域知识，理解用户的环境条件。

(5) 能够倾听他人的意见，注意发挥其它人员的作用。

(6) 具有良好的书面和口头交流表达能力。

4、可行性研究主要研究哪些问题？试说明之。

可行性研究主要做 4 个方面的研究：

经济可行性：进行成本 / 效益分析。从经济角度判断系统开发是否“合算”。

技术可行性：进行技术风险评价。从开发者的技术实力、以往工作基础、问题的复杂性等出发，判断系统开发在时间、费用等限制条件下成功的可能性。

法律可行性：确定系统开发可能导致的任何侵权、妨碍和责任。

方案的选择：评价系统或产品开发的几个候选方案。最后给出结论意见。

5、信息和信息结构有什么区别？有没有不存在信息流的系统？有没有不存在信息结构的系统？

信息是宇宙三要素（物质、能量、信息）之一。它是现实世界各种事物在人们头脑中的反映。此外，人们通过科学仪器能够认识到的也是信息。信息的特征为：可识别、可存储、可变换、可处理、可传递、可再生、可压缩、可利用、可共享。信息域就是对信息的多视角考虑。信息域包含 3 个不同的视图：信息内容和关系、信息流和信息结构。为了完全理解信息域，必须了解每一个视图。

信息结构：它是信息在计算机中的组织形式。一般表示了各种数据和控制对象的内部组织。数据和控制对象是被组织成 n 维表格，还是组织成有层次的树型结构?　在结构中信息与其它哪些信息相关?　所有信息是在一个信息结构中，还是在几个信息结构中?　一个结构中的信息与其它结构中的信息如何联系?　这些问题都由信息结构的分析来解决。

信息流：表示数据和控制在系统中传递时的变化方式。输入对象首先被变换成中间信息（数据或控制），然后再变换成输出结果信息。沿着变换路径，可能从已有的数据存储（如磁盘文件或内存缓冲区）中引入附加的信息。对数据进行变换是程序中应有的功能或子功能。两个变换功能之间的数据传递就确定了功能间的接口。

所以，没有信息流的系统相当于没有功能的系统，这样的系统的存在是毫无意义的。而没有信息结构的系统是没有信息的系统，这样的系统不是计算机能够处理的系统。

6、有人说：软件开发时，一个错误发现得越晚，为改正它所付出的代价就越大。对否？请解释你的回答。软件开发时，一个错误发现得越晚，为改正它所付出的代价就越大。这个说法是对的。在 1970 年代，GTE、TRW 和 IBM 等三家公司对此问题做了独立研究，最后它们得到相似的结论：

阶段	需求分析	软件设计	程序编码	单元测试	验收测试	维护
相对修复代价	0.1~0.2	0.5	1	2	5	20

从表中可以看出，在需求分析阶段检查和修复一个错误所需的代价只有编码阶段所需代价的 1/5 到 1/10，而在维护阶段做同样的工作所付出的代价却是

编码阶段的 20 倍。

7、软件需求分析的操作性原则和需求工程的指导性原则是什么？

软件需求分析的操作性原则指所有的需求分析方法都与一组操作性原则相关联：

必须理解和表示问题的信息域。

必须定义软件将完成的功能。

必须表示软件的行为（作为外部事件的结果）。

必须对描述信息、功能和行为的模型进行分解，能够以层次方式揭示其细节。

分析过程应当从要素信息转向细节的实现。

通过使用这些原则，分析员可以系统地处理问题。首先检查信息域以便更完整地理解目标软件的功能，再使用模型以简洁的方式表达目标软件的功能和行为，并利用自顶向下、逐层分解的手段来降低问题的复杂性。在这些处理过程中，因处理需求带来的逻辑约束和因其它系统元素带来的物理约束需要通过软件要素和视图的实现加以检验和确认。

Davis 建议了一组针对“需求工程”的指导性原则：

在开始建立分析模型之前应当先理解问题。如果问题没有很好理解就急于求成，常常会产生一个解决错误问题的完美的软件。

强力推荐使用原型。这样做可以使用户了解将如何与计算机交互，而人们对软件质量的认识常常是基于对界面“友好性”的切身体会。

记录每一个需求的起源和原因。这是建立对用户要求的可追溯性的第一步。

使用多个视图，建立系统的数据、功能和行为模型。这样做可帮助分析员从多方面分析和理解问题，减少遗漏，识别可能的不一致之处。

给需求赋予优先级。因为过短的时限会减少实现所有软件需求的可能性。因此，对需求排定一个优先次序，标识哪些需求先实现，哪些需求后实现。注意消除歧义性。因为大多数需求都是以自然语言描述，存在叙述的歧义性问题，造成遗漏和误解。采用正式的技术评审是发现和消除歧义性的好方法。

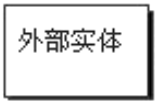
遵循以上原则，就可能做好软件需求规格说明，为软件设计奠定基础。

8、数据流图的作用是什么？它有哪些基本成份？

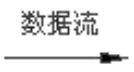
数据流图可以用来抽象地表示系统或软件。它从信息传递和加工的角度，以图形的方式刻画数据流从输入到输出的移动变换过程，同时可以按自顶向下、逐步分解的方法表示内容不断增加的数据流和功能细节。因此，数据流图既提供了功能建模的机制，也提供了信息流建模的机制，从而可以建立起系统或软件的功能模型。

数据流图的基本成份有 4 种：

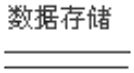
位于被建模系统之内的数据加工，输入数据在此进行变换产生输出数据，其中要注明加工的名字。



位于被建模系统之外的信息生产者（数据源点）和信息消费者（数据汇点），其中要注明源点与汇点的名字。



数据流对象，被加工数据及其流向，箭头边应注明数据流名字，可用名词或名词性短语命名。



数据存储对象，也需加以命名，用名词或名词性短语命名。

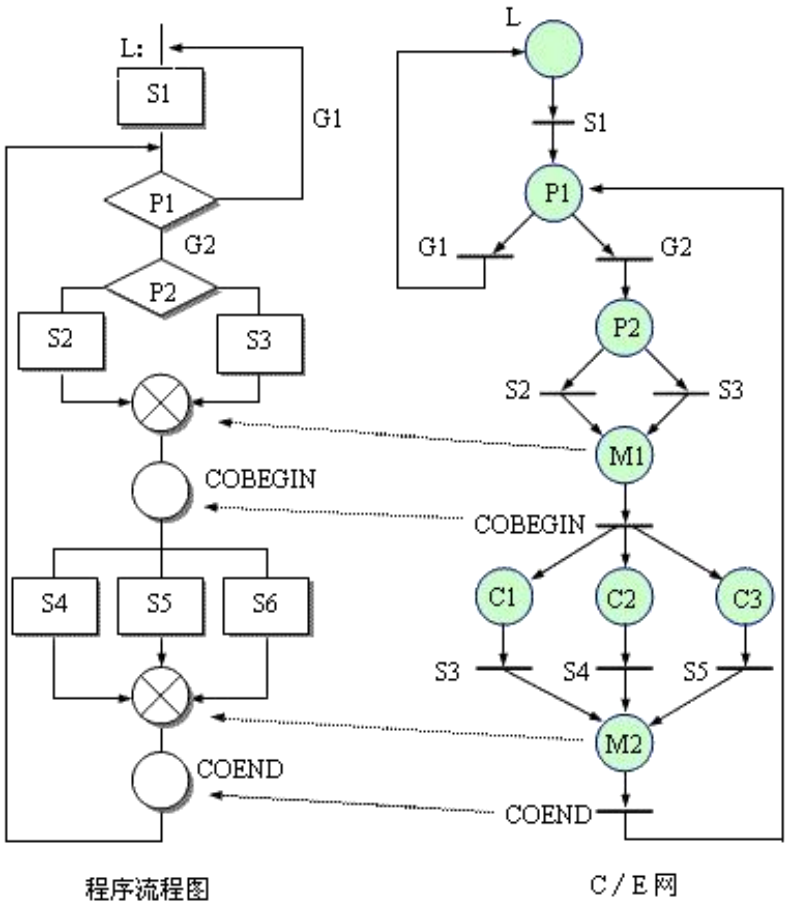
9、Petri 网可以描述计算机软件系统的执行。现有一个程序如下（类似于 Pascal 语言）

S6 是单个执行语句，COBEGIN 和 COEND 是并行执行开始和并行执行结束（即 S4，S5 和 S6 语句并行执行）。试用 Petri 网描述这段程序的执行过程。

采用条件／事件网（C／E 网，C—Condition，E—Event）式 Petri 网。

（程序：）

```
L : S1;
   WHILE  P1  DO
     BEGIN
IF  P2  THEN  S2
                ELSE  S3;
          COBEGIN
            S4;
            S5;
            S6;
          COEND
        END;
   GOTO  L;
```



程序流程图　　C／E网

10、数据词典的作用是什么？它有哪些基本词条？

分析模型中包含了对数据对象、功能和控制的表示。在每一种表示中，数据对象和控制项都扮演一定的角色。为表示每个数据对象和控制项的特性，建立了数据词典。数据词典精确地、严格地定义了每一个与系统相关的数据元素，并以字典式顺序将它们组织起来，使得用户和分析员对所有的输入、输出、存储成分和中间计算有共同的理解。

在数据词典的每一个词条中应包含以下信息：

名称：数据对象或控制项、数据存储或外部实体的名字。

别名或编号。

分类：数据对象, 加工, 数据流, 数据文件, 外部实体, 控制项（事件／状态）

描述：描述内容或数据结构等。

何处使用：使用该词条（数据或控制项）的加工。

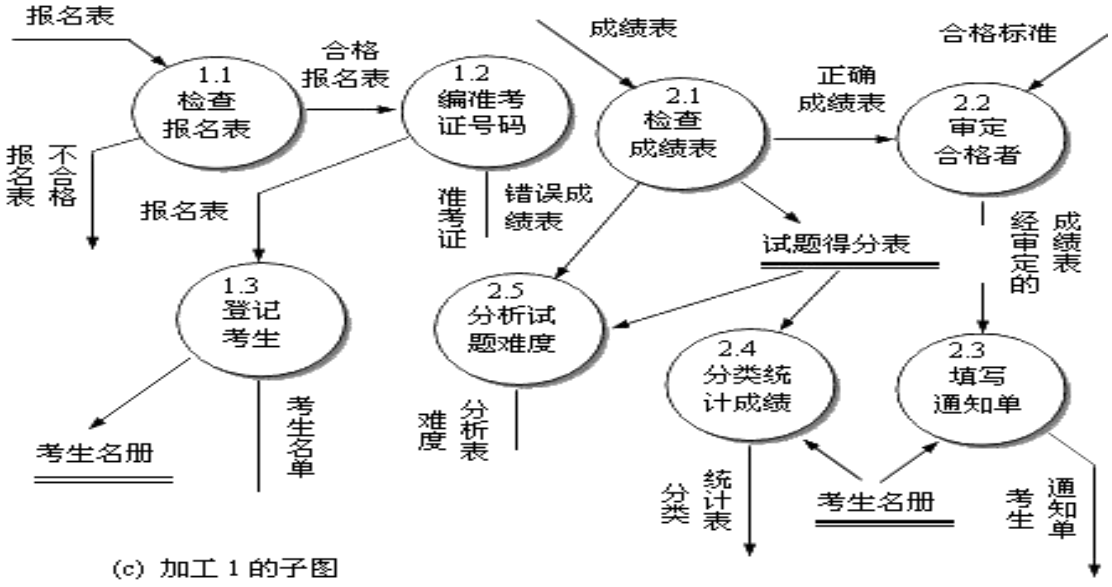
11、软件需求分析说明书主要包括哪些内容?

软件需求规格说明是分析任务的最终产物，通过建立完整的信息描述、详细的功能和行为描述、性能需求和设计约束的说明、合适的验收标准，给出对目标软件的各种需求。

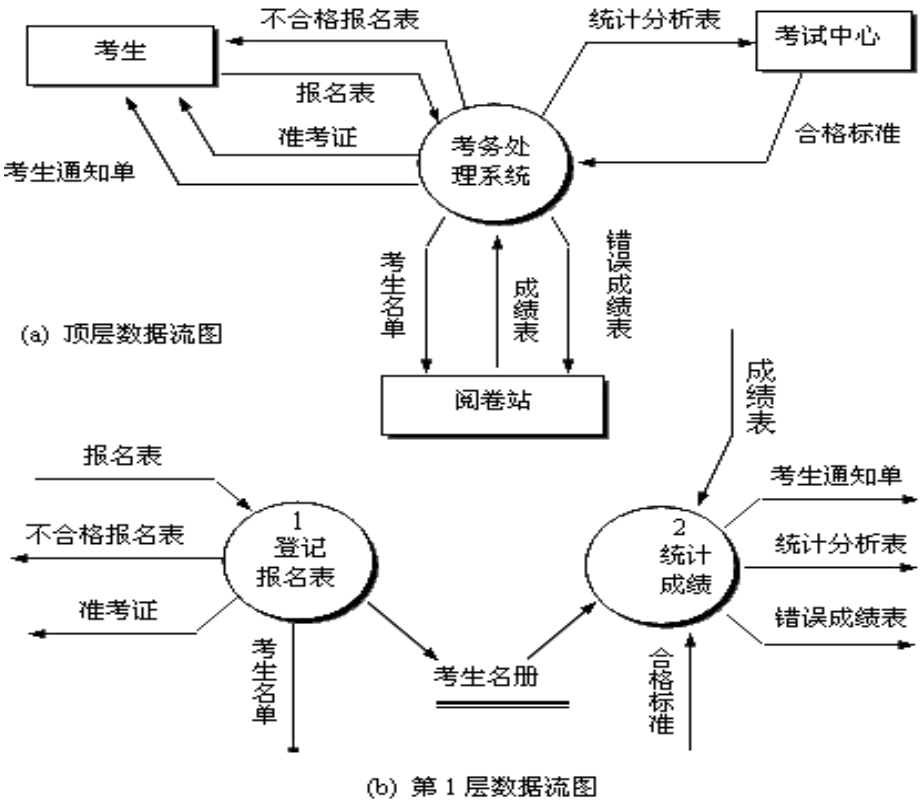
软件需求规格说明的框架如下：

Ⅰ.引言　A.系统参考文献　B.整体描述　C.软件项目约束
Ⅱ.信息描述　A.信息内容表示　B.信息流表示　i 数据流　ii 控制流
Ⅲ.功能描述 A.功能划分 B.功能描述 i处理说明 ii 限制／局限 iii 性能需求 iv 设计约束 v支撑图 C. 控制描述 i 控制规格说明 ii 设计约束
Ⅳ.行为描述　A.系统状态　B.事件和响应
Ⅴ.检验标准 A.性能范围 B.测试种类 C.期望的软件响应 D.特殊的考虑
Ⅵ.参考书目
Ⅶ.附录

12、考务处理系统的分层数据流图如下图所示。



(c) 加工 1 的子图　　(d) 加工 2 的子图



该考务处理系统有如下功能：

对考生送来的报名表进行检查；

对合格的报名表编好准考证号码后将准考证送给考生，并将汇总后的考生名单送给阅卷站；对阅卷站送来的成绩表进行检查，并根据考试中心指定的合格标准审定合格者；

填写考生通知单（内容包含考试成绩及合格／不合格标志），送给考生；

按地区、年龄、文化程度、职业、考试级别等进行成绩分类统计及试题难度分析，产生统计分析表。

（1）图(c)中，加工 1.1 的输入数据流是(　A　②　)，输出数据流是(　B　⑤　),图(b)中,加工 2 的输出数据流是(　C　①　),它是由(　D　⑥　)和(　E　⑧　)组成。

A～E.　① 统计分析表　② 报名表　③ 准考证　④ 考生通知单
⑤合格报名表　⑥ 难度分析表 ⑦ 错误成绩表 ⑧ 分类统计表

(2) 图(d)中的文件“试题得分表”是否在图(b)中漏掉了？回答是(　F　②　)。

F:① “试题得分表”没有在图(b)中画出，是错误的。
② “试题得分表”是图(b)中加工的内部文件，不必在图(b)中画出。
③ “试题得分表”是多余的。

应注意的问题：

① 适当地为数据流、加工、文件、数据的源／汇点命名。名字应反映该元素的实际含义，避免空洞的名字。如数据、信息处理、计算等名字都不好。

② 画数据流时不要夹带控制流。数据流图中各种数据的加工没有考虑时序关系，引入控制流后，加工之间就有了时序关系，这与画数据流图不考虑实现细节的初衷相违背。

③ 一个加工的输出数据流不要与该加工的输入数据流重名，即使它们的组成成分相同。例如图(c)中加工 1.1 的输入数据流“报名表”与输出数据流“合格报名表”。

④ 允许一个加工有多个数据流流向另一个加工，也允许一个加工有两个相同的输出数据流流向两个不同的加工。

⑤ 保持父图与子图的平衡。就是说，父图与它的子图的输入数据流与输出数据流应当在数量与名字上都相同。特别的是，如果父图的一个输入（或输出）数据流对应于子图中几个输入（或输出）数据流，但子图中这几个数据流中的数据项合起来正好是父图中的那个数据流，这时它们还算是平衡的。例如，图(b)中加工 2 的输出数据流“统计分析表”是由“难度分析表”和“分类统计表”组成，那么图(b)与图(d)仍满足父图与子图平衡的条件。

⑥ 在自顶向下的分解过程中，若一个文件首次出现时只与一个加工有关，那么这个文件应作为这个加工的内部文件而不必画出。例如，图(d)中的文件“试题得分表”就是图(b)中加工的内部文件，所以在图(b)中没有画出。

⑦ 保持数据守恒。就是说，一个加工的所有输出数据流中的数据必须能从该加工的输入数据流中直接获得，或者是通过该加工产生的数据。

13、阅读下列关于开发人事管理系统的交互式工作方式的叙述，再回答问题。

某大企业最近决定采用高性能微机开发人事管理系统，将四台联机终端分置于人事处的三个科室。该系统可供操作员和程序员使用，也可供人事处负责人和主管人事的副厂长等查询人事信息用。人事管理系统通过录入人事数据和修改、删除等操作，产生和更新各类人事文件，通过搜索这些文件进行各类人事信息的查询。

该企业有 3000 多个工人、干部和技术人员，大体可分成机关科室、生产车间、后勤服务和开发研制部门等几类部门。厂领导决定由计算机应用科来负责协调和开发应用系统。计算机应用科科长指示系统工程师张某负责进行系统分析。

考虑到人事处有大量的查询信息要求、频繁的人事信息修改和文件存档、查阅等特点，计算机应用科决定认真设计人机交互界面，首先设计好在终端上的交互式会话的方式。

系统工程师张某通过调查收集到如下 10 条意见：

(1) 某程序员认为：系统在屏幕格式、编码等方面应具有一致性和清晰性，否则会影响操作人员的工作效率。

(2) 某操作人员认为：在交互式会话过程中，操作人员可能会忘记或记错某些事情，系统应当提供 HELP 功能。

(3) 某操作人员认为：既然是交互式会话，那么对所有的输入都应当作出响应，不应出现击键后，计算机没有任何反应的情况。

(4) 某操作人员认为：在出错的时候，交互式会话系统应当给出出错信息，并且尽可能告诉我们出错的性质和错在什么地方。

(5) 某程序员认为：终端会话也应当符合程序员编制程序时的习惯，这样可以更高效地维护人事管理系统。

(6) 教育科干部甲认为：应当对操作员进行一些必要的培训，让他们掌握交互式会话系统的设计技巧，有助于提高系统的使用效率。

(7) 教育科干部乙认为：尽管操作人员的指法已经强化训练但在交互式会话时应尽可能缩短和减少操作员输入的信息，以降低出错概率。

(8) 某程序员认为：由于本企业中有很多较大的文件，文件的查找很费时

间，交互式会话系统在响应时间较长时应给予使用者以提示信息。

(9) 人事处干部丙认为：我们企业的人事资料相当复杂，格式非常之多，希望交互式系统使用十分清晰的格式，并容易对输入数据中的错误进行修改。

(10) 人事处干部丁认为：人事管理系统应当具有相当的保密性和数据安全性，因此在屏幕上显示出的信息应该含混一些，以免泄密。

系统工程师张某对上述调查情况和其他要求作了分析后，发现收集到的 10 条意见中有 3 条意见是不能接受的，写出编号并各用 40 字以内叙述理由。

不能接受的 3 条意见是 (5)、(6)、(10)。人机交互界面首先考虑的是用户如何使用起来方便，与编程习惯、设计技巧无关。此外，屏幕上信息应很清晰易懂，安全保密与屏幕显示无关。

14、工资计算系统中的一个子系统有如下功能：

计算扣除部分—由基本工资计算出应扣除（比如水电费、缺勤）的部分；

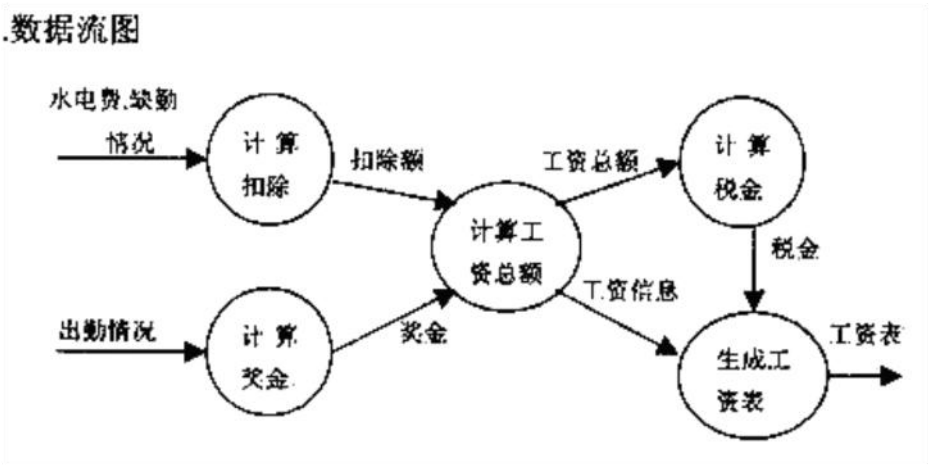
计算奖金部分—根据职工的出勤情况计算出奖励金；

计算工资总额部分—由工资总额中计算出应扣除各种税金；

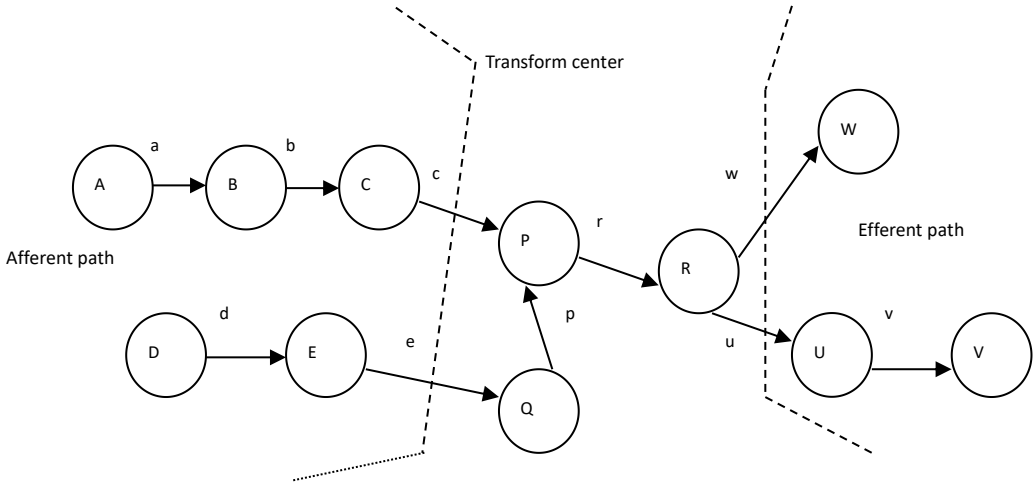
生成工资表—根据计算总额部分和计算税金部分传递来有关职工工资的详细信息生成工资表

根据要求画出该问题的数据流程图。

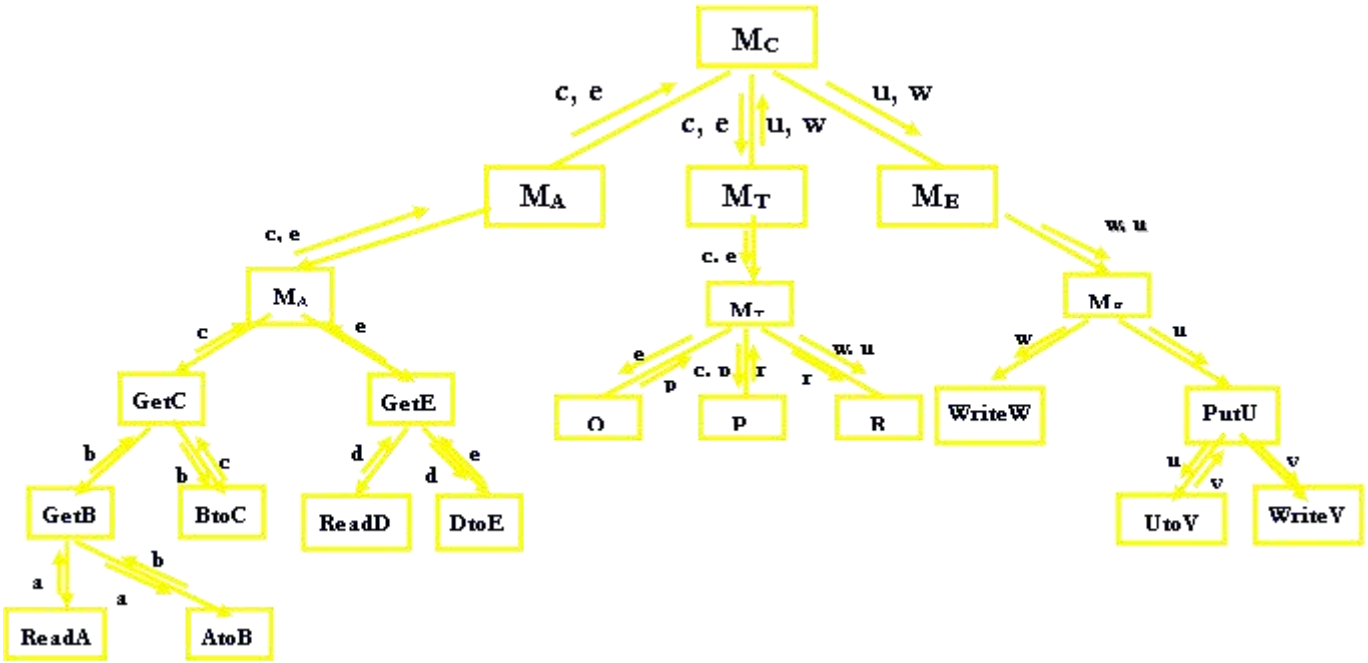
.数据流图



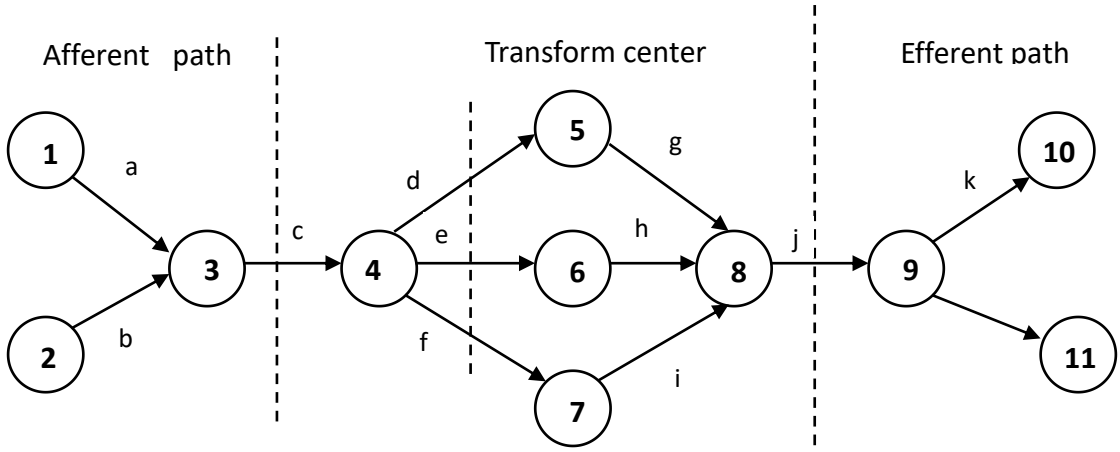
15、Please map the following data flow diagram (DFD) to structure chart (SC) using transform analysis. The afferent path, transform center and efferent path are labeled in the DFD.



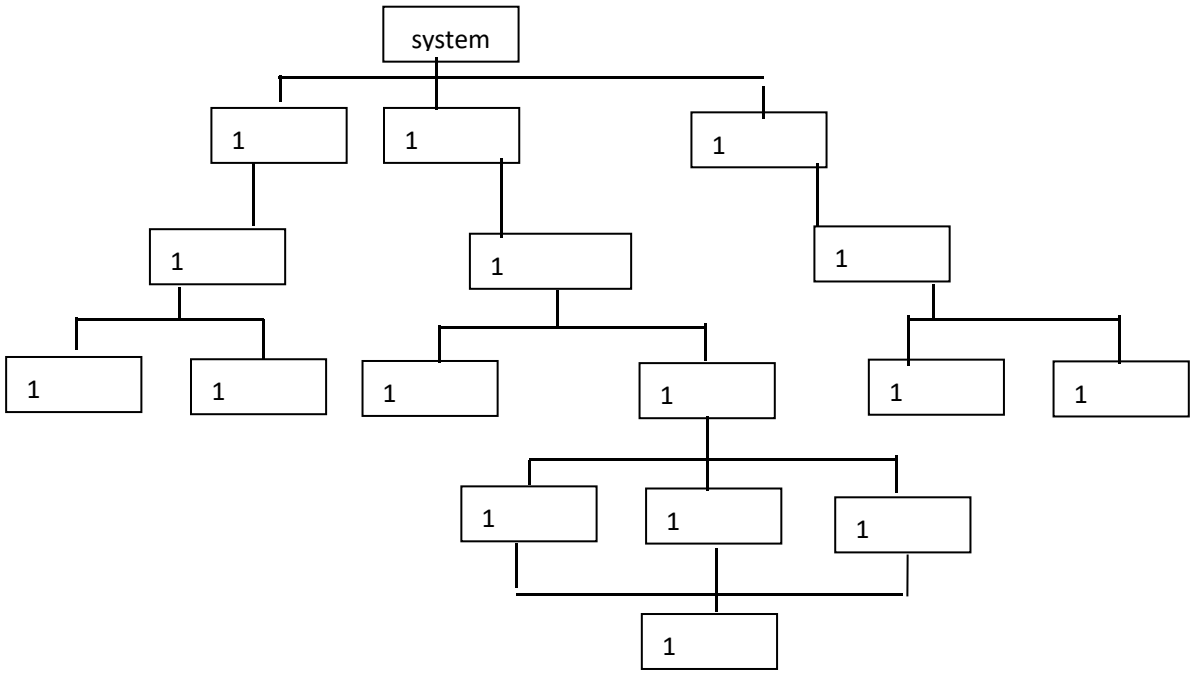
第 1 步　复查基本系统模型。
第 2 步　复查并精化数据流图。
第 3 步　确定数据流图具有变换特性还是事务特性。
第 4 步　确定输入流和输出流的边界，从而孤立出变换中心。
第 5 步　完成“第一级分解(first level factoring)”。
第 6 步　完成“第二级分解”。
第 7 步使用设计度量和启发式规则对第一次分割得到的软件结构进一步精化。



16、Please map the following data flow diagram (DFD) to structure chart (SC) using transform analysis. The afferent path, transform center and efferent path are labeled in the DFD.



第 1 步　复查基本系统模型。
第 2 步　复查并精化数据流图。
第 3 步　确定数据流图具有变换特性还是事务特性。
第 4 步　确定输入流和输出流的边界，从而孤立出变换中心。
第 5 步　完成“第一级分解(first level factoring)”。
第 6 步　完成“第二级分解”。
第 7 步　使用设计度量和启发式规则对第一次分割得到的软件结构进一步精化



17、某培训中心要研制一个计算机管理系统。它的业务是：

将学员发来的信件收集分类后，按几种不同的情况处理。

1)如果是报名的，则将报名数据送给负责报名事务的职员，他们将查阅课程文件，检查该课程是否额满，然后在学生文件、课程文件上登记。

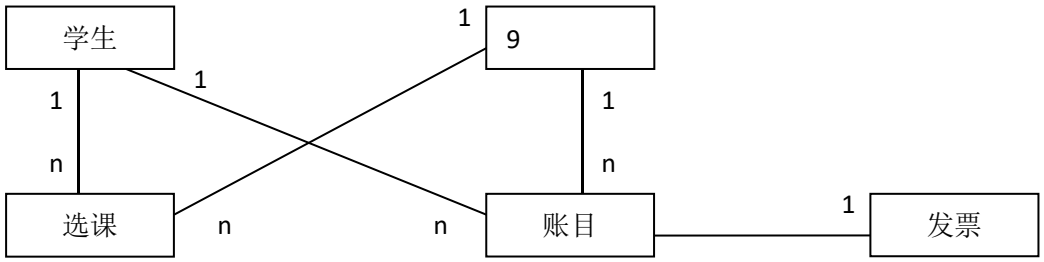
2)如果是想注销原来已选修的课程，则由注销人员在课程文件、学生文件和帐目文件上做相应的修改，并给学生注销单。

3)如果是付款的，则由财务人员在帐目文件上登记，并开出发票给学生。

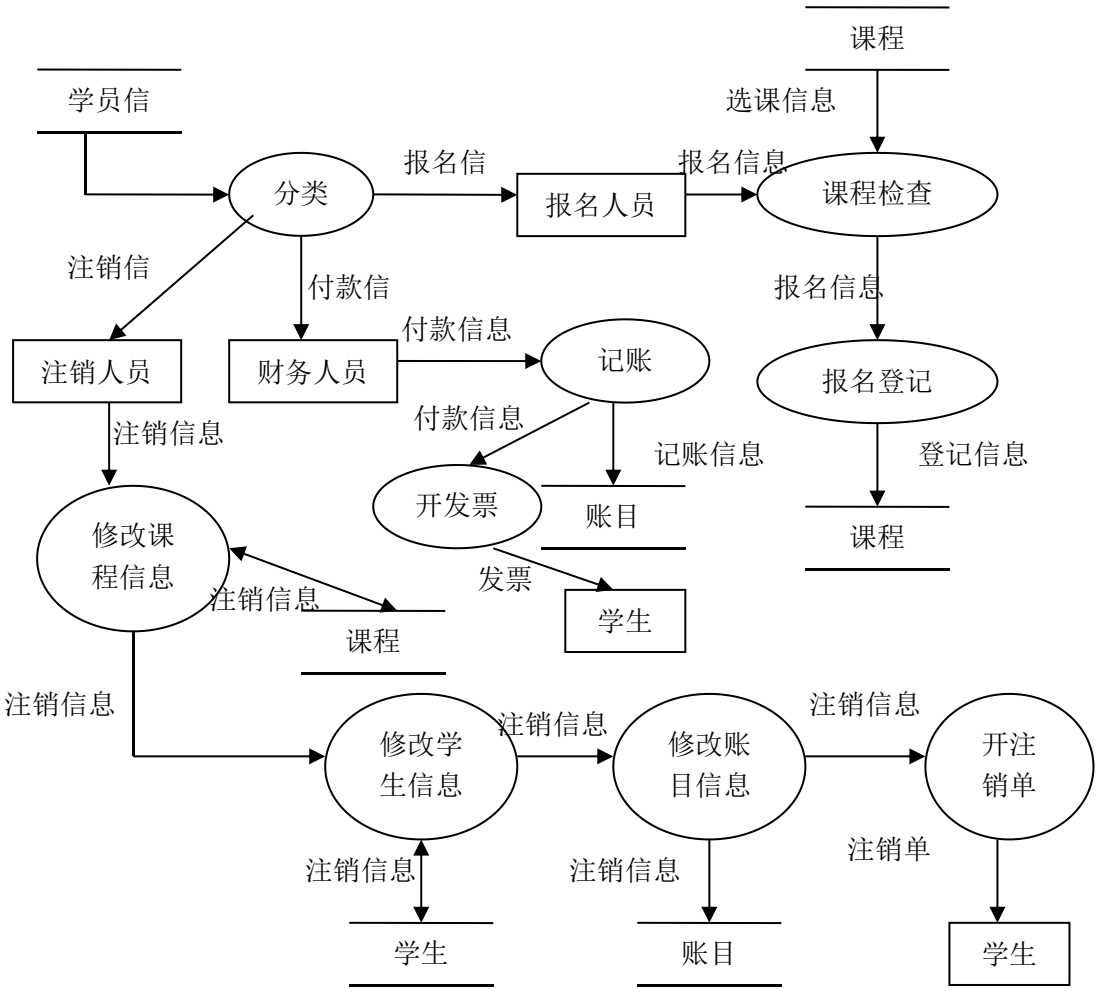
注：一个学生可选多门课程

要求：

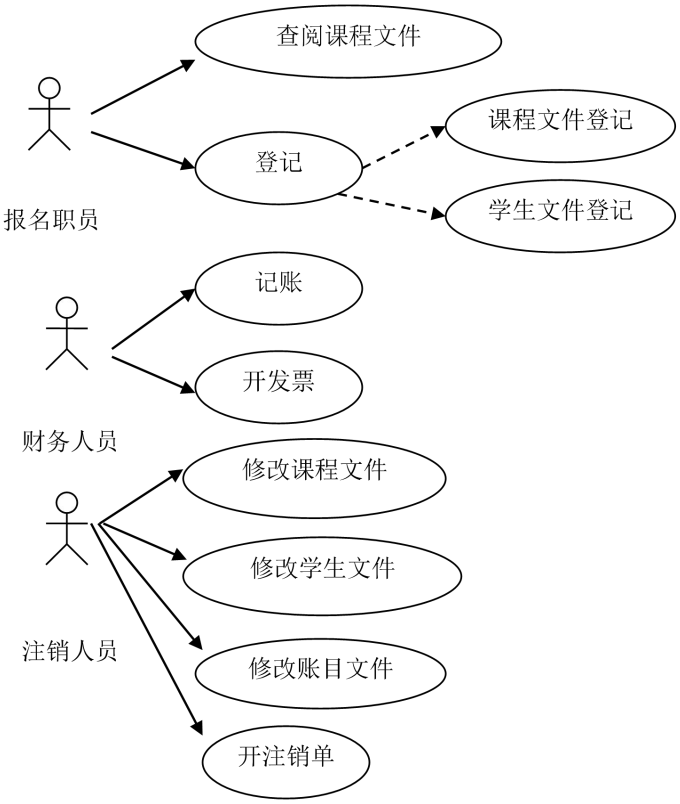
1. **画出 E-R 图**



2.对以上问题画出数据流图。



3. 画出系统的用例图。


查阅课程文件
登记
课程文件登记
学生文件登记
报名职员
记账
开发票
财务人员
修改课程文件
修改学生文件
修改账目文件
开注销单
注销人员

第五章 系统设计

一、判断题

1、 The results of decomposition form composite parts called modules or components.（T）

2、Cohesion refers to the internal “glue” with which a component is constructed.（T）

3、We say that two components are loosely coupled when there is a great deal of dependence between them.（F）

4、Design is the creative process of transforming the problem into a solution.（T）

二、解释概念

1、 what is design?

Design is the creative process to transform the problem into a solution. 设计是将问题转化成解决方案的创造性的活动 Design is the description of a solution. 是对解决方案的描述。

2、What is Coupling? States Coupling levels from low to high.

Coupling 耦合性是指模块间联系，即程序结构中不同模块之间互连程度。

耦合等级从低到高：

Uncoupled 非直接耦合：通过上级模块进行联系，无直接关联。

Data coupling 数据耦合：参数传递的是一般类型的数据。

Stamp coupling 标记耦合：参数传递的是诸如结构类型的数据。

Control coupling 控制耦合：模块间传递的是诸如标记量的控制信息。

Common coupling 公共耦合：全局结构类型的数据。

Content coupling 内容耦合：病态连接，一个模块可以直接操作另一个模块的数据（如 go to 语句的使用）。

3、What is Cohesion? States Cohesion levels from low to high.

Cohesion （内聚）：标志一个模块内各个元素彼此结合的紧密程度，是模块功能强度的度量，用来量化表示一个模块在多大程度上专注于一件事情 。一个模块内部各个元素彼此结合得越紧密，内聚度就越高。

内聚等级从低到高依次为：

Coincidental Cohesion	偶然(巧合)内聚
Logical Cohesion	逻辑内聚
Temporal Cohesion	时间内聚
Procedural Cohesion	过程内聚
Communicational Cohesion	通讯内聚
Sequential Cohesion	顺序内聚
Functional Cohesion	功能性内聚

三、填空题

1、软件设计是要把描述软件“做什么”的（**逻辑模型**）转换为描述“怎么做”的（**物理模型**），即着手实现软件的需求，并将设计结果记入软件（**设计规格说明书**）文档中。

2、软件程序系统结构的设计是以（**模块**）为基础的。以需求分析的结果为依据，从实现的角度进一步划分为（**模块**），并组成模块（**层次结构**）。

3、数据库的设计指（**数据存储文件**）的设计，主要进行（**概念设计**）、（**逻辑设计**）、（**物理设计**）的设计。

4、在数据处理系统的功能分析与设计过程中同时要进行数据分析和数据设计，数据库的概念设计和逻辑设计分别对应于系统开发的（**需求分析**）与（**概要设计**），而数据库的物理设计与模块的（**详细设计**）相对应。

5、在软件的系统结构中，，模块是可组合、可分解和可更换的单元。模块的基本属性包括**功能**、（**逻辑**）、（**接口**）和状态。

6、模块内聚与耦合是模块独立性的两个定性标准。在划分模块时，应尽可能作到（**高）内聚**、（**低）耦合**。

7、一个模块的（**作用**）范围应在其（**控制**）范围之内，且判定所在的模块应与受其影响的模块在层次上尽可能（**靠近**）。

8、如果模块之间耦合性太高，每个模块内功能不复杂，可将它们（**合并**），以减少信息的（**传递**）和（**数据公用区**）的引用。若有多个相关的模块，应对它们的功能进行（**分析**），消去（**重复的功能**）。

四、单选题

1、结构化设计方法(SD)与结构化分析方法(SA)一样，遵循（ C ）的模型，采用自顶向下，逐步细化技术。通常 SD 方法继续 SA 的工作，根据数据流图设计程序的结构。

A. 实体-关系　B. 快速原型　**C. 抽象**　D. 瀑布

2、结构化设计在软件开发中用于（ B ）。

A. 测试设计　**B. 概要设计**　C. 程序设计　D. 详细设计

3、（ D ）把已确定的软件需求转换成特定形式的软件表示，使其得以实现。

A. 系统设计　B. 逻辑设计　C. 详细设计　**D. 软件设计**

4、在进行软件模块结构设计时应当遵循的最主要的准则是（ C ）。

A. 抽象　B. 模块化　**C. 模块独立**　D. 信息隐蔽

5、（ A ）是数据说明、可执行语句等程序对象的集合，它是单独命名的并可通过名字访问。

A. 模块　B. 复合语句　C. 程序块　D. 数据块

6、模块（ C ），则说明模块的独立性越强。

A. 耦合越强　B. 扇入数越高　**C. 耦合越弱**　D. 扇入数越低

7、模块之间的信息可以做“控制信息”用，也可以作为（ D ）用。

A. 控制流　B. 数据结构　C. 控制结构　**D. 数据**

8、在多层系统结构图中，其模块的层次数称为结构图的（ A ）。

A. 深度　B. 宽度　C. 控制域　D. 粒度

9、（C）着重反映的是模块间的隶属关系，即模块间的调用关系和层次关系。

A. 程序流程图 B. 数据流图　**C. 系统结构图** D. 实体关系图

10、(C)是指把一些关系密切的软件元素物理地放置到彼此靠近的位置。

A. 信息隐蔽　B. 内聚　**C. 局部化**　D. 模块独立

11、块间联系和块内联系是评价程序结构质量的重要标准。联系的方式、共用信息的作用、共用信息的数量和界面的(C)等因素决定了块间联系的大小。

A. 友好性　B. 健壮性　**C. 清晰性**　D. 安全性

12、为了提高模块的独立性，模块之间最好是(D　)。

A. 公共耦合　B. 控制耦合　C. 内容耦合　**D. 数据耦合**

13、为了提高模块的独立性，模块内部最好是 (C)。

A. 逻辑内聚　B. 时间内聚　**C. 功能内聚**　D. 通信内聚

14、从下列有关系统结构图的叙述中选出正确的叙述(D)。

A. 系统结构图中反映的是程序中数据流的情况。

B. 系统结构图是精确表达程序结构的图形表示法。因此，有时也可将系统结构当作程序流程图使用。

C. 一个模块的多个下属模块在系统结构图中所处的左右位置是无关紧要的。

D. 在系统结构图中，上级模块与其下属模块之间的调用关系用有向线段表示。这时，使用斜的线段和水平、垂直的线段具有相同的含义。

五、选择题

1、请将下述有关模块独立性的各种模块之间的耦合，按其耦合度从低到高排列起来。

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

③、⑤、④、②、⑥、⑦、①

2、请将下述有关模块独立性的各种模块内聚，按其内聚度（强度）从高到低排列起来。

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

③、⑥、④、⑦、②、⑤、①

3、从供选择的答案中选出正确的答案填入下列叙述中的(　　)内。

模块内聚性用于衡量模块内部各成份之间彼此结合的紧密程度。

(1) 一组语句在程序中多处出现，为了节省内存空间把这些语句放在一个模块中，该模块的内聚性是(A ⑤)的。

(2) 将几个逻辑上相似的成分放在同一个模块中，通过模块入口处的一个判断决定执行哪一个功能。该模块的内聚性是(B ⑦)的。

(3) 模块中所有成分引用共同的数据，该模块的内聚性是(C ③)的。

(4) 模块内的某成份的输出是另一些成份的输入，该模块的内聚性是(D ④)的。

(5) 模块中所有成份结合起来完全一项任务，该模块的内聚性是(E ①)的。它具有简明的外部界面，由它构成的软件易于理解、测试和维护。

供选择的答案：

A～E：　①功能内聚　②信息内聚　③通信内聚　④过程内聚　⑤ 巧合内聚　⑥ 时间内聚　⑦ 逻辑内聚

4、从供选择的答案中选出正确的答案填入下面的(　　)中。

块间联系和块内联系是评价程序模块结构质量的重要标准。联系的方式、共用信息的作用、共用信息的数量和接口的(A ③)等因素决定了块间联系的大小。在块内联系中，(B ②)的块内联系最强。

SD 方法的总的原则是使每个模块执行(C ①)功能，模块间传送(D ①)参数，模块通过(E ②)语句调用其它模块，而且模块间传送的参数应尽量(F ①)。

此外，SD 方法还提出了判定的作用范围和模块的控制范围等概念。SD 方法认为，(G ①)应该是(H ②)的子集。

供选择的答案：

A： ①友好性　②健壮性　**③简单性**　④安全性

B： ①巧合内聚　**②功能内聚**　③通信内聚　④信息内聚

C： **①一个**　②多个

D： **①数据型**　②控制型　③混合型

E： ①直接引用　**②标准调用**　③中断　④宏调用

F： **①少**　②多

G～H：**①作用范围　②控制范围**

5、从供选择的答案中选出应该填入下列关于软件设计的叙述的(　　)内的正确答案。

在众多的设计方法中，SD 方法是最受人注意的，也是最广泛应用的一种，这种方法可以同分析阶段的(A ②)方法及编程阶段的(B ⑤)方法前后衔接，SD 方法是考虑如何建立一个结构良好的程序结构，它提出了评价模块结构质量的两个具体标准——块间联系和块内联系。SD 方法的最终目标是(C ③)，用于表示模块间调用关系的图叫(D ③)。

供选择的答案：

A～B：①Jackson　**②SA**　③SC　④Parnas　**⑤ SP**

C：　①块间联系大，块内联系大　②块间联系大，块内联系小

　　③块间联系小，块内联系大　④块间联系小，块内联系小

D：　①PAD　②HCP　**③SC**　④SADT　⑤ HIPO　⑥ NS

6、从供选择的答案中选出应该填入下列关于软件详细设计的叙述的(　　)内的正确答案。

软件详细设计工具可分为三类，即：图示工具、设计语言和表格工具。图示工具中，(A ②)简单而应用广泛、(B ①)表示法中，每一个处理过程用一个盒子表示，盒子可以嵌套。(C ④)可以纵横延伸，图形的空间效果好。

A～C：**①NS 图　②流程图**　③HIPO 图　**④PAD 图**

7、从供选择的答案中选出应该填入下列关于软件设计的叙述的(　　)内的正确答案。

在完成软件概要设计，并编写出相关文档之后，应当组织对概要设计工作的评审。评审的内容包括：

分析该软件的系统结构、子系统结构，确认该软件设计是否覆盖了所有已确定的软件需求，软件每一成份是否可(A ③)到某一项需求。分析软件各部分之间的联系，确认该软件的内部接口与外部接口是否已经明确定义。模块是否满足(B ②)和(　C ③)的要求。模块(　D ①)是否在其(　E　⑤)之内。

供选择的答案

A： ①覆盖　②演化　**③追溯**　④等同　⑤ 连接

B： ①多功能　**②高内聚**　③高耦合　④高效率　⑤ 可读性

C： ①多入口　　②低内聚　**③低耦合** ④低复杂度　⑤ 低强度
D～E：**①作用范围** ②高内聚　③低内聚 ④取值范围　**⑤ 控制范围**

六、简答题

1、逐步求精、分层过程与抽象等概念之间的相互关系如何？

“自顶向下，逐步求精”是 Niklaus Wirth 提出的设计策略：即将软件的体系结构按自顶向下方式，对各个层次的过程细节和数据细节逐层细化，直到用程序设计语言的语句能够实现为止，从而最后确立整个的体系结构。这样的结构实际就是一个模块的分层结构，即分层的过程。在实施时，采用抽象化的方法，自顶向下，给出不同的抽象层次。在最高的抽象层次上，可以使用问题所处环境的语言概括地描述问题的解法。而在较低的抽象层次上，则采用过程化的方法。在描述问题的解法时，可以配合使用面向问题的术语和面向现实的术语。但最后在最低的抽象层次上，应使用能够直接实现的方式来描述这个解法。

2、完成良好的软件设计应遵循哪些原则？

软件设计既是过程又是模型。设计过程是一系列迭代的步骤，使设计人员能够描述被开发软件的方方面面。设计模型体现了自顶向下、逐步细化的思想，首先构造事物的整体，再逐步细化，引导人们构造各种细节。为了给软件设计人员提供一些指导，1995 年 Davis 提出了一系列软件设计的原则如下，其中有些修改和补充：

设计过程不应受“隧道视野”的限制。一位好的设计者应当考虑一些替代的手段。根据问题的要求，可以用基本的设计概念，如抽象、逐步求精、模块化、软件体系结构、控制层次、结构分解、数据结构、软件过程、信息隐蔽等，来决定完成工作的资源。

设计应能追溯到分析模型。由于设计模型中每一单个成份常常可追溯到多个需求上，因此有必要对设计模型如何满足需求进行追踪。

设计不应当从头做起。系统是使用一系列设计模式构造起来的，很多模式很可能以前就遇到过。这些模式通常被称为可复用的设计构件。可以使用它们代替那些从头开始做的方法

设计应当缩短软件和现实世界中问题的“智力差距”，即，软件设计的结果应尽可能模拟问题领域的结构。

设计应具有一致性和集成性

使用上述的基本的设计概念，将设计构造得便于将来的修改。

应将设计构造得即使遇到异常的数据、事件或操作条件，也能平滑地、轻松地降级。

设计不是编码，编码也不是设计。即使在建立程序构件的详细的过程设计时，设计模型的抽象级别也比源代码要高。

在开始着手设计时就应当能够评估质量，而不是在事情完成之后。

应当坚持设计评审以减少概念上（语义上）的错误。在关注设计模型的语法之前，设计者应能确保设计的主要概念上的成份（的遗漏、含糊、不一致）都已检查过。

3、如何理解模块独立性？用什么指标来衡量模块独立性？

如果两个模块互相独立，那么对其中一个模块进行编码、测试或修改时可以完全不考虑另一个模块对它的影响。因此，用模块独立性作为衡量模块结构是否容易编码、容易测试、容易修改的标准是合适的。但是，在一个系统的模块结构

中没有哪两个模块可以完全独立，所以，要力争模块之间尽量独立，以得到一个质量良好的模块结构。

一般采用两个准则度量模块独立性。即模块间的耦合和模块的内聚。模块间的耦合是模块之间的相对独立性（互相连接的紧密程度）的度量。模块之间的连接越紧密，联系越多，耦合性就越高，而其模块独立性就越弱。内聚是模块功能强度（一个模块内部各个成份彼此结合的紧密程度）的度量。一个模块内部各个成份之间的联系越紧密，则它的内聚性就越高，相对地，它与其它模块之间的耦合性就会减低，而模块独立性就越强。因此，模块独立性比较强的模块应是高内聚低耦合的模块。

内聚和耦合是相互关联的。在程序结构中各模块的内聚程度越高，模块间的耦合程度就越低。但这也不是绝对的。我们的目标是力求增加模块的内聚，尽量减少模块间的耦合，但增加内聚比减少耦合更重要，应当把更多的注意力集中到提高模块的内聚程度上来。

4、模块独立性与信息隐蔽（反映模块化有效程度的属性）有何关系？

所谓“模块独立性”是指软件系统中每个模块只涉及软件要求的具体的子功能，而和软件系统中其它的模块的接口是简单的。所谓的“信息隐蔽”是指每个模块的实现细节对于其它模块来说是隐蔽的。也就是说，模块中所包含的信息（包括数据和过程）不允许其它不需要这些信息的模块使用。

如果软件系统做到了信息隐蔽，即定义和实施了对模块的过程细节和局部数据结构的存取限制，那么这些模块相互间的接口就是简单的。这组模块的独立性就比较强。事实上，衡量模块独立性的一个准则就是模块内聚，达到信息隐蔽的模块是信息内聚模块，它是高内聚情形，模块独立性当然很强了。

5、模块的内聚性程度与该模块在分层结构中的位置有关系吗？说明你的论据。

模块的内聚性与该模块在分层模块结构中的位置无关。事实上，一个好的模块化的程序系统，它所有的模块可以都是功能内聚的，即每一个模块只干了一件事。用结构化设计方法建立起来的模块结构中的每一个模块都符合这个要求。在纯面向对象范型的软件系统中，整个系统看作是一个类，它的子类可以看作是系统的子系统或高层模块，它们还可以有子类，……，这就形成一个类的层次结构。类的构造可以看成是一个抽象数据类型，实际上是信息内聚的。所以整个系统中从上到下，所有模块（对象类）都是信息内聚的模块。

6、耦合性的概念和软件的可移植性有什么关系？请举例说明你的论述。

所谓“耦合性”是指模块之间联系的紧密程度的一种度量，而软件的“可移植性”是指将一个软件系统从一个计算机系统或环境移植到另一个计算机系统或环境中运行时所需工作量的大小。可移植性是用一组子特性，包括简明性、模块独立性、通用性、可扩充性、硬件独立性和软件系统独立性等，来衡量的。如果一个软件具有可移植性，它必然耦合性低，这样模块独立性要强。反言之，模块之间的耦合都是低耦合，也对可移植性有促进。但不能讲具有低耦合性模块结构的软件一定具有可移植性，因为是否具有可移植性还有其它因素的影响。

7、递归模块（即自己调用自己的模块）的概念如何能够与软件系统设计原理与方法相适应？

递归过程在求解复杂的大型问题时非常有效。常用的求解问题的方法，如“分治”的策略和“回溯”的策略，都可以用递归方法来解决。软件设计过程中的“自顶向下，逐层分解”的做法与上述求解问题的策略是一致的。如果分解出

来的子问题（子功能、子模块）相互独立性比较强，这种分解可以降低模块的复杂性，做到模块化。所以，只要分解出来的子问题与原来问题满足递归的情况，用递归方法建立模块结构也是可行的。

8、举例说明你对概要设计与详细设计的理解。有不需要概要设计的情况吗?

软件设计是一个把软件需求变换成软件表示的过程。最初这种表示只是描绘出软件的总的框架，然后进一步细化，在此框架中填入细节，把它加工成在程序细节上非常接近于源程序的软件表示。正因为如此，所以从工程管理的角度来看，软件设计分两步完成。首先做概要设计，将软件需求转化为数据结构和软件的系统结构。然后是详细设计，即过程设计。通过对结构表示进行细化，得到软件的详细的数据结构和算法。

由于概要设计建立起整个系统的体系结构框架，并给出了系统中的全局数据结构和数据库接口，人机接口，与其它硬、软件的接口。此外还从系统全局的角度，考虑处理方式、运行方式、容错方式、以及系统维护等方面的问题，并给出了度量和评价软件质量的方法，所以它奠定了整个系统实现的基础。没有概要设计，直接考虑程序设计，就不能从全局把握软件系统的结构和质量，实现活动处于一种无序状态，程序结构划分不合理，导致系统处于一种不稳定的状态，稍一做改动就会失败。所以，不能没有概要设计。

9、解释模块化的概念；为了确保模块设计的独立性，应如何具体控制耦合的使用级别；举例说明如何使模块的作用域在其控制域内。

模块化是指将程序划分成独立命名且可独立访问的模块，每个模块完成一个子功能，把这些模块集成起来构成一个整体，可以完成指定的功能满足用户的需求。

针对耦合的设计指导原则：

尽量使用数据耦合；

少用控制耦合和标记耦合；

限制外部耦合和公共耦合；

完全不用内容耦合。

10、软件设计分为概念设计（概要设计）和技术设计（详细设计），叙述概念设计（概要设计）的内容。

概念设计的过程：

(1)设想可能的方案

(2)选取合理的方案

(3)推荐最佳方案

(4)功能分解

(5)设计软件结构

(6)数据库设计

(7)制定测试计划

(8)编写文档

(9)审查与复审

11、软件体系结构风格代表了软件体系结构设计中的惯用模式。请描述隐含调用系统风格，指出该风格的特点与应用，举一个例子来说明隐含调用系统。

隐含调用的设计模型是事件驱动的，它基于广播的概念。某个组件不直接调用一个过程，而是宣布发生了一个或多个事件。接着，其他组件会将这个程序和发生的事件联系起来（称为程序注册），由系统调用所有这些注册程序。

基于事件的隐含调用（Implicit Invocation）风格的思想是构件不直接调用一个过程，而是发布或广播一个或多个事件。

系统中的其它构件为它感兴趣的事件注册过程。

当一个事件被发布，系统自动调用在这个事件中注册的所有过程，这样，一个事件的发布就“隐式地”激发了另一模块中的过程。

隐含调用风格的特点是：

事件的发布者并不知道哪些构件会被这些事件影响。这样不能假定构件的处理顺序，甚至不知道哪些过程会被调用。

应用：

在编程环境中用于集成各种工具；

在数据库管理系统中确保数据的一致性约束；

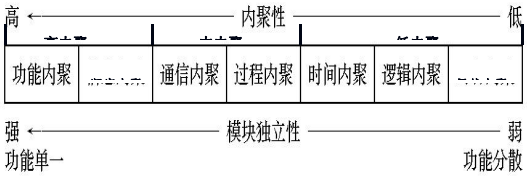
在用户界面系统中管理数据；

以及在编辑器中支持语法检查。

例如：在某系统中，编辑器和变量监视器可以登记（注册）相应 Debugge 的断点事件。当 Debugger 在断点处停下时，它声明该事件，由系统自动调用对该事件感兴趣的处理程序，如编辑程序可以卷屏到断点，变量监视器刷新变量数值。而 Debugger 本身只声明事件，不关心这些过程做什么处理。

12、解释模块的概念；为了确保模块设计的独立性，应如何具体控制内聚的使用级别；举例说明软件结构中的扇出、扇入、深度、宽度的意义。

模块：可单独命名和可编址的部分。(另:由边界元素限定的相邻程序元素的序列，而且有一个总体标识符代表它）如：procedure, function, subroutine, block，Macro



Depth 深度：软件结构中控制的层数。一般而言它与系统的复杂度和系统大小直接对应。

Width 宽度：软件结构中同一个层次上的模块总数的最大数。

fan-out 扇出：一个模块直接控制（调用）的模块数目。 扇出过大说明模块过分复杂；过小也不好，不利于系统平衡分解，3 到 9 为宜。

fan-in 扇入：一个模块的扇入是指直接控制该模块的模块数目。扇入越大说明共享该模块的上级模块越多。

第六章 对象

一、判断题

Two classes are associated when they occur together, and when the relationship must be preserved for some period of time.（T）

二、填空题

1、面向对象的特征是（**对象惟一性**）、（**分类性**）、（**继承性**）和（**多态性**）。

2、面向对象有三个基本要素，它们是（**抽象**）、（**信息隐蔽**）和（**共享性**）。

3、类具有属性，它是（**对象状态**）的抽象，用（**数据结构**）来描述类的属性。

4、类具有操作，它是（**对象行为**）的抽象，用（**操作名**）和（**操作方法**）实现来描述。

5、在类层次中，子类只继承一个父类的属性和方法，则称为（**单继承**）。子类继承了多个父类的属性和方法，则称为（**多继承**）。

6、（**UML**）不仅统一了 Booch 方法、OMT 方法、OOSE 方法的表示方法，而且对其作了进一步的发展，最终成为为国际对象组织（**OMG**）认可的同一建模语言。

7、在客观世界中有若干类，这些类之间有一定的结构关系。通常有两种主要的结构关系，即（**一般化-特殊化关系**）和整体-部分关系。

8、在面向对象设计中存在三种内聚，即（**操作内聚**）、（**类内聚**）和（**一般化-特殊化内聚**）。

三、单选题

1、对象是面向对象开发方法的基本成分，每个对象可用它本身的一组（ C）和它可以执行的一组操作来定义。

A. 服务　　B. 参数　　**C. 属性**　　D. 调用

2、在面向对象方法中，把一组具有相同数据结构和相同操作的对象的集合定义为（ B　）。此定义包括一组数据属性和在数据上的一组合法操作。

A. 聚合　　**B. 类**　　C. 结构　　D. 主题

3、面向对象技术的许多强有力的功能和突出的优点都来源于把系统组织成一个类的层次结构。一个类的上层可以有父类，下层可以有子类。这种系统的类层次结构的一个重要性质是（ D ），通过它，一个类可共享其父类的全部数据和操作。

A. 传递性　　B. 复用性　　C. 并行性　　**D. 继承性**

4、一个面向对象软件的体系结构通过它的成分对象及各对象之间的关系来确定，与传统的结构化开发方法相比，它具有（ A ）的优点。

A. 设计稳定　B. 性能稳定　C. 模块独立　D. 硬件独立

5、封装性是指所有软件部件都有明确的范围以及清楚的外部边界。每个软件部件都有友好的（ B ），软件部件的内部实现和外部使用分离。

A. 使用方式　**B. 界面**　　C. 调用　　D. 继承

6、属性指的是类中对象具有的特性（数据）。不同对象的同一属性可具有

相同的或不同的(　A)。

A. 属性值　B. 操作　C. 服务　D. 控制

7、操作是类中对象所使用的一种功能或变换。类中的各个对象可以共享操作，方法是类中操作的(B　)。

A. 别名　**B. 实现步骤**　C. 功能　D. 脚本

8、应用程序可以通过执行对象的操作来改变对象的属性值，但它必须通过(C)的传递。

A. 接口　B. 控制　**C. 消息**　D. 实例

9、在软件开发过程中，抽取和整理用户要求并建立问题论域精确模型的过程叫做(D　)。

A. 生存期 B. 面向对象分析 C. 面向对象程序设计 **D. 面向对象设计**

10、对象模型表示了静态的、结构化的系统数据性质，描述了系统的静态结构。它是从现实世界实体的相互关系的角度来描述、表现对象间的相互关系。该模型主要关心系统中对象的结构、属性和操作，使用了(B)的工具来刻划。

A. E-R 图　**B. 对象图**　C. 系统流程图　D. 系统结构图

11、组装关系是一种“整体-部分” 关系。在这种关系中，有整体类和部分类之分。组装关系中最重要的性质是(D)，它还具有逆对称性。

A. 局部性　B. 完整性　C. 一致性　**D. 传递性**

12、分类关系是“一般化-特殊化”关系。一般化类又称为父类，特殊化类又称为子类。分类关系和(C)是同时存在的。

A. 传递性　B. 逐步求精　**C. 继承性**　D. 全局性

13、继承有单继承和多继承。单继承指的是子类只有一个父类，在一个类层次结构中若只有单继承，则该类层次结构是树形结构。多继承指的是子类可以有多个父类，在一个类层次结构中若有多继承，则该类层次结构是(　C)层次结构。

A. 树形　B. 星形　**C. 网状**　D. 环形

14、动态模型描述与时间和变化有关的系统的性质。该模型描述了系统的控制结构，表示了瞬时的行为化的系统控制性质，它关心的是系统的控制、操作的执行顺序。它从系统涉及的事件和对象的(　A)出发，表现了对象及对象间的相互行为。

A. 状态　B. 属性　C. 操作　D. 控制

15、动态模型描述的系统属性是触发事件、事件序列、状态、事件和状态的组织。使用(A)作为描述工具。

A. 状态图　B. 顺序图　C. 活动图　D. 进程图

16、功能模型用来说明值是如何计算的，表明值之间的依赖关系及其相关的功能。数据流图有助于表示功能依赖关系，其中的处理对应于状态图中的活动和动作，数据流对应于对象图中的(B)。

A. 实例连接　**B. 对象或属性** C. 消息传递　D. 关联

17、操作与对象模型中的属性和关联的查询有关，与动态模型中的(A)有关，与功能模型中的加工有关。

A. 事件　B. 状态　C. 变换　D. 处理

18、面向对象设计阶段中的高层设计是要确定实现系统的策略和目标系统的(A)。

A. 体系结构 B. 算法设计 C. 类结构 D. 类设计

19、面向对象设计阶段中的类设计是要确定实现方案中的类、关联和接口形式及实现操作的(D　)。

A. 逻辑　　B. 顺序　　C. 控制　　**D. 算法**

20、状态是对象属性值的抽象，状态指明了对象对(A　)的响应。

A. 输入事件　B. 输入信息　C. 输入数据　D. 输入序列

21、活动是一种有时间间隔的操作，它是依附于状态的操作。动作是一种瞬时操作，它是与(　B)联系在一起的操作。

A. 时间　　**B. 事件**　　C. 控制　　D. 状态

22、事件可以看成是信息从一个对象到另一个对象的单向传送。因此要确定各事件的发送对象和接收对象。(A)用来表示事件、事件的接收对象和发送对象。

A. 事件追踪图 B. 进程图　C. 脚本　D. 状态序列图

四、选择填空题

1、从供选择的答案中选出与下面有关面向对象范型的叙述最适合的答案，将其编号填入相应的括号内。

对象是面向对象范型的（ A ①）。每个对象可用它自己的一组（ B ⑤）和它可以执行的一组（ C ③）来表征。应用执行对象的（ C ③）可以改变该对象的（ B ⑤）。它的应用必须通过（ D　②）的传递。可以认为，这种（ D　②）的传递大致等价于过程性范型中的函数调用。某些语言提供了特殊功能，允许对象引用自己。若一个对象没有显式地被引用，则可让该对象（ E ③）。

供选择的答案：

A.　①基本单位 ②最小单位 ③最大单位 ④语法单位

B～C. ①行为　②功能　③操作　④数据　⑤ 属性

D.　①接口　②消息　③信息　④操作　⑤ 过程

E.　①撤消　②歇着　③缺省　④隐式引用　⑤ 引用自己

2、从供选择的答案中选出与下面有关面向对象开发过程的叙述最适合的答案，将其编号填入相应的括号内。

在面向对象软件开发过程中特别重视复用。软件构件应独立于当初开发它们的应用而存在。在以后的应用开发中，可以调整这些独立构件以适应新问题的需要。因此，应使得类成为一个（ A ①）的单元。这样就有一个（ B ③）生存期问题。（ B ③）生存期有自己的步骤，与任一特定应用的开发（ C ④）。按照这些步骤，可以完整地描述一个基本（ D ②）。而不仅仅考虑当前正在开发的系统。系统开发的各个阶段都可能会标识新的类。随着各个新类的标识，(B ③)生存期引导开发工作逐个阶段循序渐进。

在设计与实现类时，应尽可能利用既存类提供为当前应用所需要的功能，利用既存类的三个可能途径是：(E ③)复用既存类；对既存类进行（ F ④）以得到满足要求的类；重新开始进行开发。

供选择的答案：

A. ①可复用　②可测试　③可适用　④可靠

B. ①应用　②寿命　③类　④软件

C. ①相关　②密切相关 ③负相关 ④无关

D. ①概念　②实体　③事件　④事情

E, F. ①修改　②更新　③照原样　④演化

3、从供选择的答案中选出与下面有关类设计的叙述最适合的答案，将其编号填入相应的括号内。

类常常被看做是一个抽象数据类型的实现，更合适的是把类看做是某种（ A ②）的一个模型。事实上，类是单个的（ B ③）单元。类的用户能够操纵的操作叫做类的（ C ①）。类定义的其余部分给出数据定义和辅助功能定义，包括类的实现。

类的实现常常包括了其它类的实例，这些实例（ D ④）被其它对象存取，包括同一个类的其它实例。类的实现可能还包括某些私有方法，实现它们的类可以使用，而其它任何对象都不能使用。

类，就它是一个数据值的聚合的意义上来看，与中的结构类似，但又有差别。类扩展了通常的记录语义，可提供各种级别的（ E ③）。类不同于记录，因为它们包括了操作的定义，这些操作与类中声明的数据值有相同的地位。

供选择的答案：

A. ①功能 ②概念 ③结构 ④数据

B. ①语法 ②词法 ③语义 ④上下文环境

C. ①界面 ②操作 ③行为 ④活动

D. ①可自由地 ②可有控制地 ③可通过继承 ④应受保护不

E. ①可移植性 ②可重复性 ③可访问性 ④继承性

4、从供选择的答案中选出与下面有关面向对象设计的叙述最适合的答案，将其编号填入相应的括号内。

在面向对象软件设计过程中，应按如下要求进行类的设计：只有类的共有界面的成员才能成为使用类的操作，这就是软件设计的(A ③)原则。当且仅当一个操作对类的实例的用户有用时，它才是类公共界面的一个成员，这是软件设计的(B ②)原则。由同属一个类的操作负担存取或加工类的数据，这是软件设计的(C ③)原则。两个类之间的交互应当仅涉及参数表，这是软件设计的(D ①)原则。每个派生类应该当做基类的特殊化来开发，而基类所具有的公共界面成为派生类的共有界面的一个子集，这是软件设计的(E ⑤)原则。

供选择的答案：

A：①过程抽象 ②功能抽象 ③信息隐蔽 ④共享性 ⑤ 连通性

B：①标准调用 ②最小界面 ③高耦合 ④高效率 ⑤ 可读性

C：①数据抽象 ②低内聚 ③高内聚 ④低复杂度 ⑤ 低强度

D：①显式信息传递 ②高内聚 ③低内聚 ④相互操作性 ⑤ 连接性

E：①动态联编 ②异质表 ③信息隐蔽 ④多态性 ⑤ 继承性

5、从供选择的答案中选出与下面有关面向对象程序设计的叙述最适合的答案，将其编号填入相应的括号内。

面向对象的程序设计语言具有数据抽象、信息隐蔽、(A ④)等特征。作为运算单位的对象应具有下列特性：(B ①)、(C ③)、(D ④)。

供选择的答案：

A：①对象调用 ②对象变换 ③非过程性 ④信息继承 ⑤ 并发性

B～D：①对象把数据和处理数据的操作结合为一体

②在程序运行时对象都处于活动状态

③对象在计算中可向其他对象发送消息

④接受消息的对象必须给消息发送者以回答

⑤ 对象的内部状态只根据外部送来的消息才操作

五、简答题

1、Use case

A use case describes particular functionality that a system is supposed to perform or exhibit by modeling the dialog that a user, external system, or other entity will have with the system to be developed.用例是用模型化会话的方式来描述系统可能会执行或显示的特定功能，这些会话发生在用户、外部系统或其它实体与将开发的系统之间。

2、List at least 7 kinds of UML

类图、对象图、用例图、顺序图、协作图、状态图、活动图、构件图、配置图(实施图)

3、 what is Use Case? What elements dose it have?

A use case describes particular functionality that a system is supposed to perform or exhibit by modeling the dialog that a user, external system, or other entity will have with the system to be developed.

Use case Diagrams have four elements:

Actors 执行者（与系统交互的实体）

Cases 实例（对系统功能某方面的描述）

Extensions 扩充（对用例的扩充）

Uses 使用（复用已定义的用例）

4、比较结构化软件设计方法与面向对象软件设计方法的特点。

从概念方面看

结构化软件是*功能*的集合，通过模块以及模块和模块之间的分层调用关系实现；

面向对象软件是*事物*的集合，通过对象以及对象和对象之间的通讯联系实现；

从构成方面看

结构化软件＝过程＋数据，以过程为中心；

面向对象软件＝（数据＋相应操作）的封装，以数据为中心；

从运行控制方面看

结构化软件采用顺序处理方式，由过程驱动控制；

面向对象软件采用交互式、并行处理方式，由消息驱动控制；

从开发方面看

结构化方法的工作重点是设计；

面向对象方法的工作重点是分析；在结构化方法中，分析阶段和设计阶段采用了不相吻合的表达方式，需要把在分析阶段采用的具有网络特征的数据流图转换为设计阶段采用的具有分层特征的结构图，在面向对象方法中则不存在这一问题。

从应用方面看

结构化方法更加适合数据类型比较简单的数值计算和数据统计管理软件的开发；

面向对象方法更加适合大型复杂的人机交互式软件和数据统计管理软件的开发。

5、什么叫面向对象？面向对象方法的特点是什么？

关于“面向对象”，有许多不同的看法。Coad 和 Yourdon 给出了一个定义：“面向对象 = 对象 + 类 + 继承 + 消息通信”。如果一个软件系统是使用这样 4 个概念设计和实现的，则认为这个软件系统是面向对象的。面向对象方法的特点是：

方法的唯一性，即方法是对软件开发过程所有阶段进行综合考虑而得到的。

从生存期的一个阶段到下一个阶段的高度连续性，即生存期后一阶段的成果只是在前一阶段成果的补充和修改。

把面向对象分析(OOA)、面向对象设计(OOD)和面向对象程序设计(OOP)集成到生存期的相应阶段。

6、什么是“对象”？识别对象时将潜在对象分成 7 类，试给出这 7 类对象的名称。

对象的定义：对象是面向对象开发模式的基本成分，是现实世界中个体或事物的抽象表示。每个对象可由一组属性和它可以执行的一组操作来定义。

可能的潜在对象有 7 类：

外部实体：它们产生或接受为目标系统所使用的信息。如各种物理设备、使用人员、其它相关的子系统。

事物：问题的信息域所涉及的概念实体。如各种报告、显示、文字、信号、规格说明等。

事件：系统运行时发生的并需要系统记忆的事件。如状态转换、物理运动等。

角色：与系统有交互的各种人员所扮演的角色。如经理、工程师、销售人员等。

场所或位置：建立系统整体环境或问题上下文的场所、位置。如基于计算机的系统的安装场所等。

部门、××组织、××组织机构：与应用有关的组织机构。

结构：定义由一组成分对象组成的聚合对象，或在极端情况下，定义对象的相关类。如传感器、四轮驱动车、计算机等。

7、什么是“类”？ “类”与传统的数据类型有什么关系？有什么区别？

把具有相同特征和行为的对象归在一起就形成了类。类成为某些对象的模板，抽象地描述了属于该类的全部对象的属性和操作。

类，就它是一个数据值的聚合的意义上来看，与 Pascal 中的记录或 C 中的结构类似，但又有差别。类扩展了通常的记录语义，可提供各种级别的可访问性。类不同于记录，因为它们包括了操作的定义，这些操作与类中声明的数据值有相同的地位。

8、按照类生存期，类的开发有哪几种方式？

按照类生存期，类的开发有三种方式。

(1) 既存类的复用

(2) 从既存类进行演化

(3) 从废弃型进行开发

9、在类的设计中需要遵循的方针是什么？三个主要的设计准则：抽象、信息隐蔽和模块化如何才能作到？

在设计类时需要遵循的方针是：

信息隐蔽：通过信息隐蔽可保护类的存储表示不被其它类的实例直接存取。

消息限制：该类实例的用户应当只能使用界面提供的操作。

狭窄界面：只有对其它类的实例是必要的操作才放到界面上。

强内聚：模块内部各个部分之间应有较强的关系，它们不能分别标识。

弱耦合：一个单独模块应尽量不依赖于其它模块。

显式信息传递：两个类之间的交互应当仅涉及显式信息传递。

派生类当做派生类型：每个派生类应该当做基类的特殊化来开发，而基类所具有的公共界面成为派生类的共有界面的一个子集。

抽象类：某些语言提供了一个类，用它做为继承结构的开始点，所有用户定义的类都直接或间接以这个类为基类。

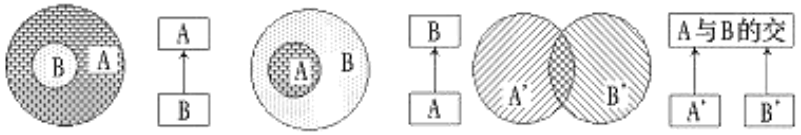
为了在类的设计中做到抽象、信息隐蔽和模块化要以类作为系统的基本模块单元，通过一般化－特殊化关系和整体－部分关系，搭建整个系统的类层次结构，实现数据抽象和过程抽象；将数据和相关的操作封装在类内部，建立共有、私有和子类型等存取级别，将数据表示定义成为类的私有成员，实现信息隐蔽。通过建立类属性(类模板)，将某些有可复用要求的类设计成在数据类型上通用的可复用的软件构件，这样有助于实现模块化。

10、在类的通过复用的设计中，主要的继承关系有哪几种？

在类的通过复用的设计中，主要的继承关系有两大类：

配置：利用既存类来设计类，可能会要求由既存类的实例提供类的某些特性。通过把相应类的实例声明为新类的属性来配置新类。

演变：要开发的新类可能与一个既存类非常类似，但不完全相同。此时可以从一个既存类演变成一个新类，可以利用继承机制来表示一般化－特殊化的关系。特殊化处理有三种可能的方式。

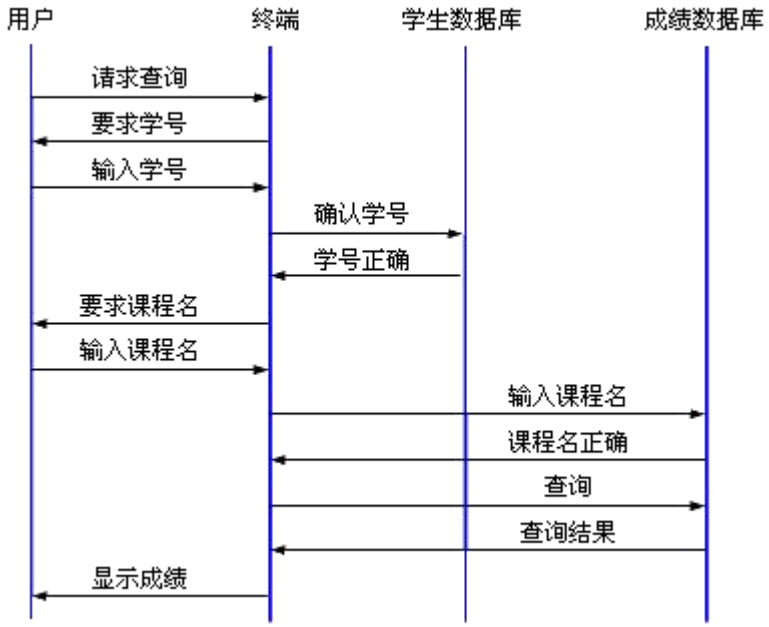


(a) 新类B中的对象是类A中对象的特殊化　(b) 新类B中的对象是类A中对象的一般化　(c) 新类B中对象与类A中对象共享某些公共属性

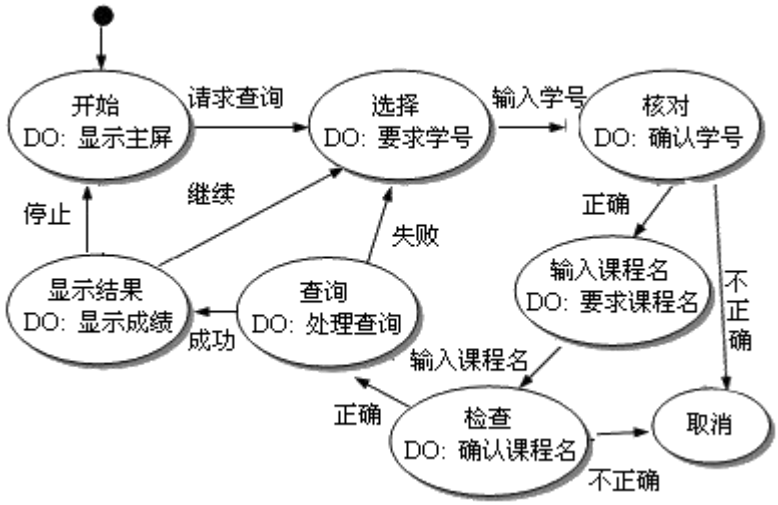
六、应用题

1、在学校教学管理系统中，学生查询成绩就是系统中的一次交互，请用状态图来描述这种交互的行为

首先建立事件追踪图，用于描述用户与系统的一次交互行为。在图中，按时间的先后次序以及事件的发送和接收顺序，自上而下画出。



根据事件追踪图建立的状态图如下：



2、开发一个学生指纹考勤系统对学生上课的出勤率进行统计，学生在每次上课前和下课后使用该系统进行指纹识别，即系统识别学生的指纹，然后将识别的指纹信息与系统中保存的学生指纹信息进行匹配，如果匹配成功则将识别出的学生身份和当前日期、时间等信息保存到学生出勤数据库中；如果匹配不成功，则返回错误信息，学生需再次进行指纹输入；教务人员可以在需要的时候使用该系统生成学生的出勤情况统计分析表。

假设在该学生指纹考勤系统中，有一个用例名为“上课登记”。此用例允许学生在上课前使用系统识别自己的指纹信息进而识别自己的身份，同时系统可以将登录信息存储在数据库中。

“上课登记”用例的主要事件流如下：

学生从系统菜单中选择“上课登记”(student select “entrance registry” from the system menu)；

系统显示指纹识别界面；(the system display the fingerprint recognize

interface)

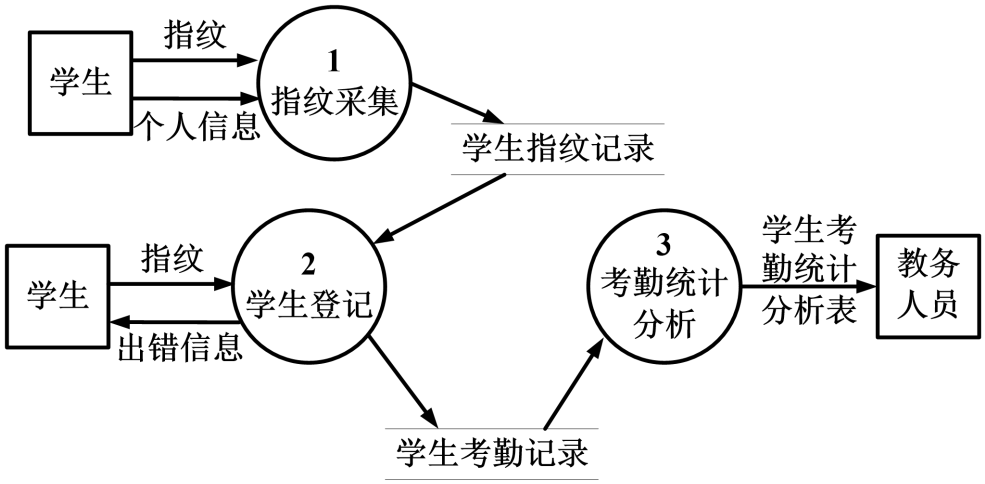
学生将手指放置于界面上；(student press the right finger on the interface)

系统捕获并识别学生的指纹，向学生返回识别的身份信息；(the system capture and recognize the student’s fingerprint and return the recognized identity information)

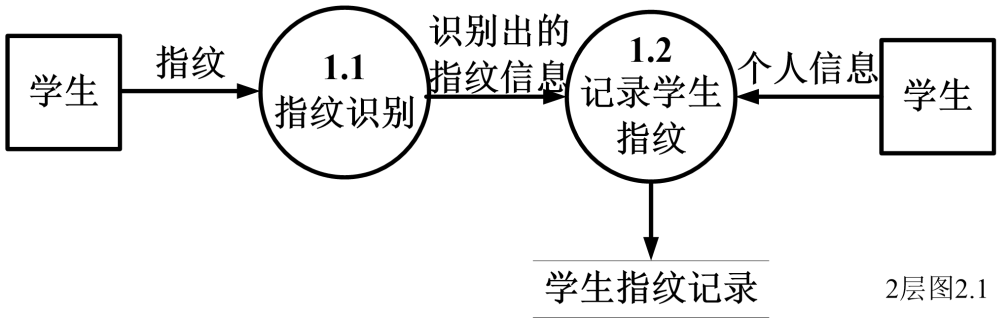
学生选择“确认”按钮；(student select the “confirm” button)

系统生成一个关于该登记学生及当前日期、时间的新记录，并将该记录保存到数据库中。(the system create a new record about the student, date and time, and save this record information to the database)

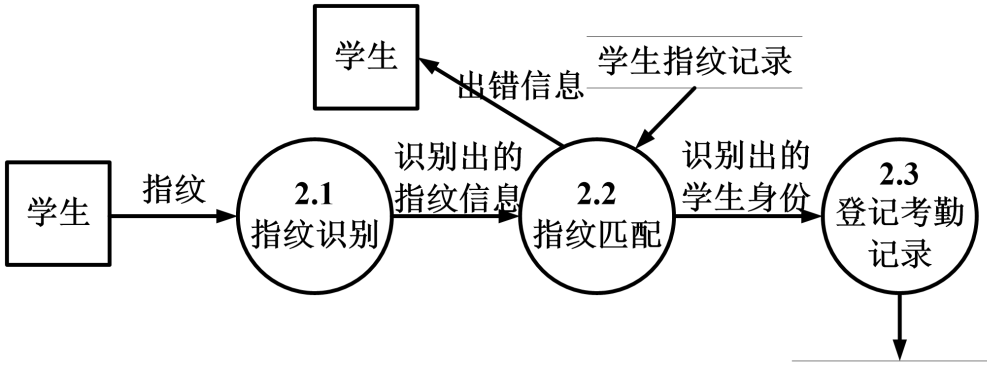
要求：（1）画出状态图；（2）画出 UML 顺序图；（3）画出分层的数据流图。



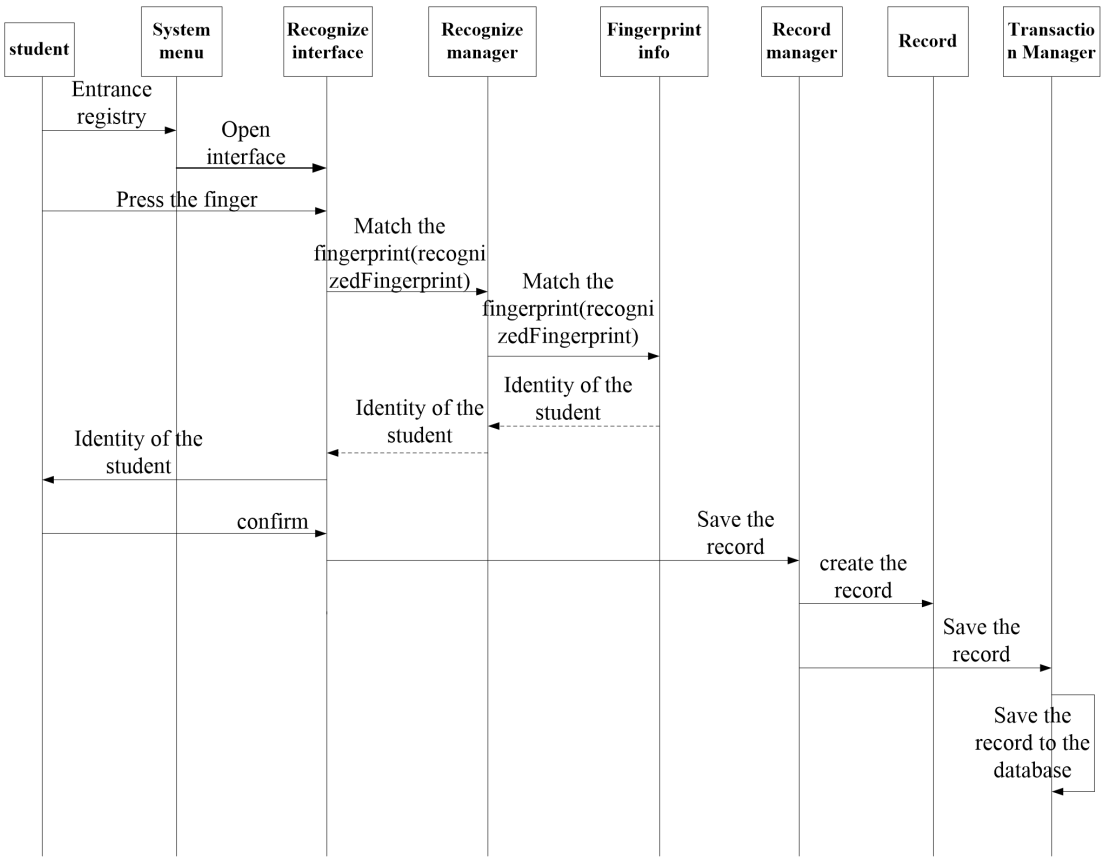
1层图



2层图2.1


学生
出错信息
学生指纹记录
学生
指纹
2.1
指纹识别
识别出的
指纹信息
2.2
指纹匹配
识别出的
学生身份
2.3
登记考勤
记录
学生考勤记录

2层图2.2


student
System menu
Recognize interface
Recognize manager
Fingerprint info
Record manager
Record
Transaction Manager
Entrance registry
Open interface
Press the finger
Match the fingerprint(recognizedFingerprint)
Match the fingerprint(recognizedFingerprint)
Identity of the student
Identity of the student
Identity of the student
confirm
Save the record
create the record
Save the record
Save the record to the database

第七章 程序的编写

一、填空

1、软件详细设计的目标不仅是逻辑上正确地实现（**每个模块的功能**），还应使设计出的处理过程 （**清晰易懂**）。结构化程序设计是实现该目标的关键技术之一;它指导人们用良好的思想方法开发易于（**理解**）、易于（**验证**）的程序。

2、结构化程序设计方法的基本要点是：①采用（**自顶向下、逐步细化**）的程序设计方法；②使用（**三种基本控制结构**）构造程序，避免 GOTO 语句的使用；③（**数据结构合理化**）。

3、任何程序都可由 （**顺序**）、（**选择**）和（**重复**）等三种基本控制结构构造。这三种基本控制结构的共同点是 （**单入口**）和 （**单出口**）。

4、程序设计风格是人们编写程序的（**特点**）、（**习惯**）和（**逻辑思路**）等。

5、语句构造的原则是（**简单直接**），不能因为追求效率而使代码（**复杂化**）。

6、追求效率建立在不损害（**程序可读性**）或（**可靠性**）的基础上。

7、提高程序效率的根本途径在于选择良好的（**设计方法**）、良好的（**数据结构和算法**），而不是靠编程时对语句进行调整。

8、为开发一个特定的项目选择程序设计语言时，必须从（**心理特性**）、（**工程特性**）和技术性能特性等几方面考虑。

9、通常考虑选用程序设计语言的因素有（**项目的应用领域**）、（**软件开发的方法**）、（**软件执行的环境**）、算法和数据结构的复杂性和软件开发人员的知识。

二、单项选择题

1、在编制程序时应采纳的原则之一是(D)。

A. 不限制 goto 语句的使用　　B. 减少或取消注解行

C. 程序越短越好　　**D. 程序结构应有助于读者理解**

2、程序控制一般分为(B)、分支和重复等三种基本控制结构。

A. 分块　**B. 顺序**　C. 迭代　D. 循环

3、以下说法正确的是(B)。

A. 所有改变循环条件的成分都在循环体外

B. 在直到型循环中，循环体至少要执行一次

C. 在当型循环中，循环体至少要执行一次

D. 基本程序结构不允许嵌套

4、源程序文档化要求在每个模块的首部加序言性注释。该注释的内容不应有(B)。

A. 模块的功能　　**B. 语句的功能**　C. 模块的接口　　D. 扇入数

5、功能性注释的作用是解释下面的语句(B)。

A. 怎么做　　**B. 做什么**　　C. 何时做　　D. 为何做

6、对于不好的程序，应当(C).

A. 打补丁　　B. 修改错误　　**C. 重新编写**　　D. 原封不动

7、程序设计语言的心理特性在语言中表现不应包括(C)。

A. 二义性　　B. 简洁性　　**C. 保密性**　　D. 传统性

8、程序设计语言的工程特性之一表现在(A)。

A. 软件的可复用性　　B. 数据结构的可描述性

C. 抽象类型的可描述性　D. 数据库的易操作性

9、程序设计语言的技术特性不应包括(D)。

A. 数据结构的可描述性　B. 抽象类型的可描述性

C. 数据库的易操作性　　**D. 软件的可移植性**

10、Lipow 证明了：当源程序少于 100 个语句时，每行代码的出错率随程序行数的增长(A)。

A. 呈线性相关关系 B.呈指数方式增长 C.呈对数方式增长 D.没有一定规律

三、选择题

1、从下列关于模块化程序设计的叙述中选出 5 条正确的叙述。

(1) 程序设计比较方便，但比较难以维护。

(2) 便于由多个人分工编制大型程序。

(3) 软件的功能便于扩充。

(4) 程序易于理解，也便于排错。

(5) 在主存储器能够容纳得下的前提下，应使模块尽可能大，以便减少模块的个数。(6) 模块之间的接口叫做数据文件。

(7) 只要模块之间的接口关系不变，各模块内部实现细节的修改将不会影响别的模块。

(8) 模块间的单向调用关系叫做模块的层次结构。

(9) 模块越小，模块化的优点越明显。一般来说，模块的大小都在 10 行以下。

正确的叙述有(2)、(3)、(4)、(7)、(8)。

2、从下列叙述中选出 5 条符合程序设计风格指导原则的叙述。

(1) 嵌套的重数应加以限制。

(2) 尽量多使用临时变量。

(3) 不滥用语言特色。

(4) 不用可以省略的括号。

(5) 使用有意义的变量名。

(6) 应尽可能把程序编得短些。

(7) 把常见的局部优化工作留给编译程序去做。

(8) 注解越少越好。

(9) 程序的格式应有助于读者理解程序。

(10) 应尽可能多用 GOTO 语句。

(1)、(3)、(5)、(7)、(9)是正确的。

3、从下面关于程序编制的叙述中，选出三条正确的叙述。

(1) 在编制程序之前，首先必须仔细阅读给定的程序说明书。然后，必须如实地依照说明书编写程序。说明书中常会有含糊不清或难以理解的地方。程序员在作业时应该对这些地方作出适当的解释。

(2) 在着手编制程序时，重要的是采用既能使程序正确地按设计说明书进行处理，又易于出错的编写方法。

(3) 在编制程序时，首先应该对程序的结构充分考虑，不要急于开始编码，

而要象写软件文档那样，很好地琢磨程序具有什么样的功能，这些功能如何安排等等。

(4) 考虑到以后的程序变更，为程序编写完整的说明书是一项很重要的工作。只要有了完整的程序说明书，即使程序的编写形式难以让他人看懂也没有什么关系。

(5) 编制程序时不可缺少的条件是，程序的输入和输出数据的格式都应确定。其他各项规定都是附带的，无足轻重。

(6) 作为一个好的程序，不仅处理速度要快，而且易读易修改等等也都是重要的条件。为了能得到这样的程序，不仅要熟悉程序设计语言的语法，还要注意采用适当的规程和单纯的表现方法，注意使整个程序的结构简洁。

(1)、(4)、(6) 是正确的。

4、从供选择的答案中选出适当的字句填入下面关于程序生产率的描述中的(　　)内。

(1) 1960 年底 Dijkstra 提倡的 (A ⑤) 是一种有效的提高程序设计效率的方法。

(2) Dijkstra 为了使程序结构易于理解，把基本控制结构限于顺序、(B ②)、(C ③)3 种，应避免使用(D ①)。

(3) (A ⑤)不仅提高程序设计的生产率，同时也容易进行程序的(E ③)。

供选择的答案：

A.①标准化程序设计 ②模块化程序设计 ③多道程序设计 ④宏语言
⑤结构化程序设计 ⑥汇编语言 ⑦表格处理语言

B～C. ①分支 ②选择 ③重复 ④计算 ⑤输入输出

D. ①GOTO 语句 ②DO 语句 ③IF 语句 ④REPEAT 语句

E. ①设计 ②调试 ③维护 ④编码

三、简答题

1、试说明下面的两个程序段的功能是什么？可否用另一些等效的程序段来代替它，以提高其可读性。

(1)
```
A[I] = A[I] + A[T];
A[T] = A[I] - A[T];
A[I] = A[I] - A[T];
```

(2)
```
for ( i = 1; i <= n; i ++ )
    for ( j = 1; j <= n; j ++ )
        V[i][j] = ( i / j ) * ( j / i );
```

(1) 的功能是对换 A[I] 与 A[T] 的内容。等效的程序段可以是：

```
TEMP = A[T];   A[T] = A[I];   A[I] = TEMP;
```

(2) 的功能是建立一个单位矩阵 V。等效的程序段可以是：

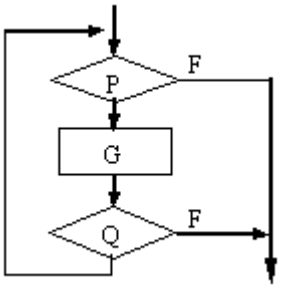
```
for ( i = 1;  i <= n;  i ++ )
    for ( j = 1;  j <= n;  j ++ )
        if ( i == j )  V[i][j] = 1;
            else V[i][j] = 0;
```

2、设下图给出的程序流程图代表一个非结构化的程序，试问：

(1) 为什么说它是一个非结构化的？

(2) 设计一个等价的使用附加标志变量 flag 的结构化程序。

(3) 设计一个使用 break(用于代替 goto)的程序。



(1) 它是一个单入口、两出口的结构，所以是一个非结构化的程序。

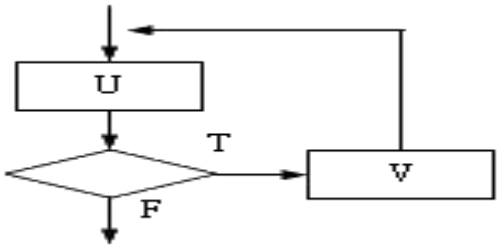
(2) 等价的结构化程序：

```
enum Boolean { false, true }
Boolean flag = true;
  while ( P && flag ) {
         do G;
         if ( !Q ) flag = false;
  }
```

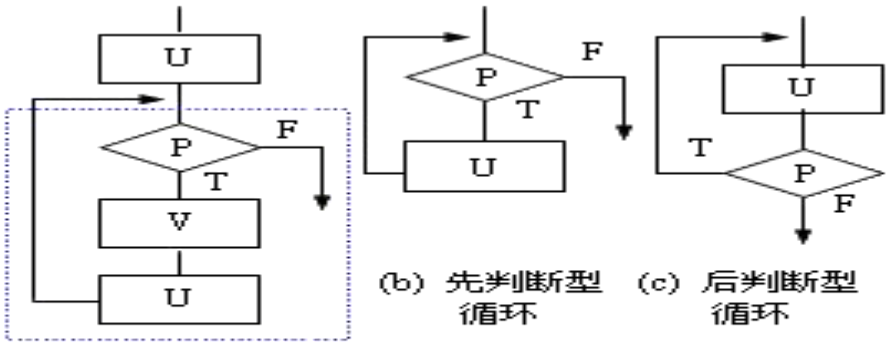
(3) 使用 break 的程序

```
    while ( P ) {
           do G;
           if ( !Q ) break;
    }
```

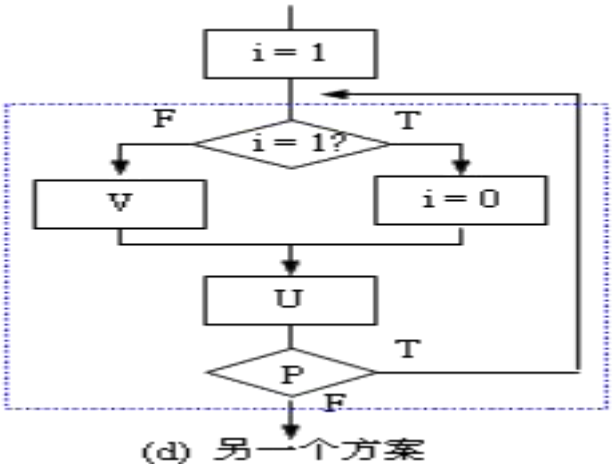
3、有一种循环结构，叫做 N+1/2 循环。其流程图如下所示。这种控制结构不属于基本控制结构：它既不是先判断型循环，又不是后判断型循环。试修改此流程图，将它改为用基本控制结构表示的等效的流程图。



等效的控制流程图如下图中 (a) 所示。

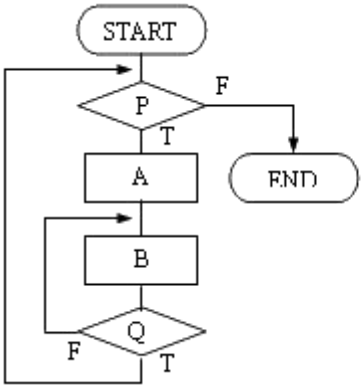


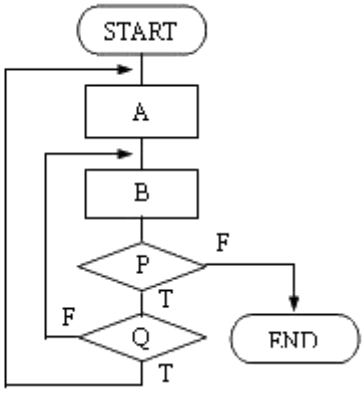
(a) 修改 N+1/2 循环



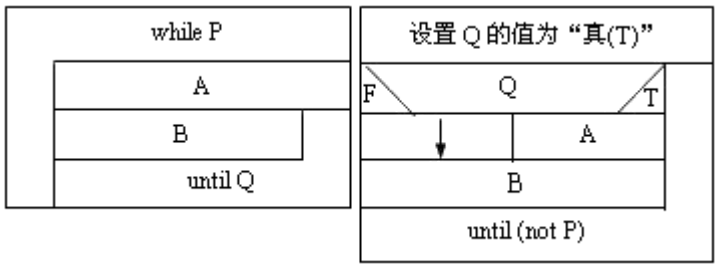
(d) 另一个方案

5、下面是两个程序流程图，试分别用 N-S 图和 PAD 表示之，并计算它们的 McCabe 复杂性度量。

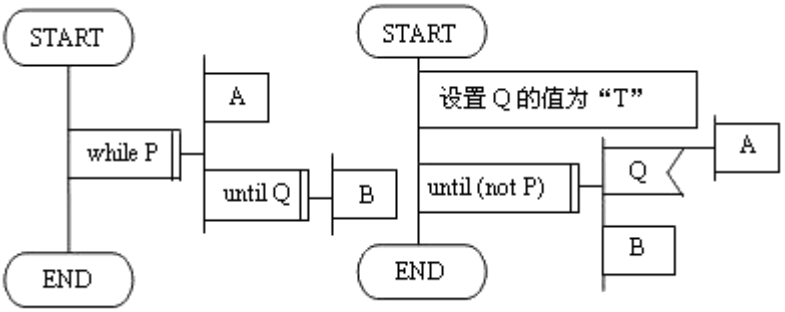




对应的 N-S 图如下。



对应 PAD 图如下。



McCabe 复杂性度量都为 3。

6、用某种软件复杂性度量算法来度量不同类型的程序时，得出的度量值是否真正反映了它们的复杂性？如果对同类型的程序进行度量，其结果是否就比较有价值?

开发规模相同，但复杂性不同的软件，花费的成本和时间会有很大的差异。因此到目前为止，还没有一个软件复杂性度量的方法能够全面、系统地度量任一软件的复杂性，某一种度量方法只偏重于某一方面。所以，用某一种软件复杂性来度量不同类型的程序，所得到的度量值不一定真正反映它们的复杂性。但对同一类型的程序，按某种视点来度量它们的复杂性，其结果还是比较有价值的。

7、软件复杂性有哪几类？软件复杂性度量模型应遵循哪些基本原则？

K.Magel 从六个方面描述软件复杂性：

①理解程序的难度；

②改错及维护程序的难度；

③向他人解释程序的难度；

④按指定方法修改程序的难度；

⑤ 根据设计文档编写程序的工作量；

⑥ 执行程序时需要资源的程度。

软件复杂性度量模型应遵循的基本原则：

⑴ 软件复杂性与程序大小的关系不是线性的；

⑵ 控制结构复杂的程序较复杂；

⑶ 数据结构复杂的程序较复杂；

⑷ 转向语句使用不当的程序较复杂；

⑸ 循环结构比选择结构复杂，选择结构又比顺序结构复杂；

⑹ 语句、数据、子程序和模块在程序中的次序对软件复杂性都有影响；

⑺ 全程变量、非局部变量较多时程序较复杂；

⑻ 参数按地址传递比按值传递更复杂；

⑼ 函数副作用比显式参数传递更难以琢磨；

⑽ 具有不同作用的变量共用一个名字时较难理解；

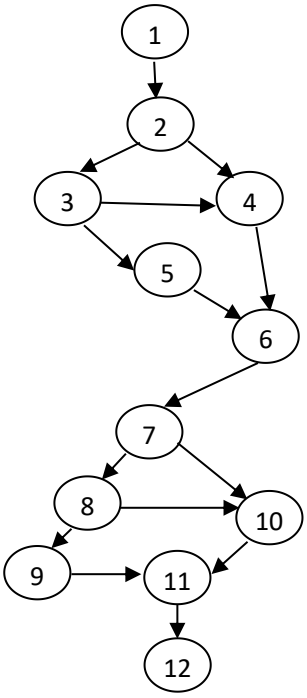
⑾ 模块间或过程间联系密切的程序较复杂；

⑿ 嵌套深度越深程序越复杂。

最典型的两种程序复杂性度量的方法中，McCabe 环路复杂性度量就是针对基本原则(2)制定的度量模型；Halstead 度量是针对程序中操作符和操作数的出现频度而制定的度量模型。

8、程序片段如下，画出其流图，用三种方法计算 V(G)的值。

```
Begin                          1
   If a or b                   2，3
      then procedure x1         4
      else procedure y1         5
   Endif                       6
   If c and d                  7，8
      then procedure x2         9
      else procedure y2         10
   Endif                       11
End                            12
```



V（G）=15-12+2=5

V（G）=p+1=4+1=5

V（G）=二项判定+1=（1+1）+（1+1）+1=5

9、程序的文档分为内部文档和外部文档，简述内部文档和外部文档的内容。

内部文档

头部注释块

其他程序注释

有意义的变量名和声明标示

增进理解的格式

记录数据

外部文档

描述问题

描述算法

描述数据

10、什么是结构化程序设计，简述结构化程序设计的分类。

结构程序设计的经典定义如下所述：“如果一个程序的代码块仅仅通过顺序、选择和循环这 3 种基本控制结构进行连接，并且每个代码块只有一个入口和一个出口，则称这个程序是结构化的。”

对经典定义的扩充：“结构程序设计是尽可能少用 GOTO 语句的程序设计方法。最好仅在检测出错误时才使用 GOTO 语句，而且应该总是使用前向 GOTO 语句。”

经典的结构程序设计：如果只允许使用顺序、IF-THEN-ELSE 型分支和 DO-WHILE 型循环这 3 种基本控制结构实现单入口单出口的程序

扩展的结构程序设计：如果除了上述 3 种基本控制结构之外，还允许使用 DO-CASE 型多分支结构和 DO-UNTIL 型循环结构

修正的结构程序设计：如果再加上允许使用 LEAVE(或 BREAK)结构

第八、九章 测试

一、判断题

1、Fault identification is the process of determining what fault or faults caused the failure, and fault correction or removal is the process of making changes to the system so the faults are removed.(T)

2、Module testing is the process of verifying that the system components work together as described in the system and program design specifications. (F)

二、填空

1、软件测试阶段的基本任务应当是根据软件开发各阶段的(**文档资料**)和程序的(**内部结构**)，精心设计一批“高产”的测试用例，利用这些测试用例(**执行程序**)，找出软件中潜藏的各种错误和缺陷。

2、测试用例不仅要选用合理的测试输入数据，还需要选用不合理的测试输入数据，这样能更多地(**发现错误**)，提高程序的可靠性。对于不合理的测试输入数据，程序应(**拒绝执行**)，并给出相应的提示。

3、动态测试指通过 (**运行程序**)发现错误。对软件产品进行动态测试时使用黑盒测试法和(**白盒测试**)法。

4、静态测试指(被测试程序)不在机器上运行，而是采用(**人工检测**)和(**计算机辅助静态分析**)的手段对程序进行检测。

5、黑盒测试依据(**软件需求规格说明**)，检查程序是否满足(**功能要求**)。因此，黑盒测试由称为功能测试或(**数据驱动**)测试。

6、白盒测试以检查处理过程的细节为基础,对程序中尽可能多的(**逻辑路径**)进行测试，检查内部(**数据结构**)和 (**运行状态**)是否有错，程序的(**语句或条件**)与预期的状态是否一致。

7、在单元测试中，驱动模块的作用是用来模拟被测模块的(**上层调用模块**)。它的工作是接受 (**为测试输入数据**)，以上层模块调用被测模块的形式 (**驱动**)被测模块，接收被测模块的(**实测结果**)并输出。

8、在单元测试中，桩模块用来代替被测模块的(**子模块**)。其作用是(**返回被测模块所需**)的信息。

9、错误的群集现象是指模块错误发现率与模块的残留错误数成 (**正比**)关系。

三、选择题

1、在软件测试中，下面说法中错误的是(A)。

A. 测试是为了发现程序中的错误而执行程序的过程

B. 测试是为了表明程序是正确的

C. 好的测试方案是极可能发现迄今为止尚未发现的错误的方案

D. 成功的测试是发现了至今为止尚未发现的错误的测试

2、软件测试的目的是(B)。

A.试验性运行软件 **B.发现软件错误**

C.证明软件正确　　D.找出软件中全部错误

3、软件测试用例主要由测试输入数据和(C　)两部分组成。

A. 测试计划 B. 测试规则 **C. 测试的预期结果** D. 以往测试记录分析

4、与设计测试用例无关的文档是(A　)。

A. 项目开发计划 B. 需求规格说明书C. 软件设计说明书　D. 源程序

5、软件测试是软件质量保证的主要手段之一，测试的成本已超过(A　)的30%以上。因此，提高测试的有效性非常重要。

A.软件开发成本　　B. 软件维护成本

C.软件开发成本和维护成本 D.软件研制成本

6、“高效”的测试是指(C　)。

A. 用适量的测试用例说明被测试程序正确无误

B. 用适量的测试用例说明被测试程序符合相应的要求

C. 用适量的测试用例发现被测试程序尽可能多的错误

D. 用适量的测试用例纠正被测试程序尽可能多的错误

7、如果想要进行成功的测试，为其设计测试用例主要依赖于(B　)。

A. 黑盒测试方法**B. 测试人员的经验** C. 白盒测试方法 D. 错误推测法

8、使用白盒测试方法时，确定测试数据应根据(A　)和指定的覆盖标准。

A. 程序的内部结构　B. 程序的复杂性 C. 使用说明书　D. 程序的功能

9、在用逻辑覆盖法设计测试用例时，有语句覆盖、分支覆盖、条件覆盖、判定-条件覆盖、条件组合覆盖和路径覆盖等。其中(D　)是最强的覆盖准则。

A. 语句覆盖　　B. 条件覆盖　C. 判定-条件覆盖　**D. 路径覆盖**

10、在设计测试用例时，(A　)是用得最多的一种黑盒测试方法。

A. 等价类划分　　B. 边界值分析 C. 因果图　　D. 功能图

11、在黑盒测试中，着重检查输入条件的组合的测试用例设计方法是(D　)。

A. 等价类划分　　B. 边界值分析　C. 错误推测法　　**D. 因果图法**

12、从下列叙述中，能够与软件开发各阶段，如需求分析、设计、编码相对应的软件测试是(D　)。

A. 组装测试、确认测试、单元测试 B. 单元测试、组装测试、确认测试

C. 单元测试、确认测试、组装测试 **D. 确认测试、组装测试、单元测试**

13、单元测试将根据在(D　)阶段中产生的规格说明进行。

A. 可行性研究与计划　B. 需求分析　C. 概要设计 **D. 详细设计**

14、组装测试计划是在(C　)阶段制定的。

A. 可行性研究与计划　B. 需求分析　**C. 概要设计** D. 详细设计

15、确认测试计划是在(B　)阶段制定的。

A. 可行性研究与计划　**B. 需求分析**　C. 概要设计 D. 详细设计

16、软件的组装测试最好是由(D　)承担，以提高组装测试的效果。

A. 该软件的设计者　　B. 该软件开发组的负责人

C. 该软件的编程者　　**D. 不属于该开发组的人员**

17、(D　)是简化了的模拟较低层次模块功能的虚拟子程序。

A. 过程　　B. 函数　　C. 仿真　　**D. 桩**

18、(A　)是指为查明程序中的错误和缺陷，可能使用的工具和手段。

A. 调试技术　　B. 测试技术　C. 跟踪法　　D. 动态测试

19、从已发现故障的存在到找到准确的故障位置并确定故障的性质，这一过

程称为(C)。

A. 错误检测　　B. 故障排除　　C. 调试　　**D. 测试**

20、统计资料表明，软件测试的工作量占整个软件开发工作量的(C)。

A. 30%　　B. 70%　　**C. 40%～50%**　　D. 95%

21、软件测试计划是一些文档，它们描述了(D)。

A. 软件的性质　　B. 软件的功能和测试用例

C. 软件的规定动作　**D. 对于预定的测试活动将要采取的手段**

22、IBM 公司的统计资料表明，使用静态测试的方法最高可以查出在测试中查出的全部软件错误的(B)。

A. 80%　　**B. 70%**　　C. 50%　　D. 35%

23、黑盒测试方法的优点是(D)。

A. 可测试软件的特定部位　B. 能站在用户立场测试

C. 可按软件内部结构测试　**D. 可发现实现功能需求中的错误**

24、白盒测试方法的优点是(C)。

A. 可测试软件的特定部位　B. 能站在用户立场测试

C. 可按软件内部结构测试　D. 可发现实现功能需求中的错误

25、等价类划分完成后，就可得出(C)，它是确定测试用例的基础。

A. 有效等价类　B. 无效等价　　**C. 等价类表**　　D. 测试用例集

26、由因果图转换出来的(A)是确定测试用例的基础。

A. 判定表　　B. 约束条件表　C. 输入状态表　　D. 输出状态表

三、选择填空题

1、软件测试的目的是（ A ②）。为了提高测试的效率，应该（ B ④）。使用白盒测试方法时，确定测试数据应根据（ C ①）和指定的覆盖标准。与设计测试数据无关的文档是（ D ④）。

软件的集成测试工作最好由（ E ④）承担，以提高集成测试的效果

供选择的答案：

A.　①评价软件的质量　　**②发现软件的错误**

　　③找出软件中的所有错误　④证明软件是正确的

B.　①随机地选取测试数据

　　②取一切可能的输入数据作为测试数据

　　③在完成编码以后制定软件的测试计划

　　④选择发现错误的可能性大的数据作为测试数据

C.　**①程序的内部逻辑**　②程序的复杂程度　③使用说明书　④程序的功能

D.　①该软件的设计人员　②程序的复杂程度　③源程序　**④项目开发计划**

E.　①该软件的设计人员　②该软件开发组的负责人

　　③该软件的编程人员　**④不属该软件开发组的软件人员**

2、程序的三种基本控制结构是（ A ②）。它们的共同点是（ B ④）。结构化程序设计的一种基本方法是（ C ④）。软件测试的目的是（ D ②）。软件调试的目的是（ E ①）。

供选择的答案：

A.　①过程，子程序，分程序　**②顺序，条件，循环**　③递归，堆栈，队列　④调用，返回，转移

B.　①不能嵌套使用　②只能用来写简单的程序　③已经用硬件实现

④只有一个入口和一个出口

C. ①筛选法 ②递归法 ③归纳法 **④逐步求精法**

D. ①证明程序中没有错误 **②发现程序中的错误** ③测量程序的动态特性 ④检查程序中的语法错误

E. **①找出错误所在并改正之** ②排除存在错误的可能性 ③对错误性质进行分类 ④统计出错的次数

3、等价类划分是一种典型的（ A ②）方法，也是一种非常实用的重要的测试方法。使用这一方法，完全不考虑程序的（ B ①）。用所有可能输入的数据来测试程序是不可能的，只能从全部可供输入的数据中选择一个（ C ②）进行测试。（ D ⑤）是指某个输入域的集合，在该集合中，各个输入数据对于揭露程序中的错误是（ E ③）。

供选择的答案：

A： ①白盒测试方法 **②黑盒测试方法**

B： **①内部结构** ②外部环境 ③顺序 ④流程

C～E： ①全集 **②子集** **③等效** ④不同的 **⑤等价类** ⑥典型集

4、①黑盒测试方法的缺点是(A)和(D)。

A.不可测试软件的特定部位 B.不能发现功能需求中的错误

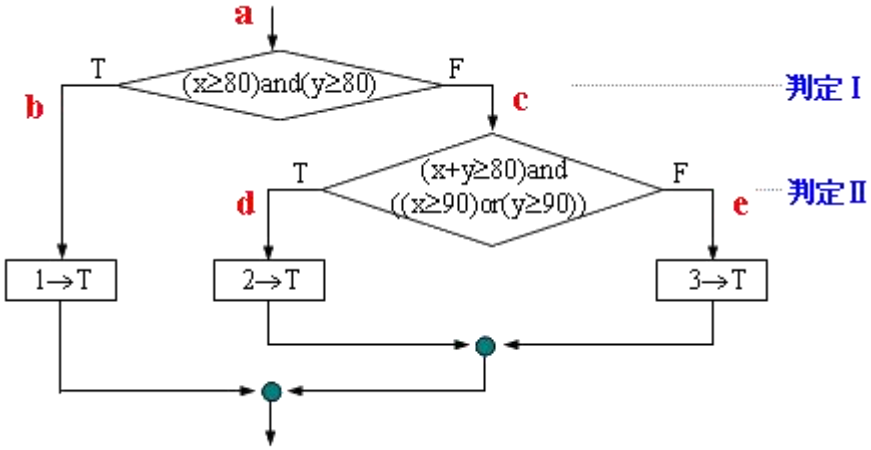
C.无法检验软件的外部特性 **D.无法测试未实现功能需求的软件的内部缺陷**

②白盒测试方法的缺点是(B)和(C)。

A. 不可测试软件的特定部位 **B. 不能发现功能需求中的错误**

C.无法检验软件的外部特性 D.无法测试未实现功能需求的软件的内部缺陷

5、如图所示的程序有三条不同的路径。分别表示为 L1(a→b)、L2(a→c→d)、L3(a→c→e)，根据所有路径测试的标准，从供选择的答案中找出满足相应标准的最小测试用例组。(用①~ ⑩回答)



供选择的答案：

① x = 90, y = 90 ② x = 50, y = 50

③ x = 90, y = 90
x = 50, y = 50

④ x = 90, y = 70
x = 40, x = 90

⑤ x = 90, y = 90 ⑥ x = 90, y = 90

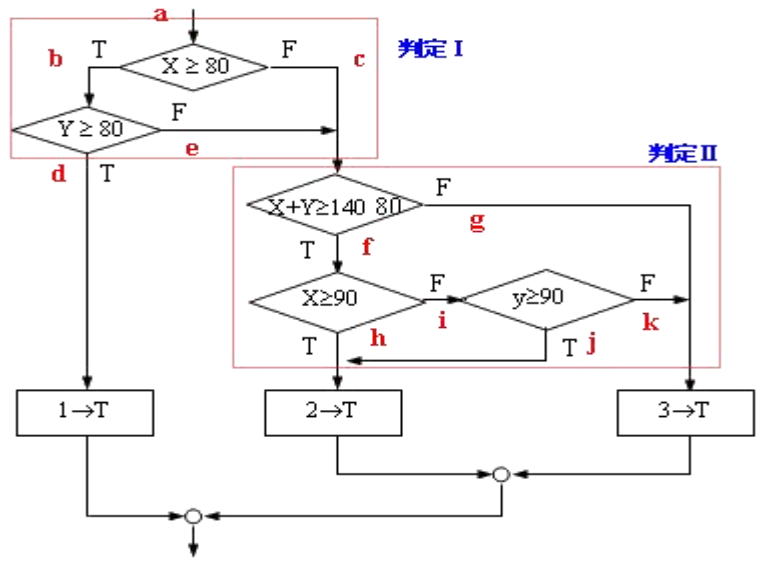
	x = 50, y = 50		x = 70, y = 90
	x = 90, y = 70		x = 50, y = 50
⑦	x = 90, y = 90	⑧	x = 90, y = 90
	x = 50, y = 50		x = 50, y = 50
	x = 80, y = 70		x = 90, y = 50
	x = 70, y = 90		x = 80, y = 80
⑨	x = 90, y = 90	⑩	x = 90, y = 90
	x = 90, y = 70		x = 80, y = 80
	x = 90, y = 30		x = 90, y = 70
	x = 70, y = 90		x = 90, y = 30
	x = 30, y = 90		x = 70, y = 90
	x = 70, y = 70		x = 30, y = 90
	x = 50, y = 50		x = 70, y = 70
	x = 50, y = 50		

解：

路径覆盖	x = 90, y = 90	x≥80=T, y≥80=T, x+y≥140=T, x≥90=T, y≥90=□, 路径 a→b，执行 1→T。
⑤	x = 50, y = 50	x≥80=F, y≥80=□, x+y≥140=F, x≥90=□, y≥90=□ 路径 a→c→e，执行 3→T。
	x = 90, y = 70	x≥80=T, y≥80=F, x+y≥140=T, x≥90=T, y≥90=□ 路径 a→c→d，执行 2→T。

注意：测试是一个程序的执行过程。对于逻辑表达式 A or B，当 A 为真时不再对 B 做判断，对于逻辑表达式 A and B，当 A 为假时不再对 B 做判断。未能做判断的条件，在解答中用“□”表示。

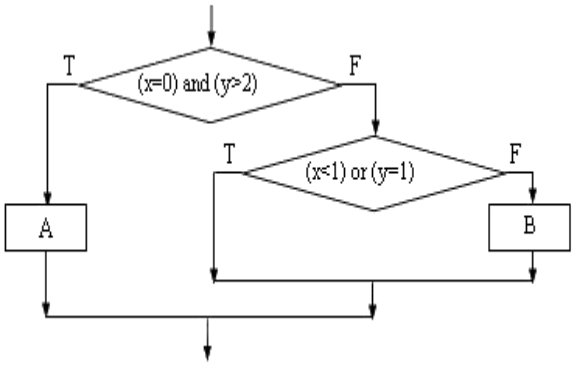
因为流程图有 3 条路径，只需 3 个测试用例就够了。如果将判定中的复合条件表达式改为单个条件的嵌套选择结构，第一个判定有 3 条路径，其中两条路径通向第二个判定。第二个判定有 4 条路径，组合起来总共应有 9 条路径。但是，其中受测试条件的限制，有 3 条路径不可达，因此，程序中应有 6 条路径，需要 6 个测试用例来覆盖它们。



路径覆盖	x = 90, y = 90	(X≥80=T)and(Y≥80=T), 路径 a→b→d
	x = 90, y = 70	(X≥80=T)and(Y≥80=F), (X+Y≥140=T)and(X≥90=T), 路径 a→b→e→f→h
	x = 90, y = 30	(X≥80=T)and(Y≥80=F), (X+Y≥140=F), 路径 a→b→e→g
	x = 50, y = 50	(X≥80=F), (X+Y≥140=F), 路径 a→c→g
	x = 70, y = 90	(X≥80=F), (X+Y≥140=T)and((X≥90=F)or(Y≥90=T)), 路径 a→c→f→i→j
	x = 70, y = 70	(X≥80=F), (X+Y≥140=T)and((X≥90=F)or(Y≥90=F)), 路径 a→c→f→i→k
	路径 a→b→e→f→i→j, a→b→e→f→i→k, a→c→f→h 不可达	

6、在白盒测试用例设计中，有语句覆盖、分支覆盖、条件覆盖、路径覆盖等，其中（ A ）是最强的覆盖准则。为了对如下图所示的程序段进行覆盖测试，必须适当地选取测试用例组。若 x，y 是两个变量，可供选择的测试用例组共有Ⅰ、Ⅱ、Ⅲ、Ⅳ四组，如表中给出，实现路径覆盖至少应采取的测试用例组是（ B）或（ C ）。

	x	y
测试用例组 I	0	3
测试用例组 II	1	2
测试用例组III	-1	2
测试用例组IV	3	1



供选择的答案：

A： ①语句覆盖 ②条件覆盖 ③判定覆盖 **④路径覆盖**

B、C： ①Ⅰ和Ⅱ组 ②Ⅱ和Ⅲ组 ③Ⅲ和Ⅳ组 ④Ⅰ和Ⅳ组 **⑤Ⅰ、Ⅱ、Ⅲ组** ⑥Ⅱ、Ⅲ、Ⅳ组 ⑦Ⅰ、Ⅲ、Ⅳ组 **⑧Ⅰ、Ⅱ、Ⅳ组**

解答：A. ④ B. ⑤ C. ⑧

判定表：

条件及其组合 \ 测试用例组	I	II	III	IV
1. 条件 x＝0	T	F	F	F
2. 条件 y＞2	T	F	F	F
3. 条件 x＜1	T	F	T	F
4. 条件 y＝1	F	F	F	T
5. 判定 (x=0)and(y>2)	T	F	F	F
6. 判定 (x<1)or(y=1)	T	F	T	T
7. 路径 (x=0)and(y>2)=T	T	F	F	F
8. 路径 (x=0)and(y>2)=F and ((x<1)or(y=1))=T	F	F	T	T
9. 路径 (x=0)and(y>2)=F and ((x<1)or(y=1))=F	F	T	F	F

为路径覆盖选取测试用例情形：总共 3 条路径，需 3 个测试用例，可选使各路径为 T 的测试用例。Ⅰ、Ⅱ、Ⅲ或Ⅰ、Ⅱ、Ⅳ均可，可选⑤或⑧。

四、简答题

1、closed-box testing

closed-box testing: view the tested object as a closed-box, testers do not think about the logical structure and characteristic inside at all, just examine whether the program’s function accord with the function instruction according as its’ demand standard instruction. (黑盒测试：测试对象看作一个黑盒子，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。

2、opened-box testing.

opened-box testing: view the tested object as an opened-box, it allows testers use the logical structure and information inside the process to design or choose the testing use case and test all the logical routes insides the process. Through examine the state of process in different points to make sure whether the actual state conforms to the expected state.（白盒测试：把测试对象看作一个透明盒子，它允许测试人员利用程序内部的逻辑结构及有关信息，设计或选择测试用例，对程序的所有逻辑路径进行测试。通过在不同点检查程序的状态，确定实际的状态是否与预期的状态一致）

3、解释驱动程序和存根程序的概念。

驱动程序：是一个“主程序”，它接收测试数据，把这些数据传送给被测试的模块，并且印出有关的结果。

存根程序代替被测试的模块所调用的模块。因此存根程序也可以称为“虚拟子程序”。它使用被它代替的模块的接口，可能做最少量的数据操作，印出对入

口的检验或操作结果，并且把控制归还给调用它的模块。

4、简述大型软件的测试过程。

模块测试、组件测试、单元测试

集成测试

系统测试（功能测试、性能测试、验收测试、安装测试）

5、解释黑盒测试、白盒测试，说明其一般用途。

黑盒测试：已知产品应该具有的功能，通过测试检验其每个功能是否都能够正常使用。又称功能测试。

用途:把程序看成一个黑盒子,仅仅考虑输入和输出的对应关系和程序接口，完全不考虑它的内部结构和处理过程。一般用于综合测试、系统测试等。

白盒测试：已知产品内部的工作过程，通过测试检验产品内部动作是否都能按照需求定义的规定正常使用。

用途：必须完全了解程序的内部结构和处理过程，才能按照程序内部的逻辑测试，以检验程序中每条路径是否正确，因此 一般用于规模较小的程序和单元测试。

6、解释软件测试、调试的概念，叙述二者间的关系。

测试：为了发现程序中的错误而执行程序的过程；

调试：确定错误的位置，并改正错误；

先测试，然后调试

7、软件测试的指导原则的内容。

所有的测试都应追溯到用户需求，从用户角度看，最严重的错误是不能满足用 户需求。

制定测试计划，并严格执行，排除随意性。测试计划在需求分析阶段就开始了，详细的测试用例在设计阶段确定。

Pareto 原则：所发现错误的 80%很可能源于程序模块的 20%中。

测试应当从“小规模”开始，逐步转向“大规模”。

穷举测试是不可能的（Exhaustive testing）。

由独立的第三方或专门的测试小组进行独立测试。

测试用例由输入数据和相应的预期输出组成。

测试用例不仅选用合理的输入数据，还要选择不合理的。

不仅检查程序是否做了应该做的事，还应该检查是否不应该做的。

长期保留测试用例，以便进行回归测试和维护。

五、计算及综合题

1、下面是快速排序算法中的一趟算法，其中 datalist 是数据表，它有两个数据成员：一是元素类型为 Element 的数组 V，另一个是数组大小 n。算法中用到两个操作，一是取某数组元素 V[i]的关键码操作 getKey ( )，一是交换两数组元素内容的操作 Swap( )：

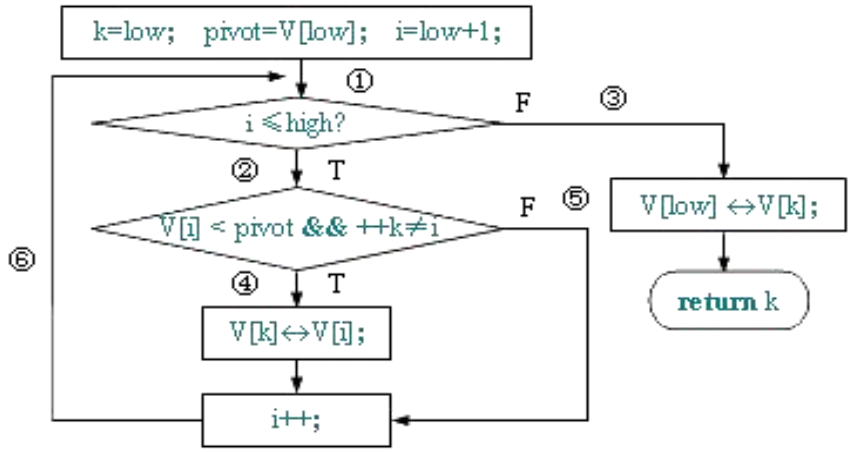
```
int Partition ( datalist &list, int low, int high ) {
//在区间[low,high]以第一个对象为基准进行一次划分，k 返回基准对象回放位置。
int  k = low;  Element pivot = list.V[low];   //基准对象
for ( int i = low+1; i <= high; i++ )       //检测整个序列，进行划分
      if ( list.V[i].getKey ( ) < pivot.getKey( ) && ++ k != i )
```

```
        Swap ( list.V[k], list.V[i] );        //小于基准的交换到左侧去
        Swap ( list.V[low], list.V[k] );      //将基准对象就位
        return k;                             //返回基准对象
位置
    }
```

(1) 试画出它的程序流程图；

(2) 试利用路径覆盖方法为它设计足够的测试用例（循环次数限定为 0 次，1 次和 2 次）。

解答：（1）流程图如下。



(2) 测试用例设计

循环	输入条件							输出结果					执行路径
次数	low	high	k	i	V[0]	V[1]	V[2]	k	i	V[0]	V[1]	V[2]	
0	0	0	0	1	–	–	–	0	1	–	–	–	①③
1	0	1	0	1	**1**	**2**	–	0	2	**1**	2	–	①②⑤⑥③
	0	1	0	1	**2**	**1**	–	1	2	1	**2**	–	①②④⑥③
	0	1	0	1	**1**	**1**	–	0	2	**1**	1	–	①②⑤⑥③
2	0	2	0	1	**1**	**2**	**3**	0	3	**1**	2	3	①②⑤⑥②⑤⑥③
	0	2	0	1	**1**	**2**	**1**	0	3	**1**	2	1	①②⑤⑥②⑤⑥③
	0	2	0	1	**2**	**3**	**1**	1	3	1	**2**	3	①②⑤⑥②④⑥③
	0	2	0	1	**3**	**2**	**1**	2	3	1	2	**3**	①②④⑥②④⑥③
	0	2	0	1	**2**	**1**	**2**	1	3	1	**2**	2	①②④⑥②⑤⑥③
	0	2	0	1	**2**	**1**	**3**	1	3	1	**2**	3	①②④⑥②⑤⑥③
	0	2	0	1	**1**	**1**	**2**	0	3	**1**	1	2	①②⑤⑥②⑤⑥③
	0	2	0	1	**2**	**2**	**1**	1	3	1	**2**	2	①②⑤⑥②④⑥③
	0	2	0	1	2	2	2	0	3	**2**	2	2	①②⑤⑥②⑤⑥③

画程序流程图是设计测试用例的关键。考虑测试用例设计需要首先有测试输入数据，还要有预期的输出结果。对于此例，控制循环次数靠循环控制变量 i

和循环终值 high。循环 0 次时，取 low = high，此时一次循环也不做。循环一次时，取 low +1 = high，循环二次时，取 low+2 = high。

2、Calculate the number of faults

A program is seeded with 100 faults. The test team finds 84 faults, in which 70 is seeded. Try to estimate how many faults are still unfound in the program.

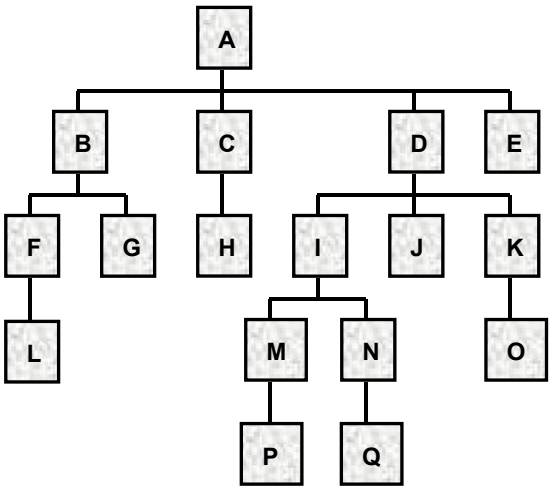
设估计固有错误数为 X，则 70/100=(84-70)/X, X=20, 故未找到的固有错误数为 20-（84-70）=6 个

3、Calculate the number of seeded faults

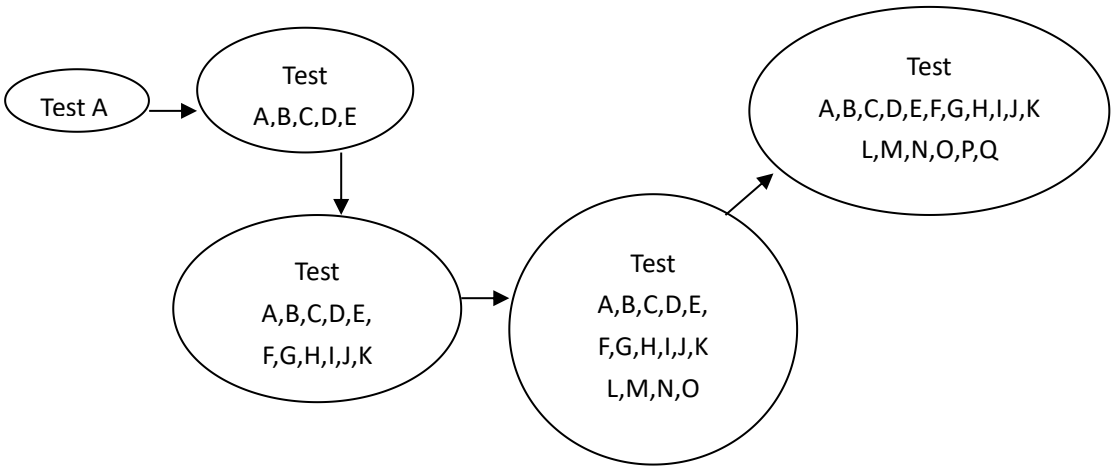
Suppose declare there are no fault in a program, to archive 98% level of confidence, how many faults should be seeded into the program?

设需植入错误数为 S，则：C=S/(S-0+1)=98/100, S=49.

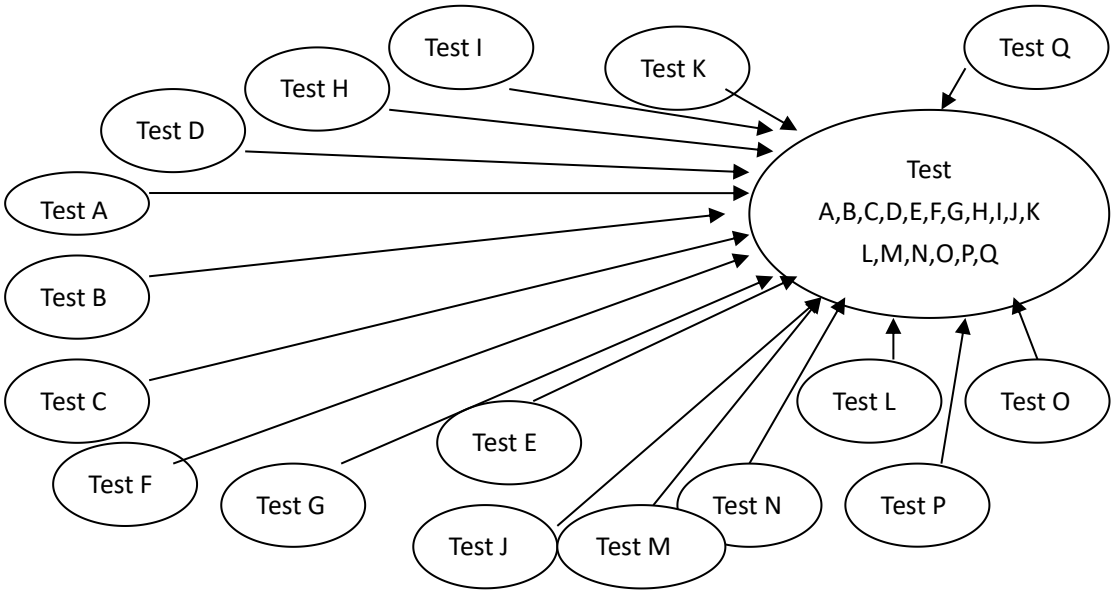
4、The follow Figure illustrates the component hierarchy in a software system. Describe the sequence of tests for integrating the components using a top-down approach; a big-bang approach; and a modified sandwich approach.



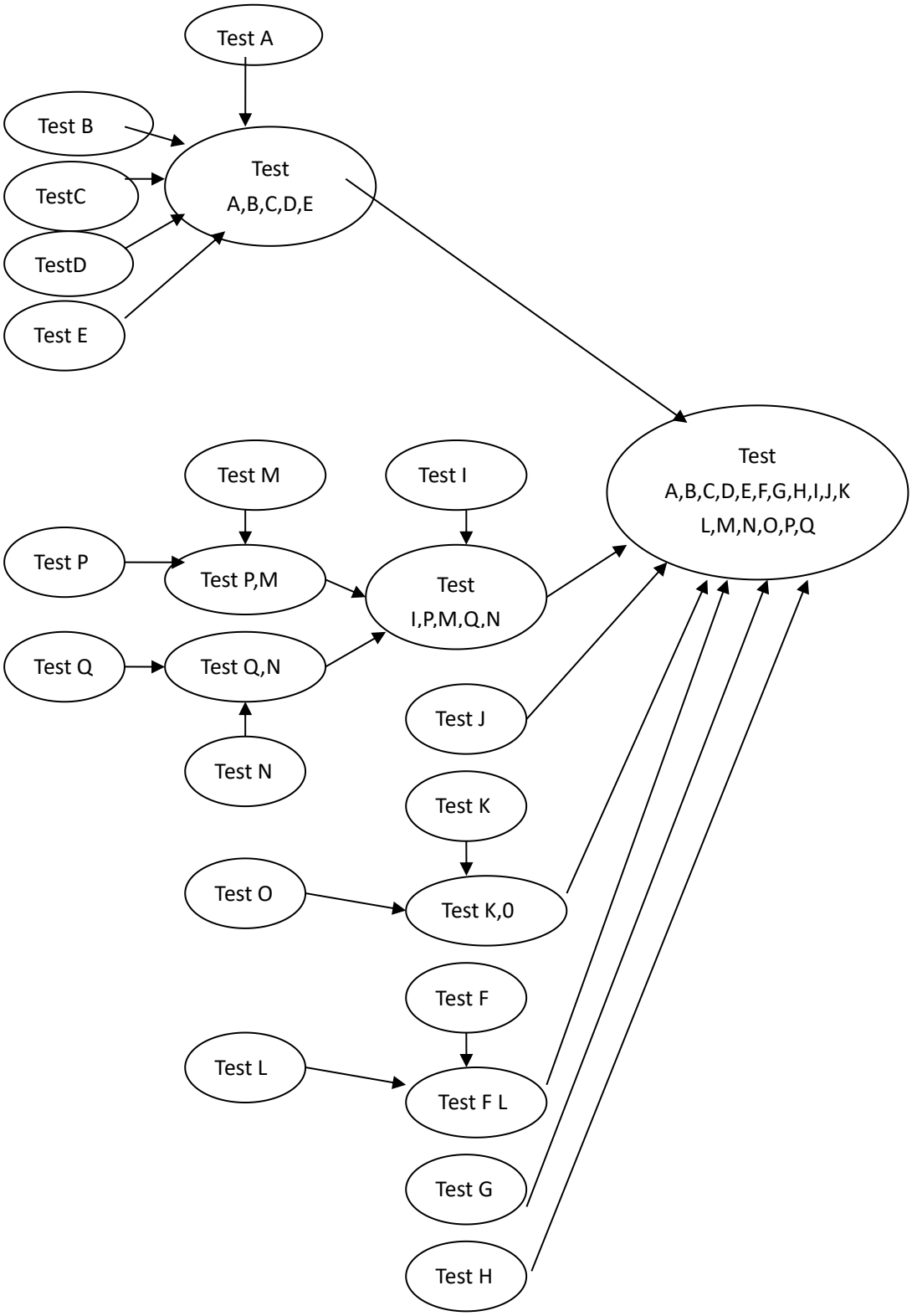
Top-down:



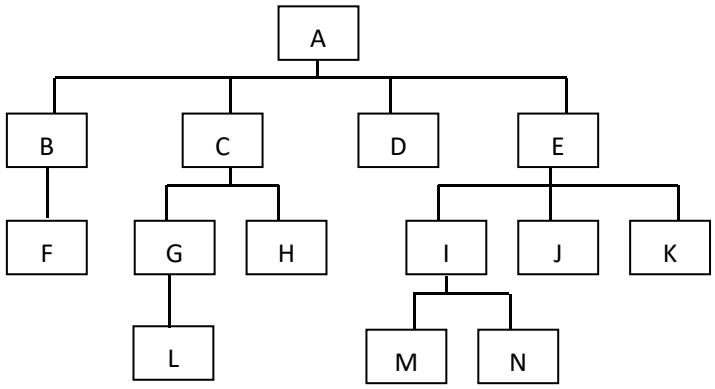
Big-Bang

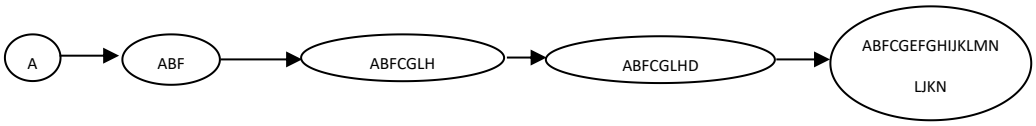


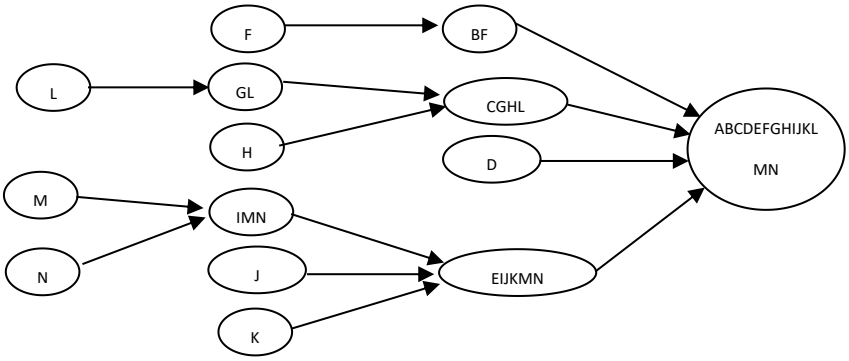
modified sandwich approach:

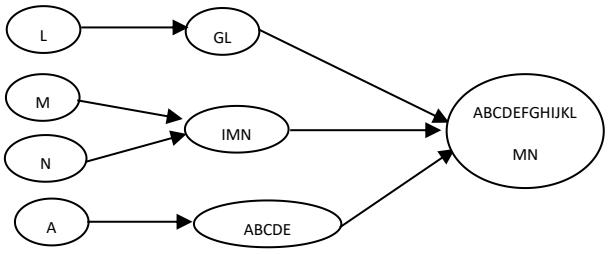

Test A
Test B
TestC
TestD
Test E
Test
A,B,C,D,E
Test M
Test I
Test P
Test P,M
Test
I,P,M,Q,N
Test Q
Test Q,N
Test N
Test J
Test K
Test O
Test K,0
Test F
Test L
Test F L
Test G
Test H
Test
A,B,C,D,E,F,G,H,I,J,K
L,M,N,O,P,Q

5、Figure illustrates the component hierarchy in a software system. Describe the sequence of tests for integrating the components using a top-down approach（depth）; a bottom-up approach; a sandwich approach（FGHIJK mesosphere）.









6、Calculate Reliability Availability, and maintainability

MTTF (Mean Time to Failure) =99

MTTR (Mean Time to Repair) =2

Reliability=(MTTF)/(1+MTTF)=99/100=0.99

Availability= (MTBF)/(1+MTBF)=(99+2)/(1+99+2)=0.9902

Maintainability= 1/(1+MTTR)=0.3333

7、下面是选择排序的程序，其中 datalist 是数据表，它有两个数据成员：一是元素类型为 Element 的数组 V，另一个是数组大小 n。算法中用到两个操作，一是取某数组元素 V[i]的关键码操作 getKey()，一是交换两数组元素内容的操作 Swap()：

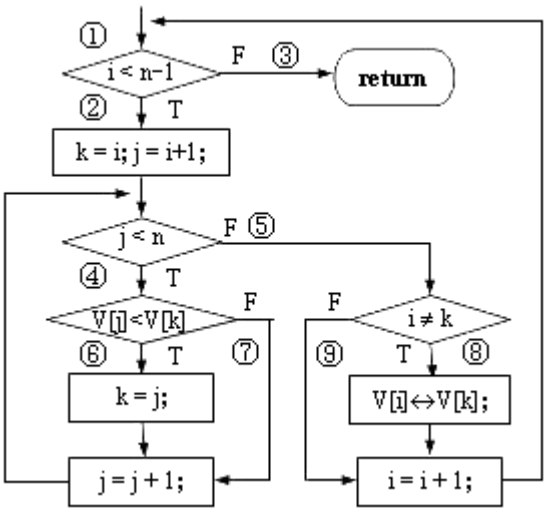
```
void SelectSort ( datalist & list ) {
      //对表 list.V[0]到 list.V[n-1]进行排序,  n 是表当前长度。
        for ( int i = 0; i < list.n-1; i++ ) {
            int k = i;
        //在 list.V[i].key 到 list.V[n-1].key 中找具有最小关键码的对象
          for ( int j = i+1; j < list.n; j++)
            if ( list.V[j].getKey ( ) < list.V[k].getKey ( ) ) k = j;
                                  //当前具最小关键码的对象
            if ( k != i ) Swap ( list.V[i], list.V[k] );      //交换
           }
      }
```

(1) 试计算此程序段的 McCabe 复杂性；

(2) 用基本路径覆盖法给出测试路径；

(3) 为各测试路径设计测试用例。

解：



(1) McCabe 环路复杂性 = 5

(2) 独立路径有 5 条：

①③

①②⑤⑧……

①②⑤⑨……

①②④⑥……

①②④⑦……

(3) 为各测试路径设计测试用例:

路径①③：取 n = 1

路径①②⑤⑧……：取 n = 2,

预期结果：路径⑤⑧③不可达

路径①②⑤⑨……：取 n = 2,

预期结果：路径⑤⑨③不可达

路径①②④⑥⑤⑧③：

取 n = 2, V[0] = 2, V[1] = 1, 预期结果：k = 1, V[0] = 1, V[1] = 2

路径①②④⑥⑤⑨③：

取 n = 2, V[0] = 2, V[1] = 1, 预期结果：k = 1, 路径⑨③不可达

路径①②④⑦⑤⑧③：

取 n = 2, V[0] = 1, V[1] = 2, 预期结果：k = 0, 路径⑧③不可达

路径①②④⑦⑤⑨③：

取 n = 2, V[0] = 1, V[1] = 2, 预期结果：k = 0, V[0] = 1, V[1] = 2

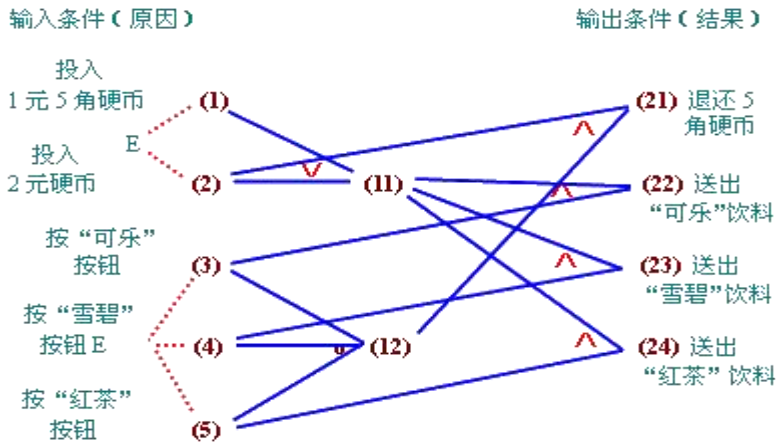
8、设要对一个自动饮料售货机软件进行黑盒测试。该软件的规格说明如下：“有一个处理单价为 1 元 5 角钱的盒装饮料的自动售货机软件。若投入 1 元 5 角硬币，按下“可乐”、“雪碧”或“红茶”按钮，相应的饮料就送出来。若投入的是 2 元硬币，在送出饮料的同时退还 5 角硬币。”

(1) 试利用因果图法，建立该软件的因果图；

(2) 设计测试该软件的全部测试用例。

解：

因果图：



测试用例设计：

			1	2	3	4	5	6	7	8	9	10	11
输入	投入 1 元 5 角硬币	(1)	1	1	1	1	0	0	0	0	0	0	0
	投入 2 元硬币	(2)	0	0	0	0	1	1	1	1	0	0	0
	按“可乐”按钮	(3)	1	0	0	0	1	0	0	0	1	0	0
	按“雪碧”按钮	(4)	0	1	0	0	0	1	0	0	0	1	0
	按“红茶”按钮	(5)	0	0	1	0	0	0	1	0	0	0	1
中间结点	已投币	(11)	1	1	1	1	1	1	1	1	0	0	0
	已按钮	(12)	1	1	1	0	1	1	1	0	1	1	1
输出	退还 5 角硬币	(21)	0	0	0	0	1	1	1	0	0	0	0
	送出“可乐”饮料	(22)	1	0	0	0	1	0	0	0	0	0	0
	送出“雪碧”饮料	(23)	0	1	0	0	0	1	0	0	0	0	0
	送出“红茶”饮料	(24)	0	0	1	0	0	0	1	0	0	0	0

每个纵列是一个测试用例。

9、输入三角形的三条边，判断其是否能构成三角形。要求自行选定测试技术，给出适度的测试用例，实现对问题的语句覆盖。

解：第一步，综合使用边界值分析、等价类划分和错误推测技术，设计出以下 11 种应测试的情况。

1 正常的任意三角形；

2 正常的等边三角形；

3 正常的等腰三角形；

4 退化的三角形；

5 三条边不构成三角形；

6 一条边长度为零；

7 两条边长度为零；

8 三条边长度全为零；

9 输入数据中包含负整数；

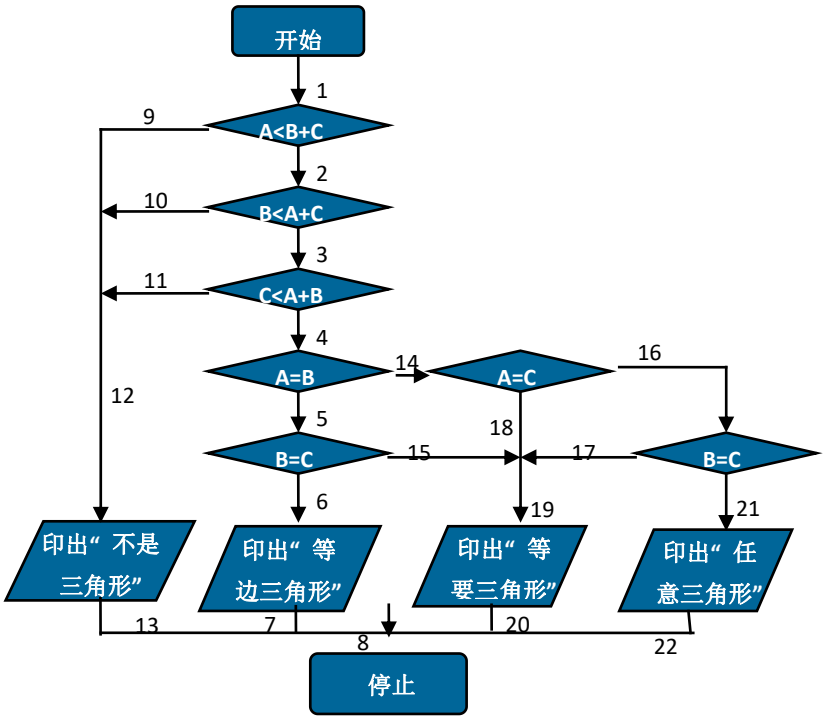
10 输入数据不全；

11 输入数据中包含非整数型的数据。

第二步，为上述 11 种情况设计测试用例

测试功能	测试数据		
1.等边	10, 10, 10	-, -, -	-, -, -
2.等腰	10, 10, 17	10, 17, 10	17, 10, 10
3.任意	8, 10, 12	10, 12, 8	12, 8, 10
4.非三角形	10, 10, 21	10, 21, 10	21, 10, 10
5.退化情况	10, 5, 5	5, 10, 5	5, 5, 10
6.零数据	0, 0 0	-, -, -	-, -, -
	0, 0, 17	0, 17, 0	17, 0, 0
	0, 10, 12	12, 0, 10	12, 10, 0
7.负数据	-10, -10, -10	-, -, -	-, -, -
	-10, -10, 17	-10, 17, -10	17, -10, -10
	-8, 10, 17	17, -8, 10	10, 17, -8
8.遗漏数据	-, -, -	-, -, -	-, -, -
	10, -, -	-, 10, -	-, -, 10
	8, 10, -	8, -, 10	-, 8, 10
9.无效数据	A, B, C	-, -, -	-, -, -

第三步，依据测试用例，检查覆盖程度。

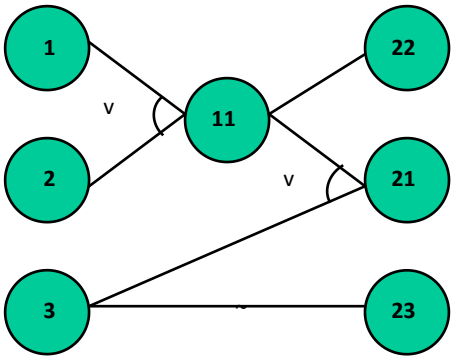


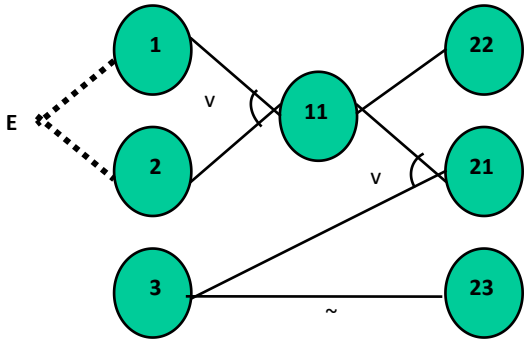
编号	测试数据	覆盖边
1	10, 10, 10	1, 2, 3, 4, 5, 6, 7, 8
2a	10, 10, 17,	1, 2, 3, 4, 5, 15, 19, 20, 8
2b	10, 17, 10	1, 2, 3, 4, 14, 18, 19, 20, 8
2c	17, 10, 10	1, 2, 3, 4, 14, 16, 17, 19, 20, 8
3a	8, 10, 12	1, 2, 3, 4, 14, 16, 21, 22, 8
3b	10, 12, 8	1, 2, 3, 4, 14, 16, 21, 22, 8
3c	12, 8, 10	1, 2, 3, 4, 14, 16, 21, 22, 8
4a	10, 10, 21	1, 2, 3, 11, 12, 13, 8
4b	10, 21, 10	1, 2, 10, 12, 13, 8
4c	21, 10, 10	1, 9, 12, 13, 8

10、程序规定第一列字符必须是 A 或 B，第二列字符必须是一个数字。在此

情况下进行文件修改。如果第一列字符不正确，输出信息 L，如果第二列字符不是数字，输出信息 M。利用因果图设计测试用例。

解：





原因有：

1 第一列字符是 A

2 第一列字符是 B

3 第二列字符是一数字

结果为：

21 修改文件

22 输出信息 L

23 输出信息 M

中间量：

11 第一列和第二列的结果

		1	2	3	4	5	6	7	8
条件(原因)	①	1	1	1	1	0	0	0	0
	②	1	1	0	0	1	1	0	0
	③	1	0	1	0	1	0	1	0
	11			1	1	1	1	0	0
动作(结果)	22			0	0	0	0	1	0
	21			1	0	1	0	0	0
	23			0	1	0	1	0	1

		1	2	3	4	5	6	7	8
条件(原因)	①	1	1	1	1	0	0	0	0
	②	1	1	0	0	1	1	0	0
	③	1	0	1	0	1	0	1	0
	11			1	1	1	1	0	0
动作(结果)	22			0	0	0	0	1	0
	21			1	0	1	0	0	0
	23			0	1	0	1	0	1
测试用例				A3 A8	AM A?	B5 B4	BN B?	C2 X6	DY P;

第十、十一章 培训与维护

一、填空题

1、为了识别和纠正错误，修改软件性能上的缺陷所需进行的确定和修改错误的维护活动称为（**改正性维护**）。

2、为了使应用软件适应计算机硬件、软件和数据环境的变化而修改软件的维护活动称为(**适应性维护**)。

3、为增加软件功能、增强软件性能、提高软件运行效率而进行的维护活动称为（**完善性维护**）。

4、为了提高软件的可维护性和可靠性而对软件进行的修改称为(**预防性维护**)。

5、因修改软件而造成的错误或其他不希望出现的情况称为(**维护的副作用**)。

6、软件再工程主要包括（**理解软件**）、（**改进软件**）和 （**获取、保存和扩充软件知识**）等三种。

7、用于软件再工程的重构技术有（**结构化设计技术**）和 （**软件复用技术**）。

8、软件再工程的风险主要包括（**过程风险**）、（**人员风险**）、（**应用问题风险**）、技术风险、工具风险和策略风险。

9、影响维护工作量的程序特征有（**系统大小**）、（**系统年龄**）、（**程序设计语言**）、数据库技术的应用、先进的软件开发技术、应用的类型、数学模型、任务的难度等。

二、单选题

1、在软件生存期中，工作量所占比例最大的阶段是(D)阶段。

A. 需求分析　B. 软件设计　C. 测试　**D. 维护**

2、在整个软件维护阶段，以(C)维护所花费的工作量所占比例最大。

A. 改正性　B. 适应性　**C. 完善性**　D. 预防性

3、一个软件产品开发完成投入使用后，常常由于各种原因需要对它做适当的变更。通常把软件交付使用后所做的变更叫做(A)。

A. 维护　B. 设计　C. 软件再工程 D. 逆向工程

4、软件工程针对维护工作的主要目标是提高软件(C)，降低维护的成本。

A. 生产率　B. 可靠性　**C. 可维护性**　D. 维护效率

5、软件可维护性是指软件能够被理解、改正、(D)功能的容易程度。

A. 变更　B. 维护　C. 修改　**D. 适应及增强**

6、软件可维护性是软件开发阶段的关键目标。软件可维护性可用下面七个质量特性来衡量，即可理解性、可测试性、可修改性、可靠性、(C)、可使用性和效率。

A. 完备性　B. 安全性　**C. 可移植性**　D. 灵活性

7、可维护性的特性中相互促进的是(A)。

A. 可理解性与可测试性　B. 效率和可移植性

C. 效率和可修改性　D. 效率和可靠性

8、可维护性的特性中相互矛盾的是(C)。

A. 可修改性和可理解性　B. 可测试性和可理解性

C. 效率和可修改性　D. 可理解性和可读性

9、在软件维护的实施过程中，为了正确、有效地修改程序，需要经历以下三个步骤：分析和理解程序、修改程序和(B　)。

A. 建立目标程序　**B. 重新验证程序**　C. 验收程序　D. 测试程序

10、在软件维护的实施过程中，为了正确、有效地修改程序，需要经历以下三个步骤：分析和理解程序、修改程序和重新验证程序。其中(C)是决定维护成败和质量好坏的关键。

A. 分析和理解程序 B. 重新验证程序 **C. 修改程序** D. 验收程序

11、在软件维护的实施过程中，为了正确、有效地修改程序，需要经历以下三个步骤：分析和理解程序、修改程序和重新验证程序。重新验证程序包括(B)确认、计算机确认和维护后的验收。

A. 动态　**B. 静态**　C. 人工　D. 自动

12、在下面的叙述中与可维护性关系最密切的是(C)。

A. 软件从一个计算机系统和环境转移到另一个计算机系统和环境的容易程度。

B. 尽管有不合法的输入，软件仍能继续正常工作的能力。

C. 软件能够被理解、改正、适应和增强功能的容易程度。

D. 在规定的条件下和规定的时间内，实现指定的功能的能力。

13、在软件维护工作的过程中，第一步是先确认(B)。

A. 维护环境　**B. 维护类型**　C. 维护要求　D. 维护者

14、不管维护类型如何，大体上要开展相同的技术工作。这些工作包括修改软件设计、(D)、单元测试、集成测试、确认测试以及验收。

A. 分析　B. 测试　C. 检验　**D. 修改代码**

15、软件生存期的(D)的工作与软件可维护性有密切的关系。

A. 编码阶段　B. 设计阶段　C. 测试阶段　**D. 每个阶段**

16、软件维护困难的主要原因是(C)。

A. 费用低　B. 人员少　**C. 开发方法缺陷**　D. 维护难

17、软件维护费用高的主要原因是(B)。

A. 生产率高　**B. 生产率低**　C. 人员多　D. 人员少

18、维护阶段的文档是(C)。

A. 软件需求说明书　B. 操作手册　**C. 软件问题报告**　D. 测试分析报告

19、产生软件维护的副作用，是指(C)。

A. 开发时的错误　B. 隐含的错误

C. 因修改软件造成的错误　D. 运行时误操作

20、在维护中，因误删除一个标识符而引起的错误是(C)副作用。

A. 文档　B. 数据　**C. 编码**　D. 设计

三、选择填空题

1、从供选择的答案中选出与下面有关软件维护的叙述最适合的答案，将其编号填入相应的括号内。

一个软件产品开发完成投入使用后，常常由于各种原因需要对它做适当的变更。在软件的使用过程中，软件原来的(A ④)可能不再适应用户的要求，需要

进行变更；软件的工作环境也可能发生变化，最常见的是配合软件工作的(B ③)有变动；还有一种情况是在软件使用过程中发现错误，需要进行修正。通常把软件交付使用后做的变更称为(C ④)。软件投入使用后的另一项工作是(D ③)，针对这类软件实施的软件工程活动，主要是对其重新实现，使其具有更好的(E ②)，包括软件重构、重写文档等。(D ③)和新的软件开发工作的主要差别在于(H ③)。我们把常规的软件开发称为(F ②)，而(G ①)是从代码开始推导出设计或是规格说明来。

供选择的答案：

A, B. ①环境 ②软件 ③硬件 ④功能和性能 ⑤ 要求

C, D, F, G. ①逆向工程 ②正向工程 ③软件再工程 ④维护 ⑤设计

E. ①可靠性 ②可维护性 ③可移植性 ④可修改性

H. ①使用的工具不同 ②开发的过程不同 ③开发的起点不同 ④要求不同

2、从供选择的答案中选出与下面有关软件维护的叙述最适合的答案，将其编号填入相应的括号内。

软件维护是软件生存期的最后一个阶段。软件工程学针对维护工作的主要目标是提高(A ④)，降低(B ⑤)。软件的(C ①)、(D ③)、(E ④)是决定软件可维护性的基本因素。软件生存期(F ④)的工作与软件可维护性有密切的关系。

供选择的答案：

A, B. ①软件的生产率 ②文档 ③软件的可靠性 ④软件的可维护性 ⑤维护的代价 ⑥维护的效率

C, D, E. ①可测试性 ②互操作性 ③可理解性 ④可修改性 ⑤可复用性 ⑥ 可管理性

F. ①编码阶段 ②设计阶段 ③测试阶段 ④每个阶段

3、从供选择的答案中选出同下列各叙述关系最密切的字句。

A. 软件从一个计算机系统或环境转移到另一个计算系统或环境的容易程度。

B. 软件在需要它投入使用时能实现其指定的功能的概率。

C. 软件使不同的系统约束条件和用户需求得到满足的容易程度。

D. 在规定的条件下和规定的一段期间内，实现所指定的功能的概率。

E. 尽管有不合法的输入，软件仍能继续正常工作的能力。

供选择的答案：

①可测试性 ②可理解性 ③可靠性 ④可移植性 ⑤可使用性

⑥兼容性 ⑦容错性 ⑧可修改性 ⑨可接近性 ⑩一致性

A. ④, B. ⑤ C. ⑥, D. ③, E. ⑦

4、从供选择的答案中选出与下面有关软件维护实施的叙述最适合的答案，将其编号填入相应的括号内。

在软件维护的实施过程中，为了正确、有效地修改，需要经历以下 3 个步骤：(A ③)、(B ①)、(C ④)。(A ③)是决定维护成败和质量好坏的关键。(C ④)包括(D ②)确认、计算机确认和维护后的(E ②)。

供选择的答案：

A～C. ①修改程序 ②建立目标程序 ③分析和理解程序 ④重新验证程序 ⑤ 验收程序

D. ①动态　②静态　③人工　④自动

E. ①验证　②验收　③检验　④存档

5、从供选择的答案中选出与下面有关软件可移植性的叙述最适合的答案，将其编号填入相应的括号内。

软件可移植性是用来衡量软件的(A ③)的重要尺度之一。为了提高软件的可移植性，应注意提高软件的(B ④)。为了提高可移植性，还应(C ①)

供选择的答案：

A. ①通用性 ②效率 ③质量 ④人机界面

B. ①使用的方便性 ②简洁性 ③可靠性 ④设备独立性

C. ①有完备的文件资料 ②选择好的宿主计算机 ③减少输入输出次数 ④选择好的操作系统

6、从下列叙述中选出 4 条与提高软件的可移植性有关的叙述。

① 把程序中与计算机硬件特性有关的部分集成在一起。

② 选择时间效率和空间效率高的算法。

③ 使用结构化的程序设计方法。

④ **尽量用高级语言编写程序中对效率要求不高的部分。**

⑤ 尽可能减少注释。

⑥ **文档资料详尽、正确。**

⑦ 在有虚拟存储器的计算机系统上开发软件。

⑧ 减少程序中对文件的读写次数。

⑨ 充分利用宿主计算机的硬件特性。

正确的叙述有 ①、③、④、⑥。

7、从供选择的答案中选出与下面有关软件再工程的叙述最适合的答案，将其编号填入相应的括号内。

软件再工程是一类软件工程活动，它能够使我们：ⅰ）增进对软件的理解；ⅱ）准备或直接提高软件自身的(A ③)、(B ④)或演化性。第ⅱ部分旨在改善软件的(C ①)，使得软件更容易为人们服务。纯粹是出于改善性能的代码优化(D ②)软件再工程。逆向工程属于上述软件再工程的第(E ②)部分。

供选择的答案：

A, B. ①可靠性 ②灵活性 **③可维护性 ④可复用性** ⑤可修改性

C. **①静态质量** ②动态质量 ③性能 ④功能

D. ①属于 **②不属于**

E. ①ⅱ **②ⅰ**

8、从供选择的答案中选出与下面有关软件再工程的叙述最适合的答案，将其编号填入相应的括号内。

关于软件再工程的定义有这样两种说法。ⅰ）软件再工程是变更系统(或程序)的(A ③)，或是系统(或程序)的(B ⑥)，而不变更其(C ⑦)的一种工程活动。ⅱ）检查并改进对象系统，按新的模式对系统进行(D ⑤)，进而实现其新的模式。

供选择的答案：

A ～D. ①外部环境 ②接口 ③内部机制 ④流程图 ⑤重构 ⑥ 数据结构 ⑦功能性 ⑧层次性

四、问答题

1、为什么软件需要维护？维护有哪几种类型？

在软件开发完成交付用户使用后，为了保证软件在一个相当长的时期能够正常运行，就需要对软件进行维护。软件维护的类型有 4 种：改正性维护、适应性维护、完善性维护和预防性维护。其中，改正性维护是要改正在特定的使用条件下暴露出来的一些潜在程序错误或设计缺陷；适应性维护是要在软件使用过程中数据环境发生变化或处理环境发生变化时修改软件以适应这种变化；完善性维护是在用户和数据处理人员使用软件过程中提出改进现有功能，增加新的功能，以及改善总体性能的要求后，修改软件以把这些要求纳入到软件之中。预防性维护是为了提高软件的可维护性、可靠性等，事先采用先进的软件工程方法对需要维护的软件或软件中的某一部分（重新）进行设计、编制和测试，为以后进一步改进软件打下良好基础。

2、改正性维护与“排错”是否是一回事？为什么？

改正性维护与“排错(调试)”不是一个概念。调试是作为测试的后继工作而出现的，是当测试发现软件中的错误后，进一步诊断和改正程序中潜在的错误的活动。而改正性维护是指在软件交付使用后，由于开发时测试的不彻底、不完全，必然会有一部分隐藏的错误被带到运行阶段来，这些隐藏下来的错误在某些特定的使用环境下就会暴露出来。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所进行的诊断和改正错误的过程。调试在程序编码阶段、测试阶段、运行和维护阶段都可以发挥作用，它实际上是一种工具或手段。在软件交付运行之后，用户实际充当了测试员的角色，一旦发现软件运行中的错误或缺陷，就会将问题报告通报软件销售商，申请软件维护。其后软件维护人员可以利用调试手段来诊断和改正软件中存在的错误。这时可能涉及的范围不只包括程序，还有文档和数据，不仅可能修改程序代码，而且可能需要修改设计。甚至需求。所以改正性维护是在更大范围中做工作。

3、简述软件演化与软件衰退的分界点是什么。

维护的成本太高

系统的可靠性不可以接受

在一个合理的时间内，系统不能再适应进一步的变化了

系统性能仍旧超出预先规定的约束条件

系统功能的作用有限

其他的系统能更好、更快、更廉价地做同样的工作

维护硬件的成本高得足以用更便宜、更新的硬件来取代

4、简述软件演化规则的内容。

连续的变化

递增的复杂性

程序演化的基本法则

组织稳定性的守恒

熟悉程度的守恒

5、简述软件维护的分类，从分类指出各类维护的优先级。

改正性维护：必须做
适应性维护：做，但可以不马上做
完善性维护：可根据自身情况决定做否
预防性维护：可以不做