# 第1章 软件测试基础

四、简答题

1、**软件缺陷处理流程**为：提交→分配→确认→处理→复测→关闭

（1）提交：测试人员发现缺陷之后，将缺陷提交给测试组长。

（2）分配：测试组长接收到测试组员提交的缺陷之后，将其移交给开发人员。

（3）确认：开发人员接收到移交的缺陷之后，会与团队甚至测试人员一起商议，确定该缺陷是否是一个缺陷。

（4）拒绝：如果经过商议之后，缺陷不是一个真正的缺陷则拒绝处理，关闭缺陷。如果经过商议之后，确定其是一个真正的缺陷，则可以根据缺陷的严重程度或优先级等立即处理或延期处理。

（5）处理：开发人员修改缺陷。

（6）复测：开发人员修改好缺陷之后，测试人员重新进行测试（回归测试）。

（7）关闭：测试人员进行回归测试之后，如果缺陷已经被正确修改，则将缺陷关闭，整个缺陷处理完成。

2、**软件测试的基本流程**为：

（1）分析测试需求

测试人员在制定测试计划之前需要先对软件需求进行分析。

（2）制定测试计划

测试计划是整个测试工作的导航图，它的制定是随着项目发展不断调整、逐步完善的过程。

（3）设计测试用例

测试用例（Test Case）指的是一套详细的测试方案，包括测试环境、测试步骤、测试数据和预期结果

（4）执行测试

测试执行就是按照测试用例执行测试的过程，这是测试人员最主要的活动阶段。在

（5）编写测试报告

测试报告是一个测试活动的总结，对项目测试过程进行总结，对测试数据进行统计，对项目的测试质量进行客观的评价文档。

# 第2章 黑盒测试

四、简答题

1、**等价类划分原则**：

（1）如果程序要求输入值是一个有限区间的值，则可以将输入数据划分为一个有效等价类和两个无效等价类，有效等价类为指定的取值区间，两个无效等价类分别为有限区间两边的值。

（2）如果程序要求输入的值是一个“必须成立”的情况，则可以将输入数据划分为一个有效等价类和一个无效等价类。

（3）如果程序要求输入数据是一组可能的值，或者要求输入值必须符合某个条件，则可以将输入数据划分一个有效等价类和一个无效等价类。

（4）如果在某一个等价类中，每个输入数据在程序中的处理方式都不相同，则应将该等价类划分成更小的等价类，并建立等价表。

2、**决策表条件项的合并规则**

在实际测试中，条件桩往往很多，而且每个条件桩都有真假两个条件项，有n个条件桩的决策表就会有2n条件规则，有些规则的取值对结果并无影响，这个问题就称为无关条件项，无关条件项使用“-”表示，忽略无关条件项，可以将这两条规则进行合并。合并之后的无关条件项（-）包含其他条件项取值，因此具有相同动作的规则还可进一步合并

3、**正交实验设计法测试用例设计步骤**：

（1）提取因子，构造因子状态表

（2）加权筛选，简化因子-状态表

（3）构建正交表，设计测试用例

# 第3章 白盒测试

四、简答题

1、请简述一下逻辑覆盖的几种方法及它们之间的区别。

（1）语句覆盖

语句覆盖的目的是测试程序中的代码是否被执行，语句覆盖在多分支的程序中，只能覆盖某一条路径，使得该路径中的每一个语句至少被执行一次，但不会考虑各种分支组合情况。

（2）判定覆盖

判定覆盖(Decision Coverage)又称为分支覆盖，其原则是设计足够多的测试用例，在测试过程中保证每个判定至少有一次为真值，有一次为假值。作用是使真假分支均被执行，

具有和语句覆盖一样的单一性。

（3）判定-条件覆盖

判定-条件覆盖(Condition/Decision Coverage)要求设计足够多的测试用例，使得判定语句中所有条件的可能取值至少出现一次，同时，所有判定语句的可能结果也至少出现一次。

它弥补了判定覆盖和条件覆盖的不足之处。但是判定-条件覆盖在仍旧存在遗漏测试的情况。

（4）条件组合覆盖

条件组合(Multiple Condition Coverage)指的是设计足够多的测试用例，使判定语句中每个条件的所有可能至少出现一次，并且每个判定语句本身的判定结果也至少出现一次。

它与判定-条件覆盖的差别是，条件组合覆盖要求让这些结果的所有可能组合都至少出现一次。

2、请简述一下目标代码插桩的三种执行模式。

目标代码插桩具有以下三种执行模式:

（1）即时模式（Just-In-Time）：原始的二进制或可执行文件没有被修改或执行，将修改部分的二进制代码生成文件副本存储在新的内存区域中，在测试时仅执行修改部分的目标代码。

（2）解释模式（Interpretation Mode）：在解释模式中目标代码被视为数据。

（3）探测模式（Probe Mode）：探测模式使用新指令覆盖旧指令进行测试。

# 第4章 性能测试

四、简答题

1、常用的性能测试指标。

（1）响应时间

响应时间（Response Time）是指系统对用户请求作出响应所需要的时间。

（2）吞吐量

吞吐量（Throughput）是指单位时间内系统能够完成的工作量，它衡量的是软件系统服务器的处理能力。

（3）并发用户数

并发用户数是指同一时间请求和访问的用户数量。

（4）TPS

TPS是指系统每秒钟能够处理的事务和交易的数量，它是衡量系统处理能力的重要指标。

（5）点击率

点击率是指用户每秒向Web服务器提交的HTTP请求数。

（6）资源利用率

资源利用率是指软件对系统资源的使用情况。

2、常见的性能测试种类。

（1）负载测试

负载测试是指逐步增加系统负载，测试系统性能的变化，并最终确定在满足系统性能指标的情况下，系统所能够承受的最大负载量。

（2）压力测试

压力测试是指逐步给系统增加压力，测试系统的性能变化，使系统某些资源达到饱和或系统崩溃，从而确定系统所能承受的最大压力。

（3）峰值测试

峰值测试是指瞬间将系统压力加载到最大，使测试软件系统在极限压力下的运行情况。

（4）配置测试

配置测试是指调整软件系统的软硬件环境，测试各种环境对系统性能的影响，从而找到系统各项资源的最优分配原则

（5）可靠性测试

可靠性测试是指给系统加载一定的业务压力，使其持续运行一段时间（如7\*24h），测试系统在这种条件下是否能够稳定运行。

（6）容量测试

容量测试是指在一定的软硬件及网络环境下，测试系统所能支持的最大用户数、最大存储量等。

3、LoadRunner的组成部分及其作用。

（1）VuGen

VuGen用于创建虚拟用户脚本的工具。

（2）Controller

Controller用于创建和控制LoadRunner场景。

（3）Analysis

Analysis是LoadRunner的数据分析工具，它可以收集性能测试中的各种数据，对其进行分析并生成图表和报告供测试人员查看。

# 第5章 安全测试

四、简答题

1、安全测试与常规测试的区别。

（1）测试目标不同

普通测试以发现BUG为目标；安全测试以发现安全隐患为目标。

（2）假设条件不同

普通测试假设导致问题的数据是用户不小心造成的，接口一般只考虑用户界面；安全测试假设导致问题的数据是攻击者处心积虑构造的，需要考虑所有可能的攻击途径。

（3）思考域不同

普通测试以系统所具有的功能为思考域；安全测试的思考域不但包括系统的功能，还有系统的机制、外部环境、应用与数据自身安全风险与安全属性等。

（4）问题发现模式不同

普通测试以违反功能定义为判断依据；安全测试以违反权限与能力的约束为判断依据。

2、安全测试基本原则。

（1）培养正确的思维方式

（2）尽量测试和经常测试

（3）选择正确的测试工具

（4）可能情况下使用源代码

（5）测试结果文档化

3、XSS攻击原理、过程及防范措施。

原理：攻击者可以利用XSS漏洞把恶意代码注入到网站中，当有用户浏览该网站时，这些恶意代码就会被执行，从而达到攻击的目的。

过程：在XSS攻击中，攻击者会通过邮件或其他方式诱使用户点击包含恶意代码的链接，用户点击链接后，浏览器会在用户毫不知情的情况下执行链接中包含的恶意代码，将用户与home.com交互的cookie和session等信息发送给攻击者，攻击者拿到这些数据之后，就会伪装成用户与真正的网站进行会话，从事非法活动。

防范措施：对于XSS漏洞，最核心的防御措施就是对用户的输入进行检查和过滤，除此之外，当向HTML标签或属性中插入不可信数据时，要对这些数据进行进行相应的编码处理。将重要的cookie标记为http only，这样javascript脚本就不能访问这个cookie，避免了攻击者利用javascript脚本获取cookie。

# 第6章 自动化测试

四、简答题

1、请简述持续集成的基本过程。

 参见持续集成过程流程图，持续集成需要版本控制工具、持续集成工具、持续集成环境部署、基础测试、阶段性测试。一旦持续集成环境搭建完成后测试人员通过集成工具获取项目代码，进行基本测试或执行相关的测试用例，得到最初的测试结果。

2、请简述传统持续集成框架和持续集成容器的区别。

 在传统持续集成框架下进行测试之需要搭建代码托管平台或者使用本地项目，准备好测试所需要的数据以及当前项目需要的服务器、测试用例等。对于持续集成容器化进行测试需要测试人员搭建容器环境、管理容器仓库等操作，搭建环境比传统持续集成测试搭建相对复杂但提高了项目部署效率以及不同场景下测试的便捷。

3、请简述自动化测试使用的技术。

 自动化测试技术可分为录制与回放技术，使用录制回放工具可以将操作过程录制下来通过回放观察是否存在问题，脚本测试可分为线性脚本、结构化脚本、共享脚本，在测试中可以使用自动化测试框架Junit、Unittest框架进行编写或者参考脚本录制回放工具生成的脚本进行测试；数据驱动测试分为关键字驱动测试、行为驱动测试。