

绪论

1.什么是软件危机？为什么会产生软件危机？

答：软件危机是指在计算机软件的开发和维护过程中遇到的一系列严重问题。

(1).软件维护费用急剧上升，直接威胁计算机应用的夸大。

(2).软件生产技术进步缓慢

2.什么是软件生产工程化？工程化生产方法与早期的程序设计方法主要差别在哪里？

答：结构化程序设计地出现，使许多产业界认识认识到必须把软件生产从个人化方式改变为工程化。采用工程的概念、原理、技术和方法开发与维护软件，把经过时间考验而证明正确的管理技术和当前能够得到的最好的技术方法结合起来，以经济地开发出高质量的软件并有效地维护它，这就是软件工程，同时这也是工程化生产方法。

3.分别说明（1）软件开发方法与开发工具；（2）软件技术与软件管理的相互关系。

答：（1）工具和方法，是软件开发技术的两大支柱，它们密切相关。当一种方法提出来并证明有效后，往往随之研制出相应的工具，来帮助实现和推行这种方法。新方法在推行初期，总有人不愿接受和采用。若将新方法融合于工具之中，使人们通过使用工具来了解新方法，就能更快促进新方法的推广。

（2）在工业生产中，即使有先进的技术和设备，管理不善的企业也不能获得良好的效益。软件在生产中不能按质按时完成计划，管理混乱往往是其中的重要原因。所以对于一个理想的软件工程环境，应该同时具备技术和管理两个方面。

4.试从你的亲身实践，谈谈软件工具在软件开发中的作用。

答：用 C++开发一个软件，是校园一卡通的模块。首先，要在编辑程序支持下在计算机中输入源程序。然后编译程序，把源程序翻译成目标程序。如果发现错误，就重新调入编辑程序对源程序进行修改。编译通过后，再调用连接程序吧所有通过了编译目标程序连同与之有关的程序连接起来，构成一个能在计算机上运行的可执行软件。编译程序，编辑程序，连接程序以及支持他们的计算机操作系统，都属于软件工具。离开这些工具，软件开发就是去了支持，变得十分困难和低效，甚至不能运行。

5.什么是软件工程环境？谈谈你对环境重要性的认识。

答：方法与工具相结合，再加上配套的软、硬件支持就形成环境。例如在批处理时代，用户开发的程序是分批送入计算机中心的计算机的，有了错误，就得下机修改。程序员对自己写的程序只能继续地跟踪，思路经常被迫中断，效率难于提高。分时系统的使用，使开发人员从此能在自己的终端上跟踪程序的开发，仅此一点，就明显提高了开发的效率。

6.何谓面向对象软件工程？简述它与传统软件工程在各型软件开发中的作用。

答：以面向对象程序设计为基础。

7.软件按规模大小可分成哪几类？简述软件工程中各型软件开发中的作用。

答：按规模分为极小、小、中、大、甚大、极大。

（1）中小型软件：软件工程对改进软件质量，提高程序员生产率和满足用户的需求，有很大的作用。

（2）大型软件：这类软件必须从头至尾坚持软件工程的方法，严格遵守标准文档格式和正规的复审制度，才能避免或减少混乱，真正开发出大型的软件。

8.什么是形式化软件开发方法？实现这类开发的困难和出路在哪里？

答：它是一种基于数学的开发技术，主要采用数学的方法来描述系统的性质（例如程序变换和程序验证等）。形式化的方法加上自动化的开发环境，可能是解决这一难题的出路。

软件开发模型

1. 什么是软件生存周期？把生存周期划分为阶段的目的是什么？

答：软件生存周期划分为计划、开发和运行 3 个时期；把整个生存周期划分为较小的阶段，给每个阶段赋予确定而有限的任务，就能够化简每一步的工作内容，使因为软件规模而增长而大大增加了软件复杂性变得交易控制和管理。

2. 传统的瀑布模型把生存周期分为哪些阶段？瀑布模型软件开发有哪些特点？

答：瀑布模型在编码以前安排了分析阶段和设计阶段；阶段间具有顺序性和依赖性。

3. 说明文档和复审对于软件质量的控制的作用。

答：每一阶段都要完成规定的文档，没有完成文档，就认为没有完成该阶段的任务。软件开发是许多人共同参加的计划，完整与合格的文档，不仅是开发时期软件人员之间互相通信的媒介，也是运行时期对软件进行维护的中要依据。每一阶段都要对已完成的文档进行复审，以便尽早发现问题，消除隐患。愈是早期潜伏下来的故障，暴露出来的时间愈晚，排除故障需付出的代价也就愈高。及时复审是保证软件质量，降低开发成本的重要措施。

4. 什么是快速原型法？其快速表现在哪里？

答：首先建立一个能够反映用户主要需求的原型，让用户实际看一看未来系统的概貌，以便判断哪些功能是符合需要的，哪些方面还需要改进。然后将原型改进，最终建立完全符合用户要求的新系统。它的快速表现在能够缩短开发周期的语言和工具，能在短时间内提供出成品，但不包括成品中的细节，然后让客户进行对比。

5. 实现快速原型法的最终系统可以有几种方法？请说明并加以比较。

答：原型系统仅包括未来系统的主要功能，以及系统的重要接口。为了尽快向用户提供原型，开发原型系统时应尽量使用能缩短开发周期的语言和工具。最终系统的形成可以采用原型废弃不用，另一方法是补充修改模型获得最终系统。方法一不包括系统的细节。后一种方法在实际工作中，由于原型系统使用的语言往往存在效率不高等原因，除了少数简单的事务系统外，大多数原型都废弃不用，仅把建立原型的过程当作帮助定义软件需求的一种手段。

6. 比较增量模型和螺旋模型的特点，有什么不同和相似的地方？

答：增量模型是瀑布模型的顺序特征与快速原型法的迭代特征相结合的产物。螺旋模型是一种迭代模型，每迭代一次，螺旋线就前进一周。增量模型每个增量具有高内聚低耦合，高度的独立性。而螺旋模型它在结合瀑布模型与快速原型的基础上还增加了风险分析。

7. 为什么利用转换模型开发软件有一定难度？什么是净室软件工程？

答：从理论上说，一个正确的，满足客户需要的形式化规格说明，经过一系列正确的程序变化后，可以确保得到这样一个形式化规格说明，目前还有较高的难度，同时，软件开发者很少具有实用形式化方法所需的背景知识，况且，转换模型开发软件现在还很费时和昂贵！净室基本思想是力求在分析和设计阶段就消除错误，确保正确，然后在无缺陷或“洁净”的状态下实现软件的制作。

8. 哪些开发模型适用于面向对象的软件开发？

答：构件集成模型

9. 比较螺旋模型和构件集成模型的异同。

答：构件集成模型利用预先对封装好的软件构件来构造应用软件系统，它融合了螺旋模型的很多特征，支持软件开发的迭代方法。

软件需求分析

1. 需求分析的任务是什么？怎样理解分析阶段的任务是决定“做什么”，而不是“怎么做”？

答：需求分析主要有两个任务：第一是通过对问题及其环境的理解、分析和综合建立分析模型；第二是在完全弄清用户对软件系统的确切要求的基础上，用“软件需求规格说明书”把用户的需求表达出来。需求分析的任务就是为了明确要开发的是一个什么样的系统，而不是去怎么去实现这个系统。

2. 需求分析要经过哪些步骤？

答：需求获取、需求提炼、需求描述、需求验证。

3. 有哪两种主要的分析模型，它们有什么联系？

答：面向对象分析模型、结构化分析模型。前者是采用面向对象的思想进行软件需求分析的建模过程，而后者模型的核心是 DD，它是设计各种数据对象的总和。他们的模型分别起到了描述数据模型，功能模型与行为模型的作用。

4. 什么是结构化分析？它的“结构化”体现在哪里？

答：是使用 DFD、DD、结构化语言、判定表和判定树等工具，来建立一种新的、称为结构化说明书的目标文档。

5. 什么是面向对象分析？其主要思想是什么？

答：OOA 面向对象的分析是采用面向对象的思想进行软件需求分析建模的过程。主要思想是采用面向对象的思想。

6. 需求说明（或需求规格说明书）由哪些部分组成？各部分的主要内容是什么？

答：引言、信息描述、功能描述、行为描述、质量描述、接口描述、其他描述。

引言：主要叙述在文体定义阶段确定的关于软件的目标与范围，简要介绍系统背景、盖帽、软件项目约束和按考资料等。

信息描述：给出对软件所含信息的详细描述，包括信息的内容、关系、数据流向、控制流向和结构等。

功能描述：对软件功能要求的说明，包括系统功能划分、每个功能的处理说明、限制和控制描述等。

行为描述：包括对系统状态变化及事件和动作的描述，据此可以检查外部事件和软件内部的控制特征。

质量描述：阐明在软件交付使用前需要进行的共更能测试和性能测试，并且规定源程序和文档应该遵守的各种标准。

接口描述：包括系统的用户界面、硬件接口、软件接口和通信接口等的说明。

其他描述：阐述系统设计和实现上的限制，系统的假设和依赖等其他需要说明的内容。

7. 为什么 DFD 要分层？画分层 DFD 要遵循哪些原则？

答：大型复杂的软件系统，其 DFD 可能含有数百乃至数千个加工，不能设想一次就将它们全部画齐。

正确的做法是：从系统的基本模型（把整个系统看成一个加工）开始，逐层地对系统进行分解。原则：由顶向下，逐步细化。

8. DFD 和 CFD 有什么区别？

答：

9. 什么是系统的行为模型，如何建立？

答：类对象模型所表示的是面向对象分析模型中的静态部分，而对象行为模型则用于描述系统的动态行为，即系统如何对应外部事件---系统的行为模型。

建立一个对象行为模型一般要经历一些过程：评估所以的用例来理解系统中的交互序列；找出驱动交互序列的事件；为每个用例创建事件轨迹；为对象创建状态转换图。

10. 选一个系统（例如工资处理系统、飞机订票系统、图书馆管理系统等），分别用 SA 方法和 OOA 方法对它进行分析，并给出分析模型。

答：

软件设计概述

1.传统软件工程把设计过程分成2步：概要设计与详细设计。试述这2个阶段个字主要完成的任务。

答：概要设计，包括结构设计和接口设计，并编写设计文档。详细设计，其任务是确定各个软件组件的数据结构和操作，产生描述个软件组件的详细设计文档。

为什么大型软件设计必须分成两步走？

答：概要设计和详细设计是软件设计的两步，概要设计确定模块的划分，模块之间的调用关系，接口等；详细设计细分模块、数据结构等；大型系统中分两步走是必要的，概要设计确定模块划分后，详细设计可把各模块交给不同的人做详细设计，大型系统中这样的分工是比较合理的，能提高效率，做到合理分工。

2.解释下列名词：（1）模块；（2）模块化；（3）模块化设计。

答：**模块**是一个拥有明确定义的、输出和特性的程序实体。

模块化是指解决一个复杂问题时自顶向下逐层把软件系统划分成若干模块的过程。每个模块完成一个特定的子功能，所有的模块按某种方法组装起来，成为一个整体，完成整个系统所要求的功能。

模块化设计是把大型软件按照规定的原则划分成一个个较小的、相对独立但又相互关联的模块。但又相互关联的模块。

3.什么是模块独立性？用什么度量？

答：模块独立性指每个模块只完成系统要求的独立的子功能,并且与其他模块的联系最少且接口简单。

模块独立的概念是模块化、抽象、信息隐蔽和局部化概念的直接结果。

独立性可以从两个方面来度量：模块本身的内聚性(Cohesion)和模块之间的耦合(Coupling)。

4.具体说明“一个模块，一个功能”的含义，并试讨论这类模块的优点。

答：

5.什么是自顶向下设计？为什么说它尤其适用于大型软件的开发？

答：设计时首先对所设计的系统有一个全面的理解。然后从顶层开始，连续的逐层向下分解，直至系统的所有模块都笑道便于掌握为止。

自底向上设计反映了软件规模较小的设计思想，随着软件规模的增长，这种方法的缺点逐渐暴露出来。而自顶向下的设计需要进行详细的可行性论证，易于修改和扩展，整体测试较易通过。

6.自顶向下逐步细化的方法也适用于编写教材。使用这种方法说明编写一本教材的过程。

答：

7.输入三角形的3条边长（假定这些边确实能组成一个三角形），用它们来鉴别三角形的性质（等腰、等边、任意或者直角），并输出结果。试用逐步细化的方法设计这一程序，并用结构化语言（汉语或英语）写出细化过程中每一步的过程描述。

答：

8.比较概要设计复审和过程设计复审的评审内容，并说明他们采取的复审方式有何异同。

答：**概要设计复审**的重点放在系统的总体结构、模块划分、内外结构等方面。例如软件结构是否满足需求？结构形态是否合理？层次是否清晰？模块的划分是不是符合优化原则？系统的人机界面、内外部接口、以及出错处理是不是合理等。

过程设计复审的重点放在模块的具体设计上。例如模块设计能否满足其功能与性能要求？选择算法与数据结构是否合理，是否符合编程语言的特点？设计描述简单、清晰等。

概要设计复审常用的复审方式是正式复审。

传统的设计方法

1.简释事务、事务型结构和变换型结构。

答：一次动作、时间或状态变化也可以成为一次事务。

事务型结构由至少一条接受路径、一个事务中心与若干条动作路径组成。

变换型结构由至少一条传入路径、变换中心和传出路径组成。

2.简述从 DFD 图到 SC 图的映射规则。

答：（1）复审 DFD 图，必要时可再次进行修改或细化；

（2）鉴别 DFD 图所表示的软件系统的结构特征，确定它所代表的软件结构是属于变换型还是事务型。

（3）按照 SD 方法规定一组规则，把 DFD 图转换为初始的 SC 图；



（4）按照优化设计的指导原则改进初始的 SC 图，获得最终 SC 图。

5.某事务系统具有下列功能：

（1）读入用户命令，并检查其有效性；

（2）按照命令的编号（1-4 号）进行分类处理

（3）1 号命令计算产品共识，能根据用户给出的各种产品数量，计算出各工种的需要工时和缺额工时；

（4）2 号命令计算材料消耗，根据产品的材料定额和用户给出的生产数量，计算各种材料的需求量

（5）3 号命令编制材料订货计划

（6）4 号命令计算产品成品

试用结构化分析和设计方法画出该系统的 DFD 图并据此到处系统的 SC 图。对动作分支中的 1 号和 2 号命令要详细描述和设计，3 号命令和 4 号命令允许从略，可用示意图表示。

答：

7.简述过程（详细）设计说明书的主要内容，怎样对它进行复审？

答：(1)为每个模块确定采用的算法。选择某种适当的工具表达算法的过程，写出模块的详细过程描述；

(2)确定每一模块使用的数据结构；(3)确定模块接口的细节，包括对系统外部的接口和用户界面，对系统内部其它模块的接口，以及关于模块输入数据、输出数据及局部数据的全部细节。

8.简化比较本章讲解的几种过程设计表达工具的优缺点。

答：流程图和 N-S 图,伪代码和 PDL 语言。

流程图具有能随意表达任何程序逻辑的有点，随着结构化程序设计方法的普及，流程图在描述程序逻辑时的随意性与灵活性恰恰变成了它的缺点。

N-S 图所有的程序结构均用方框来表示，无论并列或者嵌套，程序的结构清晰可见。容易养成良好的程序设计风格。缺点是当程序内嵌的层数增多时，内层的方块越画越小，不仅增加画图的困难，也使图形的清晰性受到影响。

伪代码工作量比画图小，又容易转换为真正的代码，PDL 具有很强的描述功能，是一种十分灵活和有用的过程表达工具。

9.任选一种排序（从小到大）算法，分别用流程图、N-S 图和 PDL 语言描述其详细过程。

10.试将下列用 PDL 伪代码表示的某种模块的过程性描述改用 N-S 图表示。

.....

```

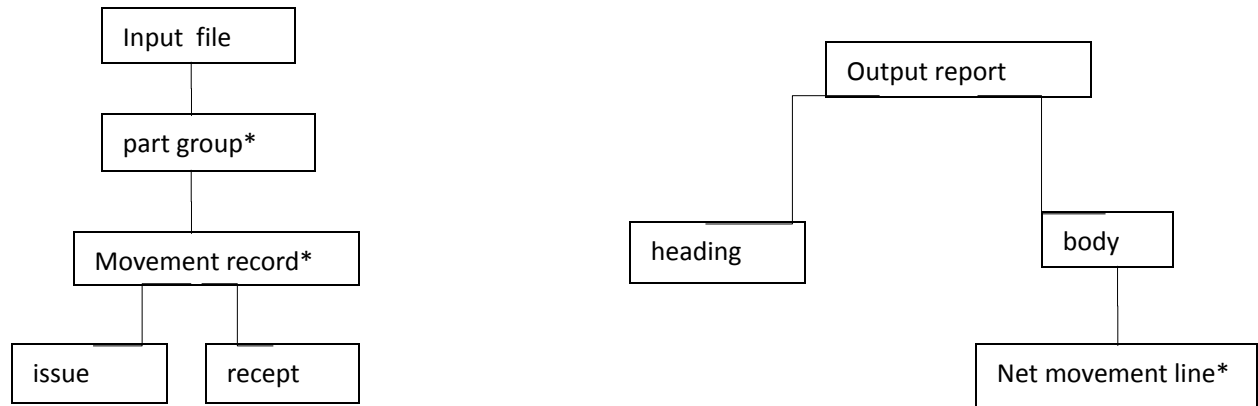
Execute process a
REPEAT UNTIL condition X8
Execute process b
IF condition X1
    THEN BEGIN
        Execute process f
IF condition X6
    THEN
        REPEAT UNTIL condition X7
        Execute process i
    ENDREP
ELSE BEGIN
    Execute process g
    Execute process h
END
ENDIF
ELSE CASE OF Xi
WHEN contdition X2 SELECT
    DO WHILE contition X5
execute process C
ENDDO
WHEN condition X3 SELECT process d
WHEN condition X4 SELECT process e
ENDCASE
ENDIF
ENDREP
Execute process j
END

```

execute process a								
	execute process b							
	X1							
	T						F	
	execute process f			X3		X4		
				T	F	T	F	
	X6			execute process d		execute process e		
	T							
	execute process i		execute process g					
	REPEAT UNTIL X7		execute process h					
	execute process j							
REPEAT UNTIL condition X8								

12.图 5-63 显示了某仓库零件收发管理程序的数据结构，用 Jackson 图表示。图中 Part 表示零件，Issue 和 Recept 分别表示零件的出库量和入库量。现要求：

- (1) 找出输入结构与输出结构之间的对应单元
- (2) 画出用 Jackson 图表示的程序结构
- (3) 列出程序所需要的操作，并加到第 (2) 步画出的程序结构图上
- (4) 用伪代码写出收发管理程序的过程性表示



13.某工资管理程序的输入/输出数据中包括：

输入数据：职工姓名，日工资率，工作天数，加班天数，病事假天数，补贴变更，扣款变更等；

输出数据：职工姓名，基本工资，各项补贴，各项扣款，事发工资等；

试用 Jackson 方法设计这一程序。

软件测试

1.软件测试的基本任务是什么？测试与纠错有什么不同？

测试的目的是发现程序错误；测试的任务是通过在计算机上执行程序，暴露程序中潜在的错误。

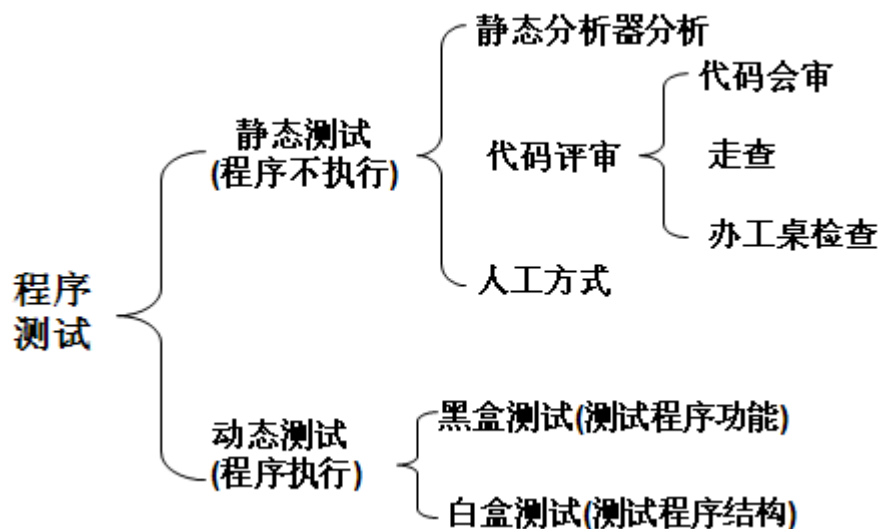
纠错的目的是定位和纠正错误；纠错的任务是软件故障，保证程序的可靠运行。

通常把一次程序执行需要的测试数据成为一个测试用例，每个测试用例产生一个相应的测试结果，如果它与期望结果不符，便说明程序中存在错误，需要用纠错来改正。

2.怎样理解下面的话所蕴含的意义：“程序测试只能证明错误的存在，但不能证明错误不存在”、“测试是为了证明程序有错，而不是证明程序无错”

揭示了测试固有的一个重要性质——不彻底性。彻底测试就是让被测程序在一切可能的输入情况下全部执行一遍，又称穷举测试。在实际情况中是根本无法实现的。这就注定了一切实际测试都是不彻底的，当然也就不能保证测试后的程序不存在遗留的错误。

◆测试的分类



◆测试方法测试用例和测试结果

测试用例={测试数据+期望结果}

测试结果={测试数据+期望结果+实际结果}

黑盒测试

1 等价测试

把输入数据的可能值划分为若干个等价类，使每类中的任何一个测试用例，都能代表同一等价类中的其它测试用例。

采用等价测试注意以下两点：

划分等价类不仅要考虑代表“有效”输入值的有效等价类，还要考虑代表“无效”输入值得无效等价类；每一无效等价类至少要用一个测试用例，不然可能漏掉某一类错误，但允许若干个有效等价类合用一个测试用例，以便进一步减少测试的次数。

【例】某工厂公开招工，规定报名者年龄在 16 周岁至 35 周岁(在 1967 年 2 月到 1986 年 3 月)。如果出生年月不在上述范围内，将拒绝接受，并显示“年龄不合格”等出错信息。试用等价分类法设计这一程序功能的测试用例。

第一步：划分等价类

输入数据	有效等价类	无效等价类
出生年月	(1)6 位数字字符	(2)有非数字字符 (3)少于 6 个数字字符 (4)多于 6 个数字字符
对应数值	(5)在 196702-198603 之间	(6)<196702 (7)>198603
月份对应数值	(8)在 1-12 之间	(9)等于“0” (10)>12

第二步：设计有效等价类需要的测试用例

测试数据	期望结果	测试范围
------	------	------

197011	输入有效	(1)、(5)、(8)
第三步：为每一无效等价类设计一个测试用例		
测试数据	期望结果	测试范围
MAY,70	输入无效	(2)
19705	输入无效	(3)
1968011	输入无效	(4)
195512	年龄不合格	(6)
196006	年龄不合格	(7)
196200	输入无效	(9)
197222	输入无效	(10)

2 边界测试

程序员在处理边界情况时，很容易因忽略或考虑不周发生编码错误。例如，数组容量、循环次数以及输入数据与输出数据在边界值附近程序出错概率往往较大。

采用边界值分析法就是要这样来选择测试用例，使得被测试程序能在边界值及其附近运行，从而更有效地暴露程序中潜在的错误。

例如程序可能设有语句

```

If(196702<=value(birthdate)<=198603)
  then read(birthdate)
  else write "invalid age"

```

将上式<=写成<，以上所有测试都不能发现该错误。

【例】上题年月日的测试用例(边界分析法)

输入等价类	测试用例说明	测试数据	期望结果	选取理由
出生年月	1个数字字符	5	输入无效	仅有一个合法字符
	5个数字字符	197505		比有效长度恰少一个字符
	7个数字字符	1986011		比有效字符恰多一个字符
	有1个非数字字符	19705A		非法字符最少
	全是非数字字符	AUGUST		非法字符最多
	6个数字字符	196702	输入有效	类型与长度均有效
对应数值	35周岁	196702	合格年龄	最大符合年龄
	16周岁	198603		最小符合年龄
	>35周岁	196701	不合格年龄	恰大于合格年龄
	<16周岁	198604		恰小于合格年龄
月份对应数值	月份为1	196801	输入有效	最小月份
	月份为12	198512		最大月份
	月份<1	196800	输入无效	恰小于最小月份
	月份>12	197413		恰大于最大月份

3 错误猜测法

猜错就是猜测被测程序放在哪些地方容易出错，然后针对可能的薄弱环节来设计测试用例。一般先用等价分类法和边界值分析法设计测试用例，然后用猜错法补充一些例子作为辅助的手段。

白盒测试

1 逻辑覆盖测试：用流程图来设计测试用例。主要考察的重点是图中的判定框(选择或循环)。按照被测试程序所作测试的有效程度，逻辑测试可由弱到强区分 5 种覆盖标准：

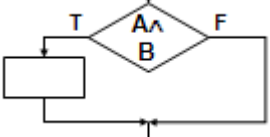
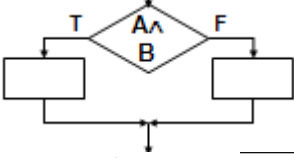
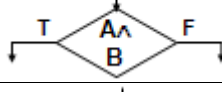
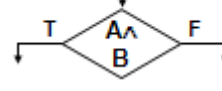
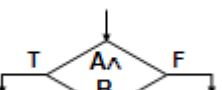
语句覆盖：每条语句至少执行一次。

判定覆盖：每一判定的每个分支至少执行一次。

条件覆盖：每一判定中的每个条件，分别按“真”、“假”至少各执行一次。

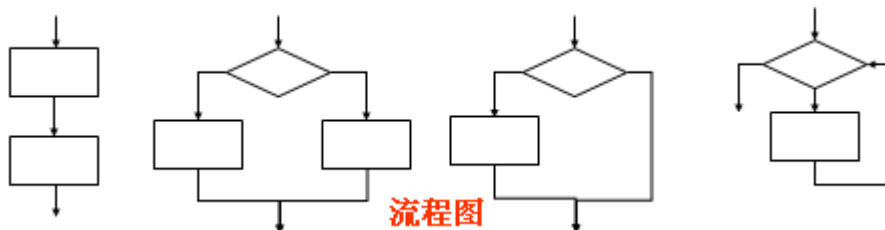
判定/条件覆盖：同时满足判定覆盖和条件覆盖的要求。

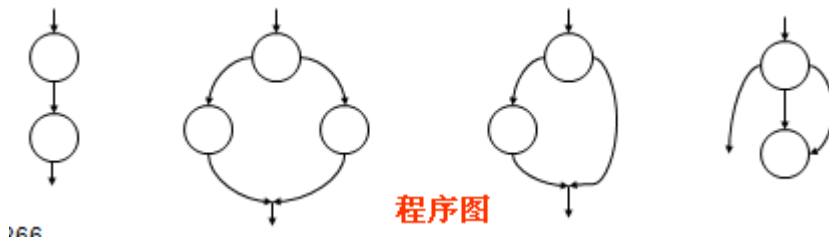
条件组合覆盖：求出判定中所有条件的各种可能组合值，每一可能的条件子集至少执行一次。

覆盖标准	程序结构举例	测试用例应满足的条件
语句覆盖		$A \wedge B = .T.$
判定覆盖		$A \wedge B = .T., A \wedge B = .F.$
条件覆盖		$A = .T., A = .F.$ $B = .T., B = .F.$
判定/条件覆盖		$A \wedge B = .T., A \wedge B = .F.$ $A = .T., A = .F.$ $B = .T., B = .F.$
条件组合覆盖		$A = .T. \wedge B = .T.$ $A = .T. \wedge B = .F.$ $A = .F. \wedge B = .T.$ $A = .F. \wedge B = .F.$

②路径测试法

程序图：是一种简化的流程图。





对程序图中每一条可能的程序执行路径至少测试一次。如果程序中含有循环(在程序中表现为环),则每个循环至少执行一次。

路径测试具有如下特征: 满足结构测试的最低要求。语句覆盖加判定覆盖是对白盒测试的最低要求,同时满足这两种标准的覆盖为“完全覆盖”。从对路径测试的要求可见,它本身就包含了语句覆盖和判定覆盖(在程序图上分别为点覆盖与边覆盖)。

软件复用

1.什么是软件复用?

答: 在构造新的软件系统的过程中,对已存在的软件人工制品的使用技术。

2.什么是领域工程?简述其活动内容。

答: 领域工程是指通过领域分析找出最优复用,把它们设计和构造为可复用构件,进而建立大规模的软件构件仓库的过程。

答: 按复用活动所应用的领域范围,复用可划分为横向复用和纵向复用。横向复用是指复用不同应用领域中的软件元素如数据结构、分类算法等。纵向复用是指在一类具有较多公共性的应用领域之间进行软部分复用。纵向复用包括以下几个方面的活动:

- (1)实施领域分析: 根据应用领域的特征及相似性,可预测软构件的可复用性,发现并描述可复用实体,进而建立相关的模型和需求规约。
- (2)开发可复用构件: 一旦确认了构件的复用价值,即可进行构件的开发,并对具有复用价值的软构件进行抽象、一般化和参数化,以便它们能够适应新的类似的应用领域。
- (3)建立可复用构件库: 将软件构件及其文档进行分类归并,形成相关的分类检索机制,成为可供后继项目使用的可复用资源。

3.什么是 CBSD? 实施软件构件技术要解决哪些问题?

答: 基于构件的软件开发(Component-Based Software Development,简称 CBSD)是在一定构件模型的支持下,复用构件库中的一个或多个软件构件,通过组合构件来构造应用软件系统的开发过程。

要解决的问题:

构件应具有的特征: 通用性、可变性、易组装性。

在建造构件时,必须考略应用领域的特征。领域构件设计框架: 标准数据、标准接口协议、程序模板。

软件维护

1.为什么说软件维护是不可避免的?

答: 满足用户对已开发产品的性能与运行环境不断提高的需要,进而达到延长软件的寿命。

2.纠错和纠错性维护有哪些异同？

答：纠错的目的是定位和纠正错误；纠错的任务是软件故障，保证程序的可靠运行。

纠错性维护是由于软件测试的不彻底性，任何大型软件交付使用后，都会继续发现潜在的错误，对它们进行诊断和改正。目的在于纠正正在开发期间未能发现的遗留错误。

5.什么是软件配置？说明搞好维护时期配置的意义与方法。

答：软件配置是一个软件在生存周期内，它的各种形式、各种版本的文档与程序的总称。

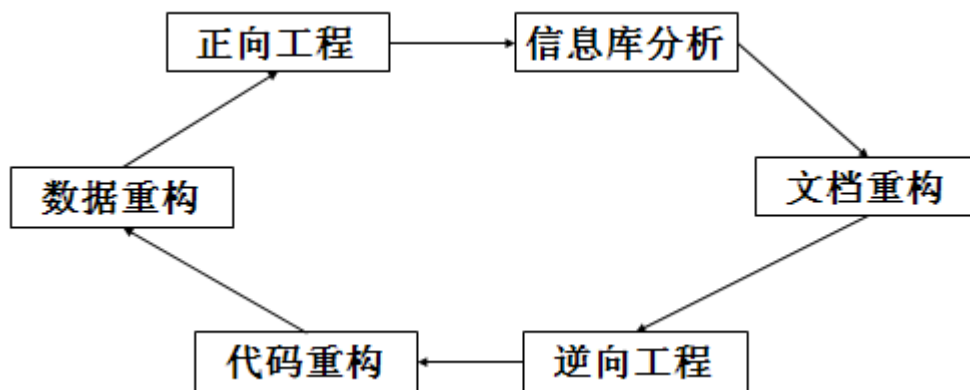
对软件配置进行科学的管理，是保证软件质量的重要手段。配置管理贯穿于整个生存周期，在运行和维护时期，其任务尤为繁重。

为了方便对多种产品和多种版本进行跟踪和控制，常常借助于自动的配置管理工具：配置管理数据库工具和版本控制库工具。

8.什么是软件再工程？软件再工程的主要活动有哪些？

答：软件再工程是将新技术和新工具应用于老的软件的一种较“彻底”的预防性维护。

主要活动有 6 类：



软件再工程与软件维护差异？

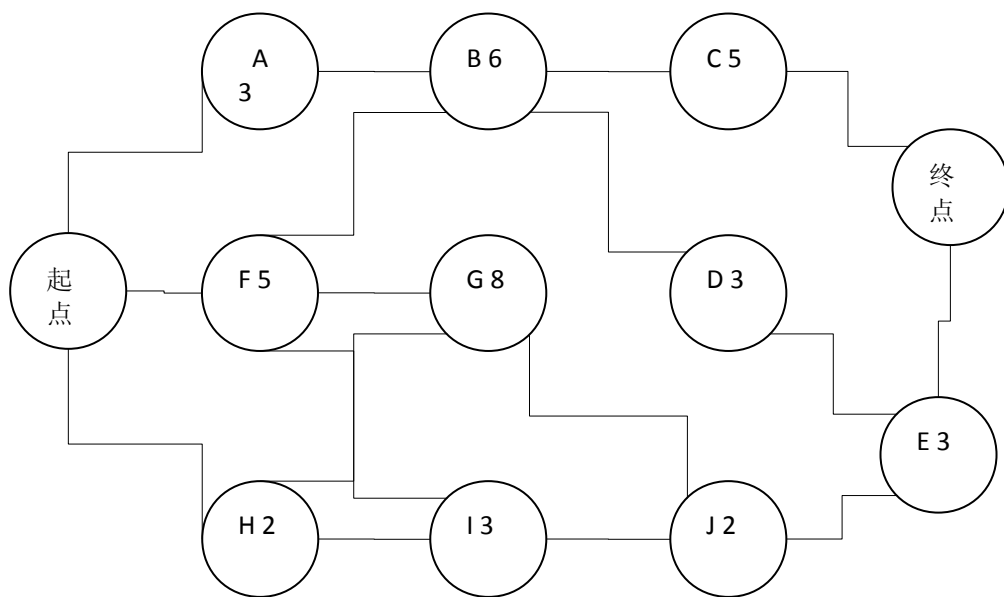
答：软件维护是局部的，以完成纠错或适应需求变化为目的；软件再工程是运用逆向工程、重构等技术，在充分理解原有软件的基础上，进行分解、综合、并重新构建软件，用以提高软件的可理解性、可维护性或演化性。

软件工程管理

2.一个 4 万行规模的应用程序，花 50 万美元可以在市场上买到。如果自己开发，则没人一月的总花费需 4000 美元。试问是购买合算呢？还是自己开发合算呢？（注：开发成本用 COCOMO 模型计算）

答：

6.下图是某软件项目的 PERT 图。



图框中的数字代表活动所需的周数。要求：

(1)找出关键路径和完成项目的最短时间；

(2)标出每项活动的最早起止时间与最迟起止时间。

答：

7.在上题中，若活动G的持续时间分别缩短为7周、6周或5周，试问，完成项目的最短时间有何变化。

答：

8.将第2题的内容改用Gantt图表示。

答：

软件质量管理

3.解释下列各队名词，并说明他们的相互关系和差别：

(1) 验证与确认

答：

(2) 软件质量保证与软件质量认证

答：软件质量保证着眼于每一个软件，保证提供给用户的产品都达到规定的质量水平。

软件质量认证注重软件企业的整体资质，目的在于全面考察企业的质量体系，判断它是否具备设计、开发和生产符合质量要求的软件产品的能力。

4.什么是软件可靠性？怎样对他进行定量表示？

答：在给定时间内，程序按照规定的条件成功地运行的概率。

$R(t)=P\{\text{在时间}[0,t]\text{内按规定条件运行成功}\}$

计算方法 1: $R(t)=e^{-\lambda t}$ ，其中， t 为程序运行时间， λ 为故障率，即单位时间内程序运行失败的次数。

计算方法 2: 平均故障时间 $MTTF=1/\lambda$ ，其中， λ 为故障率，即单位时间内程序运行失败的次数。

5.可靠性模型有几大类？他们的主要区别是什么？

答：可靠性模型分为宏观模型和微观模型 2 大类。

后者是建立在对程序语句和控制结构详细分析的基础之上的，在开发时期很难建立；前者则忽略程序方面的细节，主要从程序中残留错误的角度来建立模型，并且用统计方法确定模型中的常数。

8.什么是 CMM 软件过程能力成熟度模型？他有哪些应用？

答：“能力成熟度模型”是对于软件组织在定义、实施、度量、控制和改善其软件过程的实践中各个发展阶段的描述。是用来确定一个软件过程的成熟程度以及指明如何提高过程成熟度的参考模型。

CMM 的主要应用在能力评估和过程改善 2 个方面。