**软件测试的内容及分类**

一、软件的内容

1.1.什么是软件？

软件是计算机程序、程序所用的数据以及有关文档资料的集合。

1.2软件又分两大类：系统软件和应用软件

1.2.1系统软件:系统软件是生成、准备和执行其他程序所需要的一组文件和程序。如操作系统Windows、数据库SQL-server、驱动程序（网卡、声卡）、java语言系统编译环境等。

1.2.2应用软件：计算机用户为了解决某些具体问题而购买、开发或研制的各种程序或软件包。如APP、QQ、微信等。

提问：软件测试的对象是什么呢？

答：计算机程序、数据、有关资料文档。

1.3应用软件架构---了解

C/S与B/S架构

1.3.1 C/S：client-server 这种就是我们一定要安装一个客户端才能够用的软件，就叫C/S。

优点：系统兼容性，效率高，更安全。

缺点：每次更新，都需要更新服务端与客户端。比如超市收银系统每次更新每台电脑都必须重装客户端，特别是有分店的情况，人力物力财力都很大。----重启 业务中断

1.3.2 B/S：browser-server 只需要一个浏览器，就可以访问服务的，就是B/S。

优点：只需要更新服务器就OK，不需要去更新浏览器，用户主动性比较高，较稳定。比如：天猫、淘宝。

缺点：反应慢，功能不如C/S齐全，也不如C/S稳定。

提问：APP是什么架构的呢？

答：移动端C/S。

1. 软件测试是什么

2.1软件测试的定义

使用人工和自动手段来运行或测试某个系统的过程，其目的在于检验它是否满足规定的需求，或弄清预期结果与实际结果之间的差别。

问题：使用QQ的过程是软件测试吗？

答：不是，我们是客户的角色，是使用者，是自己在进行交流的目的，不是测试者。

2.2软件测试的目的

2.2.1为了发现程序（软件）存在的代码或业务逻辑错误。---功能问题----熟悉产品需求---bug

2.2.2为了检验产品是否符合用户需求。------用户需求

2.2.3为了提高用户的体验。-----流畅性、性能范畴

业务逻辑：例如淘宝：注册登录--浏览产品--加入购物车--提交订单--支付--付款（扣钱）--查看订单--查看物流 等等。

三、软件测试的分类

3.1按照测试技术分类

黑盒测试 白盒测试 灰盒测试

3.1.1黑盒测试：顾名思义，就是把所有的功能和逻辑接口都放在一个盒子里面，你是看不到里面的逻辑与走向的，你只能通过盒子的外表进行测试。黑盒测试是指在测试过程中只关注输入输出，如果输入一个测试数据，输出的结果是正确的，我们就认为这个功能是正确的，也叫数据驱动测试。

输入输出--数据驱动的测试。----点点点

3.1.2白合测试：基于软件内部设计和程序实现的测试方法（代码层面），不仅仅关注输入与输出的结果是否正确，同时还关注程序是如何处理的。

代码逻辑--会代码--不是测试做的，开发自测---代码审查--单元测试

3.1.3灰盒测试：大概知道代码逻辑实现，不需要看懂所有的代码。---接口测试

3.2按测试对象是否运行划分

动态测试 静态测试（文档检查、代码走查）

3.3按不同的测试手段划分

手工测试（点工--点点点） 自动化测试（工具+代码）（例jmerter）

3.4按测试包含的内容划分

功能测试 界面测试 安全测试 兼容性测试 易用性测试 性能测试

3.4.1功能测试：测试软件的功能是否符合需求，通常采用黑盒测试方法，一般由测试人员独立执行。测试业务逻辑（手工、自动化）----核心重要

3.4.2界面测试：也称UI测试，测试用户界面布局是否合理，整体风格是否一致，界面文字是否正确，命名是否统一，页面是否美观，文字、图片组合是否完美等。

3.4.3安全性测试：测试该系统防止非法入侵的能力。

3.4.4兼容性测试：测试该系统与其他软件硬件兼容的能力，（app与c/s架构软件、b/s架构软件）---软件+硬件，软件+软件，软件不同版本之间

3.4.5易用性测试：测试软件是否易用，主观性比较强，一般要根据很多用户的测试反馈信息，才能评价易用性，（同类型产品）用户使用习惯，好不好用。

---站在用户角度考虑，参考成熟产品

3.4.6性能测试：通过自动化的测试工具模拟多种正常、峰值以及异常负载条件来对系统的各项性能指标进行测试。---双十一（访问人数多），并发

3.5按测试阶段划分

单元测试 集成测试 系统测试 验收测试 α测试 β测试

3.6其他测试

回归测试 冒烟测试 探索性测试/自由测试（测试思维）

3.6.1回归测试：指错误（bug）被修正后或软件功能、环境发生变化后（开发修改）的重新测试，确认修改部分不会对其它功能造成影响。---上线之前很多轮的回归测试

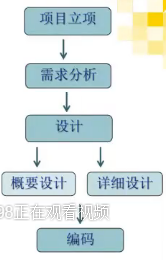
3.6.2冒烟测试：冒烟测试的对象是每一个新编译的需要正式测试的软件版本，目的是确认软件基本功能正常，可以进行后续的正式测试工作。---检验核心业务功能、主流程有没有实现

3.6.3探索性测试：是一种测试思维技术。它没有很多实际的测试方法、技术和工具，但是却是所有测试人员都应该掌握的一种测试思维方式。

**软件的生命周期及软件测试的工作流程**

1. 软件的生命周期

1.1软件生命周期（SDLC Systems Development Life Cycle ）是软件开始研制到最终被废弃不用所经历的各个阶段。---生命周期来自软件开发模型

1.2软件生命周期的阶段

1.2.1问题的定义及规划

主要确定软件的开发目的及其可行性，制定项目总体开发计划。

1.2.2需求分析

在确定软件开发可行的情况下，对软件需要实现的各个功能进行详细分析，明确客户的需求（需求评审---产品，开发，测试），输出需求规格说明书终版（原型图）。

1.2.3设计

把需求分析得到的结果转换为软件结构和数据结构，形成系统架构。

概要设计：主要是架构的实现，指搭建架构、表述各模块功能、模块接口连接和数据传递的实现等项事务。

详细设计：对概要设计中表述的各模块进行深入分析等，其中需要包含数据库设计说明。

1.2.4编码

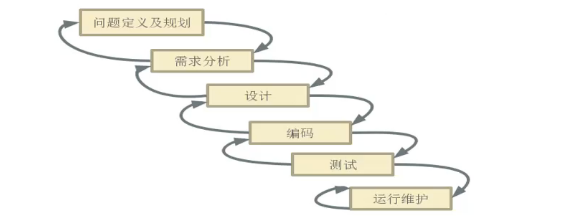
按照详细设计好的模块功能表，编程人员编写出计算机可运行的程序代码。

1. 软件开发模型

2.1瀑布型生命周期模型

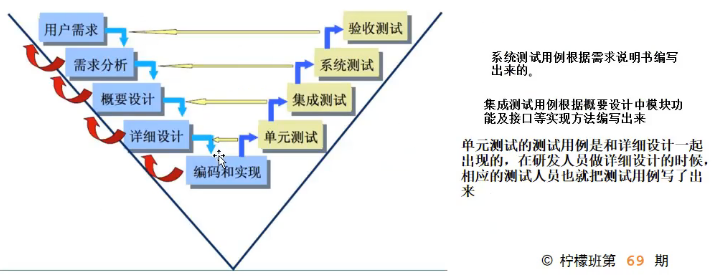
在1970年人类整理了第一个软件生命周期，即瀑布型生命周期模型也叫瀑布模型，规定了它们自上而下、相互衔接的固定次序，如同瀑布流水，逐级下落，具有顺序性和依赖性。每个阶段规定文档并需进行评审。

缺点：测试介入项目特别晚：回溯成本非常高



2.2 V模型

RAD（Rap Application Development,快速应用开发）模型是软件开发过程中的一个重要模型，由于其模型构图形似字母V，所以又称软件开发的V模型。它通过开发和测试同时进行的方式来缩短开发周期，提高开发效率。



优点：①测试介入早，可以提前对需求进行评审和测试----回溯成本减少

②测试提前准备测试文档（用例）----可以直接执行测试，节省准备文档时间----提高项目效率，周期拉短。

2.3敏捷开发模型

从1990年代开始逐渐引起广泛关注，是一种以人为核心、快速迭代、循序渐进的开发方法。强调以人为本，专注于交付对客户有价值的软件。是一个用于开发和维持复杂产品的框架。就是把一个大项目分为多个相互联系，但也可独立运行的小项目，并分别完成，在此过程中软件一直处于可使用状态。

例子：微信功能一部分一部分上线

微信：文字聊天，语音聊天，视频聊天，朋友圈，红包，小程序，零钱通，公众号===3年

第一次迭代版本：文字聊天，语音聊天====3个月，上线，拥有用户量，抢占商机

第二次迭代：视频聊天，朋友圈===2个月，用户量，吸引新用户

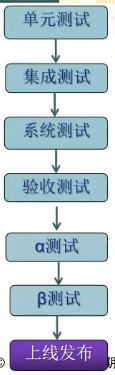
第三次迭代：红包，小程序，零钱通===3个月

……

产品不断的重复----完善，丰富

特点：弱化文档，强化沟通（站会：不需要多做准备的十分钟简短会议）。

问题1：项目周期多久？迭代周期多久？答：一周至一个月

三、软件测试阶段

软件设计完成后要经过严密的测试，以发现软件在整个设计过程中存在的问题并加以纠正。

测试的方法主要有白盒测试跟黑盒测试两种。建立详细的测试计划并严格按照计划执行。

3.1单元测试：主要是测试程序代码，为的是确保各单元模块被正确的编译，比如有具体到模块的测试，也有具体到类、函数、方法的测试等。----开发自测（白盒测试）

3.2集成测试：单元测试后，将各单元组合成完整的体系，测试软件单位之间的接口是否正确、数据能否正常传递。----接口测试

3.3系统测试：把软件系统搭建起来，按照软件规格说明书中所要求，测试软件其性能功能等是否和用户需求相符合，在系统中运行是否存在漏洞等。----最重要，常见的

3.4验收测试：主要就是用户在拿到软件的的时候，在使用现场，会根据前边所提到的需求，以及规格说明书来做相应的测试，以确定软件达到符合效果的。

四、运行维护

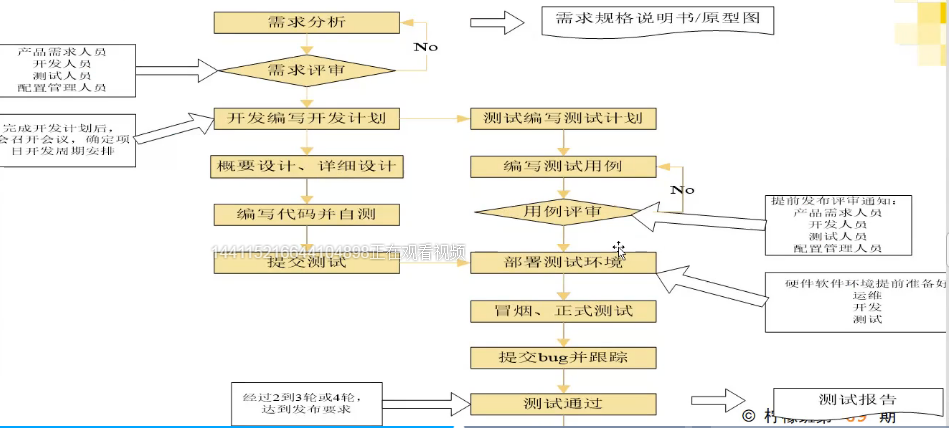
软件维护是软件生命周期中持续时间最长的阶段，在软件开发完成并投入使用后，由于多方面的原因，软件不能继续适应用户的需求，要延续软件的使用寿命，就必须对软件进行维护。软件的维护主要包括纠错性维护和改进性维护两个方面。

纠错性维护：bug修复，修改代码----新版本

改进性维护：优化，完善，改良----新版本（需求--立项--需求分析）

产品质量评估标准：①bug数量，严重级别；②测试用例执行率。-----出具测试报告

1. 软件测试工作流程



5.1测试需求分析阶段

阅读需求，理解需求，主要就是对业务的学习，分析需求点，参与需求评审会议。

5.2测试计划阶段

编写测试计划，参考软件需求规格说明书、项目总体计划，内容包括测试范围（来自需求文档）、进度的安排，人力物力的分配，整体测试策略的制定，和风险的评估与规避措施有一个制定，一般有测试负责人编写，当然我们可能也会参与相关的评审工作。

5.3测试设计阶段

主要任务是编写测试用例，会参考需求文档（原型图）、概要设计、详细设计等文档，有不明确的也会及时和开发、产品经理沟通，用例编写完成后会进行评审。

5.4测试执行阶段

首先搭建测试环境，执行预测（冒烟），以判定当前版本可测与否，如果预测通过，正式进入系统测试（2-4轮），遇到问题提交bug到缺陷管理平台，并对bug进行跟踪，直到被测软件达到测试需求要求，没有重大bug，测试结束。------完善测试用例

5.5测试评估阶段

出测试报告，对整个测试的过程和版本质量做一个详细的评估（剩余bug数量/严重程度，测试用例的覆盖率），确认是否可以上线。

5.6 UAT测试阶段

部署到UAT测试环境，由产品或者领导来验证功能。

测试阶段：测试需求分析阶段、测试计划阶段、测试设计阶段、测试执行阶段、测试评估阶段、UAT测试阶段。

测试工作流程（细化）：需求分析→需求评审→编写测试计划→编写测试用例→用例评审→部署测试环境→冒烟测试后正式测试→提交bug→测试通过→出具测试报告→发布上线

开发工作流程：需求分析→需求评审→编写开发计划→概要设计→详细设计→编写代码→自测→提交测试→修复bug

开发环境：开发自己的环境--自测（UT报告）--服务器（数据）--IP地址

测试环境：执行测试环境--服务器

生产环境：用户使用的环境（线上）

预发布：验收测试---尽量模拟用户环境

在测试环境后台添加的数据和信息在生产环境是看不到的

笔试题

1. 生命周期模型包含哪些阶段？你们开发的模型是什么？

答：生命周期模型包含的阶段：问题的定义及规划、需求分析、设计、编码、测试、运行维护

我们公司的开发模型是敏捷开发模型。

1. 测试流程包含哪些阶段？

答：测试需求分析阶段、测试计划阶段、测试设计阶段、测试执行阶段、测试评估阶段、UAT测试阶段。

面试题

1. 你们公司的开发流程是怎样的？

答：需求分析→需求评审→编写开发计划→概要设计→详细设计→编写代码→自测→提交测试→修复bug

1. 你们公司的测试流程是怎样的？各个阶段的输出是什么？

测试流程：需求分析→需求评审→编写测试计划→编写测试用例→用例评审→部署测试环境→冒烟测试后正式测试→提交bug→测试通过→出具测试报告→发布上线

输出的是：输出测试方案和测试用例、准备测试数据

1. 开发环境、测试环境、生产环境、预发布环境是什么？你在测试环境后台添加的数据和信息能够在生产环境看到么？

开发环境：开发自己的环境

测试环境：执行测试环境--服务器

生产环境：用户使用的环境（线上）

预发布：验收测试---尽量模拟用户环境

在测试环境后台添加的数据和信息在生产环境是看不到的

**测试需求分析**

1. 测试需要的内容

1.1测试需求是什么？

测试需求主要解决“测什么”的问题，一般来自需求规格说明书中原始需求。

测试需求应全部覆盖已定义的业务流程，以及功能和非功能方面的需求。（非功能：性能、界面、安全、兼容性、易用性）

1.2为什么需要软件测试需求？

只有明确了测试需求，才能知道怎么去测试？什么时候开始测？，要多少人测试？在什么环境上测试？----目的：提炼测试点，进行时间规划，人力规划，测试环境的准备。

1. 基本测试的思路

2.1明确一下这个项目是做什么的？基本业务逻辑流程；

2.2细化每一个功能，细化分析提取测试点；

2.3所有的细化模块的分析组合在一起，完成项目的测试点；-----功能

2.4非功能：界面、易用性、兼容性、安全、性能。

1. 案例分享

|  |  |
| --- | --- |
| 运行条件 | 用户未注册 |
| 输入 | 访问网站→点击注册 |
| 业务处理流程： | 主流程：   1. 单击“注册”按钮； 2. 用户输入手机号码、图片验证码、短信验证码、密码，勾选同意协议，点击“下一步”。   规则约束：   1. 图片验证码为4位字母或数字字母混合； 2. 短信验证码为4字数字，有效期为60s； 3. 密码长度为8~16位，数字、字母、符号至少包含2种。   其他流程：   1. 支持首页跳转； 2. 支持投标页面跳转。 |
| 输出 | 1. 弹框显示“欢迎加入蜂群”； 2. 选择“加入蜂群”，输入蜂群名称，点击确定，加入蜂群申请成功； 3. 选择“系统自动分配”，直接加入系统分配的蜂群； 4. 注册成功。 |

测试点思路步骤如下：正常+异常

3.1正常功能：是否可以正常提交---注册成功---单个功能进行冒烟测试；

3.2单个功能项验证（正常+异常）：

规则：按顺序从上至下，对每一个输入项进行验证。

3.2.1数据长度、数据类型验证、必填项验证、重复；

3.2.2限制约束验证；

3.2.3隐形需求：充分熟悉产品业务，挖掘隐性需求。

3.3功能交互验证

模块之间传递的信息和数据，对存在功能交互的功能项进行测试

3.4非功能性测试

界面、易用性、兼容性、安全性、性能压力

针对“注册”进行提炼测试点

|  |  |  |
| --- | --- | --- |
| 序号 | 验证项 | 测试点 |
| 1 | 注册 | 输入所有正确输入项，点击下一步，验证是否注册成功 |
| 2 | 手机号码 | 手机号码的长度是否符合要求 |
| 3 | 手机号码数据类型是否符合要求 |
| 4 | 手机号码是否为必填项 |
| 5 | 手机号码是否能重复（同一个号码重复注册） |
| 6 | 手机号码号段验证---举证1-2个不存在的号段 |
| 7 | 特殊手机号码（110）（不一定，和手机号码长度重复） |
| 8 | 国内外 境外卡验证 |
| 9 | 手机号段加前缀（86） |
| 10 | 图片验证码 | 图片验证码的长度是否符合要求 |
| 11 | 图片验证码数据类型是否符合要求 |
| 12 | 图片验证码是否为必填项 |
| 13 | 图片验证码时效性验证（刷新之前的） |
| 14 | 图片的验证码显示（初始显示，刷新之后更改） |
| 15 | 图片验证码是否区分大小写----产品确认 |
| 16 | 短信验证码 | 短信验证码的长度是否符合要求 |
| 17 | 短信验证码数据类型是否符合要求 |
| 18 | 短信验证码是否为必填项 |
| 19 | 短信验证码有效期验证 |
| 20 | 密码 | 密码的长度是否符合要求 |
| 21 | 密码数据类型是否符合要求 |
| 22 | 密码是否为必填项 |
| 23 | 密码显示是否保密状态 |
| 24 | 同意协议 | 是否可勾选 |
| 25 | 是否可以检查文档 |
| 26 | 勾选后是否跳转下一步 |
| 27 | 下一步 | 按钮能否点击 |
| 28 | 是否支持回车确认 |

另: 功能交互：注册用户是否能够正常登录（自动登录，手动登录）

非功能：界面、易用性、兼容性、安全性、性能压力

面试题：

1. 遇到隐性需求怎么办？

根据自身经验以及充分的熟悉产品业务的基础上面去分析，参考成熟的产品，最终无法确认的问题找产品经理确认需求。

1. 给你一个带有logo的水杯，你会如何去测试？
2. Logo是否清晰；
3. Logo图案、水杯形状是否美观；
4. 是否可以装水；
5. 有多大容量；
6. 是否方便拿取；
7. 是否耐摔
8. 是否耐高温
9. 是否保温
10. 是否遗留茶渍
11. 是否可以在微波炉加热
12. 你会如何去测试朋友圈，购物车等熟知的软件产品。

朋友圈：

1. 点击朋友圈按钮是否呈现朋友圈；
2. 点开朋友圈后是否可以刷新最新的朋友圈；
3. 是否可以划拉页面，查看之前发布的朋友圈；
4. 是否可以点开查看朋友发布的图片，图片是否清晰；
5. 是否可以点击播放朋友分享的音乐；
6. 是否可以给朋友发布的朋友圈点赞、评论；
7. 是否可以转发朋友发布分享的朋友圈；
8. 是否可以在朋友圈点击头像，呈现个人微信页面信息；
9. 是否可以点击朋友头像查看个人朋友圈内容；
10. 是否可以点击照相机呈现功能选择；
11. 点击拍摄是否可以拍摄；
12. 点击从个人相册中选择是否可以选择照片；
13. 点击取消是否恢复朋友圈页面呈现；
14. 选择好照片，编辑文字是否可以发布朋友圈；
15. 只有照片或者文字是否可以发布朋友圈；
16. 朋友圈呈现是否美观；
17. 朋友圈字体呈现是否清晰；
18. 点击朋友圈背景是否呈现更换相册封面和取消；
19. 点击更换相册封面是否可以从相册中选择照片更换背景；
20. 设置了查看朋友圈权限后，朋友圈是否按照设置范围呈现。

购物车：

1. 点击购物车按钮，是否可以呈现购物车页面；
2. 划拉购物车页面，是否可以查看更多购物车商品；
3. 购物车页面字体呈现是否清晰，页面呈现是否美观；
4. 购物车功能按钮呈现是否简单明了；
5. 选择商品是否可以加入购物车；
6. 添加到购物车的商品是否在购物车页面呈现；
7. 在购物车的商品是否可以点击查看详细页面；
8. 在购物车的商品是否可以点击更换尺码、种类，添加减少数量；
9. 购物车页面是否呈现的宝贝数量、收货地址，数量是否正确；
10. 购物车页面管理功能是否可以点击；
11. 点击管理功能是否可以勾选商品；
12. 点击管理功能是否呈现清理、移入收藏夹、删除功能；
13. 点击移入收藏夹商品是否收藏在收藏夹；
14. 点击删除功能商品是否被删除；
15. 购物车页面是否可以勾选部分商品，是否可以全选商品；
16. 勾选商品后点击结算是否可以跳转确认订单页面；
17. 购物车商品的领券是否可以点击打开；
18. 购物车商品的优惠券是否可以领取；
19. 勾选商品结算时结算金额是否减免优惠券金额；
20. 点击优惠明细按钮优惠金额是否呈现；
21. 是否可以在购物车添加大量商品。

**测试用例设计方法**

用例设计方法之四大金刚：

等价类划分法 边界值分析法 场景法 错误推测法（反推法）

软件测试的核心是测试用例的编写，是每个测试人员必须掌握的技能。

1. **等价类划分法**

1.1概念：等价类划分法是把所有程序输入域划分成若干个子集合（等价类），然后从每一个集合（等价类）中选取少数具有代表性的数据作为测试的输入数据。

在该子集合中，所有的输入数据对于揭露软件中的错误都是等效的。

----减少测试用例数量----提高效率

等价类划分有效等价类（正面，不会报错）和无效等价类（负面，抛出错误）。

举例：微信红包，金额区间：0.01~200 ----需求分析，设计测试用例

分析：

有效等价类：（1）【0.01，200】，（4）数字，（6）小数点后不超过2位，

无效等价类：（2）>200，（3）<0.01，（5）非数字（中文、字母、字符）（7）超过两位小数，（8）空值，（9）负数

1.2等价类划分法用例设计步骤和原则

1.2.1分析需求，先确定其有效等价类和无效等价类；

1.2.2在确立了等价类之后，建立等价类表，列出所有划分出的等价类；

1.2.3再从划分出的等价类中选择测试用例。

（1）设计一个新的测试用例数据，使其尽可能多的覆盖尚未被覆盖的有效等价类，重复这一步，直到所有的有效等价类都被覆盖为止；减少测试用例的数量，避免重复，提高效率。

（2）设计一个新的测试用例数据，使其仅覆盖一个尚未被覆盖的无效等价类，重复这一步，直到所有的无效等价类都被覆盖为止------为了确定是哪个因素触发错误，每一种错误都被正确处理。

等价类划分法的应用场景：当测试需要数据量过大，且数据操作可以分类时进行等价类划分。

1. **边界值分析法**---等价类划分法一起使用

2.1定义：边界值分析法是对等价类划分法的一个补充，边界值一般都是从等价类的边缘值去寻找。

2.2原则和步骤：确定边界：应当选取正好等于、刚刚大于或刚刚小于边界的值作为测试数据----范围相关

\*有效等价类的边界 \*无效等价类的边界

两点法 三点法 四点法

注意：次边界值：IP地址（0~255），时间规格（0~23）,2的幂值（256,1024,65535）----需求没有明说，常识

特殊边界值：0是一个特殊值，负数，空值，空格等。

2.3边界值的作用：人们从长期的测试工作经验得知，大量的错误是发生在输入或输出范围的边界上，而不是在输入范围的内部。因此针对各种边界情况设计测试用例，可以查出更多的错误！----提出更多bug

边界值的应用场景：如果需求规定了取值范围或规定了取值的个数时，可利用边界值进行测试。

2.4等价类划分法/边界值分析法常见运用场景

2.4.1输入条件规定的取值范围或值的个数的情况（类似最小<X<最大、最小<X、最大>X），比如用户名长度、红包金额数值输入范围。

2.4.2在下拉列表包含多个选项的情况，比如城市下拉选项（第一个、最后一个、中间一个）；

2.4.3在规定了输入数据必须遵守的规则的情况下，可确立一个有效等价类（符合规则）和若干个无效等价类（从不同角度违反规则）；

以上除了等价类之外同时会涉及边界值的分析。边界值还包括以下：

1. 报表数据的第一行、最后一行、中间一行；----边界值
2. 屏幕上光标在最左上、最右下位置。

例子（承接上例微信红包）：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 输入条件 | 有效等价类 | 边界值 | 无效等价类 | 边界值 |
| 红包金额 | 1）0.01~200 | 0.01,0.02,199.99,200 | 2）小于0.01 | 0 |
| 4）不能超过两位小数 |  | 3）大于200 | 200.01 |
| 6）必须数字 |  | 5）超过两位小数 | 0.009 |
|  |  | 7）非数字，字母，字符，中文 |  |

|  |  |  |
| --- | --- | --- |
| 测试用例描述 | 测试输入 | 测试覆盖 |
| 输入的红包金额0.01~200之间的不超过两位小数的数字 | 0.01,0.02,199.99,200 | 1,4,6 |
| 红包金额小于0.01 | 0 | 2 |
| 红包金额大于200 | 200.01 | 3 |
| 红包金额超过两位小数 | 99.009 | 5 |
| 红包金额非数字 | lemon！中 | 7 |

1. **场景法**

3.1什么是场景法？

通过场景描述的业务流程（业务逻辑），也包括代码实现逻辑，设计用例来遍历场景（路径），验证软件系统功能的正确性。

3.2如何使用场景法

3.2.1画出流程图----产品需求文档---画好了，需要测试自己画

矩形：表示步骤（操作、输入、输出结果）

菱形：判断条件----是、否

箭头：流向

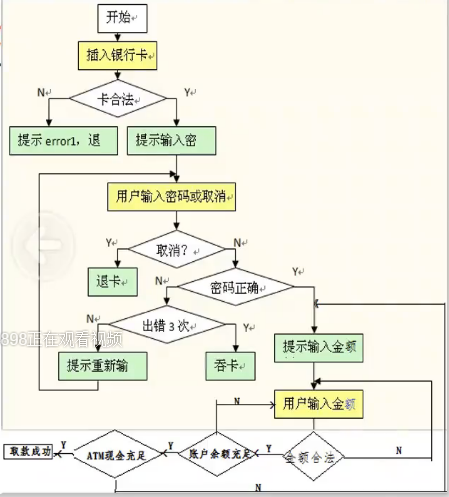
3.2.2遍历场景，提取测试用例。

1. 覆盖正常的路径---取到钱的路径----判断的地方----Y
2. 走每一个分支----判断的地方----找菱形-----N
3. 出错步骤重新回到主流程，建议多走一步正确的步骤。

注意：场景法的重点是测试流程，因此每个流程一个用例验证即可，流程测试没有问题并不能说明系统功能没有问题了，还需要针对单步的功能进行测试。

只有单个功能点和流程测试，才算是充分的测试+等价类、边界值-----细化测试

**场景法举例：**



场景一：输入合法银行卡，输入正确的密码，输入正确且充足的金额，ATM足够----取到钱；

场景二：插入不合法的卡，退卡提示错误；

场景三：插入合法银行卡，输入密码之后取消----退卡；

场景四：插入合法银行卡，输入错误的密码之后不取消，不超过3次----提示密码错误，重新输入密码；

场景五：插入合法银行卡，输入错误的密码之后不取消，出错3次-----吞卡；

场景六：插入合法银行卡，输入正确的密码之后不取消，舒输入不合法金额----提示错误，重新输入；

场景七：插入合法银行卡，输入正确的密码之后不取消，输入合法金额，账户余额不足，---提示错误，重新输入；

场景八：插入合法银行卡，输入正确的密码之后不取消，输入合法金额，账户余额充足，ATM不足余额----提示错误，重新输入。

1. 错误推测法（反推法）

基于经验和直觉推测程序中所有可能存在的各种错误，从而有针对性的设计测试用例的方法。

它的要素有三点，分别为：经验、知识、直觉。

考虑程序可能触发错误场景---不能正常运行

不单独使用-----可以作为其他方法的补充。

案例：某平台登录页面

既然是用错误猜测法，那么我们首先列出可能导致结果出错的情况----登录失败。

1. 账号密码错误；（异常字符输入/为空）
2. 验证（图片、短信）
3. 网络问题
4. 浏览器兼容性
5. 性能弱（并发大量用户）
6. 账号黑名单---举报
7. 登录失败错误次数---冻结账号
8. 服务器异常----无响应
9. 第三方登录问题
10. 单点登录---不支持多个服务器登录

**软件测试用例的编写**

1. 什么叫软件测试用例？

什么是测试用例

测试用例是为项目需求而编制的一组测试输入、执行条件以及预期结果，以便测试某个程序是否满足客户需求。----测试依据

可以总结为：每一个测试点的**数据设计**和**步骤设计**。

1. 测试用例的重要性（了解）

2.1测试用例是软件测试的核心

软件测试的重要性是毋庸置疑的，测试用例是测试工作的指导，是软件测试质量稳定的根本保障。

2.2评估测试结果的基准

测试用例的通过率以及错误率，是测试结束的一个重要依据，用来判断该软件测试结果是否通过，能否达到上线的标准。

2.3保证测试的时候不遗漏测试功能点

可以在测试人员疲累的时候起到一个牵引的作用。

2.4在编写测试用例的过程，可以熟悉需求，对系统架构或者业务流程有一个整体的，深入地了解。----深入测试，提bug

3.测试用例的八大要素（重点！）

3.1用例编号：产品名----测试阶段-----测试项--XXX（英文/拼音首字母缩写）

3.2测试项目：对应一个功能模块（细化功能）----子项目

3.3测试标题：直接对测试点进行细化得出，输入内容+结果，同一功能模块标题不能重复（来自测试点）-----建议一行写一个测试点，细致，数量越多

3.4重要级别：高（核心功能）/中（次要---异常）/低（界面，不常用场景）----high、mediun、low----P1，P2， P3， P4 ，P5 ----冒烟测试（P1），回归测试（P1，P2）

3.5预置条件：需要满足一些前提条件，否则用例无法执行-----不必须

3.6测试输入（数据）：需要加工的输入信息，根据具体情况来设计（跟步骤结合起来一定要具有指导性意义）

3.7操作步骤：明确给出每个步骤的描述，执行人员可以根据该步骤完成执行工作----要详细（小白也能看懂）

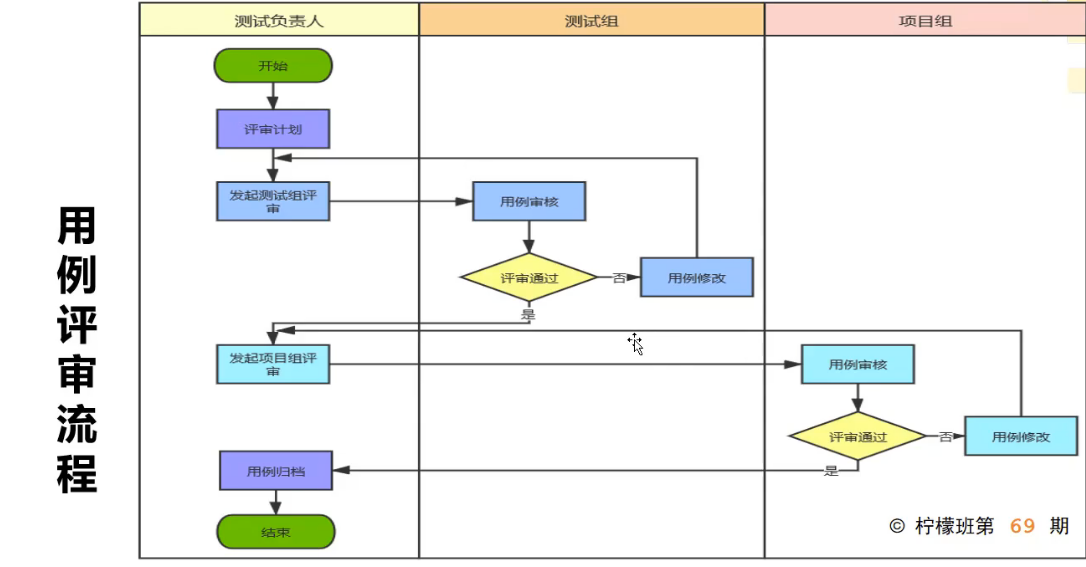
3.8预期结果：根据预期输出比对实际结果，来判断被测对象是否符合需求。（预期结果唯一，不能出现“是否”、“或者”’）

3.9实际结果

举例：注册功能

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 用例编号 | 测试项目 | 测试标题 | 重要级别 | 预置条件 | 测试输入 | 测试步骤 | 预期结果 | 实际结果 | 备注 | 版本号 |
| ZC-001 | 注册 | 输入所有正确输入项，检查是否注册成功 | P1 | 1. 网络正常 2. 系统正常访问 | 1. 网址：<http://weixin.com> 2. 手机号码：13434567890 3. 图片验证码：   58JD   1. 短信验证码：   585858  5.密码：liuyao1228 | 1.输入网址，点击注册按钮；  2.输入手机号码；  3.输入图片验证码；  4.输入短信验证码；  5.输入密码；  6.勾选同意协议；  7.点击下一步；  8.检查一下注册页面 | 1.跳转到注册页面；  2.手机号码验证通过  3.图片验证码验证成功  4.短信验证成功  5.密码验证成功  6.勾选验证成功  7.下一步点击成功  8.注册页面成功 | Failed/ Passed  通过/  不通过 | 没有执行原因，  失败bug编号 | 产品编号 |

4.用例评审流程



1. 测试用例的变更

由于需求变更，对于业务的不断深入了解和测试用例评审，测试用例是无法一次全部写好的，所以，测试用例在完成之后是不断需要修正的。

测试用例变更通常包括：

5.1需求变动

5.2执行完成后的用例完善

5.3评审后的用例修改

PS:一定要记得备份！（原1.0版本备份，更新后的1.1版本也需要备份）

笔试面试题

1.用例需要评审么？紧急情况用例也需要评审么？

答：用例需要评审，紧急情况下可以进行简单的内部评审。

2.如果被测项目很紧急，来不及写用例，怎么办？

答：先列测试点，后续进行补充测试用例。

3.遇到隐性需求如何写用例？（需求不明确）

答：1.根据自身经验，充分去熟悉产品2.参考成熟产品3.找产品确认

4.用例有没有优先级？如果一定要有优先级，根据什么来确定呢？

答：有的，优先级根据功能重要性来划分的。

5.如何编写测试用例？（此项目为基础来讲一个小模块 用例设计，手机号）

答：从以下几个方面来编写测试用例：

1. 用例编号 2)测试项目 3）测试标题4）重要级别 5)预置条件 6）数据输入7）操作步骤：

8）预期结果9）实际结果

**bug的定义＆生命周期＆状态处理＆bug处理系统禅道的使用**

1. bug的定义

你们觉得bug是什么？

软件的bug，狭义概念是指软件程序的漏洞或缺陷。

广义概念除此之外还包括测试工程师或用户所发现和提出的软件可改进的细节(增强性、建议性)，或与需求文档存在差异的功能实现等。

我们的职责就是，发现这些bug， 并提交给开发，让开发去修改。

1. bug的类型---了解

要确定一个bug的类型，需要对项目（或产品）有比较深的理解。这个划分对于开发定位问题影响很小，但对于问题类型的统计就比较重要了。---测试报告

常见的bug类型划分（禅道系统为例，可自定义）：

代码（功能）错误---最常见，优先级偏高

界面优化----UI设计，优先级偏低

设计缺陷---优化建议：需求上就不合理的---优先级偏低

按照公司具体的规定来分类

1. bug的等级---优先级

Bug等级，这个划分有分三级或四级，也有分五级的。如果是等级越高（严重），那么可能被修复的等级也会高一些，然后有些公司还会根据你提的bug数量和bug等级来考察你的绩效。很多情况下，我们提交bug大致的等级差不多即可，没有严格区分。

如何来判断bug的等级（严重），一般可以参照下面的判断条件：

3.1致命错误----blocker

3.1.1常规操作引起的系统崩溃、死机、死循环、闪退；

3.1.2造成数据泄露的安全性问题，比如恶意攻击造成的账户私密信息泄露；

3.1.3涉及金钱计算----公司巨大损失、业务；

3.1.4阻断性测试，所有测试工作进行不下去（冒烟测试）---比如登录；

3.1.5权限问题----爱奇艺会员与非会员的权限区分。

3.2严重错误----critical

3.2.1重要功能不能实现；

3.2.2错误的波及面广，影响到其他重要功能正常实现----12306购票系统--咨询；

3.2.3非常规操作导致的程序崩溃、死机、死循环、闪退；

3.2.4外观（界面）难以接受的缺陷；

3.2.5密码明文显示：前端处理bug，后端--服务器（数据库验证）---安全；

3.2.6偶现的致命性bug。

3.3一般错误----major--遇到最多

不影响产品的运行，不会成为故障起因，但对产品外观和下道工序影响较大的缺陷。

3.3.1次要功能不能正常实现；

3.3.2操作界面错误（包括数据窗口内列名定义、含义不一致）；

3.3.3查询错误，数据错误显示；

3.3.4简单的输入限制未放在前端进行控制；

3.3.5删除操作未给出提示---友好型；

3.3.6偶现性的严重性bug。

3.4细微错误----minor

程序在一些显示上不美观，不符合用户习惯，或者是一些文字的错误---用户体验

3.4.1界面不规范；

3.4.2辅助说明描述不清楚；

3.4.3提示窗口文字未采用行业术语；

3.4.4界面存在文字错误。

3.5改进建议----enhancement--新需求，下一个版本

可以提高产品质量的建议，包括新需求和对需求的改进。

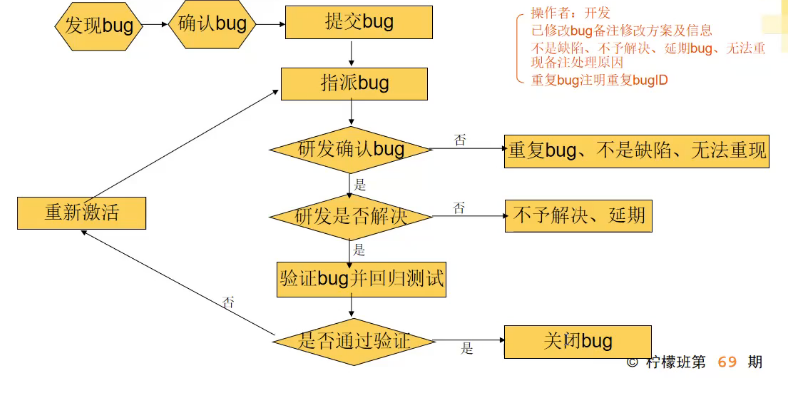
1. bug的生命周期（管理流程）！！！（重点）

这个是面试/笔试过程中经常会被问到的问题。bug的生命周期，就是一个bug被发现到这个bug被关闭的过程。

生命周期中一般缺陷状态：发现--新建（提bug）--指派--已解决--待验--关闭。--正常

如果待验的bug在验证时没有解决好，我们需要重新打开（激活）--指派--已解决--待验，循环这个过程，中间其他状态：拒绝、延期等。

1. bug的跟踪管理流程

5.1发现bug--先确认：防止环境问题，操作问题等外因---无效bug  ==专业性  
new（新建）提bug---》 （提bug者，测试老大）指派（开发，开发老大） ： 跟进（3天，push（推进）开发修改）  
5.2重复bug（duplicated）（233）：要求开发备注一下重复bug 的号（ID：122），测试确认--是== 加备注，关闭； 否== 重新激活（reopened）--指派开发 ；  
5.3不是缺陷（invalid）：by-design（设计如此），操作失误，需求的理解不一致  == 争论：（1）分析需求，从用户角度出发没找到证据--尝试说服开发 ；（2）产品（项目经理）--拍板== 是bug--开发修复； 不是--测试不纠结，留好证据（邮件据图，备注bug）；  
5.4无法复现（un-reproduced）：（1）开发复现： 确认一下测试环境再复现出来 ==可以--帮助开发复现，测试环境调试定位；（2）测试和开发环境都无法复现：尝试跟踪3-5个版本（10次+）--- 加备注（复现次数，跟踪版本数）--关闭 ==记录到小本本，研究  
5.5不予解决（wontfix）： bug级别低 == UI，minor，enhancement bug；1） 说服 开发  2） 产品确认   == 备注留证据，关闭  
5.6延期（delayed）：（1）建议性（feature/enhancement）-- 下一个版本需求 ；（2）上线之前，影响比较大（性价比）  == 分析，1）分析这个bug用户影响大小，建议他修复； 2）严重级别    === 产品、项目经理最后确认  ==不修复，备注（留证据） ==不关闭，挂起 

5.7已修复（resolved-fixed）：--测试验证  （ bug步骤，结果检查）=== 1） verified （已验证）-- closed（关闭）  2）仍然存在：reopened （重新激活）

1. bug的跟踪管理--状态处理

6.1已经指派的bug---已经指派给开发的，请大家注意自己bug的走向，随时关注并进行跟踪！如果一直未修复，提醒开发修改，以免开发忘记；如果已经修复等待测试环境更新后进行验证。催着改bug。

6.2已解决的bug---等待测试环境更新后进行验证，验证通过则关闭，验证不通过则重新打开指派给开发。

6.3重复bug---先去查看下是否跟开发指定的bug重复？如果确定的重复则关闭；如果不重复，说明原因，重新打开指派给开发。

6.4不是缺陷---再次依据需求确认，是否是bug，如果依然觉得是缺陷跟开发沟通，列举出来觉得是bug的点，沟通未达一致找产品确认，确认是bug注明情况并再次指派给开发，产品确认不是bug，就不纠结，直接关闭bug，但是会拿小本本把这个bug记录下来，等到测试任务结束后，再来研究研究。

6.5无法重现---确认开发环境是否跟测试环境一致？包括操作步骤、浏览器、环境、特定账号、输入数据等，如果多个版本验证之后，如开发所说重现不了，依据bug的严重程度跟产品、开发一起确认关闭；如果找到重现原因，注明清楚并再次指派给开发。

6.6不予解决---找产品经理进行确认，确认不予解决进行关闭；确认需要解决请备注原因并打开指派给开发。

6.7设计如此---找产品经理进行确认，确认设计如此进行关闭；确认是问题，备注原因重新指派给开发。

6.8延期修改---请看下bug严重程度，是否影响当前版本发布？与产品经理进行确认，不予延期请根据情况进行激活与情况说明；确认延期则组好记录，后续版本进行关注--不关闭。

1. bug的跟踪管理---缺陷管理工具

常见的缺陷管理平台:

禅道，我们现在做项目用的就是这个

bugzilla、jira：都还不错，也比较强大，但是搭建起来很困难

Bugfree

Readmine

Easybug:免费开源，在线网站类型的

Mantis:这个还可以用

QC、TD

不管是开源还是商业的缺陷管理工具，它们本质都是一样的，用来管理bug的生命周期。掌握其中一款工具，自然就会用其他的，稍微有一点点区别的，别人加以指点，就可以明白了。

1. bug的跟踪管理---如何提交bug

发现bug后，接下来你提交到bug管理平台，提交一个bug包含哪些内容？

bug标题——标题要清晰简洁，**写明bug描述**；如果没有选择功能模块，最好在标题中标注功能模块。让查看bug的人员清楚知道你所表达的意思。bug的**功能模块**+**bug的操作**+**bug的结果**。

重现步骤——详细写下发现bug的测试过程。能指导开发重现这个bug，附上测试数据。

**实际结果**——出现bug的结果，**粘贴bug截图**、日志截图---直观、证据（有图有真相）

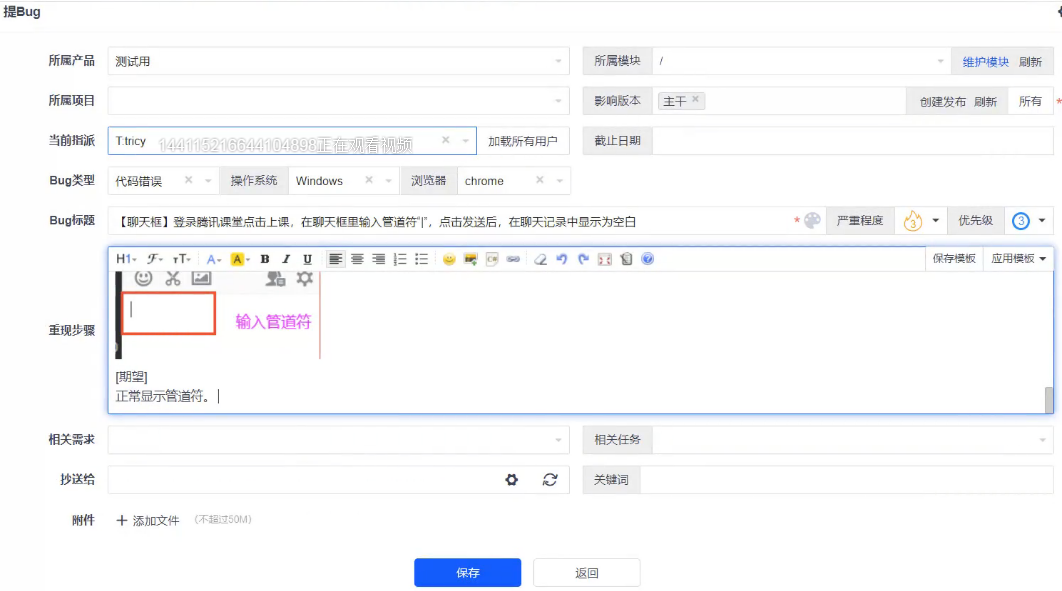
**预期结果**——记得写清楚预期---来自于测试用例中的预期结果。

bug类型和严重程度——便于后续测试结果分析，bug的统计。

Bug测试环境——例如：什么系统，哪个版本等。兼容性问题、难以重现问题。

附件——日志文件，文件**测试数据**。图片、崩溃日志文件等。（太多压缩包上传）

所有以上，参考公司前辈写的bug，依葫芦画瓢，拓展测试思维。



笔试面试题：

1. 有没有你印象深刻的bug？bug的原因/bug当时怎么解决？

等做完项目回答

1. bug的生命周期（笔试）

不是bug→关闭bug

↑

发现bug→确认bug→提交bug→开发修改bug→关闭bug

↓

开发不处理bug→沟通原因→找产品确认→不是bug→关闭bug

↓

是bug→激活重新指派开发→修改bug→关闭bug

1. 当你开了一个bug，但是开发不认为是bug，如何处理？
2. 分析需求，从用户角度说服开发；
3. 找产品经理确认，产品经理说是，就指派开发修改，产品经理说不是，就做好标记，关闭bug。
4. 你在发现bug并确认bug的过程中，对于复现率不高的bug怎么处理？

确认开发环境是否跟测试环境一致？包括操作步骤、浏览器、环境、特定账号、输入数据等，如果多个版本验证之后，如开发所说重现不了，依据bug的严重程度跟产品、开发一起确认关闭；如果找到重现原因，注明清楚并再次指派给开发

**软件测试计划&软件测试报告**

**一．软件测试计划简介**

为什么要写软件测试计划？一般是主管来写，测试计划是在做完需求分析后，整个测试工作开始之前做的一些准备计划工作，“5W+1H”去记忆。

一般包括以下内容：目的（why）、测试范围（what）、测试进度安排（when）、测试人员（who）、测试环境（where）、测试方法+测试工具（how），风险评估。

可能会有的同学问，为什么要提到测试工具？因为像一些访问量比较大的公司，可能要进行性能测试,那么就需要用到测试工具。

公司一般会有自己的软件测试计划模板，下面我们来看下软件测试计划的模板：

**微信v6.5\_ST测试计划**

1. **简介--why**

1.1项目信息

|  |  |  |  |
| --- | --- | --- | --- |
| 项目编号 | 项目经理设置 | | |
| 项目名称 | *项目经理取好名字* | | |
| 项目经理 | L+1（管理项目的人） | 业务负责人 | 潇洒哥（产品） |
| 技术经理 | 另一个自己（开发老大） | 测试负责人 | 努力（测试老大） |

1.2测试参考文档

《XXX需求说明书》

1. **测试范围 -- what**

2.1需要进行功能测试的模块及功能点

重点测试范围(参考需求文档) ====不等同于测试用例

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **产品** | **模块** | **子模块** | **功能** | **测试点** | **优先级** | **负责人** |
| 微信 | 登录 |  |  | 1、正常注册功能验证  2、长度、类型、为空、重复异常情况验证 |  | 对应开发 |
|  |  |  |  |  |

2.2测试优先级及先后顺序

2.2.1测试顺序

按照测试范围，顺序验证重点测试范围以及各个功能点。

2.2.1性能测试

性能测试不在本次ST功能测试范围中。

2.2.2安全性测试

安全性测试不在本次ST功能测试范围中。

**3.测试策略**

测试轮次安排：

冒烟测试：主要对主业务、主功能流程做验证，以执行结果正确为通过标准，所有冒烟测试案例通过则为冒烟测试通过标准；--- 用例重要级别 （优先级）

ST第一轮：覆盖所有案例；---- 发现bug （严重bug，影响功能）--催促修复bug

ST第二轮：覆盖所有案例，测试完毕后如果遗留缺陷或遗留问题等级及数量在允许范围内，则测试结果达标，转入UAT测试。否则进入第三轮。测试准出为关闭缺陷，案例100%（95%）执行通过。

用例组织：

（1）测试用例按照测试执行先后顺序进行编写，编写完成后发给开发人员、业务人员进行用例评审；--可靠性

（2）冒烟测试案例根据系统核心功能进行设计，保证开发提供版本的可测性。

**4.测试进度 --when**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **测试**  **阶段** | **测试活动** | **计划开**  **始时间** | **计划结**  **束时间** | **交付产品** |
| 测试  计划 | 熟悉需求和设计 |  |  | **/** |
| 编写计划 | 20200418 | 20200418 | 《XXX需求\_ST测试计划》 |
| 测试  准备 | 需求及用例设计 | 20200419 | 20200421 | 《XXX需求\_测试案例》 |
| 用例评审 | 20200422 | 20200422 | 《XXX需求\_测试案例》 |
| 搭建环境/验证 | 20200423 | 20200423 | / |
|  | 冒烟测试 |  |  |  |
| 测试  执行 | 第一轮执行 | 20200423 | 20200428 | Bug（执行结果的用例） |
| 第二轮执行 | 20200429 | 20200431 | Bug（执行结果的用例） |
| 测试  总结 | 测试报告编写 | 20200501 | 20200501 | 《XXX需求\_ST测试报告》 |

本项目所需工作量：工作量为测试设计工作量、测试执行和测试总结工作量的总和，以人月或人日计。软件测试工作量应为开发工作量的30%-40%。

**5.测试资源 --- who + where**

5.1人力资源 --任务分配

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **角色** | **人员** | **工作内容** | **起止时间** | **预计工作**  **量(人天)** |
| 测试主管 |  | 如：熟悉需求、确认测试范围 |  |  |
| 测试主管 | 努力 | 如：编写测试计划 |  | 1人/天 |
| 测试组 | Niki 凝凝  摆渡人 | 编写测试用例：  A:注册登录  B：聊天 |  | 3人/天 |
|  |  | 用例评审 |  |  |

5.2测试环境

|  |  |
| --- | --- |
| 需求名称 | 详细要求 |
| 操作系统 | Windows 10,8.1 ，7 |
| 运行环境 |  |
| Web浏览器 | Chrome Firefox |
| … |  |

（1）甲方爸爸需求项目：按照甲方爸爸的要求来准备测试环境

（2）自主研发项目（产品+老板）： 开发和产品沟通 -- 测试环境

原则：尽可能要跟客户环境保持一致。

硬件环境：服务器 --运维准备

|  |  |  |
| --- | --- | --- |
| 名称 | 配置 | |
| 基础配置 | Cpu |  |
| 内存 |  |
| 硬盘 |  |

**6.测试风险管理**

根据本软件产品的实际情况，填写测试风险列表，分析本软件测试过程中可能出现的风险并采取相应的措施。

**举例**：

1.人员调岗或离职 ：备份（调配人员）

2.需求变更 ：时间的预留，规划公司流程

3.问题太多，bug 太多，阻塞测试 ：时间的预留，安排； 开发--承担责任

按时 按质量发布，人员 + 时间 调配，申请延期发布

**二．软件测试报告简介**

软件测试的五个阶段中：评估阶段（测试人员必须会的）

公司一般都有自己的软件测试报告模板，看下我们的软件测试报告模板

在进行完整个测试工作后，你就要对测试产品做一个总结，提交给你老大或者是研发部门负责人，项目经理。

包括的内容：测试范围、测试环境、**遗留的bug有哪些，测试用例覆盖率有多少，bug的统计与分析**，风险有哪些，**版本测试评估，发布的建议**。

一般公司有自己的报告模板，照写就好，没有，可以用老师提供的模板。

**测试报告**

**1．引言部分**

1．1项目背景

本测试报告的具体编写目的，指出预期的读者范围。(3-4句)

项目描述 （项目内容，用户需求）

本测试报告为（系统名称）系统测试报告；本报告目的在于总结测试阶段的测试

及测试结果分析，描述系统是否达到需求的目的。

本报告预期参考人员包括测试人员、测试部门经理、项目管理人员、SQA人员和其他质量控制人员。

1．2参考资料

《需求文档》

《xxx项目\_系统测试计划》

《xx项目\_测试用例》 《缺陷记录》

**2．测试基本信息**

2．1测试范围 --针对项目真实执行情况做修改

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **产品** | **模块** | **子模块** | **功能** | **测试点** | **优先级** | **负责人** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |

2．2测试案例设计思路

根据上述测试范围测试点进行测试用例的设计。

|  |  |
| --- | --- |
| 功能测试 | 依据等价类划分、边界值分析、错误推测、场景法等设计用例 |
| 界面测试 | 满足效果图设计，保证页面风格、排版整体等一致 |
| 兼容性测试 | 满足IE9、火狐、谷歌浏览器兼容 |

**3．测试结果及缺陷分析--重点！！！**

3．1测试执行情况与记录

3．1．1测试组织

|  |  |  |  |
| --- | --- | --- | --- |
| **项目经理** | **软件工程师** | **测试工程师** | **业务负责人** |
| L+1 | 另一个自己（开发） | 努力 | 潇洒哥 |

3．1．2测试时间 --真实执行

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **测试**  **阶段** | **计划开**  **始时间** | **计划结**  **束时间** | **实际开**  **始时间** | **实际结**  **束时间** | **计划工作量（人天）** | **实际工作量（人天）** |
|  |  |  |  |  |  |

3．1．3冒烟情况

|  |  |  |  |
| --- | --- | --- | --- |
| 冒烟  测试 | 时间 | 是否通过 | 如不通过，请写原因 |
| 0.5天 | 不通过 | 因为微信文字聊天发送失败 |

3．1．4测试用例统计

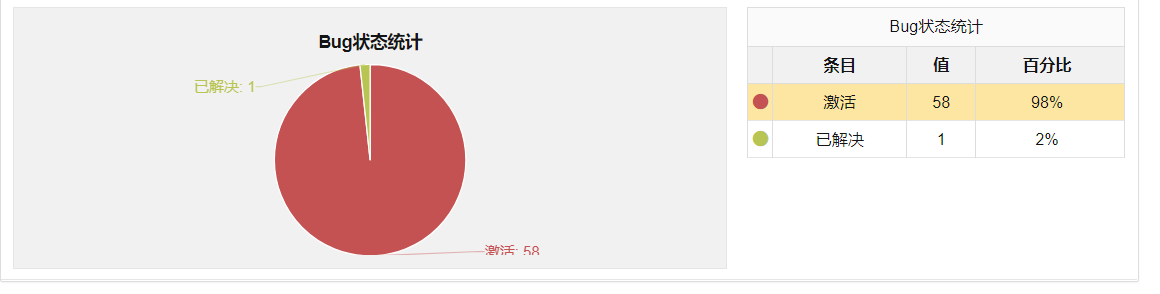
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 案例总数 | 执行个数 | 成功个数 | 失败个数 | 未执行个数 | 案例成功率 |
| 1000 | 800 | 600 | 200 | 200 | 600/800=75% |

用例覆盖率：800/1000 = 80% （95%）--充分

3．2缺陷的统计与分析 --- bug管理工具--禅道

**缺陷汇总：--bug数量现状**

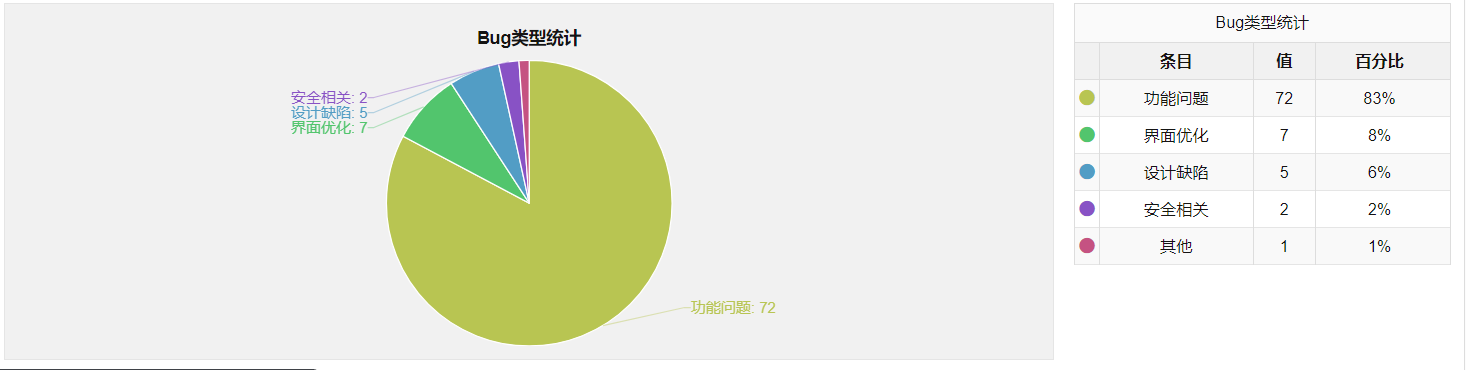
总缺陷数：384， 已解决：340，（未解决：44）延期：20 不予修改：24



结论分析：项目总工开了384 bugs，其中还是剩余44个未修复。经过跟开发和产品确认，20个确定延期，经过开发和产品商量确认，24优先级比较低，不予修复。

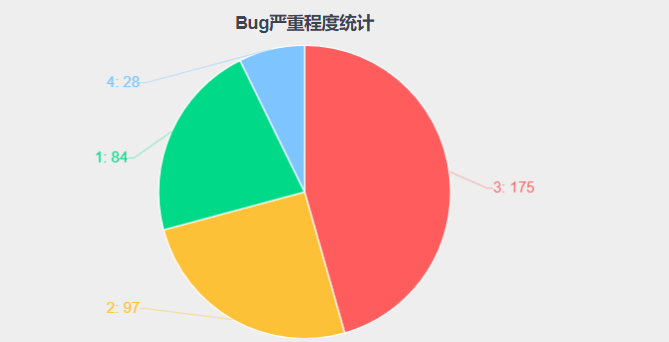
缺陷分析：

**按缺陷类型统计：**



结论分析：从以上数据得出，大量bug类型为功能问题，其次是界面问题，只有1个是性能问题。

**按严重程度统计：**



结论分析：大部分bug 都是第三级别的bug，其次是critical bug，blockers 级别的bug 占了84个。

--- blocker critical 很多：开发没有自测 （能力 + 态度）

**按功能模块统计：**--不同模块 ， 不同开发



分析：功能复杂的大模块（功能多），bug 比较多。--正常分布

**按测试阶段统计：**

分析：

第一轮测试 ，bug最多；按照版本递增，bug数依次减少。

第一轮：200， 第二轮：（300）80，第三轮：40.

数据解释：1）新需求，需求变更--- 流程把控 2）开发修复bug引起的回归问题

Bug- reopened（开发没有修好），regression（回归--开发） ，后期新bug--

测试前期漏测的

残留缺陷与未解决问题：（延期，不修复，new）--领导看

|  |  |  |  |
| --- | --- | --- | --- |
| Bug id | Bug描述 | 状态 | 未解决说明 |
| 3384 | Xxx | 延期 | 不影响发布，延期到下一版本解决 |
| 3385 |  | 不修复 | 界面的问题，可以不修复 |
|  |  |  |  |
|  |  |  |  |

**4．测试结论与建议**

4．1风险分析及建议

1）测试主要覆盖谷歌浏览器，其他浏览器兼容性未覆盖到，会存在兼容问题；建议客户使用谷歌。--产品和项目经理确认

2）因为时间的问题，经过跟项目经理确认，我们没有进行第三轮回归测试，可能会有一些风险。（优先级高中的用例）

无

4．2测试结论

本项目根据业务需求及开发人员的反馈意见，覆盖了所有的测试需求及案例（80%--原因），均已在ST环境测试完成，有效案例一共 xx个，执行率 xx%,，成功率 xx%，缺陷关闭率为xx%，目前缺陷均已修复并回归关闭；（一、二级bug全部修复并关闭，少于4%的三级、四级bug存在延期，不影响用户基本操作）

综上所述，xx项目达到ST项目测试出口标准，本项目ST测试(通过/不通过)，可以进行验收测试/发布

**5．交付文档** --项目结束后交付的文档--归档

《xxx项目\_系统测试计划》

《xx项目\_测试用例》

《xx项目\_ST测试报告》

《缺陷记录》--文档

笔试面试题：

1. 写过测试报告/计划么？内容是什么？

写过，测试报告内容是：测试范围、测试环境、遗留的bug有哪些，测试用例覆盖率有多少，bug的统计与分析，风险有哪些，版本测试评估，发布的建议。

1. 报告中的结果是怎么分析的？

先进行缺陷汇总，然后按缺陷类型统计，按严重程度统计，按功能模块统计，按测试阶段统计。

1. 你认为测试报告的侧重点？

bug的统计与分析

**Git的基本概念和使用**

1. Git是什么？

·Git：是一个开源的分布式版本控制系统，可以有效、高速的处理从很小到非常大的项目版本管理。

·GitHub：全球最大的面向开源及私有软件项目的托管平台，免费注册并且可以免费托管开源代码。

GitLab：与GitHub类似，也是属于第三方基于Git开发的产品，不同的是：GitLab可以部署在自己的服务器上，代码数据都是掌握在自己手中，适合公司内部团队开发。

1）相当于同步到服务器，对版本的控制--版本控制管理服务器--开发（提交代码）

开源：ios--苹果 Android(谷歌开发)--华为、小米

2）大仓库--存放项目---私人用多

3）仓库-公司项目---基于企业使用

2.Git和SVN的区别

·SVN版本集中管理，所有的代码都在一台服务器上；分支和主线是不一样的路径。

·Git分布式管理，去中心化，服务器和每个开发人员都拥有一个本地的代码管理仓库；分支和主线路径是一样的。

注意：本地仓库，远程仓库。

总结：Git比SVN更强大，大部分公司都在用。

svn：subversion --SVN

区别：git有很多服务器，svn只有一个服务器

3.公司都是如何使用Git的？

1）安装好git工具：

·git---命令行模式操作

·tortoiseGit---图形化界面操作

1. 公司会有一个服务器，（多个仓位，一般项目<-->仓库）,针对不同的人账号有不同的权限！

·首先，你需要有一个git的账号，公司的所有文件或者代码都放到git仓库中（仓库地址），需要有账号才能克隆相应的分支下来，得到所有的文件或者代码；

·然后，你也可以把本地推送到远程仓库中（账号有权限）。

1. 分支和主线

主线：主树干，一般放稳定的代码（最终的文档）

分支：树杈，每人（组）维护一个分支，等稳定后合并到主线。

使用分支意味着你可以把你的工作从开发主线上分离开来，以免影响开发主线。比如：每个人都有代码/文档提交到一个公共的服务器，避免互相影响，自己提交到自己的分支里，由专门的人来合并到主干分支。

Git---Linux命令

一对秘钥，密码学： 对称加密  非对称加密 -- 公钥  私钥

 对称加密 === 加密   解密 ==密码

非对称加密   -===  加密解密不同密码

一对钥匙： 公钥 （加密），私钥（解密）

**Git的常用操作**

1. Git克隆：url+本地目录，获取远端的仓库地址，并且有对应的账号权限，克隆代码/文档到本地。
2. Git拉取：远程仓库里的文件更新了，下拉到本地：git拉取→选择分支→确认；点击拉取到的差异，可以查看本次的变动。

·拉取（pull）：可以自动进行分支的合并，用的比较多；

·获取（fetch）：不可以自动合并分支，用的比较少。

1. Git推送：本地新增加了文件/代码，需要推动到远端仓库。
2. 添加（add）：添加到本地缓存区
3. 提交（commit）：提交到本地库，需要添加日志。（此时GieHub是没有文件的）本地仓库
4. 推送（push）：推送到远端服务器。（远端仓库有了本文件）
5. Git的删除：先删除本地文档
6. 提交：删除的本地库里的文件（远程文件还在）
7. 推送：删除远程仓库的文件（远程文件不存在了）
8. Git的撤回：误操作后，再未提交之前，可以revert

掌握以上操作即可。

常见面试题：

1.你工作中用过git么？用过哪些操作？

用过，克隆、拉取、推送、删除、撤回

1. git和svn的主要区别是什么？

SVN版本集中管理，所有的代码都在一台服务器上；分支和主线是不一样的URL地址路径；

Git分布式管理，去中心化，服务器和每个开发人员都拥有一个本地的代码管理仓库，分支和主线路径是一样的。-url地址，更好管理；

**浏览器兼容性测试知识**

1.兼容性测试介绍：

1.1软件+硬件平台兼容性：系统（windows macos,Android,IOS）

1.2软件+软件之间的兼容性：依赖第三方的软件：web应用浏览器兼容性。

1.3不同软件版本之间的兼容性：App升级测试---老数据，功能

2.浏览器兼容性测试介绍

产生浏览器兼容性问题的原因：

因为不同浏览器使用内核及所支持的HTML（标准通用标记语言下的一个应用）等网页语言标准不同：以及用户客户端的环境不同（如分辨率不同）造成的显示结果不能达到理想效果。

最常见的问题就是网页元素位置混乱、错位、重叠。

遇见所有客户可能遇到浏览器--兼容性-web应用

内核：决定了浏览器如何显示网页的内容以及页面的格式信息。

3.常见的浏览器有哪些？



1. 浏览器兼容性测试选择原则

3.1选择浏览器做兼容性测试的原则：

1. 用户有要求，指定浏览器；
2. 网站一般都需要做兼容，用户使用量+内核来看，选取主流浏览器。

Chrome Firefox IE9/11 QQ edge==项目时间调整

一般兼容性测试是怎么来做的？

3.2兼容性测试内容：

主要是页面的格式、字体、输入框、下拉框、复选框、按钮等的检查，页面显示正常

在功能进行中检查。

3.3什么时候测试？

1. 测试几轮测试

2-4轮系统测试 第一轮：Chrome 第二轮： Firefox 第三轮：IE

1. 任务分配

例：甲：登录注册---Chrome，次要检查其他模块的UI

乙：购物车---Firefox

丙：订单----IE

笔试面试题：

1. 小众浏览器出现问题，需不需要做兼容性测试？
2. 用户是不是主要甲方，是的话，需要做兼容性测试；
3. 如果是普通用户，小众用户，不需要做。项目发布之后，文档release note（发布文档），明确我所支持的系统或者浏览器的版本类型，申明：建议你用官方推荐的浏览器。
4. 如果一个网站分为前台访问系统，后台管理系统，是否都需要做浏览器兼容性测试？

前台：需要做兼容性测试---给用户看的；

后台：内部人员（淘宝--后台--商家），有时间的话，做兼容性测试，时间不充足的，不做也没有关系。

P2P前程贷

1. 该项目是一个什么项目？

前程贷是一家采用互联网社群理财模式的P2P信息中介网络平台。是P2P领域的创新型平台，采用蜂群理财模式，将投资理财用户交付给具有风险识别能力和理财能力的人打理，代替投资人监管P2P网贷投资风险。平台专注于中小企业主融资贷款和民间借贷，借贷业务主要有：车贷、房产质押贷款、红本赎楼、企业资金过桥，个人质押贷款。

1. 该项目包含哪些功能模块？哪些功能模块之间存在联系？具体什么联系？

**项目实战做什么**

1. 需求评审会议---产品 测试 开发 项目经理→需求规格说明书
2. Git获取需求文档----提前研读需求，整理疑点
3. 参与需求评审会议，产品确定最终会版本需求，更新到Git服务器
4. 测试Git获取最终需求，提炼测试点，编写测试用例
5. 测试计划

测试负责人（组长）编写测试计划

1. 测试设计----重中之重
2. 按照计划进行测试用例设计
3. 用例评审
4. 测试计划
5. 开发提测前2天进行测试环境的搭建，主要是项目所需软件的安装
6. 开发发布提测通知后，获取项目包，并进行测试环境部署
7. 如果没有上2步，则开发发布提测时，会附上测试环境地址
8. 在测试环境上，执行测试用例；先冒烟，后正式系统测试
9. 测试过程注意关注数据库，提交bug并跟踪；测试轮次3--6轮测试结束
10. 评估阶段
11. 测试结束输出测试报告，并发布邮件通知整个项目组
12. 开发/运维项目上线

理论 —落地 ： 结合项目 怎么用？？

1、软件测试是什么？--测试分类，测试流程、生命周期

2、需求分析 -- 可行性、测试点

3、测试计划、用例 ，评审，准备测试环境

4、开发编写代码--提测

5.1 -- 接口测试 ==集成测试

5.2执行测试： 冒烟--正式2-4轮 --- bug记录和跟踪 ==系统测试

6、评估质量--测试报告

7、验收---上线