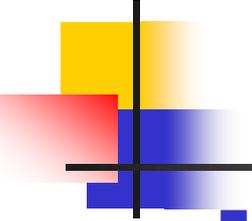


如何设计编写测试用例(浅析)

- 测试用例的概念
- 测试用例设计原则
- 测试用例的编写方法
- 实例：
纸杯的测试用例设计



- 
- 如果没有测试用例测试人员将会如何测试?



随机测试存在的问题

- 不知道是否较全面的测试了所有功能
- 测试的覆盖率无法衡量
- 对新版本的重复测试很难实施
- 无法对测试质量进行有效评估
- 无法形成有效的知识积累
-

测试用例的特征

- 最有可能抓住错误的
- 不是重复的、多余的
- 一组相似测试用例中最有效的
- 既不是太简单，也不是太复杂

测试用例的概念

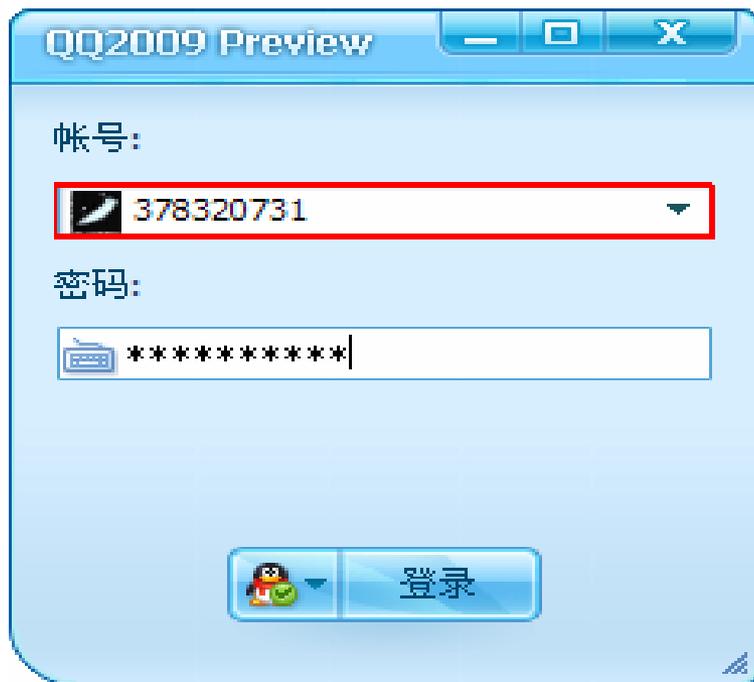
- 如何以最少的人力、资源投入，在最短的时间内完成测试，发现软件系统的缺陷，保证软件的优良品质，是软件公司探索和追求的目标
- 测试用例是测试工作的指导，是软件测试的必须遵守的准则，更是软件测试质量稳定的根本保障

测试用例的概念

- 测试用例是指为实施测试而向被测试系统提供的输入数据，操作或者各种环境设置以及期望结果的一个特定集合。
- 其实简单来说，测试用例就是解决要测什么，怎么测和如何衡量的问题。

举例

- 登录功能，说出一些简单的测试用例



QQ2009 Preview

帐号:

378320731

密码:

登录



举例

简单用例

用例编号	功能点	操作过程	预期结果
01	登录	能够正确处理用户登录	正确处理登录操作

一般的用例

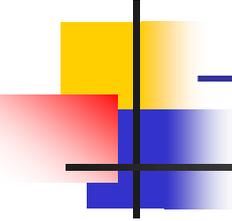
用例编号	功能点	操作过程	预期结果
01	登录	输入正确的帐号和密码	登录成功
		输入错误的帐号和密码	登录失败

举例

■ 比较详细的用例

用例编号	功能点	操作过程	预期结果
01	登录	输入正确的帐号和密码（均为6位），点击[登录]按钮	进入系统
		输入正确的帐号和密码（均为10位），点击[登录]按钮	进入系统
		输入正确的帐号和密码（均为6至8位之间），点击[登录]按钮	进入系统
		帐号为空，点击[登录]按钮	提示输入帐号
		帐号为空格，点击[登录]按钮	提示无效帐号
		帐号小于6位，点击[登录]按钮	提示无效帐号

测试用例设计原则

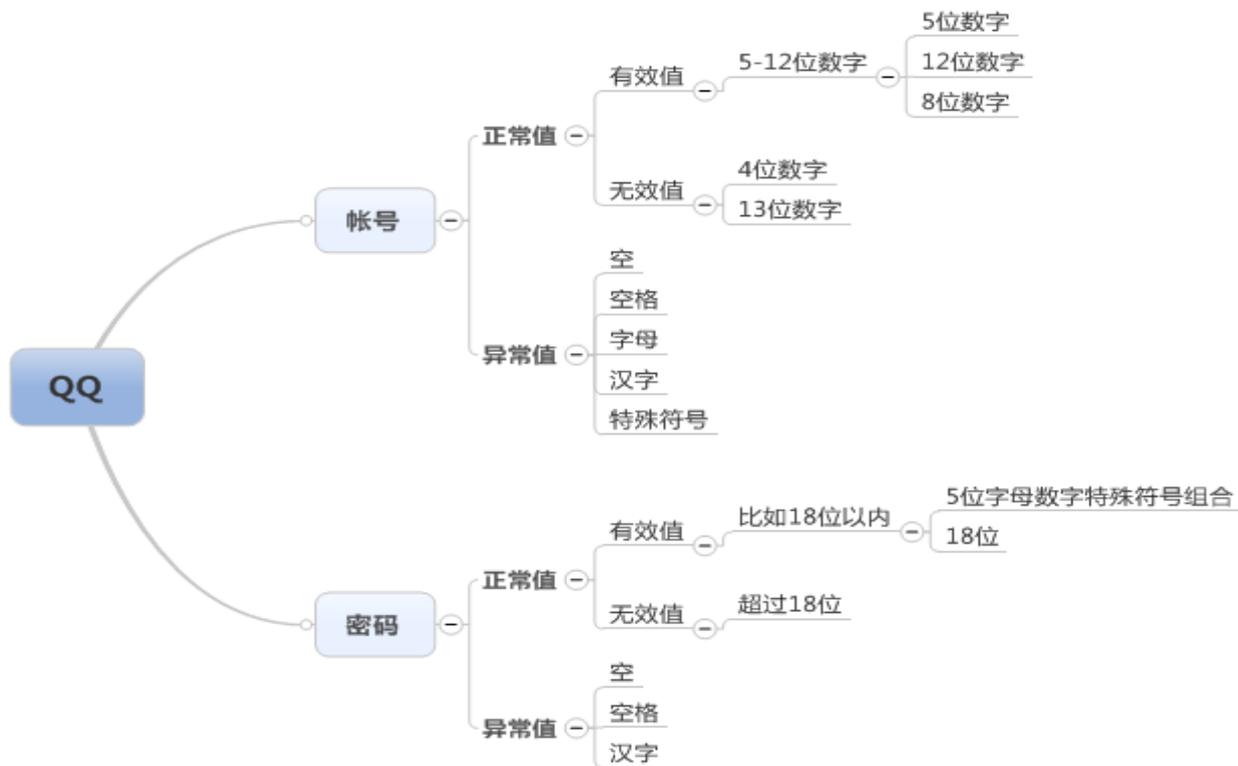


1. 测试用例对需求覆盖的完整性;
2. 测试用例的有效性;
3. 测试用例的可理解性;
4. 测试用例的清晰性;
5. 测试用例的可维护性。

需求的覆盖完整性

- 做到对需求的完全理解, 从全局上把握需求
- 对需求进行归类, 包括正常流, 异常流等, 做到对需求的100%覆盖。(其中有一个好的方法就是用mm图把需求分解了)
- 把基本路径分解出来, 将需求归类。理顺了需求, 用例写起来就顺手多了。

需求的覆盖完整性



测试用例的有效性

- 测试用例应该包含清晰的输入数据以及预期输出
- 如果环境或者业务发生变更后，测试数据必须进行更新维护
- 用例基于数据驱动

测试用例的可理解性

- 测试用例步骤必须描述清晰，不能出现模棱两可以及重复的话语
- 测试用例应该按照一定的顺序进行编写，这样执行的时候效率比较高

测试用例的清晰性

- 测试用例的验证点必须明确清晰重点突出
- 一个用例进行一个功能点的验证，一个萝卜一个坑。
- 对于流程性的用例建议按照流程顺序进行用例安排，从第一个验证点到最后一个验证点，组成流程的开始到结束，方便测试执行。
- 测试用例包含前置条件的必须将前置条件描述清楚，包括入口等。

测试用例的可维护性

- 测试用例因为业务需求发生变更的时候，需要及时更新维护测试用例，做到测试用例的实时性与有效性
- 测试用例需要细化和不断的完善，是个循序渐进的过程
- 通过测试实践检验测试用例并添加，删除，修改测试用例。

小结

- Ross Collard在"Use Case Testing"一文中说:"测试用例的前10%到15%可以发现75%到90%的重要缺陷"。如果你在项目或日常结束后,仔细的分析过我们的bug列表,那么你会觉得这句话非常适用。合理的提高我们的测试效率就是在编写测试用例的时候进行测试用例优先级的划分。
- 如何划分
 1. 用于冒烟测试的用例为最高优先级
 2. 把基本路径以及各模块主功能的测试标注为高优先级别
 3. 把你所有错误和边界值或确认测试标注为中优先级别
 4. 把可用性测试,兼容性测试等标注为低优先级别
 5. 将功能测试用例分为严重和不严重两类,对于不严重的功能测试用例降级为低优先级用例。

测试用例编写方法

等价类划分

如何测试一个两位数加法计算器的程序？

测试需求：测试两个参数的值相加后的结果是否正确。

其中：1. 输入的数值在 - 99 到 99 之间。

2. 大于99或小于- 99的输入应被拒绝，并显示错误信息。

- 根据测试需求开始测试。分别给第1个参数和第2个参数输入表中的值，然后得到测试结果。如图：

第1个参数的值	第2个参数的值	两数相加后的值
1	1	2
1	2	3
1	-1	0
1	-2	1
...

测试用例编写方法

■ 等价类划分

- 等价类划分法作为一种最为典型的黑盒测试方法，它完全不考虑程序的内部结构，而只是根据程序的要求和说明进行测试用例的设计。

■ 如何去做？

- 测试人员要对需求规格说明书中的各项需求，尤其是功能需求进行细致分析，然后把程序的输入域划分成若干个部分，从每个部分中选取少数代表性数据作为测试用例。经过这种划分，每一类的代表性数据在测试中的作用都等价于这一类中的其他值。
- 如何区分 **有效数据等价类** 与 **无效数据等价类**
- **有效数据等价类**就是由那些对程序的规格说明有意义的，合理的输入数据所构成的集合。
- **无效数据等价类**就是那些对程序的规格说明不合理的或者无意义的输入数据所构成的集合。

举例

■ 等价类表

序号	功能项	有效等价类	编号	无效等价类	编号
1	两位数加法	$-99 \leq \text{取值} \leq 99$	2	取值 < -99 取值 > 99	1 3
2

■ 测试用例表

测试用例编号	输入数值	所属等价类	预期结果
1	$-50 + 24$	2	正确输出: -26
2	-130	1	错误信息
3	125	3	错误信息

举例

- 在测试“-99<=数值<=99”的这个等价类区间的时候，会发现如
10+40, -20+30, -30+(-30)这类的正数相加，正数负数相加，负数相加也是不同的等价区间。因此可以使用更多的等价类划分。

等价类表

序号	功能项	有效等价类	编号	无效等价类	编号
1	两位数加法	-99<=取值	2	取值<-99	1
		<=0	3	取值>99	4
		0<=取值			
		<=99			
2

举例

测试用例

测试用例编号	输入数值	所属等价类	预期结果
1	$50+2$	3	正确输出: 52
2	$-63+(-20)$	2	正确输出: -83
3	$-30+10$	2,3	正确输出: -20
4	-130	1	错误信息
5	125	3	错误信息

等价类方法小结

- 等价类技术提供了一个选择哪些数值，舍弃哪些数值的测试用例设计方法。运用等价类技术，可以把相似输出，输入，操作分成组，这些组就是等价区间。只要从等价区间选择一到两个有代表性的值作为测试用例来执行就等同于测试了所有值。当然，也可能存在编程人员编写了异常处理的代码(使用多个测试用例才能发现这个错误)，但是在发现这种类型的缺陷方面存在其他更为有效的技术(比如代码审查)。
- 由之前的案例可以看出，**运用等价类方法的步骤是：**
 - 1.划分等价类(依据是需求)
 - 2.建立等价类表(有效等价类 和 无效等价类)
 - 3.设计测试用例

边界值分析

- 边界值分析也是一种黑盒测试方法，是一种和等价类相关的技术，它具有很强的发现程序错误的能力。如果软件的能力达到极限时能够运行，那么在正常情况下就不会有什么问题。长期的测试工作经验说明“**错误隐藏在角落，问题聚焦在边界上**”，大量的错误是发生在输入或者输出的边界上，而不是发生在输入输出的范围内。因此，正对各种边界值情况设计测试用例，可以查出更多的错误。

- 同样以上个程序为案例，简单设计测试用例，如图：

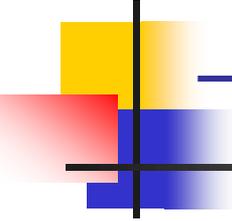
测试用例编号	输入数值	被测边界	预期结果
1	-100	-99	错误信息
2	-99+-(99)		正确输出：-198
3	-98+-(98)		正确输出：-196
4	98+98	99	正确输出：196
5	99+99		正确输出：198
6	100		错误信息

其他测试方法

- 软件测试有两个基本方法：**通过测试**和**失败测试**
- **通过测试**：主要用于验证系统和它的需求是否一致，确认软件至少能做什么，一般通过分析需求说明书设计测试用例。为了确定程序是否满足目标，就必须执行通过测试。
- **失败测试**：确信软件在普通情况下正确运行之后，就可以采取各种手段找出缺陷。纯粹为了破坏软件而设计和执行的测试用例称为失败测试或迫使出错测试。
- 失败测试主要用于证明“一个系统不做不需要它做的事情”。也就是说，要设计测试用例来考察程序超出需求规格说明的严格范围时的行为。
- 失败测试虽然与通过测试看起来相似，但是它是蓄意攻击软件的薄弱环节。

其他测试方法

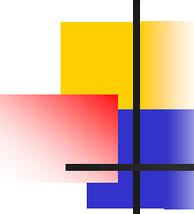
- **错误猜测:**(错误推测)本身不是一种测试技术,而是一种可以运用到所有测试技术中产生更加有效的一种测试的技能。
- 错误猜测是基于经验和直觉推测程序中所有可能存在的各种错误,从而有正对性的设计测试用例的方法。
- **随机测试:** 随机测试指测试中的所有输入数据都是随机生成的,其目标是模拟用户的操作。



测试方法的选择

一个好的测试方法将给软件测试带来事半功倍的效果。在实际的测试工作中可以按照以下原则运用以上所学习的测试技术：

- 1. 在任何情况下都必须使用边界值分析方法。经验表明，用这种方法设计出的测试用例发现程序错误的能力最强。
- 2. 用等价类划分方法补充一些测试用例。
- 3. 用错误推测法再增加一些测试用例。
- 4. 如果程序的功能说明中含有输入条件的组合，应在一开始就选择用因果图法。
- 5. 如果程序的某功能适合自动测试，则可采用自动测试方法以及随机测试方法进行测试。



帮助建议

1. 你是否感觉测试的时候思维很混乱,或者总觉得有些功能没有测到,而一些功能已经测过好几遍?

帮助建议:请明确你的需求,是否做到覆盖100%。你的用例优先级是否设置合理。

2. 在测试实践紧迫的情况下,你不知道要测什么,或者要先测哪些功能?

帮助建议:那么你需要调整自己的用例优先级,顺带回去好好整理整理需求。

3. 在编写测试用例的时候先去学习前辈们的优秀做法。在学习别人优秀成果的基础上,编写自己的用例。

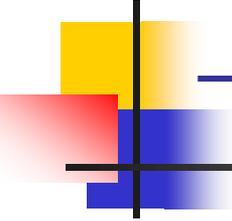
实例：纸杯的测试用例设计

- 用户需求：一个带广告图案的花纸杯



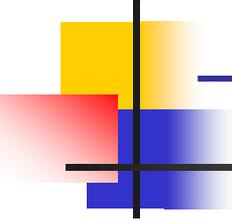
杯子特性

- 杯子的容量：能装多少升水，空杯，半杯，满杯
- 杯子的形状：圆型，上面口大，下面小。
- 杯子的材料：纸杯
- 杯子的抗摔能力：风吹是否会倒，摔一次是否会摔坏，摔多次是否会摔坏
- 杯子的耐温性：装冷水，冰水，热水



广告图案

- 广告内容与图案碰水是否会掉色
- 广告内容与图案是否合法
- 广告内容与图案是否容易剥落
- 广告内容与图案是否符合某个民族的禁忌



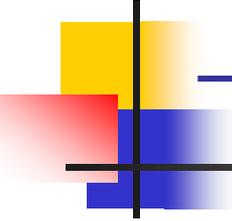
可用性及安全性

- 可用性

1. 装入液体多久后会漏水
2. 装入热水多久后可以变温，装入冰水多久后可以融化
3. 如果装入的不是液体，像石头，沙子，铁块等

- 安全性

1. 装入不同液体，是否会有化学反应。比如：可乐，咖啡等饮料
2. 装入热水杯子是不是会变型和异味
3. 可以加入当热水小于多少度(是一个确定值)时，手不能被烫伤。



易用性和性能

■ 易用性

1. 不同人群是否能适合杯子的型状，包括握杯的感觉和喝水的感觉
2. 不同人群是否能接受杯子的广告内容与图案
3. 纸杯杯壁的薄厚，杯深是否可以让消费者接受

性能

1. 杯子在50，80度的水温下可以使用多少次
2. 倒满开水后，放入冰箱冷冻结冰，取出在融化后杯子是否可以继续使用。

你有不明白的地方吗？



