声明

1. ISTQB 初级认证大纲考点分析的开发基于 ISTQB 初级大纲开发而成。
2. 感谢 ISTQB 和大纲作者的努力，对应的大纲可以从 www.istqb.org 下载获得。
3. ISTQB 初级认证大纲考点分析文档为个人开发，只能用于个人学习目的，不能用于任何商业活动。
4. 更多 ISTQB 初级认证资料，请参考：
   http://blog.csdn.net/Wenqiang_Zheng/archive/2011/04/09/6311523.aspx

从我个人的理解，在学习 ISTQB 初级大纲的时候，学员需要理解大纲中的关键句子或者段落的主要含义。针对 ISTQB 初级认证大纲的内容，考题可以有两种不同的思路：判断题的思路和问答题的思路。下面是两个例子以方便读者理解。

1）针对大纲“2.1.3 生命周期模型中的测试”中一段话“在任何生命周期模型中，良好的测试都应该具有下面几个特点：每个开发活动都有相对应的测试活动；每个测试级别都有其特有的测试目标；对于每个测试级别，需要在相应的开发活动过程中进行相应的测试分析和设计；在开发生命周期中，测试员在文档初稿阶段就应该参与文档的评审”。我们可以出类似于下面的题目，以判断选项中的内容是否有误或者哪个是描述正确的。

- 在任何开发生命周期模型中，关于良好的测试的观点，下面**错误**的是：

  a) 每个开发活动都有相对应的测试活动；
  b) 每个测试级别都有其特有的测试目标；
  c) 对于每个测试级别，需要在相应的开发活动过程中进行相应的测试分析和设计；
  d) 在开发生命周期中，测试员应该在文档发布后再参与文档的评审。

正确答案为 d。

2）针对大纲“1.2 什么是测试”中的一段话“测试可以有如下目标：发现缺陷、增加对质量的信心、为决策提供信息和预防缺陷”。我们可以出类似于下面的题目，以判断选项中的内容是否覆盖了大纲中罗列的内容。

- 软件测试的目标包括：

  (1)发现缺陷
  (2)增加对质量的信心
  (3)为决策提供信息
  (4)预防缺陷

  a) (1)
  b) (1)，(2)
  c) (2)，(3)，(4)
  d) (1)，(2)，(3)，(4)

正确答案为 d。

第 1 章 软件测试基础

1.1 为什么需要测试（K1）

1.1.1 系统测试的重要性（K1）

问答题：软件测试的不正确执行会导致哪些问题？分析：软件的不正确执行可能会导致许多问题，包括资金、时间和商业信誉等的损失，甚至导致人员的伤亡。

1.1.2 引起软件缺陷的原因（K2）

问答题：错误、缺陷和失效之间的关系？分析：所有的人都会犯错误（error， mistake），因此在由人设计的程序代码或文档中也会引入缺陷（defect， fault， bug）。当存在缺陷的代码被执行时，系统就可能无法实现期望的功能（或者实现了未期望的功能），从而引起软件失效（failure）。

判断题：并非所有的缺陷都会引起失效。

问答题：产生缺陷的原因是什么？分析：产生缺陷的原因是多种多样的：人们本身容易犯错误、时间的压力、复杂的代码、复杂的系统架构、技术的革新、以及/或者许多系统之间的交互等。

判断题：失效也可能是由于环境条件引起的。例如：辐射、电磁场和污染等都有可能引起固件中的故障，或者由于硬件环境的改变而影响软件的执行。

1.1.3 测试在软件开发，维护和运行中所担当的角色(K2)

问答题：软件测试的作用是什么？分析：对软件系统和文档进行严格的测试，可以减少软件系统在运行环境中的风险，假如在软件正式发布之前发现和修正了缺陷，可以提高软件系统的质量。进行软件测试也可能是为了满足合同或法律法规的要求，或者是为了满足行业标准的要求。

1.1.4 测试和质量（K2）

问答题：为什么测试可以提高软件质量？分析：一方面，测试过程中发现的缺陷进行修正，可以提高软件的质量。另一方面，通过分析在其他项目中发现的缺陷和引起缺陷的根本原因，可以改进开发过程，反过来可以预防相同的缺陷再次发生，从而提高以后软件系统的质量。

1.1.5 测试是否充足（K2）

问答题：从哪些方面可以判断测试是否足够？分析：在判断测试是否足够时，需要考虑下面的因素：风险以及项目在时间和预算上的限制等。

1.2 什么是测试（K2）

判断题：测试活动不仅包括测试执行活动，它只是测试的一部分。

问答题：测试的目标包括哪些？分析：测试可以有如下目标：发现缺陷；增加对质量的信心；决策提供信息；预防缺陷。

判断题：不同测试阶段，其测试目标是不同的。分析：不同的测试阶段，需要考虑不同的测试目标。比如，在开发测试中，如组件测试、集成测试和系统测试等，测试的主要目标是尽可能的发现失效，从而识别和修正尽可能多的缺陷。在验收测试中，测试的主要目标是确认系统是否按照预期工作，是建立满足了需求的信心。而在有些情况下，测试的主要目标是对软件的质量进行评估（不是为了修正缺陷），从而为利益相关者提供这样的信息：在给定的时间点发布系统版本可能存在的风险。而维护测试通常是为了验证在开发过程中的软件变更是否引入新的缺陷。在运行测试阶段，测试的主要目标是为了评估系统的特征，比如可靠性或可用性等。

问答题：调试和测试的区别是什么？调试和测试是两个不同的概念。动态测试可以发现由于软件缺陷引起的失效。而调试是一种发现，分析，清除引起缺陷原因的开发活动，随后由测试员进行的再测试是为了确认修改的代码已经解决了失效问题。每个活动的职责是截然不同的，即测试员进行测试，开发人员进行调试。

1.3 测试的基本原则（K2）

判断题（测试显示存在缺陷）：测试可以显示存在缺陷，但不能证明系统不存在缺陷。

判断题（穷尽测试是不可行的）：穷尽测试是不可能的。通过运用风险分析和不同系统功能的测试优先级，来确定测试的关注点，从而替代穷尽测试。

判断题（测试尽早介入）：为了尽早发现缺陷，在软件或系统开发生命周期中，测试活动应该尽

可能早的介入，并且应该将关注点放在已经定义的测试目标上。

判断题（缺陷集群性）：测试工作的分配比例应该与预期的和后期观察到的缺陷分布模块相适应。少数模块通常包含大部分在测试版本中发现的缺陷或失效。

判断题（杀虫剂悖论）：采用同样的测试用例多次重复进行测试，最后将不再能够发现新的缺陷。所以测试用例需要进行定期评审和修改，同时需要不断增加新的不同的测试用例。

判断题（测试活动依赖于测试背景）：针对不同的测试背景，进行不同的的测试活动。比如，对安全关键的软件进行测试，与对一般的电子商务软件的测试是不一样的。

判断题（不存在缺陷就是有用系统的谬论）：假如系统无法使用，或者系统不能完成客户的需求和期望，发现和修改缺陷是没有任何意义的。

1.4 测试的基本过程

问答题：软件测试基本过程有哪些主要活动组成？分析：计划和控制、分析和设计、实现和执行、评估出口准则和报告、测试结束活动。

1.4.1 测试计划和控制阶段（K1）

问答题：测试计划的主要活动是什么？分析：测试计划的主要活动是：识别测试任务、定义测试目标以及为了实现测试目标和任务确定必要的测试活动。

1.4.2 测试分析和设计阶段（K1）

问答题：测试分析和设计阶段的主要任务包括哪些？分析：主要任务包括如下（特别需要注意的是标注为红色的内容），

- 评审测试依据（比如需求、软件完整性级别（风险等级）、风险分析报告、系统架构、设计和接口说明）。
- 评估测试依据和测试对象的可测性。
- 通过对测试项、规格说明、测试对象行为和结构的分析，识别测试条件并确定其优先级。
- 设计测试用例并确定优先级。
- 确定测试条件和测试用例所需要的测试数据。
- 规划测试环境的搭建和确定测试需要的基础设施和工具。
- 创建测试依据和测试用例间的双向可追溯性

1.4.3 测试实现和执行阶段（K1）

问答题：测试实现和执行阶段的主要活动包括哪些？分析：通过特定的顺序组织测试用例来完成测试规程和脚本的设计，并且包括测试执行所需的其他任何信息，以及测试环境的搭建和运行测试。

问答题：测试实现和执行阶段的主要任务包括哪些？分析：

- 测试用例的开发、实现并确定它们的优先级。
- 开发测试规程并确定优先级，创建测试数据，同时也可以准备测试用具和设计自动化测试脚本。
- 根据测试规程创建测试套件，以提高测试执行的效率。
- 确认已经正确搭建了测试环境。
- 确认并更新测试依据和测试用例间的双向可追溯性

- 根据计划的执行顺序，通过手工或使用测试执行工具来执行测试规程。
- 记录测试执行的结果，以及被测软件、测试工具和测试件的标识和版本。
- 将实际结果和预期结果进行比较。
- 对实际结果和预期结果之间的差异，作为事件上报，并且进行分析以确定引起差异的原因（例如：代码缺陷、具体测试数据缺陷、测试文档缺陷、或测试执行的方法有误等）。
- 缺陷修正后，重新进行测试活动。比如通过再次执行上次执行失败的用例来确认缺陷是否已经被修正（确认测试）。执行修正后的测试用例或执行一些测试用例来确保缺陷的修正没有对软件未修改的部分造成不良影响或对于缺陷的修正没有引发其他的缺陷（回归测试）。

1.4.4 评估出口准则和报告阶段（K1）

问答题：评估出口准则和报告阶段的主要任务包括哪些？分析：

- 按照测试计划中定义的测试出口准则检查测试日志。
- 评估是否需要进行更多的测试，或是否需要更改测试的出口准则。
- 为利益相关者提供一个测试总结报告。

1.4.5 测试结束活动阶段（K1）

问答题：什么时候需要执行测试结束活动？分析：当软件系统正式发布、当一个测试项目完成（或取消）、当达到一个里程碑或当一个维护版本完成时。

问答题：测试结束活动阶段的主要任务包括哪些？分析：

- 检查提交了哪些计划的可交付产品；
- 事件报告是否关闭、或对未关闭的事件报告提交变更需求；
- 记录系统的验收；
- 记录和归档测试件、测试环境和测试基础设备，以便以后的重复使用；
- 移交测试件到维护部门；
- 分析和记录所获得的经验教训，用于以后的项目和测试成熟度改进；
- 使用为测试成熟度的提高所收集的信息。

1.5 测试的心理学（K2）

判断题：独立测试可以应用于任何的测试级别。

判断题：一定程度的独立（可以避免开发人员对自己代码的偏爱），通常可以更加高效地发现软件缺陷和软件存在的失效。

判断题：独立不能替代对软件的熟悉和经验，开发人员同样也可以高效的在他们自己

的代码中找出很多缺陷。

问答题：独立的测试级别可以有哪些？分析：

- 测试由软件本身编写的人员来执行（低级别的独立）。
- 测试由一个其他开发人员（如来自同一开发小组）来执行。
- 测试由组织内的一个或多个其他小组成员（如独立的测试小组）或测试专家（如可用性或性能测试专家）来执行。
- 测试由来自其他组织或其他公司的成员来执行（如测试外包或其他外部组织的鉴定）。

判断题：假如可以用建设性的态度对发现的缺陷或失效进行沟通，就可以避免测试员、分析人员、设计人员和开发人员之间的不愉快。这个道理不仅适用于文档的评审过程，同样也适用于测试过程。

判断题：在以建设性的方式讨论缺陷、进度和风险时，测试员和测试的负责人都需要具有良好的人与人之间沟通的能力。

问答题：如下哪些方式可以改善测试人员和其他人员之间的沟通和相互关系？分析：

- 以合作而不是争斗的方式开始项目，时时提醒项目的每位成员：共同目标是追求高质量的产品。
- 对产品中发现的问题以中性的和以事实为依据的方式来沟通，而不要指责引入这个问题的小组成员或个人。比如，客观而实际地编写缺陷报告和评审发现的问题。
- 尽量理解其他成员的感受，以及他们为什么会有这种反应。
- 确信其他成员已经理解你的描述，反之亦然。

1.6 职业道德（K2）

没有考点与之对应。

第2章 软件生命周期中的测试

2.1 软件开发模型（K2）

判断题：不同的开发生命周期模型需要不同的测试方法。

2.1.1 V 模型（顺序开发模型）（K2）

问答题：V模型的四个测试级别分别是什么？分析：组件/单元测试(component/unit testing)；集成测试(integration testing)；系统测试(system testing)；验收测试(acceptance testing)。

2.1.2 迭代-增量开发模型（K2）

判断题：在每次迭代过程中，对迭代产生的系统可能需要在不同的测试级别上进行测试。

判断题：在完成第一次迭代后，对所有的迭代进行回归测试会变得越来越重要。

判断题：验证和确认可以在每个增量模块中进行。

2.1.3 生命周期模型中的测试（K2）

问答题：良好的测试应该具有哪些特点？分析：每个开发活动都有相对应的测试活动；每个测试级别都有其特有的测试目标；对于每个测试级别，需要在相应的开发活动过程中进行相应的测试分析和设计；在开发生命周期中，测试员在文档初稿阶段就应该参与文档的评审。

判断题：根据项目的特征或系统的架构，可以对测试级别进行合并或重新进行组合。

2.2 测试级别（K2）

问答题：对于每个测试级别，都需要明确哪些内容？分析：测试的总体目标、测试用例设计需要参考的工作产品（即测试的依据）、测试的对象（即测试什么）、发现的典型缺陷和失效、对测试用具的需求、测试工具的支持、专门的方法和职责等。

2.2.1 组件测试/单元测试（K2）

问答题：组件/单元测试的测试依据有哪些？分析：组件需求说明；详细设计文档；代码。

问答题：组件/单元测试的典型测试对象有哪些？分析：组件；程序；数据转换/移植程序。

判断题：在组件测试过程中，会使用到桩、驱动器和模拟器(simulators)。

问答题：测试驱动的循环周期是什么?分析：测试用例的开发、构建和集成小块的代码，执行组件测试，修正任何问题并反复循环，直到它们全部通过测试。

2.2.2 集成测试（K2）

问答题：集成测试的测试依据是什么？分析：软件和系统设计文档；系统架构；工作流；用例。

问答题：集成测试的典型测试对象是什么？分析：子系统数据库实现；基础设施；接口。

判断题：集成测试是对组件之间的接口进行测试，以及测试一个系统内不同部分的相互作用，比如操作系统、文件系统、硬件或系统之间的接口。

判断题：对于集成测试，可以应用多种集成级别，也可以根据不同的测试对象规模采用不同的级别。

判断题：组件集成测试对不同的软件组件之间的相互作用进行测试，一般在组件测试之后进行。

判断题：系统集成测试对不同系统或软硬件之间的相互作用进行测试，一般在系统测试之后进行。

判断题：集成的规模越大，就越难在某一特定的组件或系统中定位失效，从而增加了风险并会花费额外的更多时间去发现和修理这些故障。

判断题：为了能方便快速地隔离故障和定位缺陷，集成程度应该逐步增加，而不是采用“大爆炸”式的集成。

2.2.3 系统测试（K2）

问答题：系统测试的测试依据有哪些？分析：系统和软件需求规格说明；用例；功能规格说明；风险分析报告。

问答题：系统测试的典型测试对象包括？分析：系统、用户手册和操作手册；系统配置。

判断题：系统测试关注的是在开发项目或程序中定义的一个完整的系统/产品的行为。

判断题：在系统测试中，测试环境应该尽量和最终的目标或生产环境相一致，从而减少不能发现和环境相关的失效的风险。

问答题：系统测试可能包括哪些方面的测试？分析：系统测试可能包含基于不同方面的测试：根据风险评估的、根据需求规格说明的、根据业务过程的、基于用例(use case)的、或根据其他对系统行为的更高级别描述的、根据与操作系统的相互作用的、根据系统资源等的测试。

2.2.4 验收测试（K2）

问答题：系统测试的测试依据？分析：用户需求；系统需求；用例；业务流程；风险分析报告。

问答题：系统测试的典型测试对象有哪些？分析：基于完全集成系统的业务流程；操作与维护流程；用户处理过程；结构；报告。

判断题：验收测试的目的是建立对系统、系统的某部分或特定的系统非功能特征建立信心。

判断题：发现缺陷不是验收测试的主要目标。

判断题：验收测试可以在多个测试级别上进行。

问答题：验收测试有哪几种典型的类型？分析：用户验收测试；操作（验收）测试；合同和法规性验收测试；Alpha 和Beta（或现场（field））测试。

问答题：系统操作验收测试由系统管理员来进行，测试内容主要包括哪些？分析：系统备份/恢复测试；灾难恢复测试；用户管理测试；维护任务测试；数据加载和移植活动；安全漏洞阶段性检查。

问答题：Alpha和Beta测试的区别是什么？分析：Alpha 测试通常在开发组织现场进行，但测试并非由开发团队执行。Beta 测试或实地测试，是在客户或潜在客户现场进行并由他们执行。

2.3 测试类型（K2）

问答题：不同的测试类型可能包括的测试目标有？分析：可能是测试软件所实现的功能；也可能是非功能的质量特征，比如可靠性或可用性；与软件或系统的结构或架构变更相关，如确认缺陷已被修改（确认测试）以及更改后是否引入新的缺陷（回归测试）。

2.3.1 功能测试（K2）

判断题：功能测试基于功能和特征（在文档中描述的内容或测试员自己的理解）以及专门的系统之间的交互，可以在各个级别的测试中进行。

判断题：安全性测试就是功能测试的一种。

判断题：互操作性测试是另一种功能性测试。

2.3.2 软件非功能特征测试（非功能测试）（K2）

问答题：非功能测试包括哪些？分析：性能测试、负载测试、压力测试、可用性测试、可维护性测试、可靠性测试和可移植性测试。

判断题：非功能测试可以在任何测试级别上执行。

2.3.3 软件结构/架构测试（结构测试）（K2）

判断题：可以在任何测试级别上进行结构测试（白盒测试）。

判断题：结构测试方法也可以运用到系统、系统集成或验收测试级别（比如业务模型或菜单结构）。

2.3.4 与变更相关的测试（再测试和回归测试）（K2）

判断题：当发现和修改了一个缺陷后，应该重新进行测试以确定已经成功的修改了原来的缺陷，这称之为再测试（或确认测试）。

判断题：调试（缺陷修复）是一种开发活动，不是一种测试活动。

判断题：回归测试是对已被测过的程序在修改缺陷后进行的重复测试，以发现在这些变更后是否有新的缺陷引入或被屏蔽。

判断题：当软件发生变更或者应用软件的环境发生变化时，需要进行回归测试。

判断题：回归测试的规模可以根据在以前正常运行的软件中发现新的缺陷的风险大小来决定。

判断题：回归测试可以在所有的测试级别上进行，同时适用于功能测试、非功能测试和结构测试。

判断题：回归测试套件一般都会执行多次，而且通常很少有变动，因此将回归测试自动化是很好的选择。

2.4 维护测试（K2）

判断题：维护测试是在一个现有的运行系统上进行，且一旦对软件或系统进行修改、移植或退役处理时，就需要进行维护测试。

判断题：为移植（如从一个平台移植到另外一个平台）而进行的维护测试应该包括新环境的运行测试(operational testing)，以及对变更以后的软件的运行测试。

判断题：为系统退役而进行的维护测试应该包括数据移植测试，或当数据要长时间的保存时还须存档测试。

判断题：除了对已变更的部分进行测试外，维护测试还包括对系统没有发生变更的其他部分进行大范围的回归测试。

问答题：维护测试的范围的确定取决于哪些因素？分析：维护测试的范围取决于变更的风险、现有系统的规模和变更的大小。

判断题：维护测试可以在某一测试级别或所有测试级别和测试类型上进行。

判断题：确定变更如何影响现有系统的过程，称之为影响分析，它有助于决定实施回归测试的广度和深度。

判断题：如果规格说明遗失、过时或测试人员没有具备领域知识，进行维护测试将是一件困难的事情。

第 3 章 静态技术

3.1 静态技术和测试过程（K2）

判断题：可以对任何软件工作产品进行评审，包括需求规格说明、设计规格说明、代码、测试计划、测试规格说明、测试用例、测试脚本、用户指南或 WEB 页面等。

问答题：软件评审的主要好处包括哪些？分析：软件评审的主要好处有：尽早发现和修改缺陷、改善开发能力、缩短开发时间、缩减测试成本和时间、减少产品生命周期成本、减少缺陷以及改善沟通等。评审也可以在工作产品中发现一些遗漏的内容，例如发现需求有遗漏，而这在动态测试中是很难被发现的。

判断题：评审、静态分析和动态测试具有共同的目标：识别缺陷。它们之间是互补的：不同的技术可以有效和高效地发现不同类型的缺陷。

判断题：与动态测试相比，静态技术发现的是软件失效的原因而不是失效本身。

问答题：与动态测试相比，通过评审更容易发现哪些典型的缺陷？分析：与动态测试相比，通过评审更容易发现如下典型缺陷：与标准之间的偏差、需求内的错误、设计错误、可维护性不足和错误的接口规格说明等等。

3.2 评审过程（K2）

问答题：评审过程的正式性和哪些因素相关？分析：评审过程的正式性和以下因素相关：开发过

程的成熟度、法律法规方面的要求或审核跟踪(audit trail)的需要。

3.2.1 正式评审的阶段（K1）

问答题：典型的正式评审有哪几个阶段组成？分析：计划阶段、预备会阶段、个人准备阶段、评审会议阶段、返工阶段和跟踪结果阶段。

问答题：计划阶段的主要活动是什么？分析：定义评审标准、选择人员、分配角色、为更加正式的评审类型（比如审查）制定入口和出口准则、选择需要进行评审的文档的内容。

问答题：预备会阶段的主要活动是什么？分析：分发文档、向评审参与者解释评审的目标、过程和文档、核对入口准则（针对更正式的评审类型）。

问答题：个人准备阶段的主要活动是什么？分析：先行评审文档，为评审会议做准备；标注可能的缺陷、问题和建议；

问答题：评审会议阶段的主要活动是什么？分析：检查/评价/记录结果（评审会议）；讨论和记录，并留下文档化的结果或会议纪要（针对更正式的评审类型）；标注缺陷、提出关于处理制作的缺陷决定缺陷有关的建议；检查/评价和记录任何形式的会议，或跟踪任何类型的电子通信。

问答题：返工阶段的主要活动是什么？分析：修复发现的缺陷（通常由作者来进行），记录下缺陷更新的状态（在正式评审中）。

问答题：跟踪结果阶段的主要活动是什么？分析：检查缺陷是否已得到解决、收集度量数据、核对出口准则（针对更正式的评审类型）。

3.2.2 角色和职责（K2）

问答题：典型的正式评审主要有哪些角色组成？他们各自的职责是什么？分析：经理、主持人、作者、评审员、记录员。其各自的职责如下：

- 经理：决定是否需要进行评审，在项目计划中分派时间，判断是否已达到评审的目标。
- 主持人：主持文档或文档集的评审活动，包括策划评审、召开会议和会议后的跟踪。假如需要，主持人可能还需要进行不同观点之间的协调。主持人通常是评审是否成功的关键。
- 作者：待评审文档的作者或主要责任人。
- 评审员：具有专门技术或业务背景的人员（也称为检查员)或审查员）。他们在必要的准备后，标识和描述被评审产品存在的问题（如缺陷）。所选择的评审员应该在评审过程中代表不同的观点和角色，并且应该参与各种评审会议。
- 记录员：记录所有的事件、问题，以及在会议过程中识别的未解决的问题。

3.2.3 评审类型（K2）

问答题：非正式评审的主要特点是什么？分析：

- 没有正式的过程。
- 可以是由程序员的同行们或技术负责人对设计和代码进行评审。
- 评审结果可以文档化。
- 根据不同的评审者，评审作用可能会不同。
- 主要目的：以较低的成本获得收益。

问答题：走查的主要特点是什么？分析：

- 由作者主持开会。
- 以场景、演示的形式和同行参加的方式进行。
- 开放式模式。
- 评审会议之前的准备、评审报告、发现的问题清单和记录员（不是作者本人）都是可选的。
- 在实际情况中可以是非常正式的，也可能是非常不正式的。
- 主要目的：学习、增加理解、发现缺陷。

问答题：技术的主要特点是什么？分析：

- 文档化和定义的缺陷检测过程，需要包含同行和技术专家。
- 可能是没有管理者参与的同行评审。
- 理想情况下由专门接受过培训的主持人（不是作者本人）来领导。
- 会议之前需要进行准备。
- 使用检查表是可选项。
- 准备评审报告，包括发现问题的列表、软件产品是否符合需求的判断，与发现的问题合适的建议。
- 在实际情况中可以是在不正式的和非常正式的之间。
- 主要目的：讨论、作决策、评估候选方案、发现缺陷、解决技术问题、检查与规格及标准的符合程度。

问答题：审查的主要特点是什么？分析：

- 由接受过专门培训的主持人（不是作者本人）来领导。
- 通常是同行检查。
- 定义了不同的角色。
- 引入了度量。
- 根据入口、出口规则的检查列表和规则定义正式的评审过程。
- 会议之前需要进行准备。
- 出具审查报告和发现问题列表。
- 正式的跟踪过程。
- 过程改进和读者是可选的。
- 主要目的：发现缺陷。

3.2.4 评审成功的因素（K2）

问答题：评审成功的因素有哪些？分析：

- 每次评审都有预先明确定义的目标。
- 针对评审目标，有合适的评审人员的参与。
- 测试人员参加评审不但有利于提高评审质量，还可以通过评审了解产品，为测试尽早开始做准备。
- 对发现的缺陷持欢迎态度，并客观地描述缺陷。
- 能够正确处理人员之间的问题以及心理方面的问题（比如对作者而言，能让他觉得有积极正面的体验）。
- 评审应该在一种信任的气氛中进行；并且结果不应用于对参与者的评价。
- 采用的评审技术适合于要达到的目标、软件工作产品的类型和级别以及参与评审人员。
- 选用合适的检查表或定义合适角色，可以提高缺陷识别的有效性。
- 提供评审技术方面的培训，特别是针对正式的评审技术，比如审查。

- 管理层对良好评审过程的支持（如在项目计划中安排足够的时间来进行评审活动）。
- 强调学习和过程的改进。

3.3 静态分析的工具支持（K2）

判断题：与评审一样，静态分析通常发现的是缺陷而不是失效。

问答题：静态分析的好处有哪些？分析：

- 在测试执行之前尽早发现缺陷。
- 通过度量的计算（比如高复杂性测量），早期警示代码和设计可能存在问题的方面。
- 可以发现在动态测试过程不容易发现的一些缺陷。
- 可以发现软件模块之间的相互依赖性和不一致性，例如链接。
- 改进代码和设计的可维护性。
- 在开发过程中学习经验教训，从而预防缺陷。

问答题：通过静态分析工具发现哦缺陷类型有哪些？分析：

- 引用一个没有定义值的变量。
- 模块和组件之间接口不一致。
- 从未使用的变量。
- 不可达代码(unreachable code)或死代码(dead code)。
- 逻辑上的遗漏与错误（潜在的无限循环）。
- 过于复杂的结构。
- 违背编程规则。
- 安全漏洞。
- 代码和软件模型的语法错误。

第 4 章 测试设计技术

4.1 测试开发过程（K2）

问答题：测试的正式程度主要受哪些因素的影像？分析：正式的程度是依赖于测试的背景，包括组织的架构、测试及开发过程的成熟度、项目时间的限制、安全或规范需求以及什么样的人员参与等。

判断题：将测试条件定义为能通过一个或多个测试用例进行验证的一个条目或事件（比如功能、事务处理、质量特征或结构元素等）。

判断题：测试规程（或者手工测试脚本）描述了测试用例执行的顺序。如果使用测试执行工具(test execution tool)进行测试，这种测试的动作顺序将在测试脚本中描述（自动化的测试规程）。

4.2 测试设计技术的种类（K2）

判断题：使用测试设计技术的目的是为了识别测试条件和开发测试用例。

判断题：黑盒测试设计技术（也称为基于规格说明的测试技术）是依据分析测试基础文档来选择测试条件、测试用例或测试数据的技术。它包括了功能和非功能的测试。黑盒测试，顾名思义，不需要使用任何关于被测组件或系统的内部结构信息。

判断题：白盒测试设计技术（也称为结构化或基于结构的测试技术）是基于分析被测组件或系统的结构的测试技术。

判断题：使用黑盒还是白盒测试技术取决于开发人员、测试人员和用户对所测内容的经验。

问答题：基于规格说明的测试技术具有哪些特点？分析：使用正式或非正式的模型来描述需要解决的问题、软件或其组件等；根据这些模型，可以系统地导出测试用例。

问答题：基于结构的技术的共同特点有哪些？分析：根据软件的结构信息设计测试用例，比如软件代码和软件设计；可以通过已有的测试用例测量软件的测试覆盖率，并通过系统化的导出设计用例来提高覆盖率。

问答题：基于经验的方法具有哪些共同特点？分析：测试用例根据参与人员的经验和知识来编写；测试人员、开发人员、用户和其他的利益相关者对软件、软件使用和环境等方面所掌握的知识作为信息来源之一；对可能存在的缺陷及其分布情况的了解作为另一个信息来源。

4.3 基于规格说明或黑盒测试技术（K3）

4.3.1 等价类划分（K3）

判断题：等价类划分（或等价类）可以分为两种类型的数据：有效数据（即应该被系统接受的数据）和无效数据(即应该被系统拒绝的数据)。

判断题：等价类划分也可以基于输出、内部值、时间相关的值（例如在事件之前或之后）以及接口参数（在集成测试阶段）等进行

判断题：等价类划分可以应用在所有测试级别。

4.3.2 边界值分析(K3)

判断题：每个划分的最大和最小值就是它的边界值。

判断题：有效部分的边界就是有效边界值，无效部分的边界就是无效边界值。

判断题：边界值分析可以应用于所有的测试级别。

4.3.3 决策表测试（K3）

问答题：什么是决策表？分析：决策表是一种很好的方法，它可以识别含有逻辑条件的系统需求，还可以将内部系统设计文档化。这种方法可以用来记录一个系统要实施的复杂的业务规则。通过分析规格说明，来识别系统的条件和系统的动作。输入条件和动作通常以“真”或“假”（布尔变量）的方式进行表述。决策表包含了触发条件，通常还有各种输入条件真或假的组合以及各条件组合相应的输出动作。决策表的每一列对应了一个业务规则，该规则定义了各种条件的一个特定组合，以及这个规则相关联的执行动作。

判断题：决策表测试的优点是可以生成测试条件的各种组合，而这些组合可能利用其他方法会无法被测试到。

判断题：决策表适用于所有当软件的行为由一些逻辑决策所决定的情况。

4.3.4 状态转换测试（K3）

判断题：一个状态表描绘了状态和输入之间的关系，并能显示可能的无效状态转换。

判断题：根据系统当前的情况或先前的情况（如系统先前的状态），系统可能会产生不同的响应。这种情况下，系统的特征可以通过状态转换图来表示。

判断题：测试员可以根据软件的状态、状态间的转换、触发状态变化（转换）的输入或事件以及从状态转换导致的可能的行动来进行测试。

判断题：用例描述了参与者（包括用户与系统）之间的相互作用，并从这些交互产生一个从系统用户或客户的角度所期望和能观察到的结果。

判断题：通常可以在抽象层（商业用例、不受技术限制商业流程层面）或系统层（系统功能层面的系统用例）来描述用例。

判断题：每个用例都有测试的前置条件，这是用例成功执行的必要条件。每个用例结束后都存在后置条件，这是在用例执行完成后能观察到的结果和系统的结束状态。

判断题：用例通常有一个主场景（即最有可能发生的场景）和可选的分支。

判断题：用例基于系统最可能使用的情况描述了过程流，因此从用例中得到的测试用例，在真实世界中的系统使用过程流中能最有效的发现系统的缺陷。

4.4 基于结构的或白盒技术（K4）

判断题：组件级别的软件组件的结构可以包括：语句、判定、分支或每个不同的路径。

判断题：集成级别的结构可能是调用树（模块之间相互调用的图表）。

判断题：系统级别的结构可能是菜单结构、业务过程或WEB 页面结构

4.4.1 语句覆盖和覆盖率（K4）

判断题：语句覆盖率取决于被（设计或执行）用例覆盖的可执行语句数量除以被测代码中所有可执行语句数量。

4.4.2 判定覆盖和覆盖率（K4）

判断题：判定覆盖，和分支测试相关，是指评价在一个测试用例套中已经执行的判定（例如if 语句的true 和false 选项）输出的百分比。

判断题：判定覆盖率取决于被（设计或执行）的测试用例覆盖的所有判定出口数目除以被测代码中所有可能的判定出口数目。

判断题：判定覆盖比语句覆盖更全面，100％的判定覆盖可以保证100％的语句覆盖，反之则不行。

4.4.3 其他的基于结构的技术（K1）

判断题：除了判定覆盖，还有程度更高的基于结构的覆盖，如条件覆盖和多重条件覆盖。

4.5 基于经验的技术（K2）

判断题：基于经验的测试是根据测试人员对相似的应用或技术的经验以及知识和直觉来进行测试的。

判断题：基于经验的测试技术依据测试员的经验，所以产生的效果会有极大的不同。

判断题：错误推测法的一个结构化方法是列举可能的错误，并设计测试来攻击这些错误，这种系统的方法称之为缺陷攻击。

判断题：探索性测试是指依据包含测试目标的测试章程来同时进行测试设计、测试执行、测试记录和学习，并且是在规定时间内进行的。

4.6 选择测试技术（K2）

问答题：影像测试技术选择的因素有哪些？分析：系统类型、法律法规标准、客户或合同的需求、风险的级别、风险的类型、测试目标、文档的可用性、测试员的技能水平、时间和成本预算、开发生命周期、用例模型和以前发现缺陷类型的经验等。

判断题：在建立测试用例时，测试人员通常会组合多种测试技术并结合流程、规则和数据驱动技术来保证对测试对象足够的覆盖率。

第 5 章 测试管理

5.1 测试的组织结构（K2）

问答题：可能的独立测试的类型有哪些？分析：

- 没有独立的测试人员，开发人员测试自己的代码。
- 开发团队内独立的测试人员。
- 组织内独立的测试小组或团队，向项目经理或执行经理汇报。
- 来自业务组织、用户团体内的独立测试人员。
- 针对特定测试类型的独立测试专家，例如：可用性测试人员、安全性测试人员或认证的测试人员（他们根据标准和法律法规对软件产品进行认证）。
- 外包或组织外的独立测试人员。

问答题：独立测试的优点是什么？分析：独立的测试员可以做到没有偏见，可以发现一些其他不同的缺陷；独立的测试员可以验证在系统规格说明和实现阶段的一些假设。

问答题：独立测试的缺点是什么？分析：与开发小组脱离（如果完全独立）；开发人员可能失去对软件质量的责任感；独立的测试员可能是项目的瓶颈或者要为软件发布延时负责。

5.1.2 测试组长和测试员的任务（K1）

问答题：测试组长的主要任务包括哪些？分析：

- 与项目经理以及其他人共同协调测试策略和测试计划。
- 制定或评审项目的测试策略和组织的测试方针。
- 将测试的安排合并到其他项目活动中，比如集成计划(integration planning)。
- 制定测试计划（考虑背景，了解测试目标和风险等）。计划包括选择测试方法、估算测试的时间、工作量和成本、资源的获取、定义测试级别、测试周期和计划事件管理。
- 创建测试规格说明、测试准备、测试实施和测试执行，监督测试结果并检查出口准则。
- 根据测试结果和测试进度（有时记录在状态报告中）调整测试计划，必要时采取必要的措施对存在的问题进行补救。
- 对测试件进行配置管理，保证测试件(testware)的可追溯性。
- 引入合适的度量项以测量测试进度，评估测试和产品的质量。
- 决定哪些测试用例可以自动化执行，自动化的程度，如何实现。
- 选择测试工具支持测试，并为测试员组织测试工具的培训。
- 决定测试环境的实施。
- 根据在测试过程中收集的信息编写测试总结报告。

问答题：测试员的主要任务有哪些？分析：

- 评审和参与测试计划的制定。
- 分析、评审和评估用户需求、规格说明书及模型的可测试性。
- 创建测试规格说明书。
- 建立测试环境（通常需要与系统管理员，网络管理员协同完成）。
- 准备和获取测试数据。
- 进行各种级别的测试，执行并记录测试日志，评估测试结果，记录和预期结果之间的偏差。
- 根据要求使用测试管理工具和测试监督(test monitoring)工具。
- 实施自动化测试（可能需要开发人员或测试自动化专家的支持）。
- 在可行的情况下，测量组件和系统的性能。
- 对他人的测试进行评审。

判断题：根据测试级别以及产品和项目相关的风险，测试员可以有由不同的人员担任测试员的角色，以保持一定程度的独立性。在组件和集成测试的级别，测试员可能是开发人员，进行验收测试的测试员一般是业务方面的专家和用户，进行运行验收测试的一般是将来使用软件的操作者。

5.2 测试计划和估算（K2）

5.2.1 测试计划（K2）

问答题：测试计划的制定会收到哪些因素的影响？分析：组织的测试方针、测试的范围、测试目标、风险、约束、危险程度、可测试性和资源的可用性)等。

判断题：测试计划是个持续的活动，需要在整个生命周期过程和活动中进行。从测试中得到的反馈信息可以识别变化的风险(changing risks)，从而对计划作相应的调整。

5.2.2 测试计划活动（K3）

问答题：测试计划主要包括哪些活动？分析：

- 确定测试的范围和风险，明确测试的目标。
- 定义测试的整体方法（测试策略），包括测试级别的定义、入口和出口准则的定义。
- 把测试活动集成和协调到整个软件生命周期活动中去：收集，准备，开发，运行和维护。
- 决定测试什么？测试由什么角色来执行？如何进行测试？如何评估测试结果？
- 为测试分析和设计活动安排时间进度。
- 为测试实现、执行和评估安排时间进度。
- 为已定义的不同测试任务分配资源。
- 定义测试文档的数量、详细程度、结构和模板。
- 为测试准备和执行的监控、缺陷解决和风险问题选择度量项。
- 确定测试规程的详细程度，以提供足够的信息支持可重复的测试准备和执行。

5.2.3 入口准则（K2）

问答题：入口准则通常包含哪些方面的内容？分析：测试环境已经准备就绪并可用、测试环境中的测试工具已经准备就绪、可测的代码可用、测试数据可用。

5.2.4 出口准则（K2）

问答题：出口出在通常包含哪些方面的内容？分析：完整性测量，比如代码、功能或风险的覆盖率。对缺陷密度或可靠性度量的估计。成本。遗留风险(residual risks)，比如没有被修改的缺陷

或在某些区域缺少测试覆盖等。进度表(schedules)如基于交付到市场的时间。

5.2.5 测试估算（K2）

问答题：测试工作量估算的两种方法是什么？分析：

- 基于度量的方法：根据以前或相似项目的度量值来进行测试工作量的估算，或者根据典型的数据来进行估算。
- 基于专家的方法：由任务的责任人或专家来进行测试任务工作量的估算。

问答题：测试工作量估算会受到哪些因素的影响？分析：

- 产品的特点：测试模型(即如测试基础)使用的规格说明和其它信息的质量、产品的大小、问题领域的复杂度、可靠性和安全性方面的需求、对文档的需求等。
- 开发过程的特点：组织的稳定性、使用的工具、测试过程、参与者的技能水平和时间紧迫程度等。
- 测试的输出：发现的缺陷数量和需要返工的工作量。

5.2.6 测试策略、测试方法（K2）

问答题：测试方法的选择会受到哪些因素的影响？分析：测试方法的选择取决于实际环境，应当包括风险、危害和安全、可用资源和人员技能、技术、系统的类型（比如客户定制与商业现货软件的比较）、测试对象和相关法规。

问答题：典型的测试方法包括哪些？分析：

- 基于模型的方法，比如利用失效率的统计信息(如；可靠性增长模型)或使用统计信息(如：运行概况)来进行随机测试。
- 系统的方法，比如基于失效的（包括错误推测和缺陷攻击） 方法， 基于检查表的方法和基于质量特征的方法。
- 基于与过程或标准一致的方法，比如在行业标准中规定的方法或各类敏捷的方法。
- 动态和启发式的方法，类似于探索性测试，测试很大程度上依赖于事件而非提前计划，而且执行和评估几乎是并行进行的。
- 咨询式的方法，比如测试覆盖率是主要根据测试小组以外的业务领域和/或技术领域专家的建议和指导来驱动的。
- 基于面向可重用的方法，比如重用已有的测试材料，广泛的功能回归测试的自动化，标准测试套件等。

判断题：不同的测试方法是可以结合使用的，例如基于风险的动态测试方法。

5.3 测试进度的监控（K2）

5.3.1 测试进度监控（K1）

问答题：常用的测试度量项有哪些？分析：

- 测试用例准备阶段工作所占时间的百分比（或按计划已编写的测试用例的比例）。
- 测试环境准备阶段工作所占时间的百分比。
- 测试用例执行量(如:执行的测试用例数/没有执行的测试用例数,通过/失败的测试用例数)。
- 缺陷信息（例如：缺陷密度、发现并修改的缺陷、失效率、重新测试的结果）。
- 需求、风险或代码的测试覆盖率。
- 测试员对产品的主观信心。

- 测试中确定的里程碑的具体日期。
- 测试成本，包括寻找下一个缺陷或执行下一轮测试所需成本与收益的比较。

5.3.2 测试报告（K2）

问答题：测试报告需要对哪些方面的内容进行总结？分析：测试报告是对测试工作和活动等相关信息的总结，主要包括：

- 在测试阶段发生了什么？比如达到测试出口准则的日期。
- 通过分析相关信息和度量可以对下一步的活动提供建议和做出决策，比如对仍然存在的缺陷的评估、继续进行测试的经济效益、存在的突出风险以及被测试软件的置信度等。

问答题：在进行和完成某个测试级别时需要对哪些方面进行度量和评估？分析：

- 该测试级别的测试目标的充分性。
- 采用的测试方法的适当性。
- 针对测试目标的测试的有效性。

5.3.3 测试控制（K2）

问答题：下面哪些是测试控制措施的例子？分析：

- 基于测试监控信息来做决策。
- 如果一个已识别的风险发生（如软件发布延期），重新确定测试优先级。
- 根据测试环境可用性，改变测试的时间进度表。
- 设定入口准则：规定修改后的缺陷必须经过开发人员重新测试（确认测试）后才能将它们集成到版本中去。

5.4 配置管理（K2）

判断题：配置管理的目的是在项目和产品的生命周期内建立和维护软件或系统产品（组件、数据和文档）的完整性。

问答题：配置管理对于测试的主要作用是什么？分析：

- 标识出所有测试件，版本控制，跟踪相互之间有关联以及和开发项（测试对象）之间有关联的变更，从而在测试过程中可以维持可追溯性。
- 在测试文档中，所有被标识的文档和软件项能被清晰明确的引用。

5.5 风险和测试（K2）

判断题：风险可以定义为事件、危险、威胁或情况等发生的可能性以及由此产生不可预料的后果，即一个潜在的问题。

判断题：风险级别由出现不确定事件的可能性和出现后所产生的影响（事件引发的不好的结果）两个方面来决定。

5.5.1 项目风险（K2）

判断题：项目风险是指关于项目按目标交付的能力方面的风险。

问答题：下面哪些风险是属于项目风险？分析：项目风险是指关于项目按目标交付的能力方面的风险，比如：

- 公司组织因素：
  - 技能、培训和人员的不足。
  - 个人问题。
- 政策因素，比如
  - 与测试员进行需求和测试结果沟通方面存在的问题。
  - 测试和评审中发现的信息未能得到进一步跟踪（如未改进开发和测试实践）。
  - 对测试的态度或预期不合理（如：认为在测试中发现缺陷是没有价值的）。
- 技术方面的问题：
  - 不能定义正确的需求。
  - 给定现有限制的情况下，没能满足需求的程度。
  - 测试环境没有及时准备好。
  - 数据转换、移植计划，开发和测试数据转换/移植工具造成的延迟。
  - 低质量的设计、编码、配置数据、测试数据和测试。
- 供应商的问题：
  - 第三方存在的问题。
  - 合同方面的问题。

5.5.2 产品风险（K2）

判断题：在软件或系统中的潜在失效的区域（即将来可能发生的不利事件或危险）称之为产品风险。

问答题：下面哪些风险是属于产品风险的？分析：产品风险是针对产品质量而言的，比如：

- 易错(failure-prone)的软件交付使用。
- 软件/硬件对个人或公司造成伤害的可能性。
- 劣质的软件特征（比如功能性、可靠性、可用性和性能等）。
- 低劣的数据完整性和质量（例如：数据迁移问题、数据转换问题、数据传输问题、违反数据标准问题）。
- 软件没有实现既定的功能。

问答题：在基于风险的测试中，识别的风险的主要作用包括哪些？分析：

- 决定采用何种测试技术。
- 决定要进行测试的范围。
- 为了尽早的发现严重的缺陷，确定测试的优先级。。
- 决定是否可以通过一些非测试的活动来减少风险（比如对缺乏经验的设计者进行相应的培训）。

问答题：系统化的风险管理包括哪些活动？分析：为了确保产品失效机会最小化，风险管理活动提供了一些系统化的方法：

- 评估（和定期重新评估）可能出现错误处（风险）。
- 确定哪些风险需要处理。
- 实施处理这些风险的措施。

5.6 事件管理（K3）

判断题：在软件产品的开发、评审和测试、以及软件使用的过程中都会产生事件。它们可能是在代码内或在使用的系统内或以任意方式在文档内（包括需求文档、开发文档、测试文档和用户信

息如“帮助”或安装手册等)。

问答题：事件报告的主要目的和作用是什么？分析：

- 为开发人员和其他人员提供问题反馈，在需要的时候可以进行识别、隔离和纠正。
- 为测试组长提供一种有效跟踪被测系统的质量和测试进度的方法。
- 为测试过程改进提供资料。

问答题：事件报告的主要内容有什么？分析：

- 提交事件的时间，提交的组织和作者。
- 预期和实际的结果。
- 识别测试项（配置项）和环境。
- 发现事件时软件或系统所处的生命周期阶段。
- 为了能确保重现和解决事件需要描述事件（包括日志、数据库备份或截屏）。
- 对利益相关者的影响范围和程度。
- 对系统影响的严重性。
- 修复的紧迫性/优先级。
- 事件状态（例如：打开的、延期的、重复的、待修复的、修复后待重测的或关闭的等）。
- 结论、建议和批准。
- 全局的影响，比如事件引起的变更可能会对系统的其他部分产生影响。
- 变更历史纪录，比如针对事件的隔离、修改和确认已修改，项目组成员所采取的行动顺序。
- 参考(references)，包括发现问题所用的测试用例规格说明的标识号。

第 6 章 软件测试工具

6.1 测试工具的类型（K2）

6.1.1 理解使用测试工具支持测试的意义和目的（K2）

问答题：测试工具可以用于支持哪些测试活动？分析：直接用于测试的工具，如测试执行工具、测试数据生成工具和结果对比工具；帮助管理测试流程的工具，如用于管理测试、测试结果、数据、需求、事件、缺陷等等，并且可以报告和监控测试执行；用于观测的工具，或简单地说就是探索（例如：监视应用程序文件活动的工具）；任何对测试有帮助的工具（从这个角度看，电子表格也可以是测试工具)。

问答题：根据实际情况，工具支持测试的目的有哪些？分析：在自动化重复性的任务时可以改善测试活动的效率，或支持手动测试活动，如测试计划、测试设计、测试报告与监控；手动进行可能需要大量资源，则可考虑自动执行（例如，静态测试）；无法手动完成的测试可以将其自动化（例如：CS 应用程序的大规模性能测试）；增加测试的可靠性（例如：进行大量数据的自动比较或是行为模拟)。

问答题：术语“测试框架”被大量的应用于工业界，它的主要的三个含义是什么？分析：可重用、可扩展的测试库，可用于搭建测试工具（也称为测试用具）；设计测试自动化的类型（例如：数据驱动、关键字驱动）；测试执行全流程。

6.1.2 测试工具分类（K2）

判断题：有些工具仅仅支持一种测试活动；有些可以支持多种测试活动。

判断题：某些类型的测试工具本身是植入式的，因工具本身会影响测试对象的行为。比如，使用

不同的性能测试工具测出来的实际时间特征会有所不同，或使用不同的覆盖率分析器工具可能测量出不同的覆盖率。一个工具的植入式特征也称之为探测影响(probe effect)。

6.1.3 测试管理的工具支持（K1）

判断题：管理工具适用于整个软件生命周期中的所有测试活动。

问答题：测试管理工具的主要作用？分析：这些工具除了支持测试执行、缺陷跟踪和需求管理提供接口外，还提供定量分析和报告测试对象。它还支持追溯测试对象到需求规格说明并可提供独立的版本控制能力或提供一个外部接口。

问答题：需求管理工具的主要作用？分析：需求管理工具储存了需求描述、需求的一些属性（包括优先级），并提供统一的标识符以及从需求到相应测试的可追溯性。这些工具也可以帮助识别自相矛盾或遗漏的需求。

问答题：事件管理工具（缺陷跟踪工具）的主要作用？分析：这类工具存储并管理事件报告，即缺陷、失效、变更请求或察觉到的问题和异常，并协助管理事件的整个生命周期，为静态分析提供了可能。

问答题：配置管理工具的主要作用？分析：严格的来说，配置管理工具并不能算是测试工具，但对测试件和相关软件版本进行存储和管理时却是必要，尤其是当配置一个以上硬件/软件环境的时候，比如：多个操作系统版本、多种编译器、多种浏览器等等。

6.1.4 静态测试的工具支持（K1）

问答题：评审工具的主要作用？分析：这类工具可支持评审过程、检查表、评审指导方针，存储和交流评审意见、缺陷和工作报告。也可为庞大的或分布于不同地区的团队提供在线评审。

问答题：静态分析工具的主要作用？分析：这类工具通过实施的编码规范（包括安全编码）、结构和其相关性的分析，从而帮助开发和测试员在动态测试之前就找到缺陷。通过对代码进行度量（例如：复杂度）可以帮助计划或风险分析。

问答题：建模工具的主要作用？分析：建模工具可以用来确认软件模型（例如：关系数据库的物理数据模型“physical data model-PDM”），可以发现其不一致性和缺陷。他们通常也用于生成一些基于模型的测试用例。

6.1.5 测试规格说明的工具支持（K1）

问答题：测试设计工具的主要作用？分析：测试设计工具能够根据需求、图形用户界面（GUI）、设计模型（状态、数据或对象）或代码生成测试输入或可执行的测试和/或测试准则。

问答题：测试数据准备工具的主要作用？分析：测试数据准备工具用来处理数据库、文件或数据传输，并且生成可以在测试执行过程中使用的测试数据。这种类型的工具的优点是可以保证传输到测试环境中的实时数据是匿名的，从而提供数据保护。

6.1.6 测试执行和记录工具（K1）

问答题：测试执行工具的主要作用？分析：测试执行工具存储了输入和预期结果并通过脚本语言使测试能够自动或半自动进行并记录每一轮测试。它也可用于录制测试，并支持基于脚本语言或图形用户界面数据的参数化配置和其他自定义设置。

问答题：测试用具/单元测试框架工具的主要作用？分析：单元测试用具或测试框架用桩和驱动

器作为测试对象的虚拟替代品，使在模拟环境中的组件测试或部分系统测试能够达成目标。

问答题：测试比较器的主要作用？分析：测试比较器能够确定文件、数据库或测试结果之间的差异。测试执行工具通常包括动态比较器，但执行后的比较也许会由一个分离的独立比较器完成。一个测试比较器可能会使用测试准则，尤其是自动化进行的时候。

问答题：覆盖率测量工具的主要作用？分析：覆盖率测量工具可以是植入式和非植入式两种，代码覆盖率工具测量已执行的特定代码结构类型（比如语句、分支或判定以及模块或功能调用）所占百分比。

问答题：安全测试工具的主要作用？分析：这类工具用于评估软件的安全特性，包括评估软件保护数据机密性、完整性、身份验证、授权、可用性和不可否认性的能力。

6.1.7 性能和监控工具（K1）

问答题：动态分析工具的主要作用？分析：动态分析工具仅能发现那些只有在软件执行过程中才显现的错误，比如与时间依赖(time dependencies)有关的问题，或内存泄漏(memory leaks)等。它们通常应用在组件和组件集成测试以及中间件的测试。

问答题：性能测试/负载测试/压力测试工具的主要作用？分析：性能测试工具监测和报告系统在模拟多种大量用户同时使用系统时的性能表现、持续增加的模型、处理各业务的频率和相应的处理百分比。系统的负载只是创造了虚拟用户，用来执行一个所选的业务，这些虚拟用户分布在不同的测试机上，通常理解为负载生成器。

问答题：监测工具的主要作用？分析：监测工具持续地分析、验证和报告特定系统资源的使用情况，对可能存在的服务问题提出警告。

6.1.8 特定应用领域的测试工具（K1）

问答题：数据质量评价工具的主要作用？分析：数据是有些项目的关键，例如：数据转换/数据迁移项目和数据仓库（data warehouse）应用程序，它们会在不同临界和容量时表现出不同的特性。在有些情况下，需要使用工具评价数据质量，来评审和验证数据转换和迁移规则，以确保处理的数据是正确的、完整的并遵循一个预定义的、符合特定背景的标准。

6.2 有效使用工具：潜在的收益与风险（K2）

6.2.1 测试工具的潜在收益和风险（针对所有工具）（K2）

问答题：使用工具的潜在收益是什么？分析：减少重复性的工作（比如，执行回归测试，重新输入相同测试数据，按代码标准检查）；更好的一致性和可重复性（比如，用工具执行测试，从需求导出测试）；客观的评估（比如，静态测量、覆盖率）；容易得到测试和测试的相关信息（比如，关于测试进展的统计和图表，事件发生率和性能）

问答题：使用工具存在的风险有哪些？分析：对工具抱有不切实际的期望（包括功能性和易用性）；低估首次引入工具所需的时间、成本和工作量（包括培训和获取外部的咨询）；低估从工具中获得较大和长久收益需要付出的时间和工作量（包括更改测试过程并不断改进工具使用方式的需要）；低估对测试工具生成的结果进行维护所需的工作量；对测试工具过分依赖（替代测试设计或者对一些更适合手工测试的方面却使用自动测试工具）；忽视了使用工具时对测试相关文档的版本控制；忽视了多个重要工具之间的关系和互操作性，例如：需求管理工具、版本控制工具、事件管理工具、缺陷跟踪工具和其他从不同供应商获得的工具；工具供应商破产、工具退役或不同供应商所提供的工具的风险；工具供应商缺少对工具的支持，对升级和缺陷修复反应迟缓；开

源/免费工具项目中止的风险；其他不可预知的风险，例如不能支持新平台。

6.2.2 一些工具类型的特殊考虑（K1）

判断题：数据驱动的方法是将测试输入（测试数据）与测试用例分离，并将测试输入存放在一个电子表格中，这样可以使用不同的数据进行相同的测试。

判断题：在关键字驱动的测试方法中，电子表格含有描述系统要采取的行为的关键字（也称为行为字 action words）和测试数据。测试员（即使不熟悉脚本语言）也能用这些关键字来定义测试，并可根据被测程序进行调整和适配。

判断题：无论使用什么脚本技术都需要储存每个测试的预期结果，便于以后能与实际的结果进行比较。

判断题：静态分析工具检查源代码是否遵循编码标准(coding standards)。

判断题：测试管理工具需要有与其他工具和表格的接口以便产生符合组织所需格式的有用信息。

6.3 组织内引入工具（K1）

问答题：为组织选择一个工具所需要考虑的关键点有哪些？分析：评估组织的成熟度(maturity)、分析引入工具的优点和缺点和认识引入工具能改善测试过程的可能性；根据清晰的需求和客观的准则进行评估；概念验证(proof-of-concept)，在评估阶段要确认在现有的情况下使用工具对被测软件是否有足够效果，或为了有效使用工具，目前的基础设施需要如何改变；对工具提供商进行评估（包括培训、提供的支持及其他商业方面考量）或针对非商业性工具要考虑供应商提供的服务；确定在工具使用方面应提供的指导和内部培训需求； 评估培训需求时需要考虑现有测试团队的自动化测试技能；根据实际的情况估算成本-收益比。

问答题：将选择的工具引入组织要从一个试点项目开始，试点项目的主要目的有哪些？分析：对工具有更多的认识；评价工具与现存的过程以及实践的配合程度，确定哪些方面需要作修改；决定工具和由工具生成/应用的结果的使用、管理、保存和维护的使用工具标准（比如，文件和测试的命名规则、创建数据库和定义测试套件(test suites)）；评估在付出合理的成本后能否得到收益。

问答题：在组织内成功部署工具的因素有哪些？分析：逐步在组织的其他部门推广工具；调整并改进过程来配合工具的使用；为新使用者提供培训和指导；定义使用指南；在实际运用中实施收集工具使用情况的方法；监测工具的使用和收益情况；为测试团队使用工具提供支持；在所有团队内收集经验和教训。