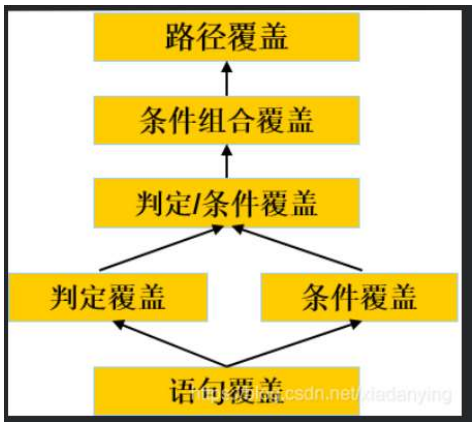
期末补充



语句覆盖

- 设计若干条测试用例，使程序中每条可执行语句（所有的方框）至少执行一次。

判定覆盖（分支覆盖）

- 设计测试用例，使程序中的每个逻辑判断的取真和取假的分支至少经历一次。无法确定判定内部条件的错误。

条件覆盖

- 设计若干测试用例，使程序的每个判定中的每个条件的可能取值至少满足一次。

判定-条件覆盖

- 使判定中每个条件的可能取值至少满足一次，并且使每个判定分支至少执行一次。
- 同时满足判定、条件两种覆盖标准。
- 缺点在于没有考虑两个条件的组合情况。

	A B X	路径	覆盖分支	覆盖条件
Case 7	2 0 3	a c e	c e	T1 T2 T3 T4
Case8	1 1 1	a b d	b d	F1 F2 F3 F4

条件组合覆盖

- 使得每个判断表达式中条件的各种可能组合都至少出现一次。

①	A > 1, B = 0	T1 T2	判定一为真
②	A > 1, B≠0	T1 F2	判定一为假
③	A≤1, B = 0	F1 T2	
④	A≤1, B≠0	F1 F2	
⑤	A = 2, X > 1	T3 T4	判定二为真
⑥	A = 2, X≤1	T3 F4	
⑦	A≠2, X>1	F3 T4	
⑧	A≠2, X≤1	F3 F4	判定二为假

环形复杂度

- 概念
  - 环形复杂度也称为圈复杂度，它是一种为程序逻辑复杂度提供定量尺度的软件度量。
  - 环形复杂度的应用–可以将环形复杂度用于基本路径方法，它可以提供程序基本集的独立路径数量。
- 计算环形复杂度的方法
  - 控制流图中区域的数量对应于环形复杂度。
  - 给定控制流图G的环形复杂度V(G)，定义为V(G) = E-N+2，其中E是控制流图中边的数量，N是控制流图中的节点数量。
  - 给定控制流图G的环形复杂度V(G)，也可定义为V(G) = P+1，其中P是控制流图G中判定节点的数量。

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。

（2）例子2
设有一个档案管理系统，要求用户输入以年月表示的日期。假设日期限定在1990年1月~2049年12月，并规定日期由6位数字字符组成，前4位表示年，后2位表示月。现用等价类划分法设计测试用例，来测试程序的"日期检查功能"。

1、划分等价类并编号,下表等价类划分的结果👇

输入等价类	有效等价类	无效等价类
日期的类型及长度	①6位数字字符	②有非数字字符、③少于6位数字字符、④多于6位数字字符
年份范围	⑤在1990~2049之间	⑥小于1990、⑦大于2049
月份范围	⑧在01~12之间	⑨等于00、⑩大于12

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。

（3）例子3：NextDate函数的边界值分析测试用例

在NextDate函数中，隐含规定了变量mouth和变量day的取值范围为1≤mouth≤12和1≤day≤31，并设定变量year的取值范围为1912≤year≤2050 。

测试用例	mouth	day	year	预期输出
Test1	6	15	1911	1911.6.16
Test2	6	15	1912	1912.6.16
Test3	6	15	1913	1913.6.16
Test4	6	15	1975	1975.6.16
Test5	6	15	2049	2049.6.16
Test6	6	15	2050	2050.6.16
Test7	6	15	2051	2051.6.16
Test8	6	-1	2001	day超出[1…31]
Test9	6	1	2001	2001.6.2
Test10	6	2	2001	2001.6.3
Test11	6	30	2001	2001.7.1
Test12	6	31	2001	输入日期超界
Test13	6	32	2001	day超出[1…31]
Test14	-1	15	2001	Mouth超出[1…12]
Test15	1	15	2001	2001.1.16
Test16	2	15	2001	2001.2.16
Test17	11	15	2001	2001.11.16
Test18	12	15	2001	2001.12.16
Test19	13	15	2001	Mouth超出[1…12]

Scrum: Backlog; Sprint; Product increment

SDP（软件开发计划）

	Product Backlog	Sprint Backlog
详细度	比较详细	非常详细
估算单位	Story Points	小时
文档归属	Product Owner	开发团队
更新频率	一次/周	一次/工作日
持续长度	整个项目周期	一个Sprint
文档名	Product Backlog workbook	Iteration Backlog workbook

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。

产品订单

产品订单（**product backlog**）是整个项目的概要文档。产品订单包括所有所需特性的粗略的描述。产品订单是关于将要创建的什么产品。产品订单是开放的，每个人都可以编辑。产品订单包括粗略的估算，通常以天为单位。估算将帮助产品负责人衡量时间表和优先级（例如，如果"增加拼写检查"特性的估计需要花3天或3个月，将影响产品负责人对该特性的渴望）.

冲刺订单

冲刺订单（**sprint backlog**）是大大细化了的文档，包含团队如何实现下一个冲刺的需求的信息。任务被分解为以小时为单位，没有任务可以超过16个小时。如果一个任务超过16个小时，那么它就应该被进一步分解。冲刺订单上的任务不会被分派，而是由团队成员签名认领他们喜爱的任务。

燃尽图

燃尽图（burn down chart）是一个公开展示的图表，显示当前冲刺中未完成的任务数目，或在冲刺订单上未完成的订单项的数目。不要把燃尽图与挣值图相混淆。燃尽图可以使'冲刺(sprint)'平稳的覆盖大部分的迭代周期，且使项目仍然在计划周期内。

角色

产品负责人 Product Owner： 负责维护产品订单的人，代表利益相关者的利益。

Scrum主管 Scrum Master： 为Scrum过程负责的人，确保scrum的正确使用并使得Scrum的收益最大化。一般不翻译。

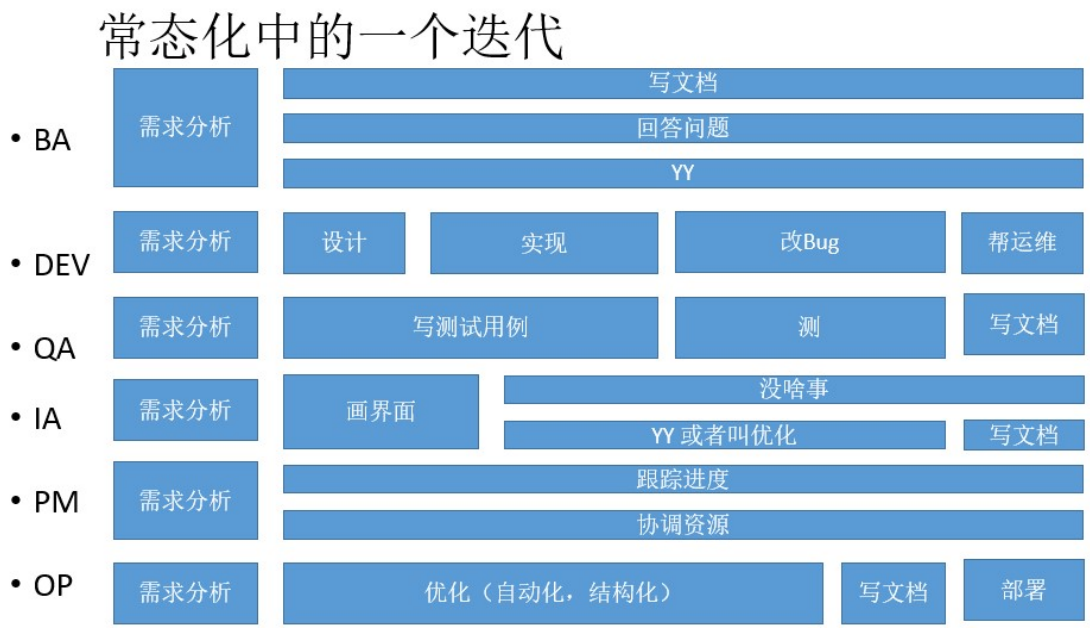
开发团队 Team: 由负责自我管理开发产品的人组成的跨职能团队。

工件

产品列表 Product Backlog：根据用户价值进行优先级排序的高层需求。

冲刺订单 Sprint Backlog：要在冲刺中完成的任务的清单。

产品增量 Increment：最终交付给客户的内容



产品经理（Product Manager），英文缩写PM

业务分析是（Business Analyst），英文缩写BA

项目经理（Project Manager），英文缩写PM

软件开发人员（Development），英文缩写Dev

软件测试人员，英文缩写QA

IT运维技术人员（Operations），英文缩写Ops

最重要的是 DEV 和 QA 之间的配合