**软件测试用例阶段总结**

　　**测试****用例**，简单来说，就是一个文档，描述输入、动作和期望的结果，作为一个半路转行的小白来说，刚开始，觉得测试用例有点多余，脑图画画思路，有个梗概即可，但是越来越多的坑告诉我，测试用例，必不可少，那么，大家可能会遇到以下几个问题：

　　1.测试用例繁杂，如何维护？

　　2.怎样高效利用测试用例来开展测试工作？

　　3.测试过程中涉及的范围如何定义？

　　接下来，就分享一下自己的方案

　**1.高效维护**

　　刚开始，虽然是模块化，但是在一个sheet里，每次下拉几百行，定位太浪费时间，不知道你们的用例评审模式和我们的是否一样，我们的用例写完之后产品都要先过一遍，所以，每次都会被各种吐槽，所以，商量了一下

　　解决方案：分模块，一个模块一个sheet，新增功能或优化按大模块细分增加进去，最近的优化发给产品之前用颜色标注一下本次的修改，再发消息说明总的变动，用例用版本工具维护，第三方工具也可，我们是用SVN维护，可以找回历史版本

　**2.测试范围**

　　之前，同事问我评审测试用例的意义何在，答：距离项目评审已经一段时间，可能会有遗忘，再碰一下，看测试和产品之间对于需求理解有没有偏差，查漏补缺。

　　再问，那为什么要叫开发一起呢，答：开发也可能会忘。。。

　　说完我自己都感觉搞笑

　　完了同事提出了正解：对于初级测试来说，我们对于一个功能可能影响的范围是按照主观来看的，但是实际上可能还影响了其他的模块，这个就要开发帮我们分析一下，他们在改代码的时候可能会动到哪一块，总而言之，两点

　　1>思维导图自己梳理功能，和产品达成一致

　　2>用例评审过细节，**记录**开发的问题点，联想可能涉及到的模块

　**3.必测项**

　　刚开始接触这个概念，是基础用例，不过我感觉叫必测项更好理解，字面意思，就是不管上什么功能，你都要保证这部分功能不会受影响

　　八个字：由繁至简，由少到多

　　一开始，不太确定范围，可以把一些核心流程都测一遍，完了预估时间，多次之后，把一些不会变动且没问题的流程去掉，慢慢形成自己测试的一套必测项，然后，在这个过程中，可能会出现一些新的问题，慢慢增加，这个，是个不断优化的过程

　　**4.总结**

　　一件事情刚开始做的时候，可能会觉得很繁琐，但是框架搭起来之后，你会发现，以后的工作会方便很多，所以，磨刀不误砍柴工