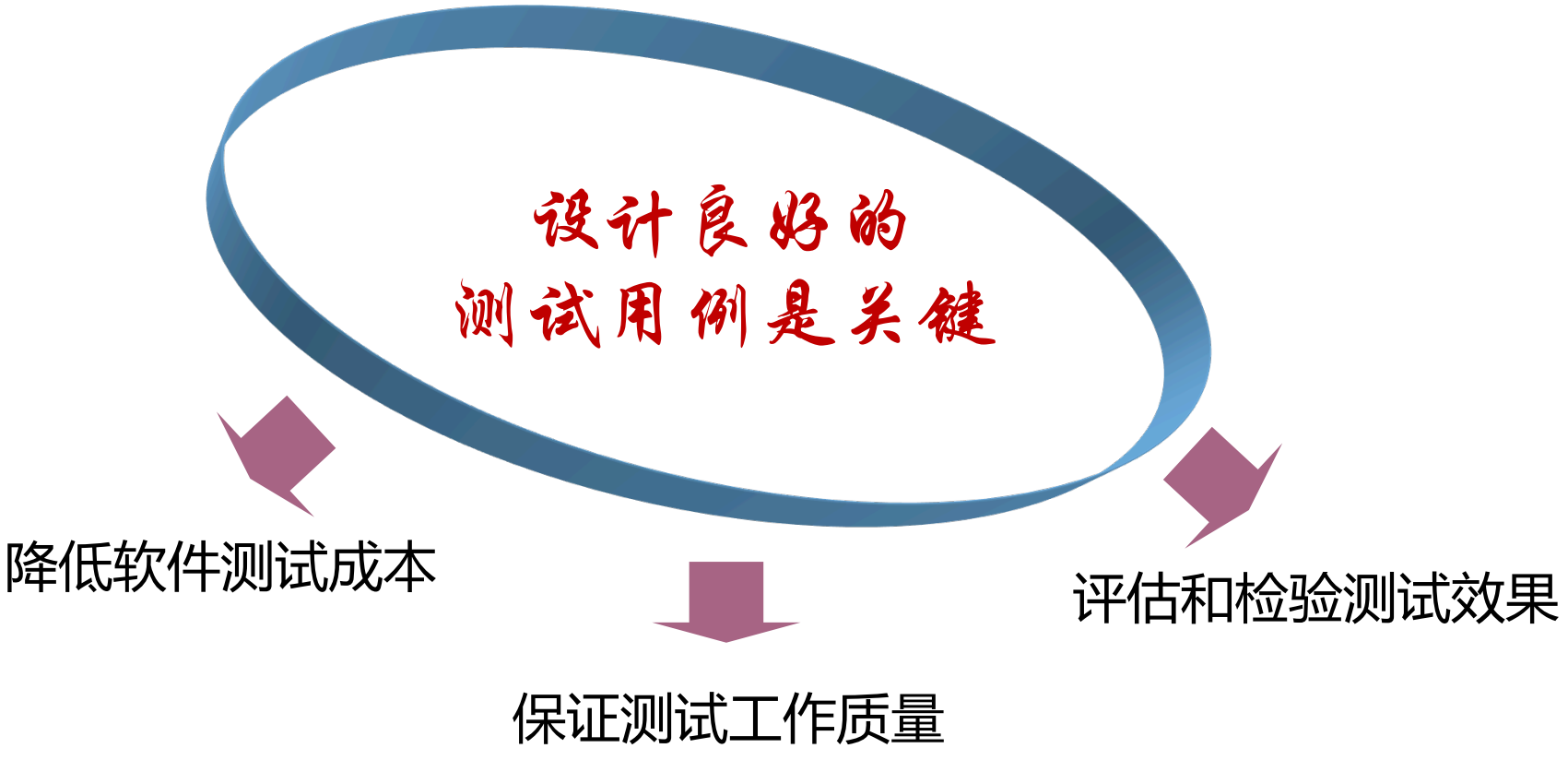
黑盒测试方法

清华大学软件学院 刘强

测试用例的重要性



测试用例的概念

测试集
测试用例值的集合

测试用例值 → 完成被测软件的某个执行所需的输入值

期望结果 → 当且仅当程序满足其期望行为，执行测试时产生的结果

前缀值 → 将软件置于合适的状态来接受测试用例值的任何必要的输入

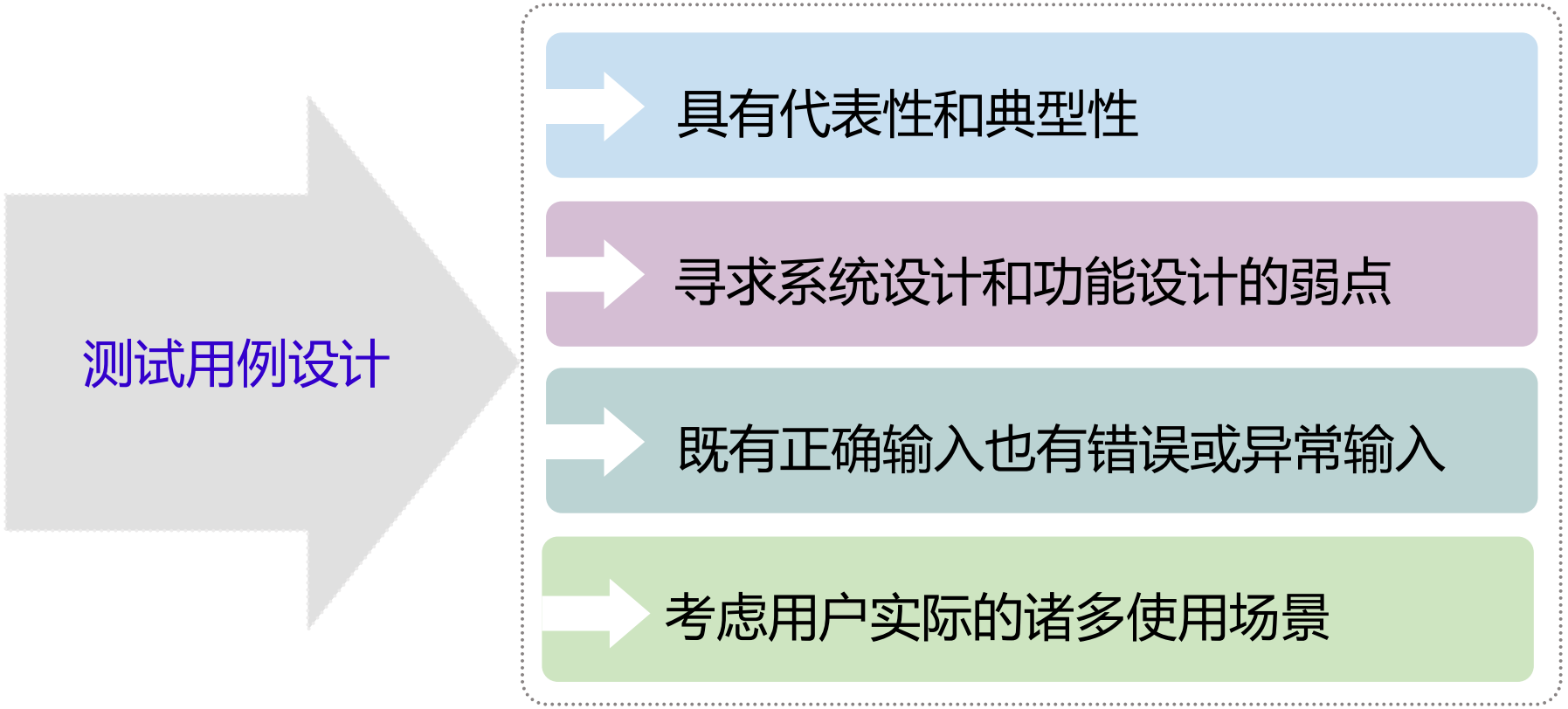
后缀值 → 测试用例值被发送以后，需要被发送到软件的任何输入

验证值：查看测试用例值结果所要用到的值
结束命令：终止程序或返回到稳定状态所要用到的值

测试用例的概念

- 测试用例值：电话号码
- 期望结果：接通（或未接通）
- 前缀值：电话开启并进入拨号界面
- 后缀值：按下“呼叫”或“取消”按钮

测试用例设计的要求

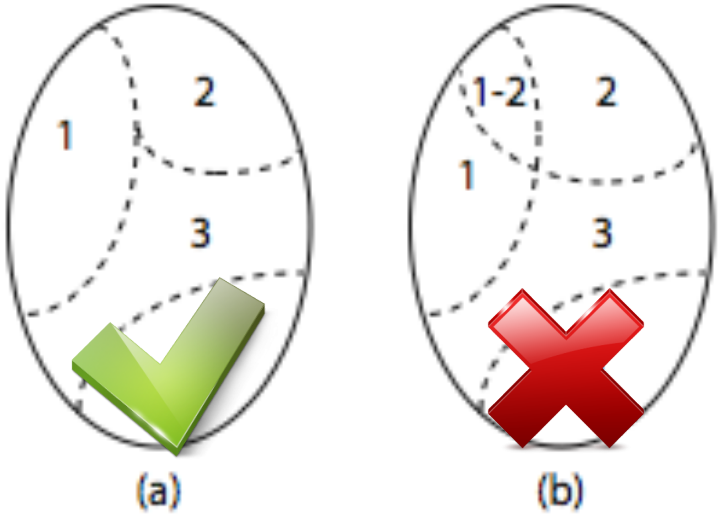

测试用例设计
具有代表性和典型性
寻求系统设计和功能设计的弱点
既有正确输入也有错误或异常输入
考虑用户实际的诸多使用场景

黑盒测试技术

黑盒测试是将测试对象看做一个黑盒子，完全不考虑程序内部的逻辑结构和内部特性，只依据程序的需求规格说明书，检查程序的功能是否符合它的功能说明。

等价类划分

等价类划分是将输入域划分成尽可能少的若干子域，在划分中要求每个子域两两互不相交，每个子域称为一个等价类。

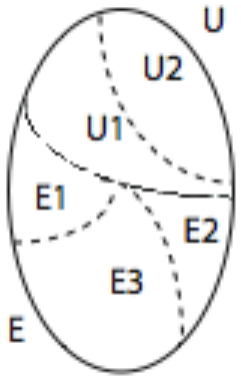


- 同一输入域的等价类划分可能不唯一
- 只需从每一个等价类中选取一个输入作为测试用例
- 对于相同的等价类划分，不同测试人员选取的测试用例集可能是不同的

等价类类型

有效等价类是对规格说明有意义、合理的输入数据构成的集合，能够检验程序是否实现了规格说明中预先规定的功能和性能。

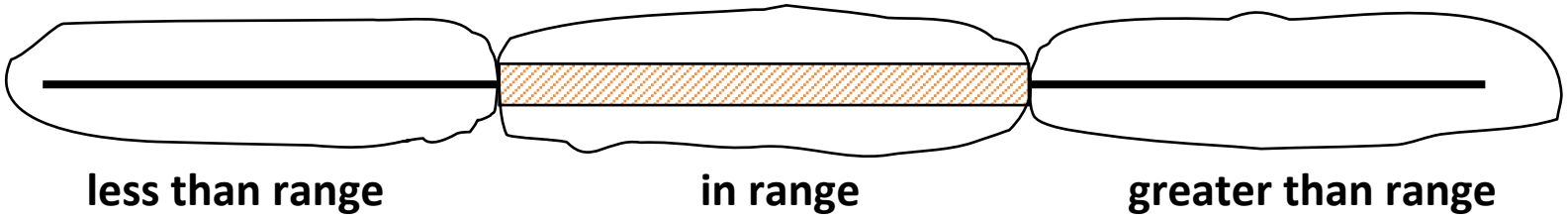
无效等价类是对规格说明无意义、不合理的输入数据构成的集合，以检查程序是否具有一定的容错性。



- E 表示所有正常和合法的输入
- U 表示所有异常和非法的输入

变量的等价类

取值范围：在输入条件规定了取值范围的情况下，可以确定一个有效等价类和两个无效等价类。



举例：程序的输入参数 x 是小于100大于10的整数。

1个有效等价类：$10 < x < 100$

2个无效等价类：$x \leq 10$ 和 $x \geq 100$

变量的等价类

字符串：在规定了输入数据必须遵守的规则情况下，可确定一个有效等价类（符合规则）和若干个无效等价类（从不同角度违反规则）。

举例：姓名是长度不超过20的非空字符串，且只由字母组成，数字和其他字符都是非法的。

1个有效等价类：满足了上述所有条件的字符串

3个无效等价类：

- 空字符串
- 长度超过20的字符串
- 包含了数字或其它字符的字符串（长度不超过20）

变量的等价类

枚举：若规定输入数据是一组值（假定N个），并且程序要对每一个输入值分别处理，可确定N个有效等价类和一个无效等价类。

举例：某程序根据不同的学历分别计算岗位工资，其中学历可以是专科、本科、硕士、博士等四种类型。

4个有效等价类：专科、本科、硕士、博士

1个无效等价类：其他学历

如果将专科、本科、硕士、博士按一种方式计算岗位工资，这时应如何划分等价类？

变量的等价类

数组：数组是一组具有相同类型的元素的集合，数组长度及其类型都可以作为等价类划分的依据。

举例：假设某程序的输入是一个整数数组 `int oper[3]`

1个有效等价类：所有合法值的数组，如 `{-10,20}`

2个无效等价类：

- 空数组
- 所有大于期望长度的数组，如 `{-9,0,12,5}`

如果对数组元素有其他附加约束，例如数组`oper`元素的取值范围是`[-3,3]`，则需要增加相应的等价类。

变量的等价类

复合数据类型：复合数据类型是包含两个或两个以上相互独立的属性的输入数据，在进行等价类划分时需要考虑输入数据的每个属性的合法和非法取值。

举例：

```
struct student {
        string name;
        string course[100];
        int grade[100];
    }
```

对复合数据类型中的每个元素进行等价类划分，再将这些等价类进行组合，最终形成对软件整个输入域的划分。

等价类组合

测试用例生成：测试对象通常有多个输入参数，如何对这些参数等价类进行组合测试，来保证等价类的覆盖率，是测试用例设计首先需要考虑的问题。

所有有效等价类的代表值都集成到测试用例中，即覆盖有效等价类的所有组合。任何一个组合都将设计成一个有效的测试用例，也称正面测试用例。

无效等价类的代表值只能和其他有效等价类的代表值（随意）进行组合。因此，每个无效等价类将产生一个额外的无效测试用例，也称负面测试用例。

举例：判断三角形类型

输入三个整数a、b、c，分别作为三角形的三条边，现通过一个程序判断这三条边构成的三角形类型，包括等边三角形、等腰三角形、一般三角形（特殊的还包括直角三角形）以及构不成三角形。

现在要求输入的三个整数a、b、c必须满足以下条件：

条件1：1≤a≤100

条件2：1≤b≤100

条件3：1≤c≤100

条件4：a<b+c

条件5：b<a+c

条件6：c<a+b

请使用等价类划分方法，设计该程序的测试用例。

举例：判断三角形类型

等价类划分：

① 按输入取值划分

{0，>0，<0} 或 {0，1，>1，<0}

② 按输出的几何特性划分

{等腰且非等边三角形，等边三角形，一般三角形，非三角形}

你会选择哪一种方法划分等价类

举例：判断三角形类型

标准等价类测试用例

序号	测试用例描述	输入参数			期望输出
		a	b	c	
1	a>0，b>0，c>0 a+b>c，b+c>a，a+c>b a=b=c	10	10	10	等边三角形
2	a>0，b>0，c>0 a+b>c，b+c>a，a+c>b a=b≠c 或 b=c≠a 或 a=c≠b	10	10	5	等腰三角形
3	a>0，b>0，c>0 a+b>c，b+c>a，a+c>b a≠b≠c	3	4	5	一般三角形
4	a>0，b>0，c>0 a+b≤c 或 b+c≤a 或 a+c≤b	4	1	2	非三角形

举例：判断三角形类型

健壮等价类测试用例

序号	测试用例描述	输入参数			期望输出
		a	b	c	
1-4	有效等价类同前面（略）				
5	a<0，b>0，c>0	-1	5	5	a 值越界
6	a>0，b<0，c>0	5	-1	5	b 值越界
7	a>0，b>0，c<0	5	5	-1	c 值越界
8	a>100，b>0，c>0	101	5	5	a 值越界
9	a>0，b>100，c>0	5	101	5	b 值越界
10	a>0，b>0，c>100	5	5	101	c 值越界

边界值分析

边界值分析是对输入或输出的边界值进行测试的一种方法，它通常作为等价类划分法的补充，这种情况下的测试用例来自等价类的边界。

- 先确定边界：通常输入或输出等价类的边界就是应该着重测试的边界情况。
- 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。

实践表明：大多数故障往往发生在输入定义域或输出值域的边界上，而不是内部。因此，针对各种边界情况设计测试用例，通常会取得很好的测试效果。

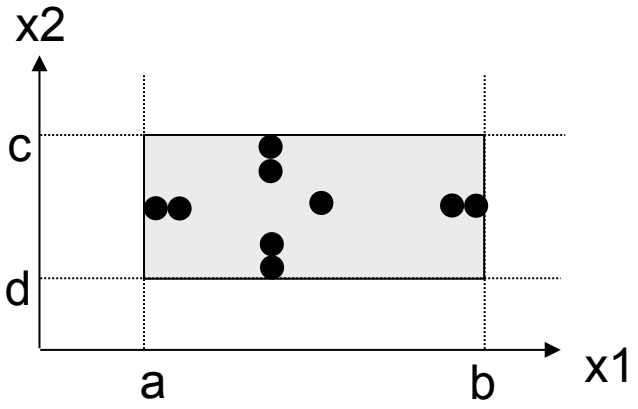
边界值分析

输入项	边界值	测试用例的设计思路
字符	起始 - 1个字符 结束 + 1个字符	假设一个文本输入区域允许输入1到255个字符，输入1个和255个字符作为有效等价类； 输入0个和256个字符作为无效等价类，这几个数值都属于边界条件值。
数值	最小值 - 1 最大值 + 1	假设某软件要求输入5位十进制整数值，可以使用10000作为最小值、99999作为最大值；然后使用刚好小于5位和大于5位的数值作为边界条件。
空间	小于空余空间一点 大于满空间一点	例如在用U盘存储数据时，使用比剩余磁盘空间大一点（几 KB）的文件作为边界条件。

边界值分析

基本思想：故障往往出现在程序输入变量的边界值附近

- 边界值分析法是基于可靠性理论中称为“单故障”的假设，即有两个或两个以上故障同时出现而导致失效的情况很少。
- 对程序中的每个变量重复：每次保留一个变量，让其余的变量取正常值，被保留的变量依次取 min、min+、nom、max- 和 max。



$<x1_{nom}, x2_{min}>$ $<x1_{nom}, x2_{min+}>$ $<x1_{nom}, x2_{nom}>$

$<x1_{nom}, x2_{max}>$ $<x1_{nom}, x2_{max-}>$ $<x1_{min}, x2_{nom}>$

$<x1_{min+}, x2_{nom}>$ $<x1_{max}, x2_{nom}>$ $<x1_{max-}, x2_{nom}>$

举例：判断三角形类型

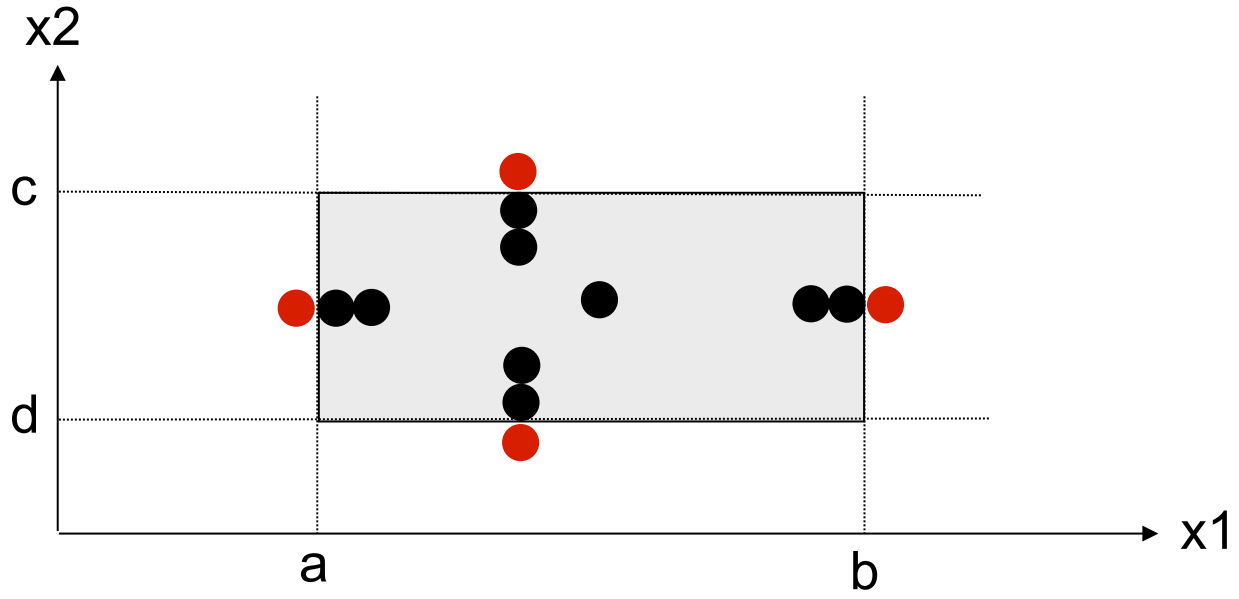
序号	测试用例描述	输入参数			期望输出
		a	b	c	
1	a>0，b>0，c>0 a+b>c，b+c>a，a+c>b a=b=c	60	60	60	等边三角形
2	a>0，b>0，c>0 a+b>c，b+c>a，a+c>b a=b≠c 或 b=c≠a 或 a=c≠b	60	60	1	等腰三角形
3		60	60	2	
4		50	50	99	
5		60	1	60	
6		60	2	60	

举例：判断三角形类型

序号	测试用例描述	输入参数			期望输出
		a	b	c	
7	a>0，b>0，c>0 a+b>c，b+c>a，a+c>b a=b≠c 或 b=c≠a 或 a=c≠b	50	99	50	等腰三角形
8		1	60	60	
9		2	60	60	
10		99	50	50	
11	a>0，b>0，c>0 a+b≤c 或 b+c≤a 或 a+c≤b	50	50	100	非三角形
12		50	100	50	
13		100	50	50	

健壮性测试

健壮性测试是作为边界值分析的一个简单的扩充，它除了对变量的5个边界值分析取值外，还要增加一个略大于最大值（max+）以及略小于最小值（min-）的取值，检查超过极限值时系统的情况。



错误推测法

错误推测法是人们根据经验或直觉推测程序中可能存在的各种错误，从而有针对性地编写检查这些错误的测试用例的方法。

- 软件缺陷具有空间聚集性，80%的缺陷常常存在于20%的代码中。因此，应当记住常常光临代码的高危多发“地段”，这样发现缺陷的可能性会大得多。
- 列举程序中所有可能的错误和容易发生错误的特殊情况，根据可能出现的错误情况选择测试用例。

谢谢大家！

THANKS