

**等价类划分法：**

一、方法简介  
（1）定义  
把所有可能的输入数据,即程序的输入域划分成若干部分（子集）,然后从每一个子集中选取少数具有代表性的数据作为测试用例。该方法是一种重要的,常用的黑盒测试用例设计方法。

（2）划分等价类：  
等价类是指某个输入域的子集合。在该子集合中,各个输入数据对于揭露程序中的错误都是等效的，并合理地假定：测试某等价类的代表值就等于对这一类其它值的测试，因此,可以把全部输入数据合理划分为若干等价类,在每一个等价类中取一个数据作为测试的输入条件就可以用少量代表性的测试数据取得较好的测试结果。等价类划分可有两种不同的情况：有效等价类和无效等价类。

1. 有效等价类  
   是指对于程序的规格说明来说是合理的、有意义的输入数据构成的集合。利用有效等价类可检验程序是否实现了规格说明中所规定的功能和性能。
2. 无效等价类  
   与有效等价类的定义恰巧相反。无效等价类指对程序的规格说明是不合理的或无意义的输入数据所构成的集合。对于具体的问题，无效等价类至少应有一个，也可能有多个。

设计测试用例时,要同时考虑这两种等价类。因为软件不仅要能接收合理的数据,也要能经受意外的考验，这样的测试才能确保软件具有更高的可靠性。

（3）划分等价类的标准

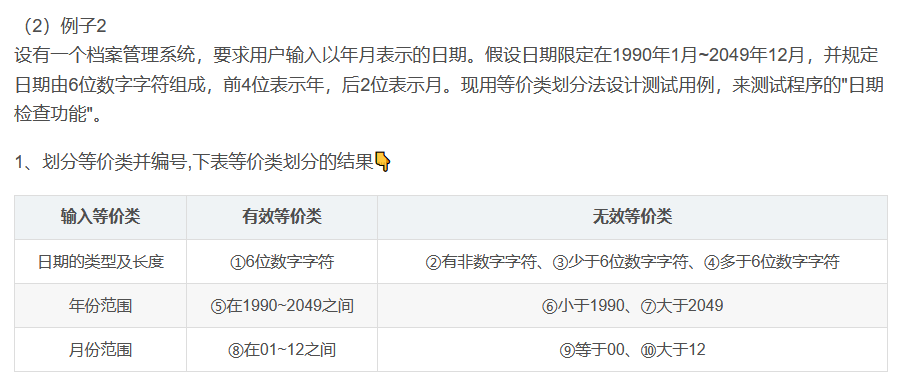
1. 完备测试、避免冗余;
2. 划分等价类重要的是：集合的划分，划分为互不相交的一组子集，而子集的并是整个集合;
3. 并是整个集合：完备性;
4. 子集互不相交：保证一种形式的无冗余性;
5. 同一类中标识（选择）一个测试用例，同一等价类中，往往处理相同，相同处理映射到"相同的执行路径"。

（4）划分等价类的方法

1. 在输入条件规定了取值范围或值的个数的情况下,则可以确立一个有效等价类和两个无效等价类。如：输入值是学生成绩，范围是0～100；
2. 在输入条件规定了输入值的集合或者规定了"必须如何"的条件的情况下,可确立一个有效等价类和一个无效等价类；
3. 在输入条件是一个布尔量的情况下,可确定一个有效等价类和一个无效等价类。
4. 在规定了输入数据的一组值（假定n个）,并且程序要对每一个输入值分别处理的情况下,可确立n个有效等价类和一个无效等价类。例：输入条件说明学历可为:专科、本科、硕士、博士四种之一，则分别取这四种这四个值作为四个有效等价类，另外把四种学历之外的任何学历作为无效等价类。
5. 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）；
6. 在确知已划分的等价类中各元素在程序处理中的方式不同的情况下,则应再将该等价类进一步的划分为更小的等价类。

(5)设计测试用例  
在确立了等价类后,可建立等价类表,列出所有划分出的等价类输入条件：有效等价类、无效等价类，然后从划分出的等价类中按以下三个原则设计测试用例：

1. 为每一个等价类规定一个唯一的编号；
2. 设计一个新的测试用例,使其尽可能多地覆盖尚未被覆盖地有效等价类,重复这一步，直到所有的有效等价类都被覆盖为止；
3. 设计一个新的测试用例,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。



**边界值分析方法**

一、方法简介  
（1）定义  
边界值分析法就是对输入或输出的边界值进行测试的一种黑盒测试方法。通常边界值分析法是作为对等价类划分法的补充，这种情况下，其测试用例来自等价类的边界。

（2）与等价划分的区别

1. 边界值分析不是从某等价类中随便挑一个作为代表，而是使这个等价类的每个边界都要作为测试条件。
2. 边界值分析不仅考虑输入条件，还要考虑输出空间产生的测试情况。

（3）边界值分析方法的考虑  
长期的测试工作经验告诉我们，大量的错误是发生在输入或输出范围的边界上，而不是发生在输入输出范围的内部。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。

使用边界值分析方法设计测试用例，首先应确定边界情况。通常输入和输出等价类的边界，就是应着重测试的边界情况。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，而不是选取等价类中的典型值或任意值作为测试数据。

（4）常见的边界值

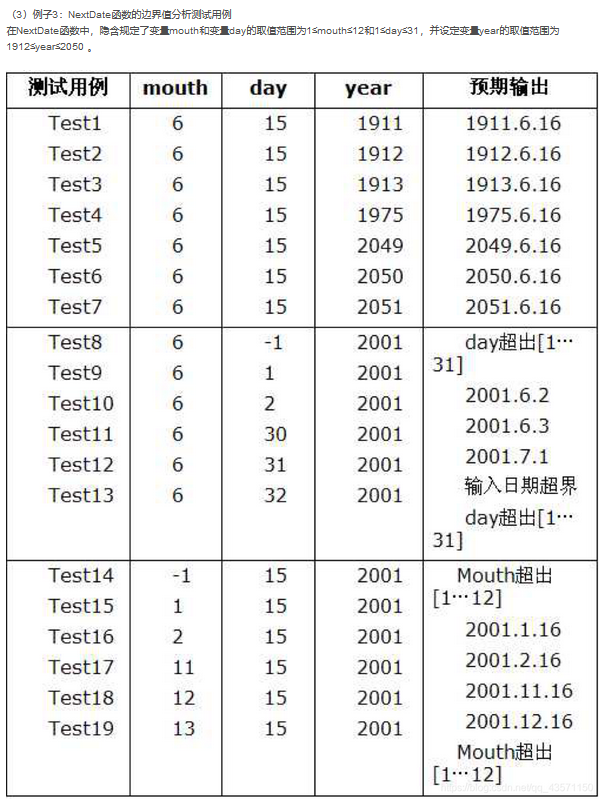
1. 对16-bit 的整数而言 32767 和 -32768 是边界
2. 屏幕上光标在最左上、最右下位置
3. 报表的第一行和最后一行
4. 数组元素的第一个和最后一个
5. 循环的第 0 次、第 1 次和倒数第 2 次、最后一次

（5）边界值分析

1. 边界值分析使用与等价类划分法相同的划分，只是边界值分析假定错误更多地存在于划分的边界上，因此在等价类的边界上以及两侧的情况设计测试用例。  
   例：测试计算平方根的函数  
   –输入：实数  
   –输出：实数  
   –规格说明：当输入一个0或比0大的数的时候，返回其正平方根；当输入一个小于0的数时，显示错误信息"平方根非法-输入值小于0"并返回0；库函数Print-Line可以用来输出错误信息。
2. 等价类划分：  
   I.可以考虑作出如下划分：  
   a、输入 (i)<0 和 (ii)>=0  
   b、输出 (a)>=0 和 (b) Error  
   II.测试用例有两个：  
   a、输入4，输出2。对应于 (ii) 和 (a) 。  
   b、输入-10，输出0和错误提示。对应于 (i) 和 (b) 。
3. 边界值分析：  
   划分(ii)的边界为0和最大正实数；划分(i)的边界为最小负实数和0。由此得到以下测试用例：  
   a、输入 {最小负实数}  
   b、输入 {绝对值很小的负数}  
   c、输入 0  
   d、输入 {绝对值很小的正数}  
   e、输入 {最大正实数}
4. 通常情况下，软件测试所包含的边界检验有几种类型：数字、字符、位置、重量、大小、速度、方位、尺寸、空间等。
5. 相应地，以上类型的边界值应该在：最大/最小、首位/末位、上/下、最快/最慢、最高/最低、 最短/最长、 空/满等情况下。
6. 利用边界值作为测试数据

(6)基于边界值分析方法选择测试用例的原则

1. 如果输入条件规定了值的范围,则应取刚达到这个范围的边界的值,以及刚刚超越这个范围边界的值作为测试输入数据。例如，如果程序的规格说明中规定：“重量在10公斤至50公斤范围内的邮件，其邮费计算公式为……”。作为测试用例，我们应取10及50，还应取10.01,49.99,9.99及50.01等。
2. 如果输入条件规定了值的个数,则用最大个数,最小个数,比最小个数少一,比最大个数多一的数作为测试数据。比如，一个输入文件应包括1~255个记录，则测试用例可取1和255，还应取0及256等。
3. 将规则1）和2）应用于输出条件，即设计测试用例使输出值达到边界值及其左右的值。例如，某程序的规格说明要求计算出"每月保险金扣除额为0至1165.25元"，其测试用例可取0.00及1165.24、还可取一0.01及1165．26等。再如一程序属于情报检索系统，要求每次"最少显示1条、最多显示4条情报摘要"，这时我们应考虑的测试用例包括1和4，还应包括0和5等。
4. 如果程序的规格说明给出的输入域或输出域是有序集合,则应选取集合的第一个元素和最后一个元素作为测试用例。
5. 如果程序中使用了一个内部数据结构,则应当选择这个内部数据结构的边界上的值作为测试用例。
6. 分析规格说明,找出其它可能的边界条件。



**Scrum: Backlog; Sprint; Product increment**

**SDP（软件开发计划）**



Scrum过程

Scrum是一个包括了一系列的实践和预定义角色的过程骨架（是一种流程、计划、模式，用于有效率地开发软件）。

在每一次冲刺（一个15到30 天周期 ，长度由开发团队决定），开发团队创建可用的（可以随时推出）软件的一个增量。每一个冲刺所要实现的特性来自产品订单（product backlog，我觉得翻译成“产品目标”更恰当）， 产品订单（产品目标）是指按照优先级排列的需要完成的工作的概要的需求（目标）。哪些订单项（目标项目）会被加入一次冲刺，由冲刺计划会议决定。 在会议中，产品负责人告诉开发团队他需要完成产品订单中的哪些订单项。开发团队决定在下一次冲刺中他们能够承诺完成多少订单项。 在冲刺的过程中，没有人能够变更冲刺订单（sprint backlog），这意味着在一个冲刺中需求是被冻结的。

管理Scrum过程有很多实施方法，从白板上的即时贴到软件包。Scrum最大的好处是它非常容易学习，而且应用Scrum不需要太多的投入。

**敏捷方法之极限编程(XP)和 Scrum区别**

区别之一： 迭代长度的不同

XP的一个Sprint的迭代长度大致为1~2周, 而Scrum的迭代长度一般为 2~ 4周。

区别之二: 在迭代中, 是否允许修改需求

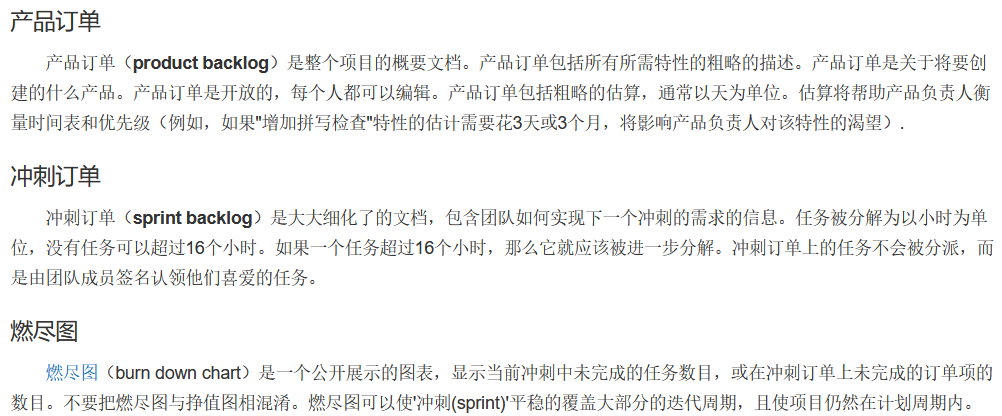
XP在一个迭代中，如果一个User Story(用户素材, 也就是一个需求)还没有实现， 则可以考虑用另外的需求将其替换， 替换的原则是需求实现的时间量是相等的。而Scrum是不允许这样做的，一旦迭代开工会完毕, 任何需求都不允许添加进来，并有Scrum Master严格把关，不允许开发团队受到干扰。

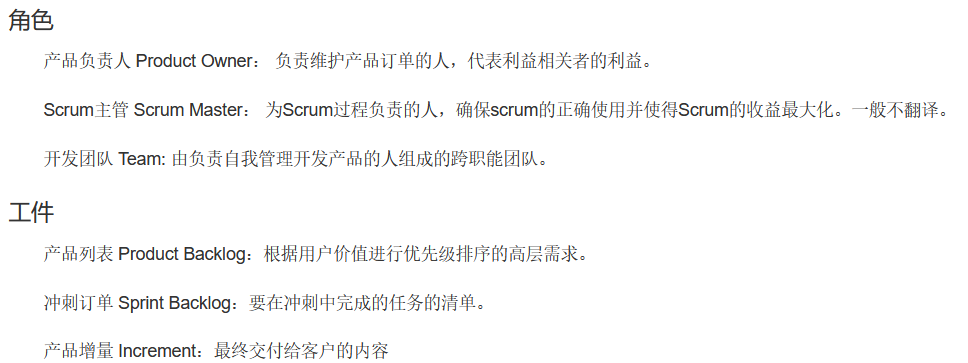
区别之三: 在迭代中，User Story是否严格按照优先级别来实现

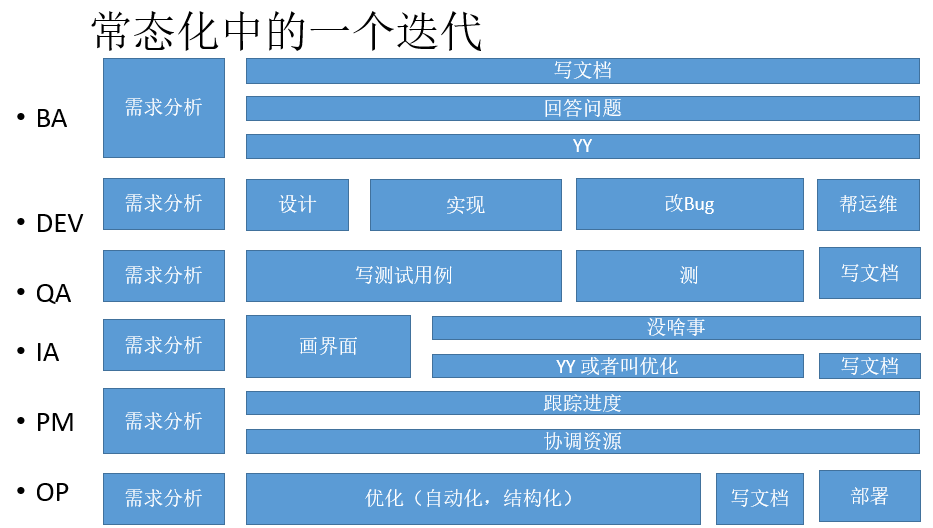
XP是务必要遵守优先级别的。但Scrum在这点做得很灵活，可以不按照优先级别来做，Scrum这样处理的理由是：如果优先问题的解决者，由于其它事情耽搁，不能认领任务，那么整个进度就耽误了。另外一个原因是，如果按优先级排序的User Story #6和#10，虽然#6优先级高，但是如果#6的实现要依赖于#10，则不得不优先做#10。

区别之四:软件的实施过程中，是否采用严格的工程方法，保证进度或者质量

Scrum没有对软件的整个实施过程开出工程实践的处方，要求开发者自觉保证。但XP对整个流程方法定义非常严格，规定需要采用TDD、自动测试、结对编程、简单设计、重构等约束团队的行为。









**最重要的是DEV和QA之间的配合**