

**软件定义**：能够完成预定功能和性能的可执行的指令(计算机程序);使得程序能够适当地操作信息的数据结构；  
描述程序的操作和使用的文档.

**软件测试定义:**IEEE/ANSI标准:在既定的状况条件下，运行一个系统或组建，观察记录结果，并对其某些方面进行评价的过程。《软件测试技巧》:软件测试是为了发现错误而运行程序的过程。

广义软件测试的定义：由确认、验证、测试3方面组成。

**软件开发过程**:需求分析(可行性报告，项目初步开发计划，需求规格说明，用户手册概要，测试计划);设计(**概要**：建立系统总体结构，划分功能模块；定义各功能模块接口；数据库设计；指定组装测试计划.**详细**:设计各模块具体实现算法;确定模块间的详细接口;指定模块测试方案)[设计说明书,测试计划]**编码**(编程,进行模块调试和测试,编写用户手册)[调试报告,用户手册]**测试**、**维护**(纠错、适应、增强、预防)

**软件质量**:与软件产品满足规定的和隐含需求的能力有关的特性

**1.运行质量成本**：企业内部损失成本、

鉴定成本、预防成本和外部损失成本

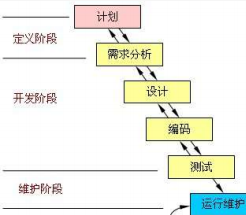
**软件= 程序(数据) + 文档+ 服务**

**2.软件产品组成部分**：程序代码、帮助文件、用户手册、样本和示例、标签、产品支持信息、图表和标志、错误信息、广告与宣传材料、软件的安装、软件说明文件、测试错误提示信息

**3.软件特征**：功能的多样性、实现的多样性、能见度低、结构的合理性差

4.**项目MM**：全程负责整个软件项目的开发。**系统设计师**：设计整个系统构架或软件构思。**程序员**：负责设计、编写程序，并修改软件中的缺陷。  
**软件测试员(**QA)：负责找出并报告软件产品的问题，与开发组密切合作，进行测试并报告发现的问题。  
技术制作、用户助手、用户培训员、手册编写和文件档案专员：负责编写软件产品附带的文件和联机文档。  
**结构管理和制作人员**：负责将程序员编写的全部文档资料合并成一个软件包

5.软件生命周期：



6.**维护阶段可能遇到四类修改要完成**：纠错、适应、增强、预防

7.**软件开发模型：**

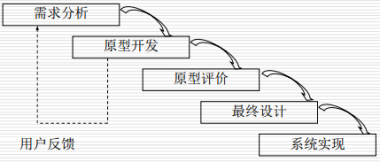
**a.瀑布模式(**优点:保证整个软件产品较高的质量和系统在整体上的充分把握,使系统具备良好的扩展性和可维护性**)**:易于理解、调研开发的阶段性、强调早期计划及需求调查、确定何时产生可交付的产品及何时进行评审和审查、强调产品测试

**大棒模式(**既没有规格说明,没有经过设计，软件随客户需要不断被修改**)**：优点(简单)几乎无计划。可能是开发者的“突发奇想”，项目成员精力都花在开发软件和编写代码上。  
缺点：开发过程非工程化，随意性大。最终的软件产品是什么样不可知。

**边写边改模式(**在大棒模式基础上考虑了产品的要求。项目成员通常只有粗略的想法就进行简单的设计，然后开始漫  
长的编码、测试、修复这样一个循环的过程。在认为无法更精细的描述软件产品要求时，就发布产品**):**优点:能够较为迅速展现成果，适合需要快速制作而且用完就扔的小项目.**缺点**：编码和测试可能是长期循环往复的过程。



**原型模式(**基本需求分析后,快速开发出产品原型，然后基于这个原型，更好了解客户需求，不断修改这个原型，到双方认可的程度，再详细分析、设计和编程，最终开发出令客户满意的产品**)**





**快速应用(RAD)模式(**通过使用基于构件的开发方法来缩短产品开发周期，提高开发速度。实现的前提是做好需求分析，且项目范围明确，这点正好和  
原型模型相反**)**

**增量模式**：描述软件产品不同阶段是按产品所具有的功能进行划分，先开发主要功能或用户最需要的功能，随着时  
间推进，不断增加新的辅助功能或次要功能，最终开发出强大的、功能完善的、高质量的、稳定的产品。

**迭代模式：**描述软件产品不同阶段是按产品深度或细化程度来划分。先将产品的整个框架建立起来，在系统初期，  
已具有用户所需求全部功能。随着时间推进，不断细化已有功能或完善已有功能，这个过程好像是一个迭代的过程。最终实现一个强大、功能完善、高质量、稳定的产品

**螺旋模式：**开始时不必详细定义所有细节，从小开始，定义重要功能，尽量实现，接受客户反馈，进入下一阶段，并重复上述过程，直到获得最终产品。

**敏捷开发敏捷方法论**采用迭代/增量开发的过程模型：以人为核心、迭代、循序渐进的开发方法。  
◼组织上，软件项目的构建被切分成多个子项目，各个子  
项目的成果都经过测试，具备集成和可运行的特征。  
◼时间上，相对于传统的瀑布式开发，迭代开发把软件生  
命周期分成很多个小周期（一般不大于2个月，建议2周）每一次迭代都可以生成一个可运行、可验证的版本并确保软件不断的增加新的价值

**敏捷开发方法：**精益开发(LeanDevelopment)、极限编程(XP)[编写用户案例，架构规范，实施规划，迭代计划，代码开发、单元测试、验收测试]、

**Scrum(**是一个敏捷开发框架,由一个开发过程，几种角色以及一套规范的实施方法组成.可运用于软件开发,项目维护,也可用来作为一种管理敏捷项目的框架**)定义了四种角色：**产品拥有者、利益相关者、Scrum专家、团队成员

进入sprint开发周期，在此周期内，每天需要召开DailyScrum meeting

**敏捷开发原则和方法：**迭代式开发、增量交付、开发团队和用户反馈推动产品开发、持续集成、开发团队自我管理

**MSF：经验知识库**(若干原则和准则：基础原则:观念、项目管理准则、风险管理准则、就绪管理准则二种模型：小组模型、监管模型)

**UML代表着软件建模的发展趋势**

**8.敏捷宣言：**个体和交互胜过过程和工具、可以工作的软件胜过面面俱到的文档、客户合作胜过合同谈判、响应变化胜过遵循计划

**MSF检查点**主要检查站和临时检查点

**MSF准则**：风险管理：风险识别、分析、计划、跟踪、控制、学习  
就绪管理：定义、评估、变更（培训、进度跟踪），评价项目管理：项目范围、变更控制、预算、成本控制和时间表、沟通、供应商管理、

**9.RUP**：面向对象且基于网络的程序开发方法论

RUP软件开发生命周期是一个二维的软件开发模型

RUP迭代式开发(生命周期)：周期、阶段、迭代和里程碑

**周期(四阶段)：**初始、细化、构造、提交 (动态结构：每个阶段分解为迭代)

**RUP迭代开发过程(迭代的个数、每个迭代的延续时间、目标)是受控的**

**静态结构：**RUP采用以下四个基本模型元素，组织和构造系统开发过程

Workers(角色)、Activities(明确目的的独立工作单元)、Artifact (多种形式存在：模型、源代码、可执行文件和文档)、Workflows(产生一些有价值的Artifacts)

两种组织WF的方式：**核心工作流**(9个(6个核心过程+3个核心支持)，被轮流使用，在每一次迭代中以不同重点和强度重复)和迭代工作流

**RUP裁剪**分以下几步：1) 确定本项目需要哪些工作流。2) 确定每个工作流需要哪些制品。3)以风险控制为原则确定4个阶段之间如何演进。4) 确定每个阶段内的迭代计划。规划RUP的4个阶段中每次迭代开发的内容。5) 规划工作流内部结构。通常用活动图的形式给出

**最佳软件工程实践**：迭代式开发、管理需求、基于组件的体系结构、可视化建模、验证软件质量、控制软件变更

**10.不同的项目需要不同的方法论，一个项目的最佳过程是这个项目所能负担的最小过程自我定义的软件过程是最好的**

**11.模型的是指：对现实的简化**

混合模型、

12.**建模目标**：便于展现系统、允许指定系统的结构或行为、提供构造系统的模板、记录决策

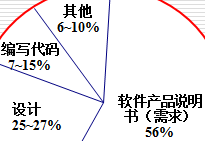
**两种建模方法**：基于算法(难维护)、面向对象(主要模块是对象或类)

13**.模型图**：结构类(类图、对象图、组件图、配置图)、行为类(用例图、序列图、行为图、协作图、状态图)、模型管理类(软件包、子系统、模型)

14.**缺陷(**破坏程序正常运行**)error**：在软件生存期内不希望或不可接受的人为错误，其结果是导致软件缺陷的产生。**bug**：是存在于软件（文档、数据、程序）之中不希望或不可接受的偏差。其结果是软件运行于某一特定条件时出现软件故障，这时**称软件缺陷被激活**。**fault**：运行过程中出现的不希望或不可接受的内部状态。此时若无适当措施(容错)加以及时处理，便产生软件失效。**failture**：是指软件运行时产生的一种不希望或不可接受的外部行为结果

15.**错误分类(软测观点)**：功能错误、系统错误、加工错误、数据错误、代码错误**逻辑错误**(按生存期不同阶段分)：问题定义错误、规格说明错误、设计错误、编码错误

**16.缺陷构成**(技术、团队、软件本身)



**17.软件缺陷状态**：激活(问题还没解决)、已修正(问题解决或通过单元测试)、关闭或非激活(bug已改)、Hold(无法解决的bug)、Differed(不需解决)

**18.可靠性指标**：**MTBF**:平均故障间隔时间 (t失效时间) **MTTF**:平均故障时间

**软件测试则是保证软件质量、提高软件可靠性的最重要手段**

**测试与开发**:**项目规划**:负责从单元测试到系统测试的整个测试阶段的监控。**需求分析**:确定测试需求分析、系统测试计划的制定.评审后成为管理项目**设计**：确保集成测试计划和单元测试计划完成。**编码**:由开发人员进行自己负责部分测试代码.项目较大时,由专人进行编码阶段的测试任务。**测试(**单元集成系统):依据测试代码进行测试并提交相应测试状态报告和测试结束报告

**软件错误不可避免**

**19．测试步骤：**详细规定了如何设置、执行、评估特定的测试用例。

**精确和准确：**准确是指得到的测试结果与真实值之间的接近程度。

精确是指同样环境下重复测测试所得到的结果之间的重现性。

**确认和验证：**确认是保证软件符合产品说明书的过程。

验证是保证软件满足用户要求的过程。

**20.测试分类**：按测试过程：单元、集成、确认、系统、验收。按测试用例设计方法分：**白盒**(结构或逻辑驱动测试[语句覆盖、判定覆盖(支覆盖)、条件覆盖、判定/条件覆盖、条件组合覆盖、基本路径测试])、**黑盒**(功能或数据驱动测试)、灰盒测试(介于黑白之间)

**按实施对象分:Alpha测试(**企业内部测试)：由用户在开发环境下进行测试，也可由公司内部的用户在模拟实际操作环境下进行的受控测试。**Beta测试**(最终用户测试)：是软件的多个用户在实际环境下进行的测试。**第三方测试(**独立测试)**按执行方式分:人工测试**：手工执行的测试; 自动化测试

**按测试方式分:静态测试**:只对被测程序进行特性分析(文档评审、代码检查、代码度量)包括检查单和静态分析方法测试的内容与选择的测试方法有关**动态测试**：必须真正运行被测试的程序(主要是白盒为主和黑盒为辅)通常对软件单元的功能、性能、接口、局部数据结构、独立路径、出错处理、边界条件和内存使用情况进行测试

**按测试形态分:建构性测试(**当程序还是处于建设阶段时所进行的测试；是属于前置性的测试,它主要是偏重于程序端的功能测试,以确保程序执行运行正常**)**;**系统测试(**针对概要设计**,**是针对系统的行为来做测试；是属于中后期的整合测试,所进行的测试是以使用者的观点为主,模拟外界使用者会如何的使用产品**)**; **特殊测试**:根据产品的本质特性来安排或剔除特殊测试

**完全测试是不可能的，测试需要终止**

**测试原则**:尽早和不断测试；测试用例要有测试输入和对应的输出；程序员避免检查自己的程序;设计测试用例应包含合理的和不合理的输入条件;妥善保存测试计划,测试用例,出错统计和最终分析报告;测试需要终止;进行正确判断；对每一个测试做全面的检查;时间服从质量;合理取舍，根据风险分析决定哪些故障必须修复，哪些故障可以不修复

**21.好测试用例4特性**：检测软件质量的有效性，是否能发现缺陷，或至少可能发现缺陷；可仿效的测试用例可以测试很多内容，因而减少测试用例的数量；经济性，测试用例的执行、分析和调试是否经济；测试用例的可修改性，每次软件修改后对测试用例的维护成本

22.**V模型价值**：非常明确标明测试过程中的不同级别,并且描述了这些测试阶段和开发过程各阶段的对应关系

**23.(每阶段要检测)测试对象是程序**

需求分析验收测试

概要测试 (开发之后测试)系统测试

详细设计集成测试

编码单元测试

**价值**明确标明测试过程中不同级别,并描述了这些测试阶段和开发过程各阶段的对应关系**问题**：测试在开发之后、测试对象是程序本身、易导致需求阶段的错误直到系统测试阶段才被发现

**24.W模型(整个开发周期测试对象：程序需求设计)**: 是V模型的改进，主要思想:尽早和不断地进行软件测试。

优点：强调了测试计划等工作的先行和对系统需求和设计的测试。

缺点：没有对软件测试流程予以说明



**25.H模型**(将测试作为独立流程,贯穿整个开发周期，与其他流程并行，同时测试准备和测试执行分离)**特性**:测试不仅仅指测试的执行,还包括许多其他活动;测试是一个独立流程,贯穿产品整个生命周期；与其它流程并发进行;测试要尽早准备,尽早执行;测试是根据被测对象的不同而分层进行.**意义**:测试准备和测试执行分离,有利于资源调配,降低成本,提高效率;充分体现测试过程(不是技术)的复杂性;有组织、结构化的独立流程,有助于跟踪测试投入的流向



26．**前置测试**：(V模型和X模型是当前被测试专家所推崇的主要的测试模型。前置测试从V模型和X模型中汲取其中精华，并设法弥补其不足之处。虽然前置测试并不完美，但以带来明显益处)开发和测试相结合,标识了项目生命周期从开始到结束之间的关键行为,对每个交付内容测试：每个交付的开发结果都必须通过一定的方式进行测试

**设计阶段是做测试计划和测试设计的最好时机(较低的成本及早发现错误)**

**验收test应独立于技术测试(针对码)**

**27.结论**：实测中，灵活运用各模型优点，通常在W模型框架下，运用H模型的思想进行独立测试。当有变更时，按X模型和前置模型的思想进行处理

27.**测试用例**：为特定目的而设计的一组测试输入、执行条件和预期的结果。

**执行的最小实体编写标准：**:标识符、测试项、测试环境要求、输入标准、输出标准、测试用例之间的关联

**28.设计测试用例准则**：用例代表性、结果的可判定性、结果的可再现性

**29．黑盒测试**(已知软件产品所具有的功能的基础上) 测试的错误类型：①不正确或遗漏的功能；②接口、界面错误；③性能错误；④数据结构或外部数据访问错误；⑤初始化或终止条件错误

**30.黑盒测试分为**：功能测试(等价类划分、边界值分析、因果图法、判定表法、场景法、正交实验法，随机测试法、错误推测法)、非功能测试(配置/安装测试、兼容性测试、互操作性测试、文档和帮助测试、性能测试、可靠性测试、易用性测试和界面测试)

**31.等价类特性**:完备性、无冗余性

确认等价类(列出等价类)、确认UC

有效等价类(实现规格说明中预先规定的功能和性能)、无效等价类(无意义的、不合理的输入数据)

**划分原则**：按区间、按输入限制、按输入布尔量、按数值、按限制条件或划分规则、细分等价类

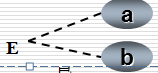
**等价类划分是边界值分析的补充**

**弱等价类测试(单缺陷假设)**

**强等价类测试(多缺陷假设)**

**32.因果图法**：约束(输入约束)：

E约束(异):a,b不能同时为1



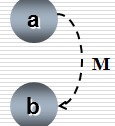
I约束(或)：a、b 和c不能同时为0。

O约束(唯一)；a和b有且仅有1个为1。

④R约束(要求)：a是1时，b必须是1，即不可能a是1时b是0



输出约束：M约束(强制)若结果a是1，则结果b强制为0

生成判定表

**基于判定表的测试是最为严格、最具有逻辑性的测试方法**

**33．判定表组成**：条件桩(问题的所有条件)、动作桩(问题规定可能采取的操作)、条件项(左列条件的可能取值-真假)、动作项(应采取的动作)

**34.规则**:任何一个条件组合的特定取值及其相应要执行的操作称为规则

35**.正交试验**：正交表的形式：

行数(Runs)：正交表中行数，即试验次数，也是正交实验法设计测试用例的。

因素数(Factors) ：正交表中列的个数，即要测试的输入数。

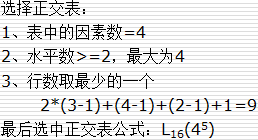
水平数(Levels)：任何单个因素取值的最大个数。正交表中的包含的值为从0到“水平数-1”，或从1到“水平数”



**36.正交表的正交性**：

**整齐可比性**：每个因素（列）的每个水平值出现的次数是完全相同的

**均衡分散性**：任意两列的水平搭配（横向形成的数字对）是完全相同的。



**37.评审是一种保证质量的方法**

**评审步骤**：计划、概述、准备、检查（评审会议）、返工和跟踪

**评审会议通用准则**:评审会议限制2小时内；如有评审人缺席或准备不充分,主持人有权取消或中止会议;检查对象是被提交的文档,而非作者;评审人必须注意他们的言语及表达方式;作者不应为自己或文档辩护;主持人不应同时作为评审人;不讨论方针之外的问题;开发方案和对应的讨论不是评审团队的任务；每个评审人员必须有机会充分表达各自论点；会议纪要必须完整表达评审人的意见；问题不应以命令的形式写给作者；问题必须划分为不同的权重：严重缺陷、重要缺陷、一般缺陷、好的；评审团队应对评审对象给出最后意见：接受(无需修改)、有条件接受(需修改，但不需进一步评审)、不接受(需进一步评审或其他的检查);要有会议纪要及总结，包括会议中讨论的问题或发现问题的列表，评审总结报告等。

**返工**：经理决定接受评审团队意见修正缺陷，或选择另外的方法(经理必须对此全权负责)

**跟踪**:专人跟踪缺陷的修改

**38.评审角色职责**：**经理(**确保文档、必需资源可用、选择评审人)，经理不一定得是管理层人员。**主持人**(管理评审有关工作：计划、准备并保证评审有序进行且满足目标，收集评审数据、发布评审报告).**作者(**文档的创建者，如有多人应为主要负责人)。**评审人**(最多5个。应能识别并描述评审对象中存在的问题)。为保证有效的覆盖率，可给评审人分配制定的评审主题。**记录员**(记录所有的发现：问题、采取的措施、决定和建议等。文字应简短和准确,最好由文档作者来担当)

**39.评审失败可能原因**：需要的人没空或不具备必须的资格和技术技能。

管理层在资源计划时不准确的估计

准备不足。文档不足

缺少管理层支持

**40:。代码检查(**检查代码和设计的一致性,在编译和动态测试前进行**)**：桌面检查(程序员自查)、代码审查(若干程序员和测试员组成一个会审小组，对程序静态分析.两步：阅读材料、召开审查会)、代码走查(分两步。避免现场修改)

41.**覆盖分类(**覆盖标准从低到高**)**：语句、判定、条件、判定/条件、组合和路径

42**.基本路径覆盖**：覆盖程序中所有可能的路径。这是最强的覆盖准则

43.**条件测试(**穷举或分支**)**：测试程序条件错误和程序的其他错误

**域测试**：对大于小于等于值的测试策略**数据流分析**：此方法在程序代码经过的路径上检查数据的用法，以期发现异常

44.**三种变量状态**：已定义的(d)：变量已赋值；引用的(r)：访问变量；没有定义的(u)：变量没有定义具体的值

45.**循环测试**(白盒的一种)：注重于循环构造的有效性。简单循环，串接(连锁)循环(相互独立和非独立)，嵌套循环(逐步外推)和不规则循环。

46.**类测试**：功能性测试(以类的规格说明为基础)和结构性测试(从程序出发)

**47.测试分类**：单元、集成、配置项(也称软件合格性试或确认)、系统、验收

**48.测试过程四步**：测试策划、测试设计、测试执行、测试总结。

**49.工作角色具体职责**  
**测试项目负责人**管理监督测试项目，提供技术指导，获取适当资源，制定基线，技术协调，负责项目的安全保密和  
质量管理**测试分析员**确定测试计划、测试内容、测试方法、测试数据生成方法、测试（软、硬件）环境、测试工具，评价测试工作的有效性  
**测试设计员**设计测试用例·确定测试用例的优先级，建立测试环境  
**测试程序员**编写测试辅助软件  
**测试员**执行测试、记录测试结果  
**测试系统管理员**对测试环境和资产进行管理和维护

**50. 测试充分性准则**单调性、非复合性、非分解性、复杂性、回报递减率

**51.单元测试(**针对编码过程中可能存在的问题**)**技术要求(模块测试)：对软件单元进行动态测试之前，应对软件单元的源代码进行静态测试：语句覆盖率达到100%、分支覆盖率要达到100%,  
对输出数据及其格式进行测试

**多个模块可平行地独立进行单元测试。**

**52.独立路径**指在程序中至少引进一个新处理语句集合或新条件的任一路径

**53．好的设计**：应能预见各种出错条件，并预设各种出错处理通路

**边界值是单元测试最后最重要的任务**

53.单元测试的文档：计划、说明、报告、记录、问题报告

**54.驱动模块**：模拟被测模块的上级模块**桩模块**：模拟被测模块的调用模块

**55.集成测试目的(**组装测试、联合测试**)**发现与接口有关的问题

**56.集成策略**：一次性组装(先分散测试，再集中起来一次完成集成测试。应避免此方式)、增值组装(自顶向下、自底向上、混合增值式)

**57.回归测试**：已进行过测试的某些子集再重新进行一遍，以保证上述改变不传播无法预料的副作用或引发新问题、

**回归测试集**包括三种类型的测试用例：  
1)能够测试软件的所有功能的代表性测试用例2)专门针对可能会被修改而影响软件功能的附加测试  
3)针对修改过的软件成分的测试

**58.确认测试**：验证软件功能和性能及其特性是否与用户要求一致

对象：包括有效性测试和软件配置审查

**59.有效性测试**(功能测试)：验证被测软件是否满足需求规格说明列出的需求

**61.效率有关术语**：响应时间：完成一项规定任务所需的时间周转时间：从发出一条指令开始到一组相关的任务完成的时间；周转时间极限：指在最大负载条件下，系统完成一项任务所需要时间的极限。

**62.负载类型**：flat测试(一次加载所有负载,在预定时间段内持续运行,取平均值-产生波动效应)、ramp-up测试(负载是交错上升,无法产生精确和可重现的平均值)

对企业级系统性能测试方法有：基准(关键是要获得一致的、可再现的结果)、性能规划、渗入测试、峰谷测试

**63.验收测试(**交付测试,UC根据需求分析规格说明设计,并在实际使用环境下来运行**)**：是否满足软件合同中的确认标准**对象**：是完整的、集成的计算机系统**步骤**:确定测试计划;建立测试环境;准备测试数据,执行测试用例,记录测试结果,分析测试结果。

**与系统测试区别**：组织机构、测试地点、覆盖范围、实施人员

**64.产品说明包含**：标识和标示功能性、可靠性、易用性、效率陈述可维护性陈述可移植性陈述使用质量陈述

**65.文档编写原则**：完备性、正确性、一致性、易理解性、易学性、可操作性

**66.软件质量要求**：功能、可靠、易用性、效率、维护、可移植性、使用质量

**66.自动化测试**(模拟手动测试步骤，控制被测软件的执行):一般是指软件测试的自动化。自动化测试可理解为测试过程自动化和测试结果分析自动化，包括测试活动的管理与实施、测试脚本的开发与执行。

**意义**：测试量大，重复性高，手工测试局限性；软件自动化测试所带来的好处。

**好处**：缩短测试周期，可让产品更快投放市场，提高测试效率，充分利用硬件资源。节省人力资源，降低测试成本。

增强测试的稳定性和可靠性。提高软件测试的准确度和精确度，增加软件信任度。测试工具使测试工作相对较容易，且能产生更高质量的测试结果。手工不能做的事情，自动化测试能做，如负载、性能测试

**原理**：通过设计的特殊程序模拟测试人员对计算机的操作过程、操作行为。

**方法**：直接对代码进行静态和动态分析(针对不同语言构造分析工具,在工具中定义相关规则)；测试过程的**捕获**(将用户每步操作都记录)和**回放**(将脚本语言所描述的过程转换为屏幕上的操作)[捕获和回放时黑盒测试技术,能大大减少工作量];测试脚本技术(一组测试工具执行的指令集)、虚拟用户技术和测试管理技术

**存在的问题**：不正确的观念或不现实的期望；缺乏良好素质和有经验的测试人才；测试工具本身的问题影响测试的质量；没有进行有效充分的培训；不考虑公司的实际情况、盲目引入测试工具；没有形成一个良好的测试工具使用环境；其他技术问题和组织问题

**限制**:不能取代手工测试;手工测试比自动测试发现的缺陷更多;对测试质量的依赖性极大;测试自动化不能提高有效性;测试自动化可能会制约软件开发;工具本身并无想象力。

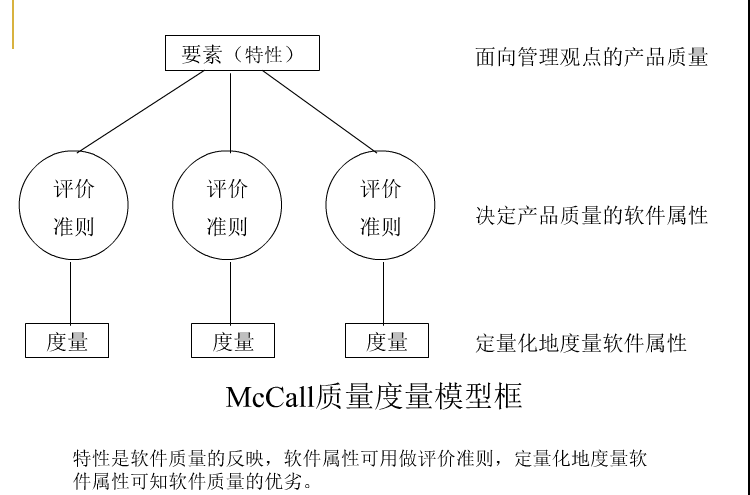
67.ISO9000:2000质量管理体系文件结构:**质量手册(QM)**:将管理层的质量方针及目标以文件形式告诉全体员工或顾客.**程序文件(**QP):指导员工如何进行及完成质量手册内容所表达的方针及目标的文件.**作业指导书(**WI):详细说明特定作业是如何运作的文件.**记录表格(**F):证实产品或服务是如何依照所定要求运作的文件。

**八项原则:**以顾客为关注焦点,领导作用,全员参与,过程方法,管理的系统方法,持续改进,基于事实的决策方法,互利的供方关系

68.**CMM**:软件过程策划**两个阶段**：组织标准生产过程策划和项目产品策划。**三个关键过程域**实现过程策划:组织过程定义,软件项目策划,软件产品工程.**成熟度等级**:初始、可重复性、已定义、已管理、优化

**69.测试与调试区别**:测试是为发现错误,贯穿于整个开发过程,调试是找出错误原因和错误发生地方并排除错误,是编码活动的一部分；指导原则、方法、操作者不同；

**70.McCall质量模型**包括质量要素、准则、度量三层次:

**1)McCall质量模型软件质量要素（11个分为3类）产品修正**：可维护性、可测试性、灵活性**、产品转移**:互联性、可移植性、复用性、**产品运行**：正确性、可使用性、完整性、可靠性、效率

**2) 质量要素评价准则:**可审查性、准确性、通信通用性、完全性、简明性、一致性、数据通用性、容错性、执行效率、可扩充性、通用性、硬件独立性、检测性、模块化、可操作性、安全性、子文档化、简单性、软件系统独立性、可追踪性、易培训性

**3)软件质量度量:**确定软件质量需求、确定度量、分析度量结果确认质量度量

**71.ISO-9126软件质量模型：**包括SQRC(软件质量需求评价准则)**、**SQDC(软件质量设计评价准则)**、**SQMC(软件质量度量评价准则)，对应于McCall质量模型的质量要素、准则、度量。

**Iso9126**

**1)8个要素**：正确性、可容性、效率、安全性、可用性、可维护性、适应性、连接性**2）23个评价准则**：和McCall一样

**3）6个质量特性4)质量特性使用**：定义软件质量需求、评价软件产品。软件质量观点（用户、开发者、管理者）**5）质量评价。目的：**1.确定产品是否通过验收 2.与其他类似产品相比较选择 3.评估产品正面和负面影响 4.确定何时优化。**步骤：**1.质量要求定义：据软件需求定义软件质量特性和可能的子特性，将用户的质量要求转化为软件开发不同阶段的质量要求，并及时分解为软件产品组成部分的质量要求。2.评价准备：（1）选择质量度量；（2）定义等级；（3）定义评估准则；3.评价过程：（1）测量：把选定的度量应用到软件产品上（2）评级（3）评估。

**7.软件质量管理:**

**1）ISO9000：2000：**本标准表述了质量管理体系的基础。1 质量管理体系的理论说明2 质量管理体系要求与产品要求3 质量管理体系方法4 过程方法5 [质量方针](http://wiki.mbalib.com/wiki/%E8%B4%A8%E9%87%8F%E6%96%B9%E9%92%88)和质量目标6 最高管理者在质量管理体系中的作用7 文件 8 质量管理体系评价9 持续改进10 统计技术的作用11 质量管理体系与其他管理体系的关注点 12 质量管理体系与优秀模式之间的关系

**1．什么是软件质量？请写出ISO/IEC9126中如何评价和度量软件质量的。**

ISO 8402中把软件质量定义为“反映实体满足明确和隐含需要的能力和特性总合”。即软件质量是软件一些特性的组合。

在ISO/IEC9126中将质量特性定义为6（包括个21个子特性），即：1)功能性：在指定条件下使用时，软件产品提供满足明确和隐含需求功能的能力；2)可靠性：在指定条件下使用时，软件产品维持规定的性能级别的能力；3)易用性：在指定条件下使用时，软件产品被理解、学习、使用及其吸引用户的能力；4)效率：在规定条件下，相对于所用资源的数量，软件产品可提供适当性能的能力；5)易维护性：软件产品可被修改的能力，修改可能包括修正、改进或者适应环境、需求和功能规约的变化；6)易移植性：软件产品从一种环境迁移到另一种环境的能力；

这6个特性作为软件质量的内部度量和外部度量，同时ISO/IEC9126还定义了使用质量度量，包括4个特性：1)有效性：软件产品在指定使用环境下，使用户准确、完整地获得规定目标的能力；2)生产率：软件产品在指定使用环境下，使用户花费合适的与有效性相关的资源数量的能力；3)安全性：软件产品在指定使用环境下，获得可接受的损害人类、商务、软件、财产或环境风险级别的能力；4)满意度：软件产品在指定使用环境下，使用户满意的能力。

**2. 怎样做好文档测试**

文档测试：是检查用户文档(如需求、用户手册、安装手册等)的正确性、清晰性和精确性，检验文档是否和实际应用存在的差别。 对用户文档中所使用的例子必须在测试中一一试过，确保叙述正确无误。

对文档的测试一般从正确性、完备性和易理解性几个方面去测试，具体包括以下内容：

（1）检查产品说明书属性、（2）检查是否完整 、（3）检查是否准确、（4）检查是否精确、（5）检查是否一致、（6）检查是否贴切、（7）检查是否合理、（8）检查代码无关、（9）检查可测试性。

**3. 请简要描述几种主要的软件测试模型** 22.24.25.26

**4. 请描述实现软件自动化测试用例的几个主要技术。**

▪ [QTP](http://baike.baidu.com/view/1303916.htm#2_1)

1. ▪ [WinRunner](http://baike.baidu.com/view/1303916.htm#2_2)
2. ▪ [Rational Robot](http://baike.baidu.com/view/1303916.htm#2_3)
3. ▪ [AdventNet QEngine](http://baike.baidu.com/view/1303916.htm#2_4)
4. ▪ [SilkTest](http://baike.baidu.com/view/1303916.htm#2_5)
5. ▪ [QA Run](http://baike.baidu.com/view/1303916.htm#2_6)
6. ▪ [Test Partner](http://baike.baidu.com/view/1303916.htm#2_7)
7. ▪ [Holodeck](http://baike.baidu.com/view/1303916.htm#2_8)
8. ▪ [Telelogic TAU](http://baike.baidu.com/view/1303916.htm#2_9)
9. ▪ [AutoRunner](http://baike.baidu.com/view/1303916.htm#2_10)
10. ▪ [Phoenix Framework](http://baike.baidu.com/view/1303916.htm#2_11)

**5. 请描述对一个嵌套循的环循环测试。**

**6. 什么是黑盒测试?黑盒测试主要采用的技术有哪些?** 30

黑盒测试又称功能测试、数据驱动测试或基于需求说明书的测试，是从用户观点出发的测试。30

**1、问：为何说软件缺陷的最大来源是产品说明书？答：**软件需求规格说明书描述了系统应该具有哪些功能/不应该具有哪些功能/功能的操作性如何/性能如何等等具体规格, 是开发初期最重要的过程文档, 也是后期开发与测试的重要依据, 可以说是开发流程与测试流程的输入, 计划、说明、报告、记录、测试问题报告

**2、软件质量保证和软件测试的关系：**软件测试能够找出软件缺陷，确保软件产品满足需求。测试可以查找错误并进行修改，从而提高软件产品的质量。软件质量保证测试避免错误以求高质量，并且还有其他方面的措施以保证质量问题。

**3、问：什么是单元测试**？答：对软件中的最小可测试单元进行检查和验证。

**4、问：单元测试主要采用什么测试方法？**答：代码走读（Code review)，静态分析（Static analysis)和动态分析

**5、等价类划分的原则**：集合的划分，划分为互不相交的一组子集，而子集的并是整个集合

等价类划分法步骤：1）确定等价类，列出等价类表。2）确定测试用例。

划分等价类可分为两种情况：（1）有效等价类（2）无效等价类

**六条确定等价类的原则**：①按照区间划分②按照输入限制划分③按照输入布尔量划分④按照数值划分⑤按照限制条件或规则划分⑥细分等价类

**6、简述边界值分析法的原则**：1）如果输入条件规定了值的范围,则应取刚达到这个范围的边界的值,以及刚刚超越这个范围边界的值作为测试输入数据。2）如果输入条件规定了值的个数,则用最大个数,最小个数,比最小个数少一,比最大个数多一的数作为测试数据。3）将规则1和2应用于输出条件，即设计测试用例使输出值达到边界值及其左右的值。4）如果程序的规格说明给出的输入域或输出域是有序集合（如有序表、顺序文件等） ,则应选取集合的第一个元素和最后一个元素作为测试用例。5）如果程序中使用了一个内部数据结构,则应当选择这个内部数据结构的边界上的值作为测试用例。6）分析规格说明,找出其它可能的边界条件。

**7、什么是测试桩？什么是驱动模块？**

在我们进行单元测试的时候，单元本身无法构成一个切实可运行的程序系统，所以我们需要为单元测试来开发桩模块和驱动模块，从而完成我们的单元测试目的，这是桩模块和驱动模块的作用。

测试桩：被测试的模块在调用的方法/模块称为桩；驱动模块：用来调用被测试模块的模块。

**8、简述验收测试的步骤**：（1）确定测试计划。（2）建立测试环境。（3）准备测试数据、执行测试用例，记录测试结果。（4）分析测试结果。

**9、简述面向对象软件测试和传统软件测试的不同**。以往采用的是过程性的测试方法，程序单元即为功能单元，系统由一系列相关联的序单元构成。可以认为程序具有单一入口和单一出口，程序之间的关系是调用关系。而面向对象的系统由一些相互关联的对象构成，对象之间靠消息传递信息，对象由数据和服务组成，程序的基本单元是“类”。

**10、测试结束的依据有那些？**

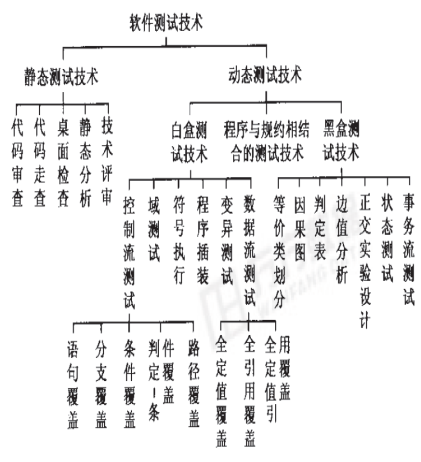
1第一类标准：测试超过了预定时间，则停止测试。

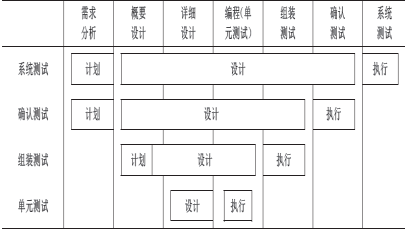
2第二类标准：执行了所有的测试用例，但并没有发现故障，则停止测试。

3第三类标准：使用特定的测试用例设计方案作为判断测试停止的基础。

4第四类标准：正面指出停止测试的具体要求，即停止测试的标准可定义为查出某一预订数目的故障。

5第五类标准：根据单位时间内查出故障的数量决定是否停止测试。

**11、榨汁机**1、 榨汁机可以榨汁的水果/蔬菜等被榨产品类型2榨汁机的工作效率3榨汁后得到的榨汁的效果4榨汁机的额定功率/刀片的旋转速度5榨汁机的破坏性测试，比如放入一些坚硬的被榨对象6榨汁机的易用性7榨汁机的可维护性，比如，榨汁机是否方便清洗8榨汁机的使用寿命，包括每种具体的刀片的使用寿命和整机的使用寿命9榨汁机的容量测试10榨汁机的噪声11榨汁机的耗电量12榨汁机的使用安全性测试13在电压过高或过低的状态能否正常使用榨汁机



**WEB测试 一、功能测试**

**1、链接测试**

链接是Web应用系统的一个主要特征，它是在页面之间切换和指导用户去一些不知道地址的页面的主要手段。链接测试可分为三个方面。首先，测试所有链接是否按指示的那样确实链接到了该链接的页面；其次，测试所链接的页面是否存在；最后，保证Web应用系统上没有孤立的页面，所谓孤立页面是指没有链接指向该页面，只有知道正确的URL地址才能访问。链接测试可以自动进行，现在已经有许多工具可以采用。链接测试必须在集成测试阶段完成，也就是说，在整个Web应用系统的所有页面开发完成之后进行链接测试。

**2、表单测试**

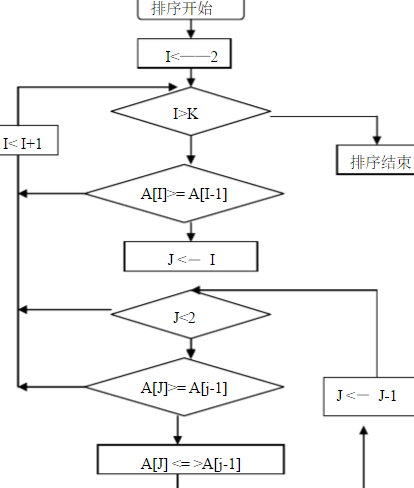
当用户给Web应用系统管理员提交信息时，就需要使用表单操作，例如用户注册、登陆、信息提交等。在这种情况下，我们必须测试提交操作的完整性，以校验提交给服务器的信息的正确性。例如：用户填写的出生日期与职业是否恰当，填写的所属省份与所在城市是否匹配等。如果使用了默认值，还要检验默认值的正确性。如果表单只能接受指定的某些值，则也要进行测试。例如：只能接受某些字符，测试时可以跳过这些字符，看系统是否会报错。**3 、Cookies测试4、设计语言测试5、数据库测试**

**二、性能测试**

**1、连接速度测试**用户连接到Web应用系统的速度根据上网方式的变化而变化，他们或许是电话拨号，或是宽带上网。当下载一个程序时，用户可以等较长的时间，但如果仅仅访问一个页面就不会这样。如果Web系统响应时间太长（例如超过5秒钟），用户就会因没有耐心等待而离开。另外，有些页面有超时的限制，如果响应速度太慢，用户可能还没来得及浏览内容，就需要重新登陆了。而且，连接速度太慢，还可能引起数据丢失，使用户得不到真实的页面。**2、负载测试**负载测试是为了测量Web系统在某一负载级别上的性能，以保证Web系统在需求范围内能正常工作。负载级别可以是某个时刻同时访问Web系统的用户数量，也可以是在线数据处理的数量。例如：Web应用系统能允许多少个用户同时在线？如果超过了这个数量，会出现什么现象？Web应用系统能否处理大量用户对同一个页面的请求？

**3、压力测试**负载测试应该安排在Web系统发布以后，在实际的网络环境中进行测试。因为一个企业内部员工，特别是项目组人员总是有限的，而一个Web系统能同时处理的请求数量将远远超出这个限度，所以，只有放在Internet上，接受负载测试，其结果才是正确可信的。进行压力测试是指实际破坏一个Web应用系统，测试系统的反映。压力测试是测试系统的限制和故障恢复能力，也就是测试Web应用系统会不会崩溃，在什么情况下会崩溃。黑客常常提供错误的数据负载，直到Web应用系统崩溃，接着当系统重新启动时获得存取权。压力测试的区域包括表单、登陆和其他信息传输页面等。

**12圈复杂度**：判定节点+1 或 边-点+2



由以上分析可知，排序程序具有双重嵌套循环结构。其内外循环体各包含一条选择语句，用于在条件满足时提前推出循环。程序中得4个判断是测试时考察得重点。以下分别列出按不同覆盖标准设计得测试用例： 语句覆盖。稍作分析便不难看出，只要送入先大后小得两个数，程序执行时就可以遍历流程图中的所有框。因此，仅需选用一组测试数据如{A＝{8,4}，K＝2}，就能实现语句覆盖。这类覆盖发现错误得能力不强，例如若将程序中得两个“>=”均误写为“＝”，用上述得测试数据就不能发现。 判定覆盖。选用上述得测试数据，内、外层循环都是从正常得循环出口退出得。要实现判定覆盖，还需在语句覆盖得基础上，增加两个能使程序从非正常出口退出的测试数据。例如，用以下两组数据： {A＝{8,4,8}，K＝3}和{A＝{8,4,4}，K＝3}或{A＝{8,4,8,4}，K＝4} 则程序将在满足A[I]=A[I-1]或A[J]=A[J-1]的条件下通过非正常出口，也能实现判定覆盖。但又可能出现另一种偏向，掩盖把“>=”误写为“＝”的错误，造成更加严重得测试漏洞。 条件覆盖。从以上分析很容易想到，必须选取足够得测试，使复合条件占的每个条件分别按“真”、“假”出现一次，才能克服前述的缺点，进一步提高发现错误的能力。测试用例：{A＝{8,4,9,6}，K＝4} {A＝{8,4,8,4}，K＝4} 就能对程序实现条件覆盖。此时A[I](或A[J])大于、等于或小于A[I-1](或A[J-1])的3中情况将分别至少出现一次，无论把“>=”误写为“>”或“=”，都可用这两组数据检查出来。 其它覆盖。本例中得两个复合条件，其组成条件都不是互相独立的。如果其中有一个条件（例如A[I]=A[I-1] ）为真，则另一个条件（例如 A[J]=A[J-1]）必然为假。所有就本例来说，判定条件覆盖及条件组合覆盖都没有实际意义，可以不必讨论。 由此可见，本例宜选择条件覆盖，以便得到较强得查错能力。测试数据可选择 {A＝{8,4,9,6}，K＝4} {A＝{8,4,8,4}，K＝4}或合成一组：{A＝{8,4,8,4,9,6}，K＝6}