Software Testing

软件测试技术

刘瑞康

QQ：2972370818

考核要求

平时成绩 30%

实验报告 35%

课程报告 35%

本章内容

- 软件质量
- 软件测试
- 软件缺陷
- 软件测试分类

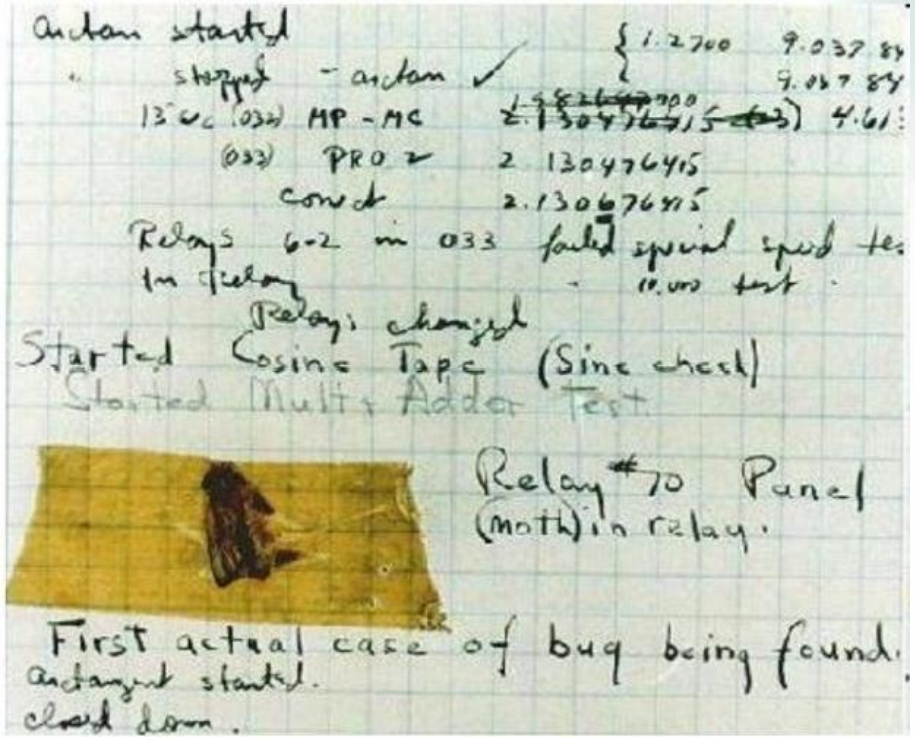
问题引入

- 软件缺陷与软件故障案例
  - 第一个Bug 的故事
  - “千年虫”问题
  - Intel 奔腾处理芯片缺陷
  - 火星登陆事故
  - 爱国者导弹防御系统
  - 迪斯尼的圣诞节礼物
  - 北京奥运门票系统

第一个Bug 的故事

- 1945.9, Grace Hopper

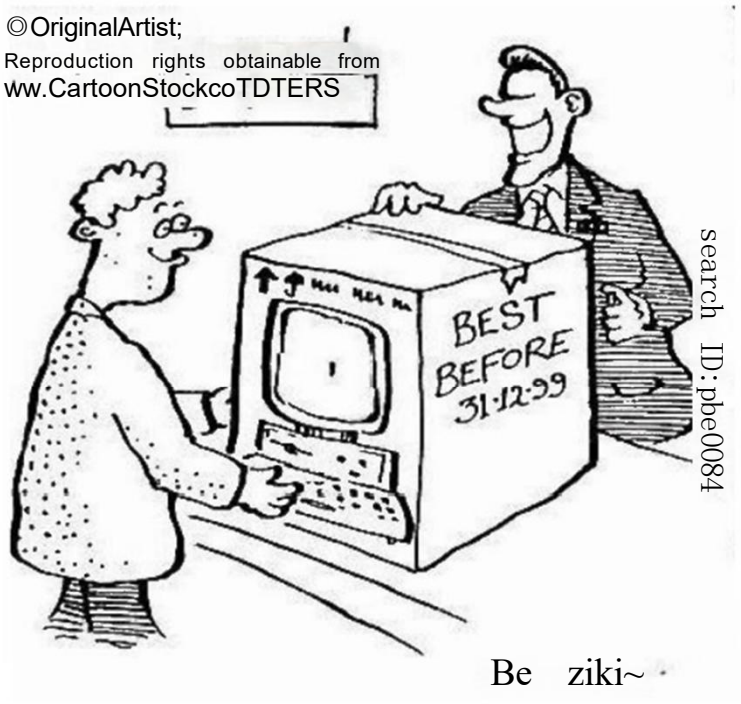


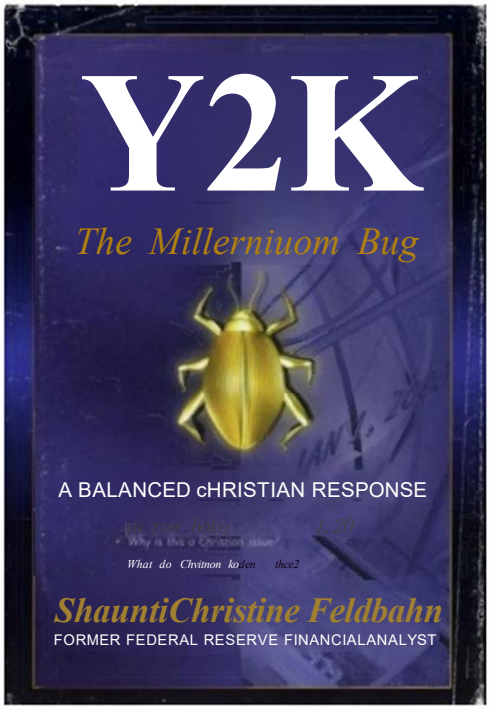


格蕾斯. 哈珀：计算机科学家、美国海军将军、编译器的发明者
COBOL语言的开发负责人

千年虫问题(Y2K)

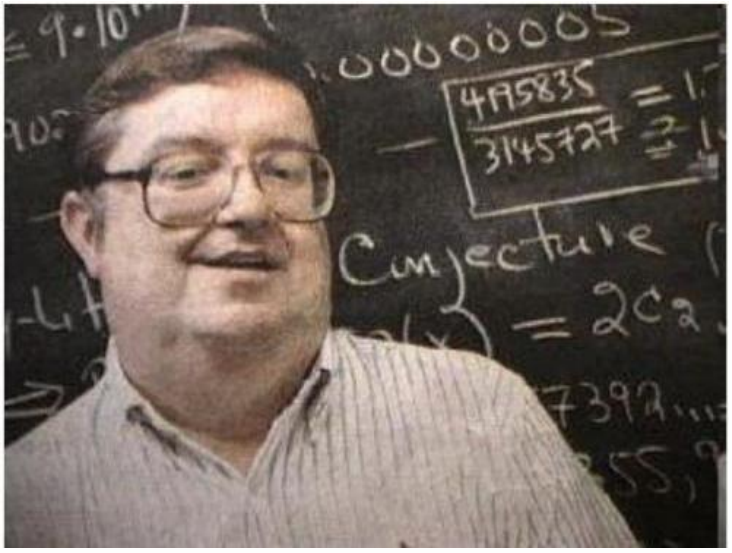
- 20世纪70年代，计算机存储空间小，为节省字节，将四位的年份用两 位表示。
- 只有到数十年后的200 0年1月1日才会出现问题，这期间肯定会升级或更改系统。
- 各种系统中这类问题的解决费用估计超过数亿美元。



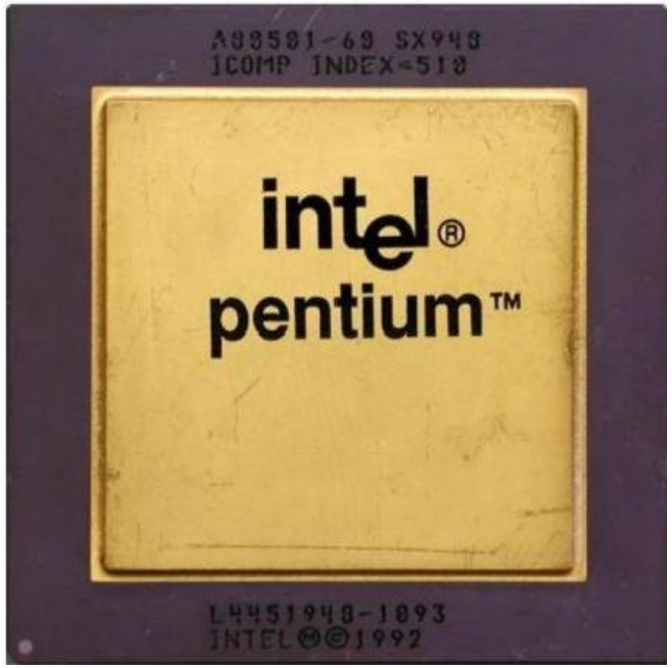


Intel奔腾处理芯片缺陷(Pentium FDIV bug)1994年

- (4195835/3145727)*3145727 - 4195835算式的结果应为0；如果结果不为0，则有浮点除法软件缺陷。
- 由于只有在精度要求很高的计算中才会出错(发生的概率是几十亿分之一)，Intel 管理层认为不严重，未修正。
- 更换芯片的费用：4亿美元。



弗吉尼亚州数学教授尼斯利



火星登陆事故(Mars Polar Lander)

- 1999年12月3日，美国航天局的火星基地登陆飞船在试图登陆火星表面时失踪。
- 原因：控制登陆推进器的某个数据位在未着地时就意外更改，导致推进器在未着陆时就被关闭，飞船坠毁。







MARS SURVEYOR 98

爱国者导弹防御系统1991年

- 在海湾战争中对抗伊拉克飞毛腿导弹的防御战，其中一枚在沙特阿拉伯的多哈击毙了28名美军士兵。
- 原因：一个很小的系统时钟错误累积起来就可能延迟14小时，造成跟踪系统失去准确度。在多哈袭击战中，系统被延迟100多个小时。





迪斯尼的圣诞礼物：1994-1995

- 94年秋，迪斯尼公司发布了面向儿童的游戏“Lion King Animated Storybook”（狮子王历险记），进行了大量的宣传和促销，销售额可观。12月26日，开始收到大量投诉：游戏无法正常运行，舆论哗然。

- 原因：没有对市场上的各种PC机型进行测试，该软件只能在少数系统中正常工作，但在大众常用的系统中不行。

北京奥运订票网站瘫痪2007年

事故过程：订票系统半小时即瘫痪



火车票订票网站：12306

- 12306网站于2011年6月1日正式上线。
- 2012年春节期间和2012年中秋国庆长假来临前因访问量过大而瘫痪。



火车票订票网站：12306

◆ 2014年春运购票首日，万众瞩目的铁总12306还是出了故障：出现串号问题。

凤凰资讯 凤凰网资讯>滚动新闻>正文

春运售票首日12306爆出“串号门”

2013年12月29日03:37
来源：南方都市报

原标题：春运售票首日12306爆出“串号门”



为什么需要软件测试

- 有缺陷的软件带给用户的问题
  - 不好的感觉，产生不信任感；
  - 无法正常使用；
  - 系统崩溃，损害用户利益；

为什么需要软件测试

- 有缺陷的软件带给软件企业的问题
  - 企业形象下降，产生信任危机
  - 高额的维护费用
  - 产生隐性的商业损失

1.1软件质量

- **Nokia:** 科技，以人为本
- IBM: 把产品质量作为最重要的因素去考虑”
- 微软：足够好的软件
- 华为：为客户服务是华为存在的唯一理由

首先、最后和永远让客户满意

共同的主线一-产品质量

软件质量

- 软件质量具体表现在软件产品固有的特性，如产品的功能性、可靠性、易使用性、效率、可维护性、可移植性和安全性等。

高质量软件具备的特性

- 满足用户的需求；
- 合理处理进度、成本、功能的关系；
- 具备一定的可扩展性和灵活性，适应需求变化；
- 具备一定的可靠性，能有效处理例外情况；
- 保持成本和性能的平衡。

质量成本

- 预防成本；
- 质量评估成本；
- 缺陷修复成本；

1.2软件测试

1. 软件测试的发展历程

20世纪50－60年代

- 软件测试才开始与调试区别开来，成为一种发现软件缺陷的活动

+ 70年代以后

- 软件技术的成熟和完善使得软件测试的规模和复杂度加大，软件测试也逐渐形成了一套完整的体系，逐渐走向规范化。

1. 软件测试的发展历程(续)

- 20世纪80年代早期
  - “质量”的号角才开始吹响
- 20世纪90年代
  - 测试工具终于盛行起来



2. 软件测试的现状

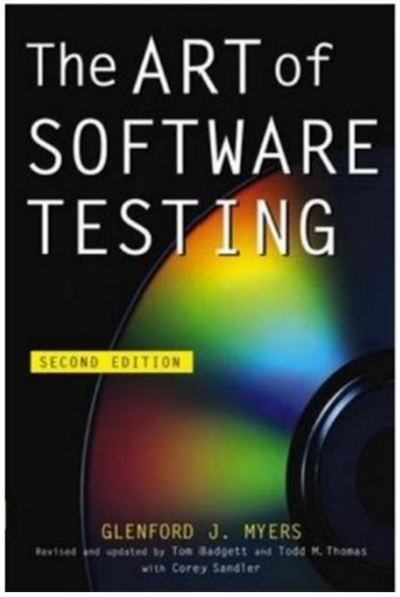
- 与一些发达国家相比，国内测试工作还存在一定的差距。
- 国内测试人员所占比例小，但是，在软件测试实现方面都是相当的，而且向产业化方向发展。

1.2.1软件测试的定义

Myers 认为（1979年）

+ “软件测试是为了发现错误而执行程序的过程”，明确提出了“寻找错误”是测试目的。





1983年IEEE对软件测试的定义

- “使用人工或自动手段运行或测定某个系统的过程，其目的在于检验它是否满足规定的需求或是弄清预期结果与实际结果之间的差别”。
- 明确地提出软件测试以检验是否满足需求为目标。

- 2002年，软件测试的定义
  - 使用人工或者自动手段来运行或测试被测试件的过程，其目的在于检验它是否满足规定的需求并弄清预期结果与实际结果之间的差别。 它是帮助识别开发完成的计算机软件的正确度(correctness)、 完全度(completeness) 和质量(quality) 的软件过程；

软件测试的狭义观点

- 在代码完成后，通过运行程序或软件来查找程序代码或者软件系统中的错误。
- 局限性：不能及时发现软件在需求、设计等上的缺陷，不利于保证软件质量。

软件测试的广义观点

- 为了尽早发现问题，降低软件质量成本，将传统的软件测试范围延伸到需求评审、设计评审、代码评审等活动中去。
- 广义观点将软件测试分为
  - 静态测试
  - 动态测试

软件测试辩证统一观点

- (1)测试是为了证明程序有错，而不是证明程序无错；
- (2)一个好的测试用例在于它发现至今没有发现的错误；
- (3)一个成功的测试是发现了至今未发现的错误的测试；

软件测试的经济成本观点

- “一个好的测试用例在于发现至今未发现的错误”，体现了软件测试的**经济成本**观点。
- 不充分的测试是不负责任的。
- 过分的测试是一种资源的浪费。



- 测试和调试的区别
  - 测试：找错误(证明程序有错)
  - 调试：改错误(使程序正确)

1.2.2软件测试的目的

- 软件测试的目标是尽快尽早地将被测件中所存在的缺陷找出来，并尽快解决这些缺陷，并评估被测件的质量水平。



- 一个成功的测试是发现了至今未发现的缺陷。

测试的“成败”就在于：能否发现缺陷！

注意：

- 测试只能说明软件中存在缺陷，而不能证明软件中不存在缺陷。

主要原因是：

- 测试覆盖率几乎不可能达到100%；
- 去除现有的缺陷可能会产生新的缺陷；
- 测试工程师对产品的理解不能完全代表用户的理解；
- 测试的模拟环境不能完全代表用户的实际使用环境；

1.2.3软件测试的原则

1.可追溯性

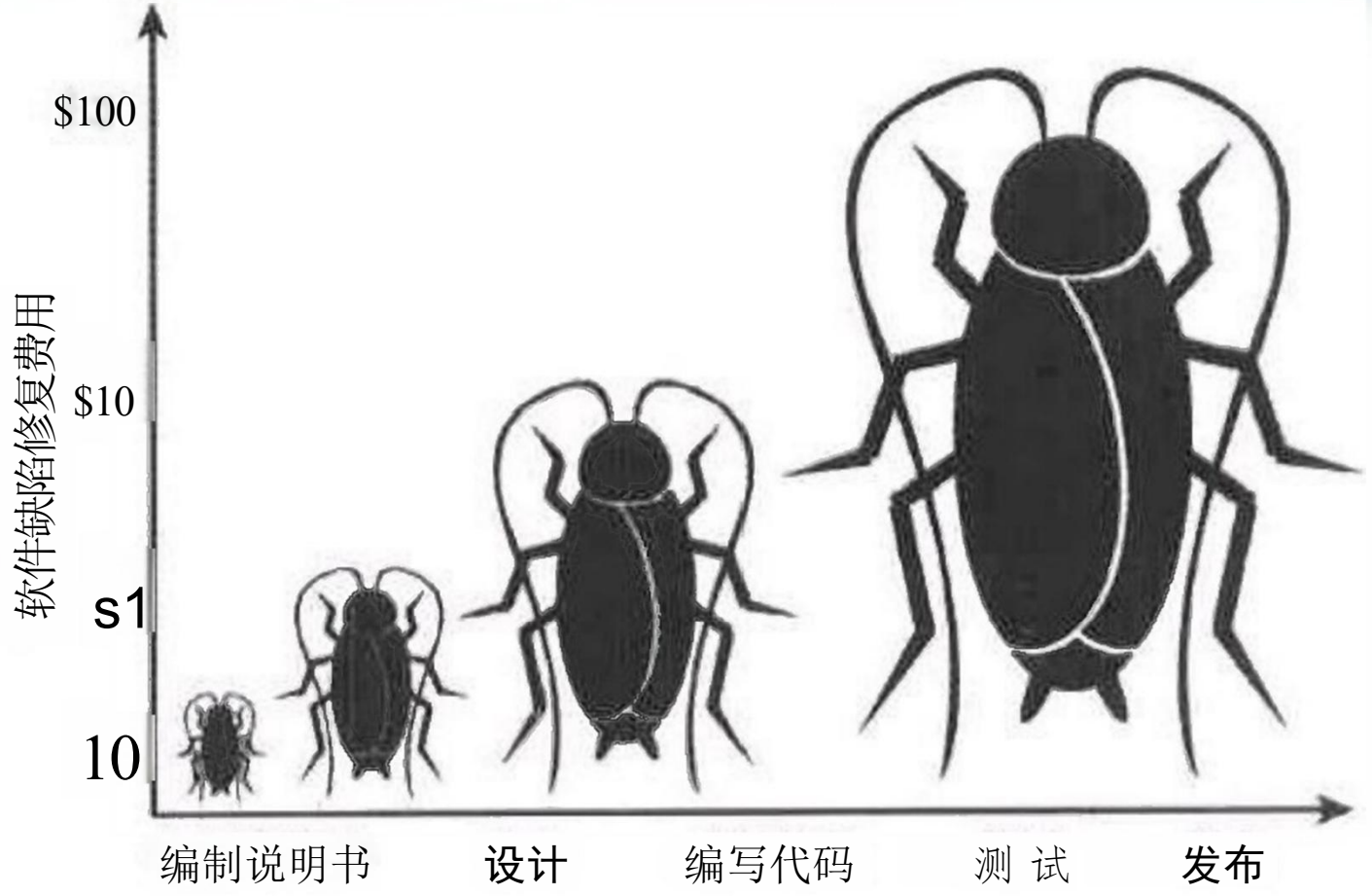
- 所有的测试都应追溯到用户的需求。
- 系统中最严重的错误是那些导致程序无法满足用户需求的错误。



2. 尽早开展预防性测试

- 在软件生命周期各阶段都可能产生错误；
- 缺陷的修改成本随着阶段的推移将急剧上升；
- 缺陷具有放大的特点；

缺陷存在放大趋势



问题发现越早，解决问题的代价就越小。

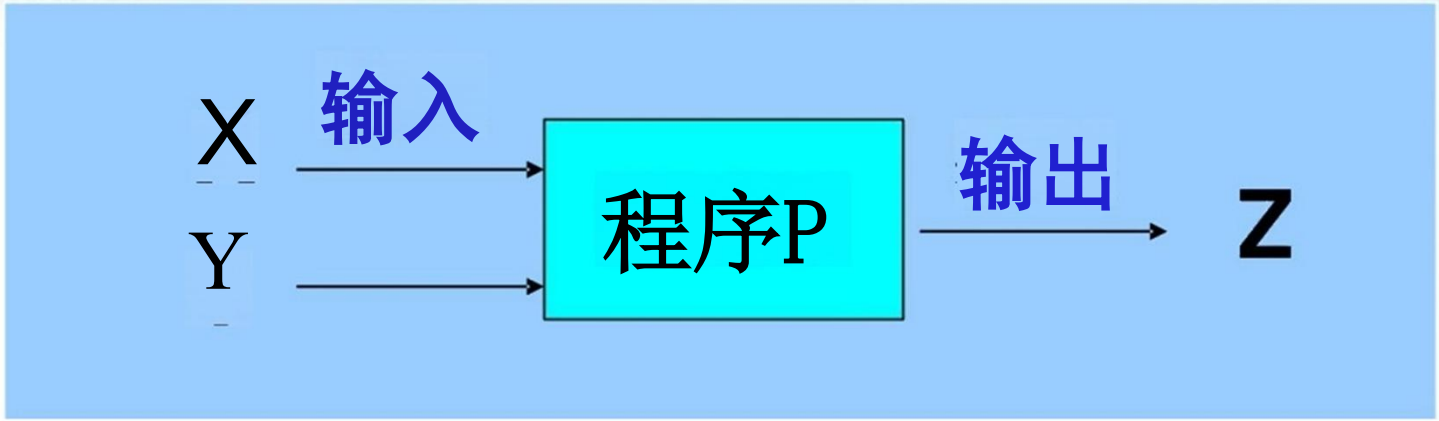
3. 不可能完全的测试

- 输入量太大
- 执行路径太多

软件测试最致命的缺陷就是：

不能进行彻底的测试。

例1:

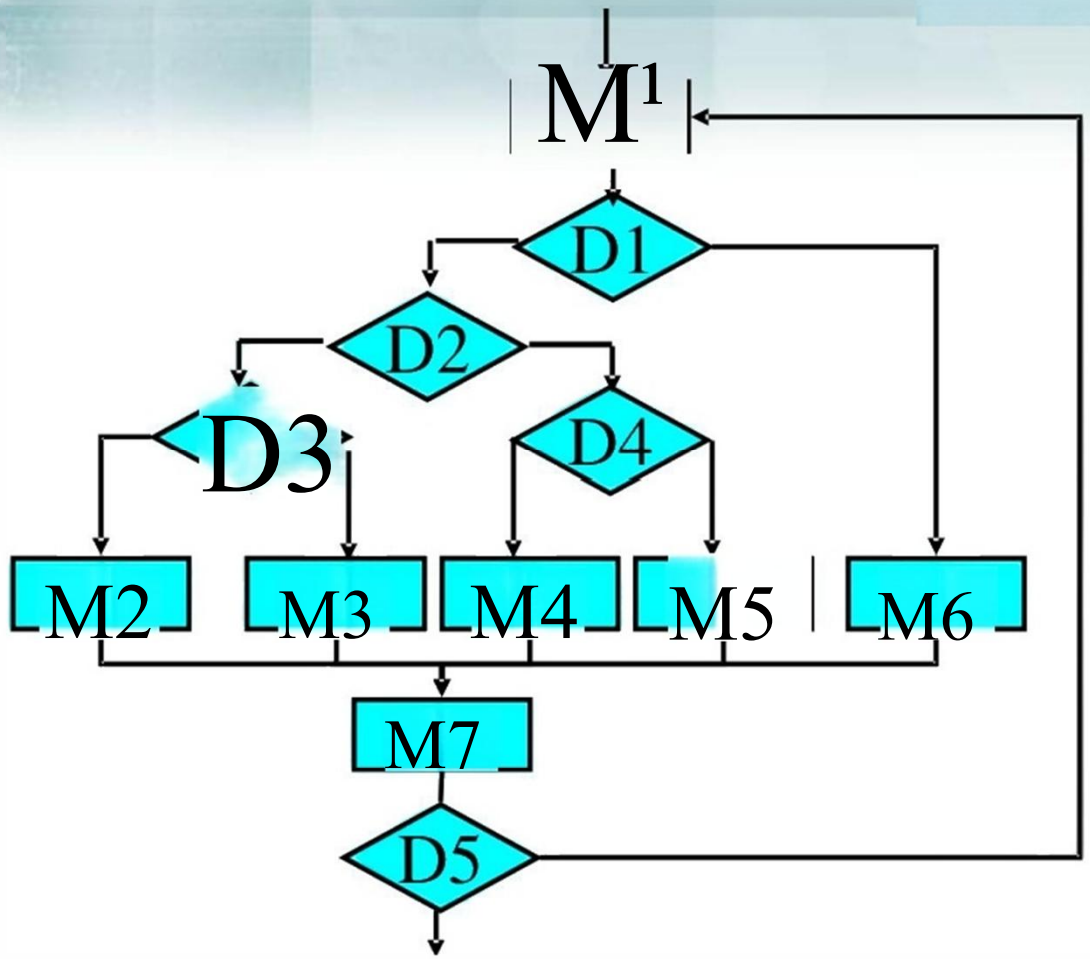


- 若X、Y为所有可能的整数，在字长32位机上测试：

$$X_1、Y_1 \rightarrow Z_1$$
$$\vdots$$
$$Xn、Yn \rightarrow Zn$$

测试次数：$n=2^{32}\times 2^{32}=264\approx 1.84\times 1019$

例2：



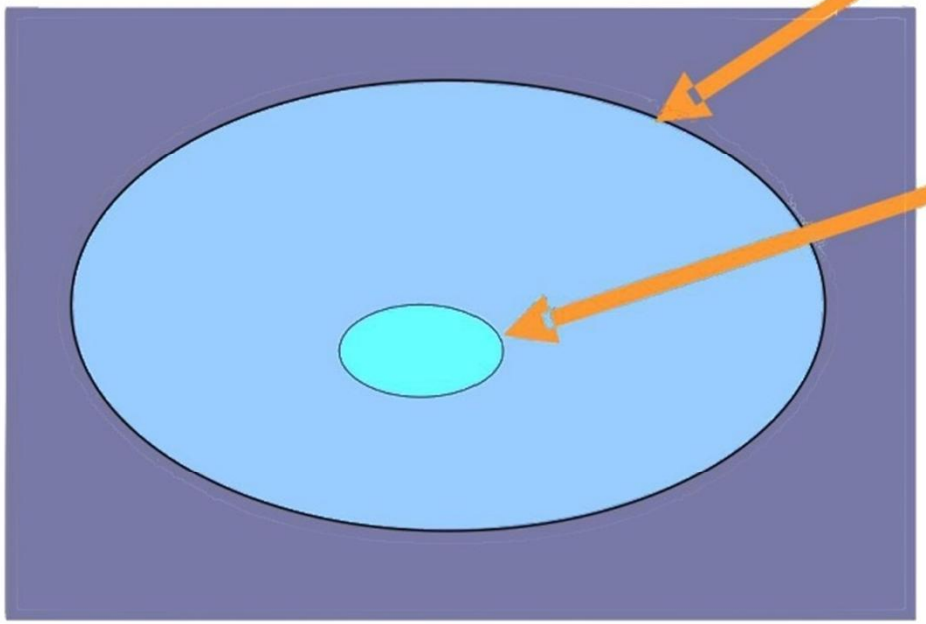
<=20次

循环次数　　01　　2…　　…20

独立路径数　$5^1+5^2+5^3+\cdots\cdots+5^{21}\approx10^14$

每个测试用例(考虑、执行、验证结果)5分钟

共需测试时间　　10亿年



- 应该针对用户最容易遇到的错误进行测试，以便改进测试的有效性。

4. 投入/产出原则

- 不充分的测试是不负责任的；
- 过分的测试是一种资源的浪费；

在满足软件预期的质量标准时，确定质量的投入/产出比。

5. 80-20原则

- 测试发现的错误中80%很可能起源于20%的模块中。应孤立这些疑点模块重点测试。
- 在所测程序段中，若发现错误数目多，则残存错误数目也比较多。

6. 设立独立的测试机构或委托第三方测试

避免测试自己的程序

1. 程序员轻易不会承认自己写的程序有错误；

2. 程序员的测试思路有局限性，做测试时很容易受到编程思路的影响；

3. 程序员测试不具有典型性。

7. 回归测试

- 程序修改后必须进行回归测试，避免引入新的错误。

8. 设计周密的测试用例

- 软件测试的本质就是针对要测试的内容 确定一组**测试用例**。
- 测试用例至少应包括：
  - 执行测试用例前，应满足的前提条件
  - 输入
  - 预期输出
- 设计测试用例时，应当包括合理的输入 条件 和**不合理的输入条件**。

人们对软件测试的误解

- **误解一：测试是开发后期的一个阶段**
  - 软件测试不等于程序测试，软件测试应该涵盖整个软件生命周期；
- **误解二：测试技术要求不高，比编程容易多了**
  - 好的测试人员不仅要学握各种测试技术和测试工具，还要具备丰富的编程经验和对BUG的敏感。

- **误解三：**
  - 软件测试是测试人员的事，与开发人员无关。
- **误解四：**
  - 好的测试一定要使用很多的测试工具。
- **误解五：**
  - 软件测试就是运行一下软件，看看结果对不对。
- **误解六：**
  - 软件出了重大问题，是测试人员的事。

1.3软件缺陷

- 缺陷：最终产品同用户的期望不一致。
  - 功能错误；
  - 功能遗漏；
  - 超出需求的部分；
  - 性能不符合要求；
  - ……

世上不存在没有缺陷的软件

软件缺陷的范围很广：

1．软件未达到产品说明书要求的功能；

2．软件出现了产品说明书指明不会出现的错误；

3．软件功能超出产品说明书指明范围；

4．软件未达到产品说明书虽未指出但应达到的目标；

5．软件难以理解、不易使用、运行速度缓慢、最终用户认为不好等。

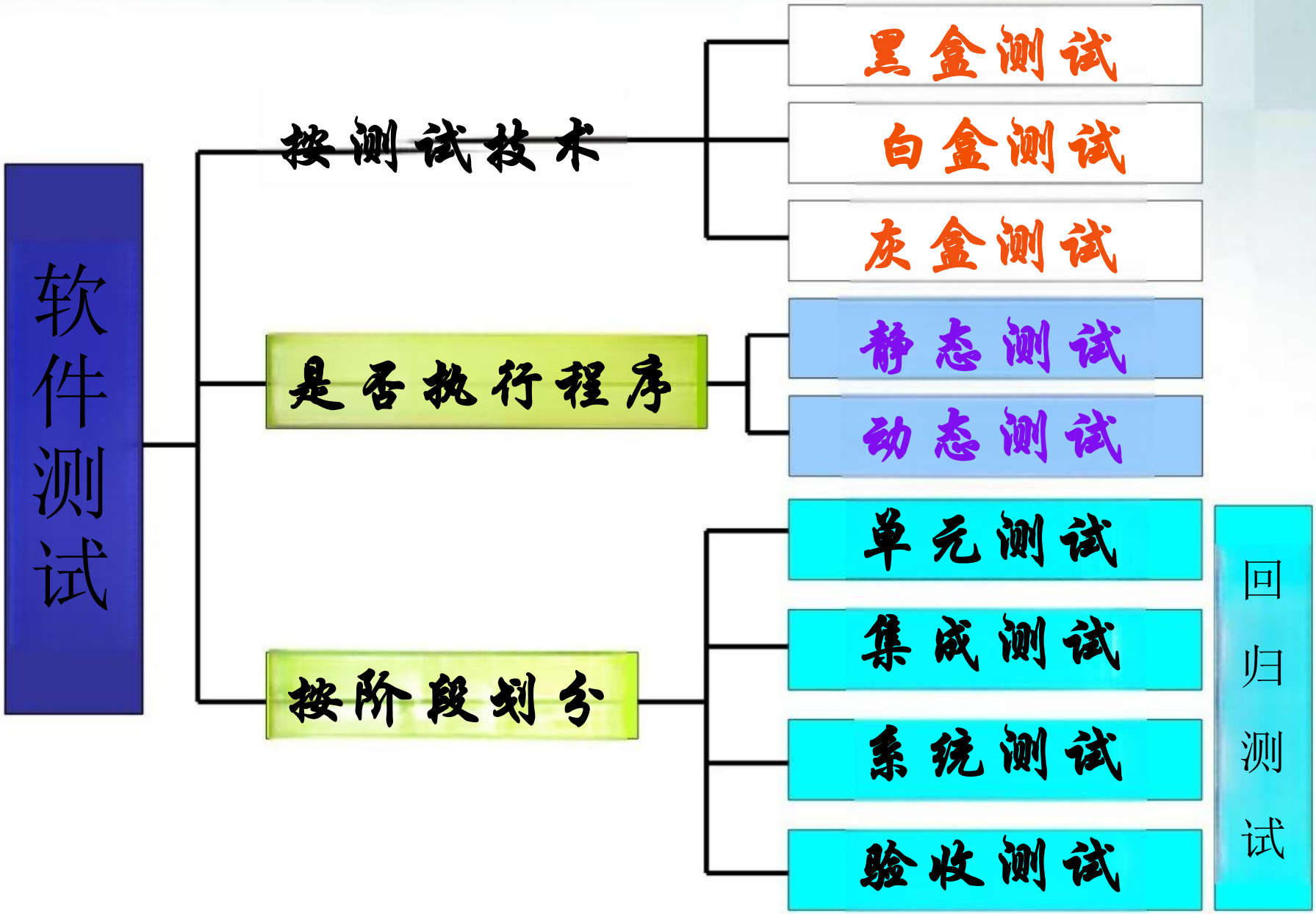
难以说清的软件缺陷

- 如果软件中的缺陷没有被人发现它算不算软件缺陷？

- 古谚语：“一片树叶飘落在森林中没有人听见，谁能说它发出了声音？”

眼见为实

1.3 软件测试分类


软件测试
按测试技术
黑盒测试
白盒测试
灰盒测试
是否执行程序
静态测试
动态测试
按阶段划分
单元测试
集成测试
系统测试
验收测试
回归测试

- 一、按照开发阶段划分：
  - 单元测试(Unit Testing)
  - 集成测试(Integration Testing)
  - 系统测试(System Testing)
  - 验收测试(Verification Testing)

二、按测试实施组织划分

- 开发方测试
- 用户测试（β测试）
- 第三方测试

三、按是否执行软件（程序）

十静态测试

不运行被测程序，而是通过在对软件进行分析、检查和审阅达到测试目的的

十动态测试

- 通过运行软件来检验软件的动态行为和运行结果的正确性

静态测试

①代码检查：代码审查、代码走查、桌面检查

②静态分析；

③技术评审；

- 静态测试除了进行人工测试外，还可以借助于计算机辅助分析。
- 静态测试工具： Logiscope、PcLint

例 如 ：

```
PI=3.14;
a=0;
 while(a<100){
        b=b+100;
        C=b*PI;
}
```



- 动态测试

  (按是否了解源代码)

  - 黑盒测试 (Black-Box Testing)
  - 白盒测试 (White-Box Testing)
  - 灰盒测试 (Gray-Box Testing)

黑盒测试

- 边界分析；
- 等价类划分；
- 因果图；
- 判定表；
- 正交试验设计
- 状态测试；
- 场景测试；

- 白盒测试
  - 逻辑覆盖
  - 基路径测试
  - 数据流测试
  - 域测试
  - 程序插装
  - 变异测试

- 四、按是否使用工具划分
  - 手工测试
  - 自动化测试

五、按测试行为划分

- 有计划测试
- 随机测试

测试类型

- 按照企业中实际工作需要，测试包含下面的类型：

功能测试

- 功能测试的基本方法是构造一些合理输入(在需求范围之内)，检查输出是否与期望的相同。

性能测试

- 性能测试主要测试软件的性能，包括负载测试，强度测试，数据库容量测试，基准测试以及基准测试

用户界面测试

- 测试用户界面的风格是否满足客户要求，文字是否正确，页面是否美观，文字，图片组合是否完美，操作是否友好等等。

数据和数据库完整性测试

- 测试关系型数据库完整性原则以及数据合理性测试。

安全性和访问控制测试

- 应用程序级别的安全性(对数据或业务功能的访问)
- 系统级别的安全性(对系统的登录或远程访问)

健壮性测试

- 健壮性是指在异常情况下，软件还能正常运行的能力。

兼容性测试

- 测试软件在一个特定的硬件/软件/操作系统/网络等环境下的性能如何。

安装/反安装测试

- 安装测试包括测试安装代码以及安装手册。

故障转移和恢复测试

- 指当主机软硬件发生灾难时候，备份机器是否能够正常启动，使系统是否可以正常运行。

十文档测试

- 文档测试主要包括需求文档测试，设计文档测试等。

十本地化测试(语言测试)

- 是指为各个地方开发产品的测试，如英文版，中文版等等，包括程序是否能够正常运行，界面是否符合当地习俗等等，特别测试在A语言环境下运行B语言软件，是否正常。

十分辨率测试

- 测试在不同分辨率下，界面的美观程度。

α测试

- α测试是指软件开发公司组织内部人员模拟各类用户行对即将面市软件产品(称为α版本)进行测试。α测试的关键在于尽可能逼真地模拟实际运行环境和用户对软件产品的操作。经过α测试调整的软件产品称为β版本。

β测试

- β测试是指软件开发公司组织各方面的典型用户在日常工作中实际使用β版本，并要求用户报告异常情况、提出批评意见。然后软件开发公司再对β版本进行改错和完善。

练习

- 请例举出10各你在生活中遇到的缺陷，并简单说明给你带来的影响或不便。

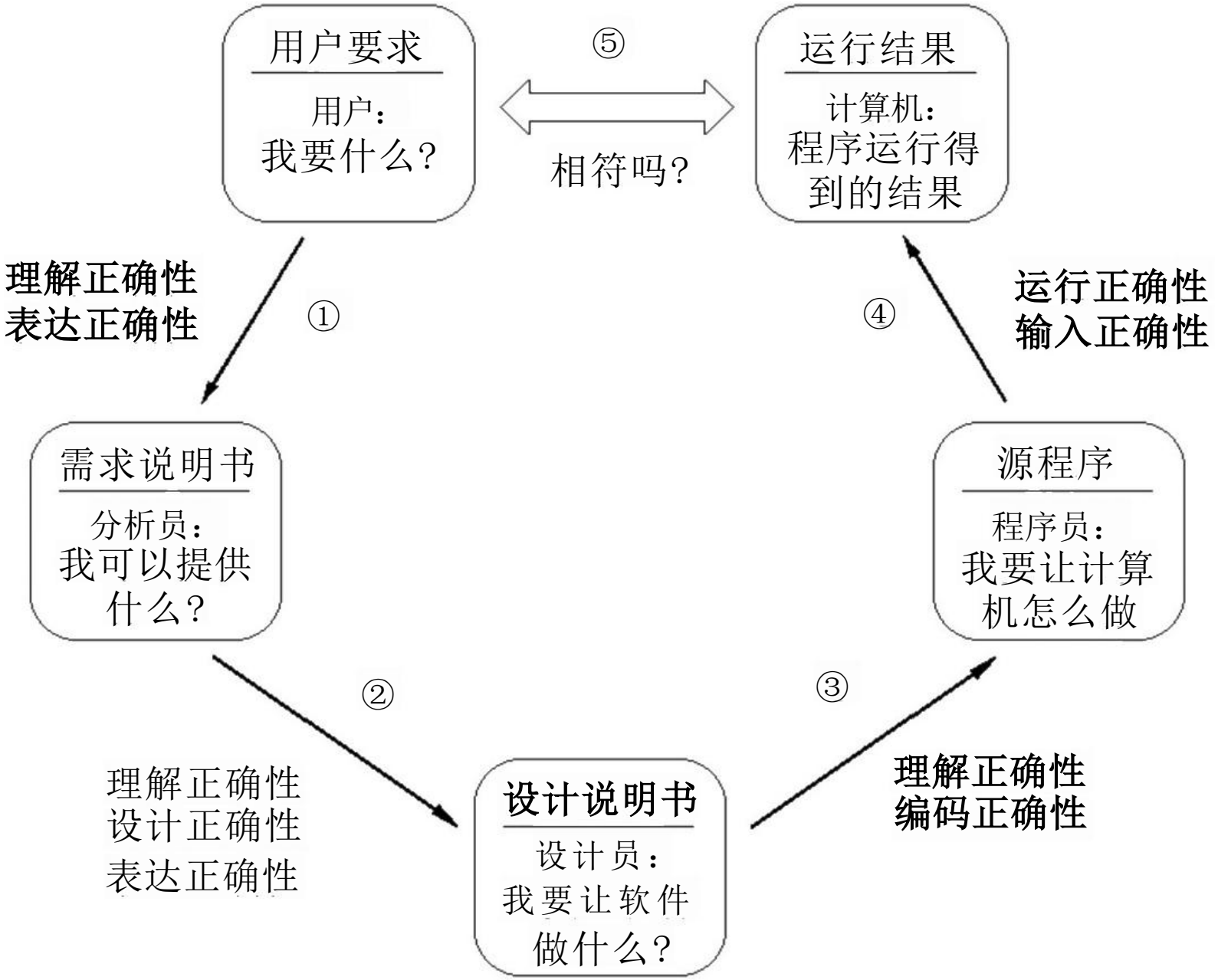


问题与讨论

本章内容

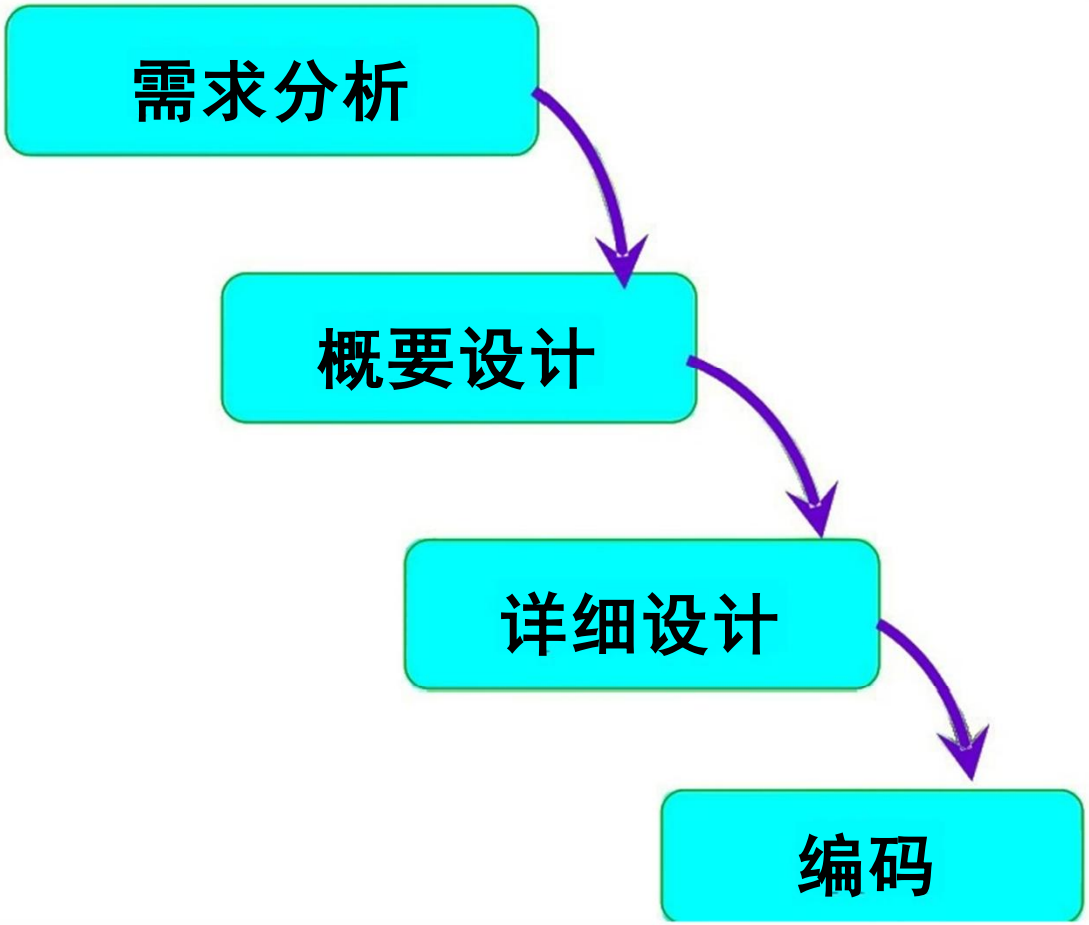
- 软件测试模型
- 软件测试流程

软件的一般实现过程

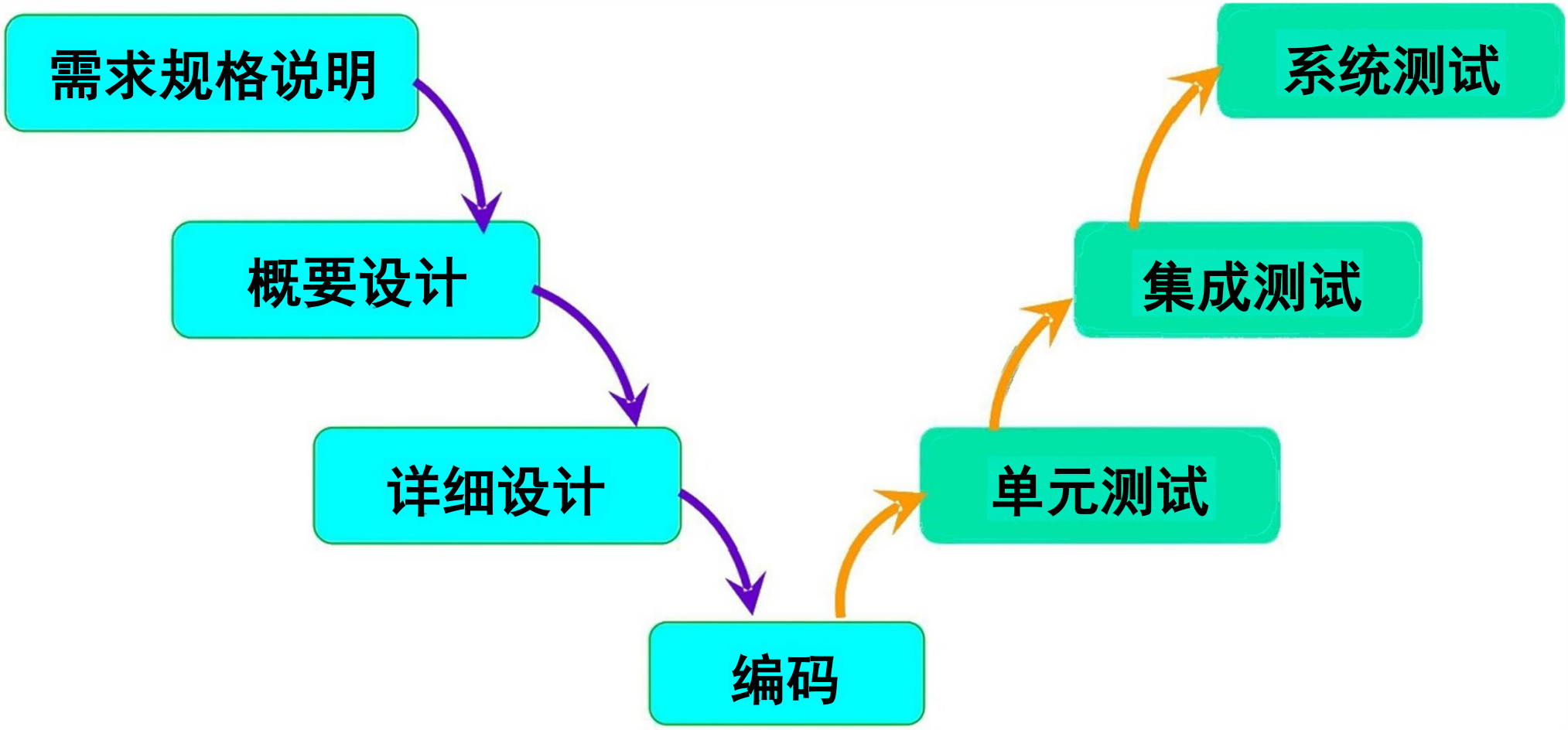

用户要求
用户：
我要什么？
⑤
相符吗？
运行结果
计算机：
程序运行得
到的结果
理解正确性
表达正确性
①
④
运行正确性
输入正确性
需求说明书
分析员：
我可以提供
什么？
源程序
程序员：
我要让计算
机怎么做
②
③
理解正确性
设计正确性
表达正确性
设计说明书
设计员：
我要让软件
做什么？
理解正确性
编码正确性

测试与软件开发各阶段的关系

- 软件开发是一个自顶向下，逐步细化的过程。



测试过程是依相反顺序安排的，是自底向上，逐步集成的过程。



瀑布式生命周期

2.1软件测试模型

V模 型

W模 型

H 模 型

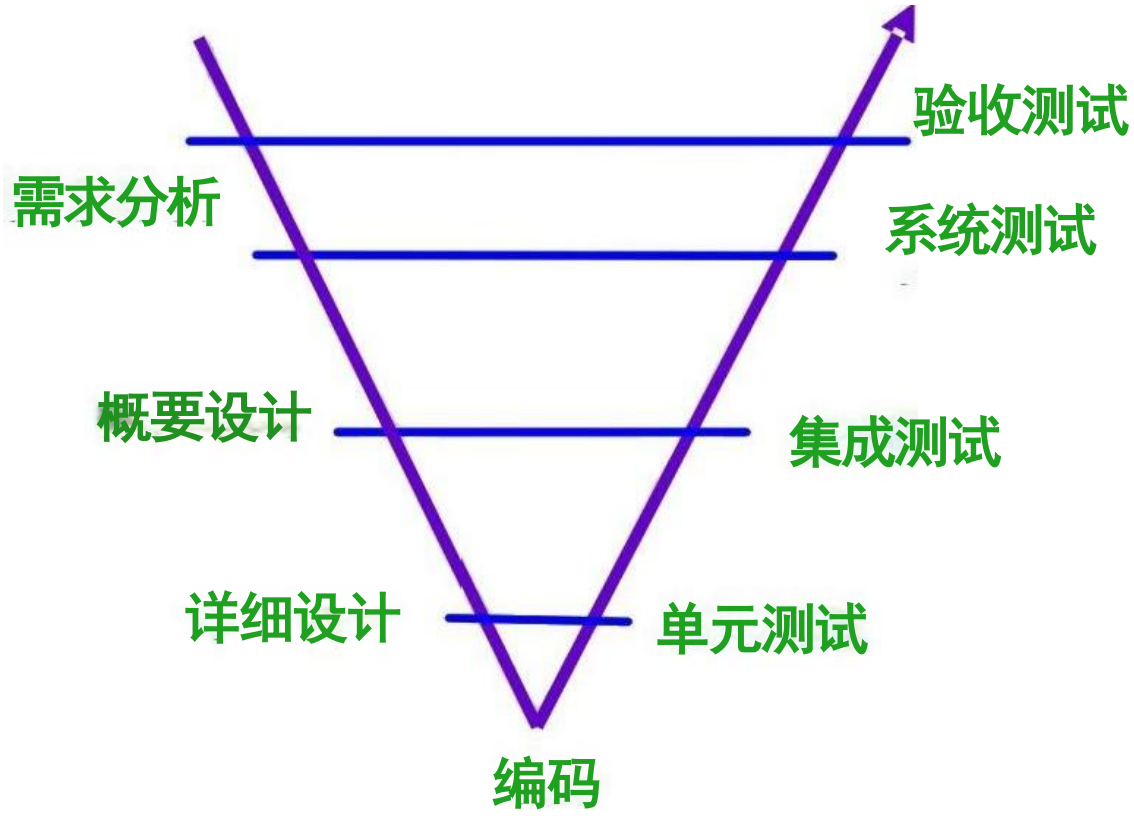
X 模 型

测试前置模型(测试驱动模型)

思考

- 软件是什么？
- 软件测试需要测什么？

V 模型

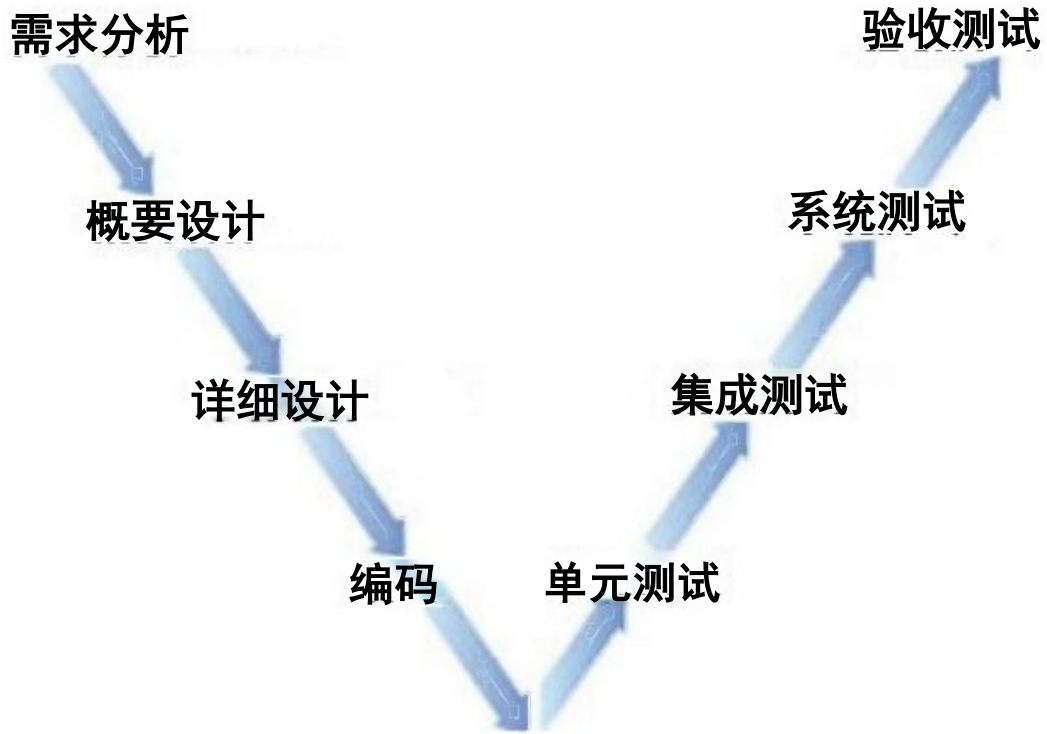


- 由Paul Rook在20世纪80年代后期提出的。

  从左到右，描述了基本的开发过程和测试行为。

- 非常明确地标明了测试过程中存在的不同级别，描述了这些测试阶段和开发过程期间各阶段的对应关系。

V 模型(测试与开发阶段对应关系)



- 单元和集成测试检测程序的执行是否满足软件设计的要求；
- 系统测试检测系统功能、性能的质量特性是否达到系统要求的指标；
- 验收测试确定软件的实现是否满足用户需要或合同的要求。

V 模型的缺陷

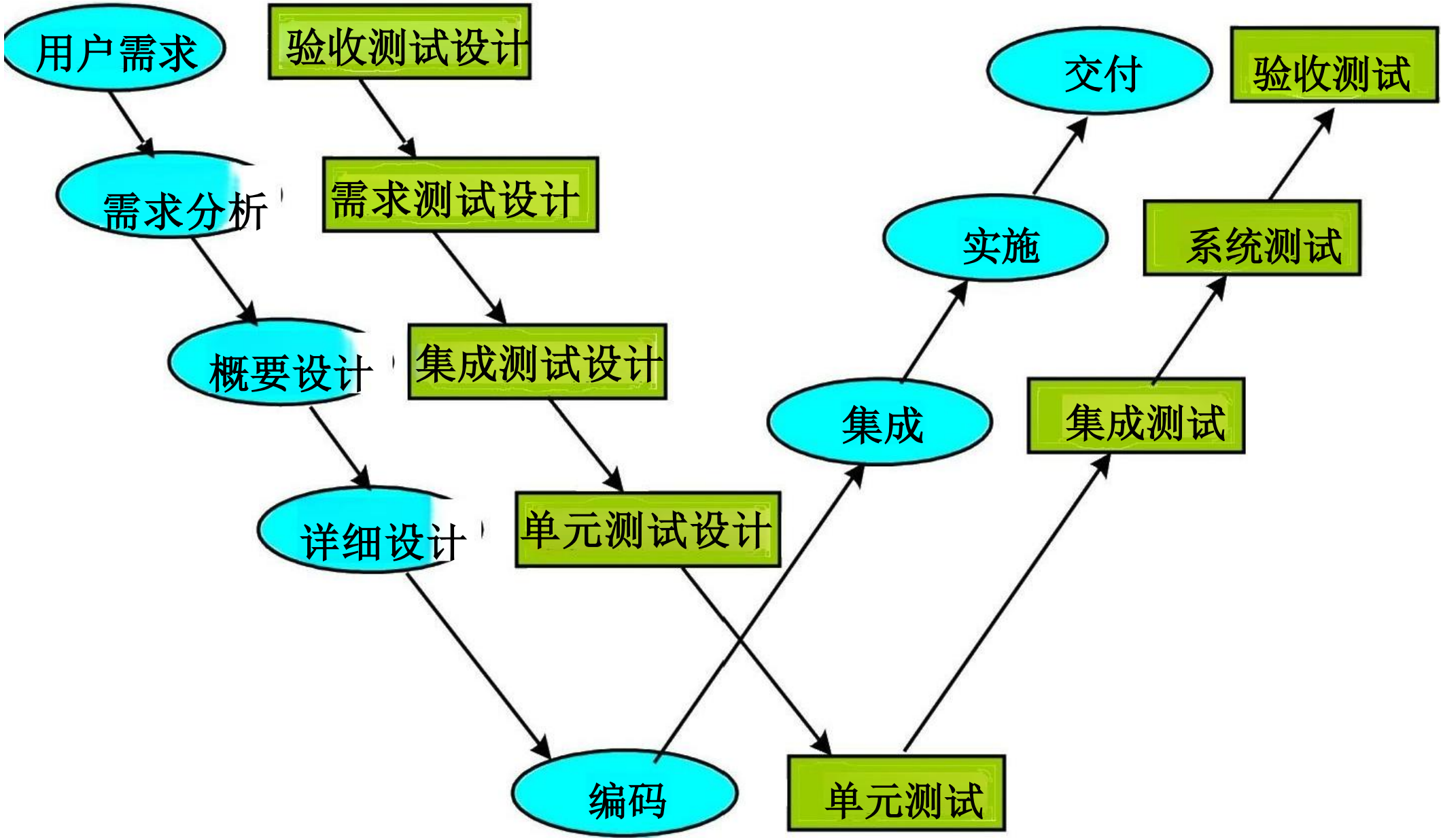
- 仅仅把测试过程作为在需求分析、系统设计及编码之后的一个阶段。

  只针对程序进行的寻找错误的活动，忽视了测试活动对需求分析，系统设计等活动的验证和确认的功能，直到后期的验收测试才被发现。

W模型

- W模型由Evolutif公司提出。
- W模型从V模型演化过来，实际上开发是V，测试也是与此并行的V。
- 相对于V模型，W模型增加了软件各开发阶段中应同步进行的验证和确认活动。

W 模型


用户需求
验收测试设计
交付
验收测试
需求分析
需求测试设计
实施
系统测试
概要设计
集成测试设计
集成
集成测试
详细设计
单元测试设计
编码
单元测试

W 模型优点

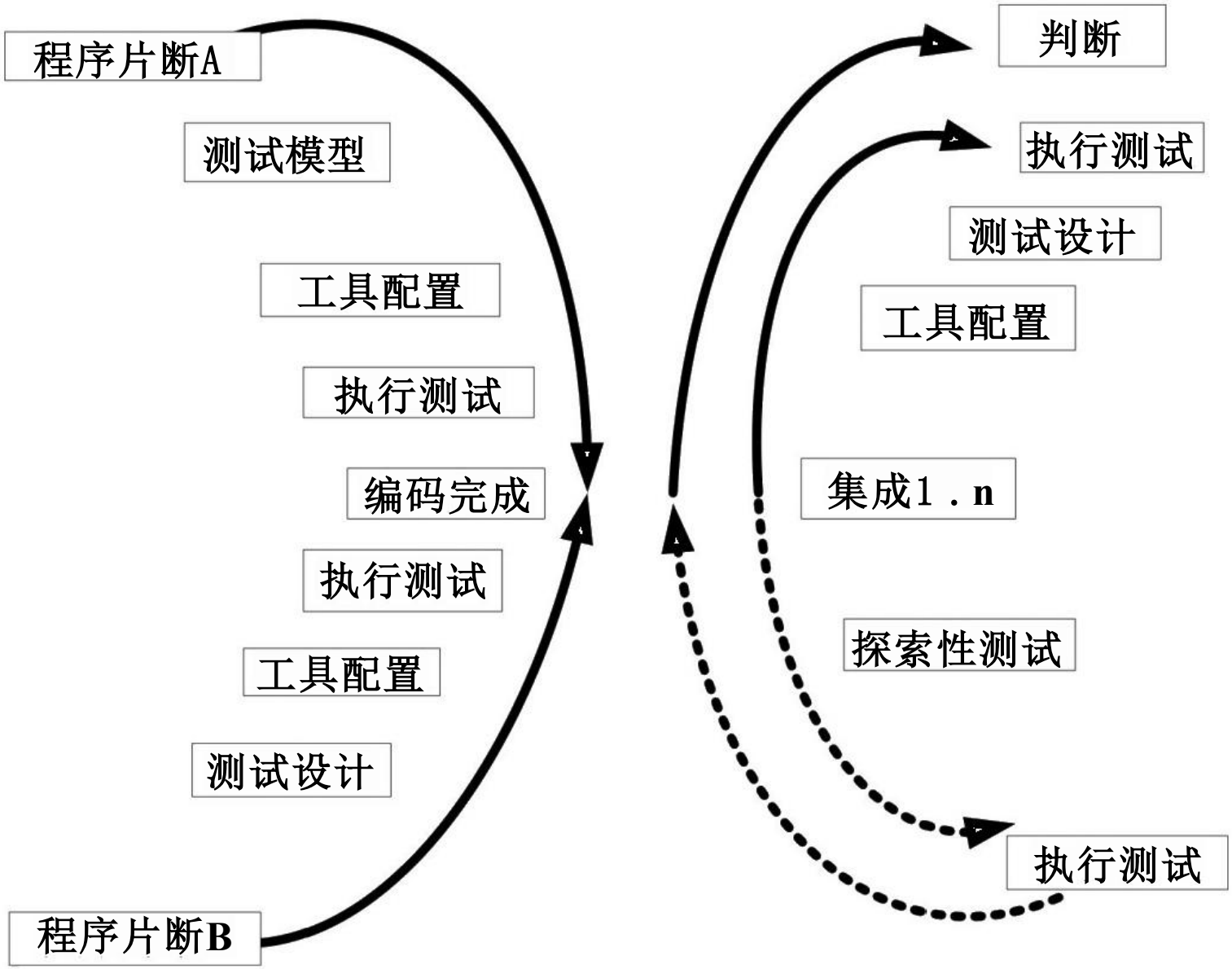
- 软件测试贯穿于整个开发周期；
- 测试与开发可同步进行；
  - 每个软件开发活动结束后就可以执行相应的测试，如：在需求分析结束后，就可以进行需求分析测试。

W模型缺点

- 在W模型中，需求、设计、编码等活动被视为串行的，同时，测试和开发活动也保持着一种线性的前后关系，上一阶段完全结束，才可正式开始下一个阶段工作。这样就无法支持迭代的开发模型。

  对于当前软件开发复杂多变的情况，W模型并不能解除测试管理面临的困惑。

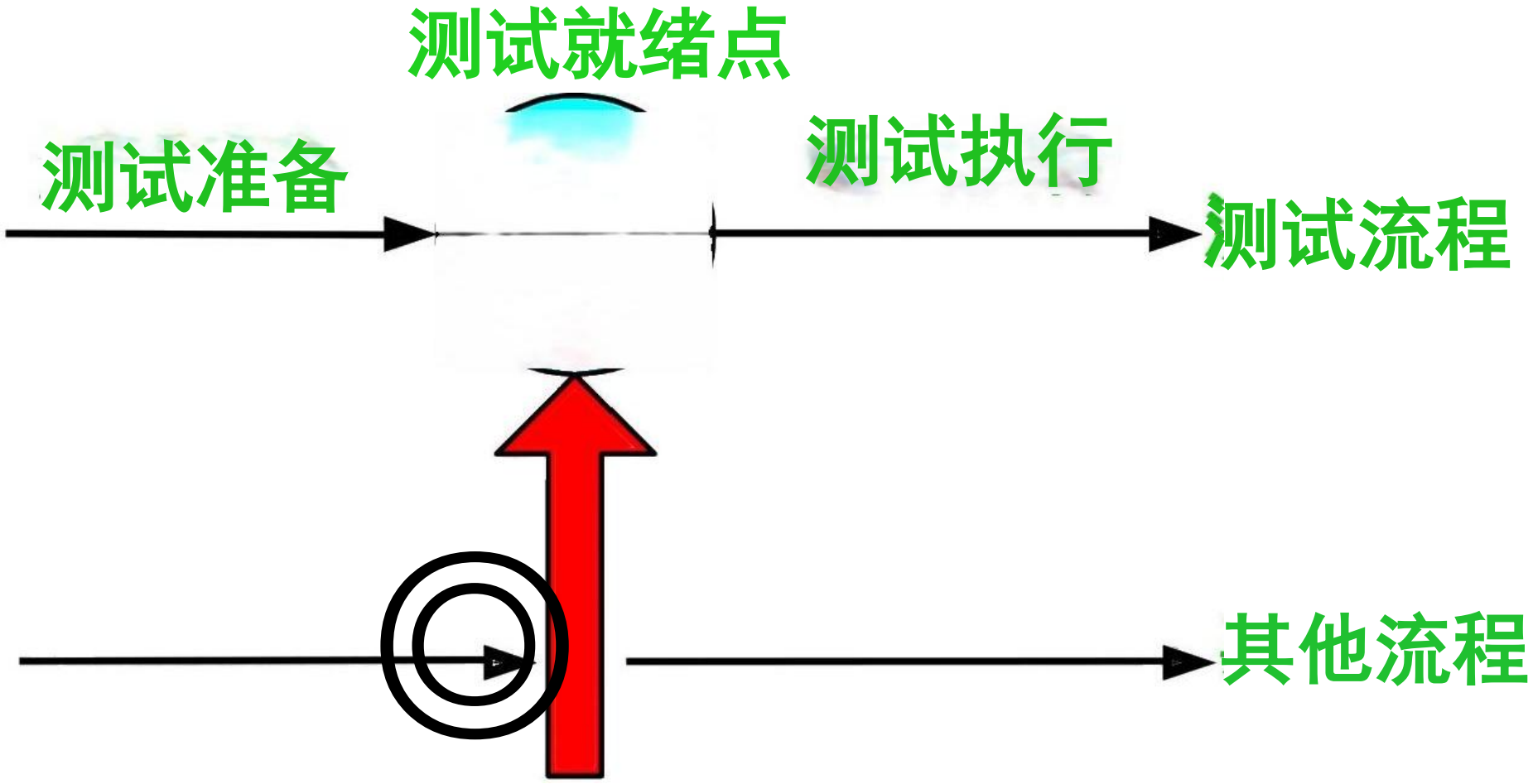
X模 型


程序片断A
测试模型
工具配置
执行测试
编码完成
执行测试
工具配置
测试设计
程序片断B
判断
执行测试
测试设计
工具配置
集成1．n
探索性测试
执行测试

X模型

- 很好地处理测试与开发的交接过程(交接的过程是一个时间段，而不是一个点)
- 左边描述的是针对单独程序片段所进行的相互分离的编码和测试，此后将进行频繁的交接，通过集成最终合成为可执行的程序，然后再对这些可执行程序进行测试。已通过集成测试的成品可以进行封装并提交给用户，也可以作为更大规模和范围内集成的一部分。多根并行的曲线表示变更可以在各个部分发生。
- X模型还定位了探索性测试，这是不进行事先计划的特殊类型的测试，给有经验的测试人员在测试计划之外发现更多的软件缺陷。

H 模型



H模型示意图

H模型特点

H模型充分地体现了测试过程。H模型说明：

1、软件测试不仅仅指测试的执行，还包括很多其他的活动。

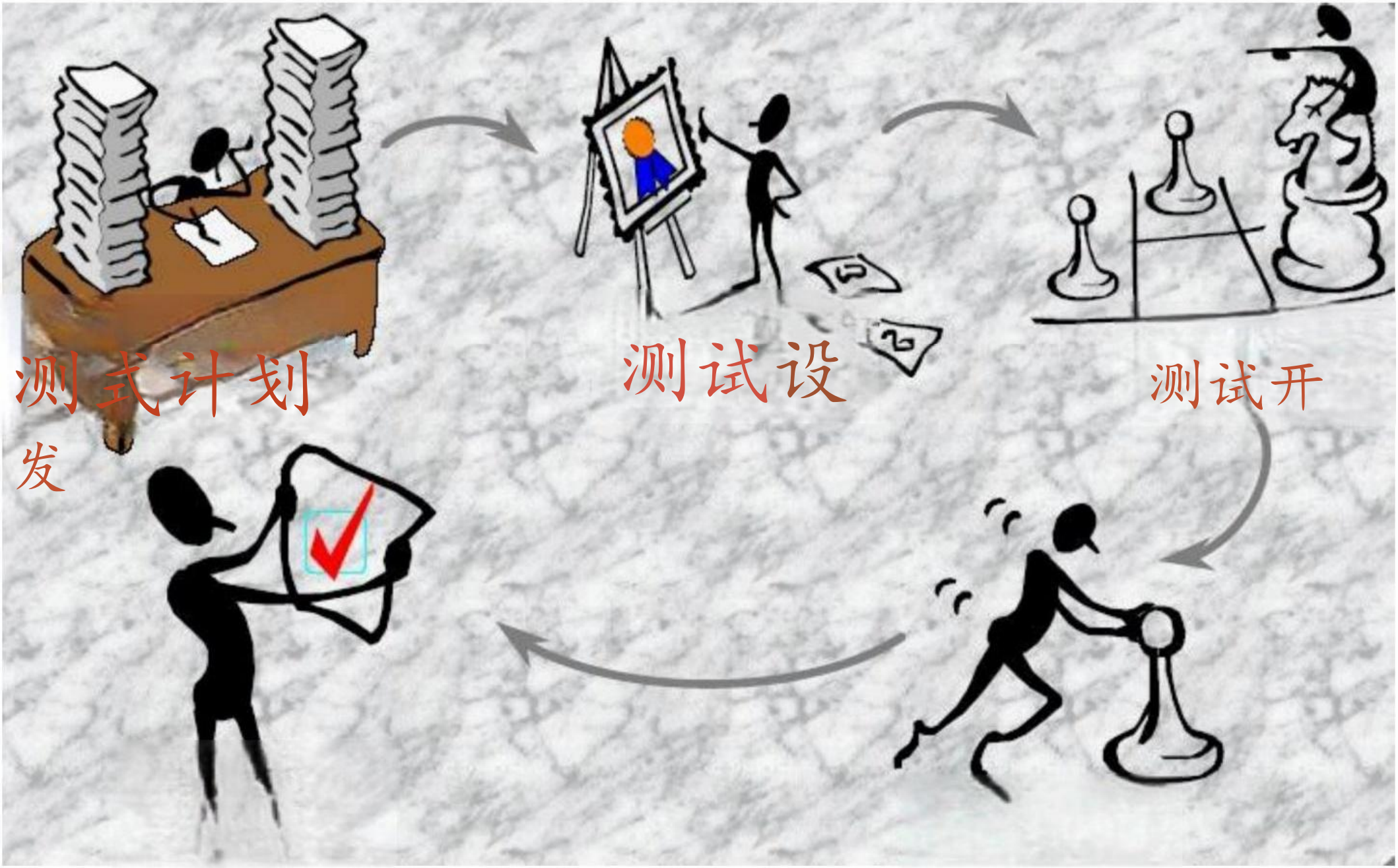
2、软件测试是一个独立的流程，贯穿产品的整个开发周期，与其它流程并发进行。

+ 3、软件测试要尽早准备，尽早执行。

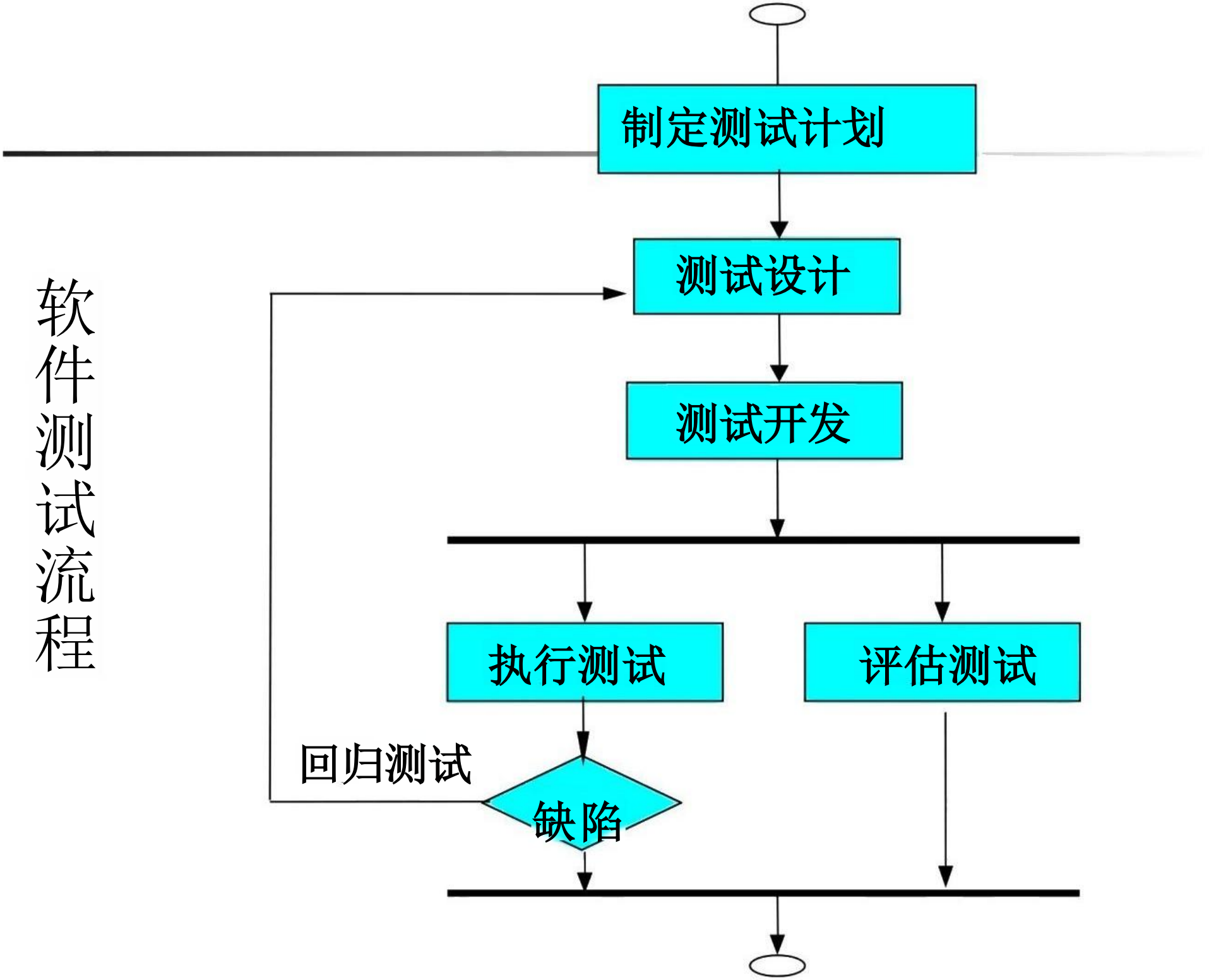
2.2软件测试流程

- 软件测试工作必须要通过制定测试计划、测试设计、测试开发、测试执行、测试评估几个阶段来完成。

软件测试流程



软件测试流程



（1）制定测试计划

- 测试计划是对每个产品，或是对各个开发阶段的产品开展测试的策略。
- 一个完整的测试计划应该包含以下几个方面：

  对测试范围的界定；

  风险的确定；

  资源的规划；

  时间表的制定；

（2）测试设计

- 测试设计阶段要设计测试用例和测试过程，要保证测试用例完全覆盖测试需求。

- 测试设计阶段最重要的是如何将测试需求分解，如何设计测试用例。

（3）测试开发

- a. 准备测试资源
  - (1) 测试技术准备
  - (2)配置软件、硬件环境
  - (3) 人 员
- b. 获得测试数据
  - (1) 正常事务的测试
  - (2)使用无效数据的测试

C. 测试脚本

十所谓脚本，是完整的一系列相关终端的活动。

一般测试脚本有5个级别：

单元脚本

- 并发脚本
- 集成脚本
- 回归脚本
- 强度/性能脚本

d. 辅助测试工具

优秀的办公处理软件

错误跟踪系统

- 自动测试工具
- 软件分析工具
- 好的操作系统
- 多样化平台

(4) 测试执行

- 测试执行的过程由4个部分组成

  输入：要完成工作所必须的入口标准或可交付的结果。

  + 执行过程：从输入到输出的过程或工作任务。
  + 检查过程：确定输出是否满足标准的处理过程。

  输出：推出标准或工作流程产生的可交付的结果。

(5）评估测试

- 软件测试的主要评测方法包括：
  - 覆盖评测
  - 质量评测
  - 性能评测

a. 覆盖评测

覆盖指标提供了“测试的完全程度如何？”这一问题的答案。

最常用的覆盖评测是基于需求的测试覆盖和基于代码的测试覆盖。

b. 质量评测

在测试过程中，已发现缺陷的评估提供了最佳的软件质量指标。

C. 性能评测

+ 评估测试对象的性能时，侧重于获取与行为相关的数据，如响应时间、事务处理数、内存占用率、操作可靠性等。

测 试 用 例

测试用例：是为某个特定测试目标而设计的，是测试操作过程序列、条件、期望结果及相关数据的一个特定的集合。

使用测试用例的好处

避免盲目测试，提高测试效率；

估算测试工作量；

- 降低工作强度、缩短项目周期；
- 方便书写软件测试缺陷报告；

测试用例包括：

测试目标：为什么测试

- 测试对象：测什么

测试环境：执行测试用例运行的环境

测试前提：什么条件下测试

输入数据

操作步骤

预期结果

设计时需考虑如下基本因素：

十 测试用例应具有代表性、典型性；

十 考虑正确的输入， 错误或异常的输入；

不仅检查程序是否实现预期功能，还应检查程序是否作了不该做的事；

考虑用户实际使用场景；

■ 测试用例要基于用户实际的可能场景，从用户角度模拟系统的输入

■ 测试用例模板

课堂 练 习 ：

用户名：

密　码：



说 明 ：

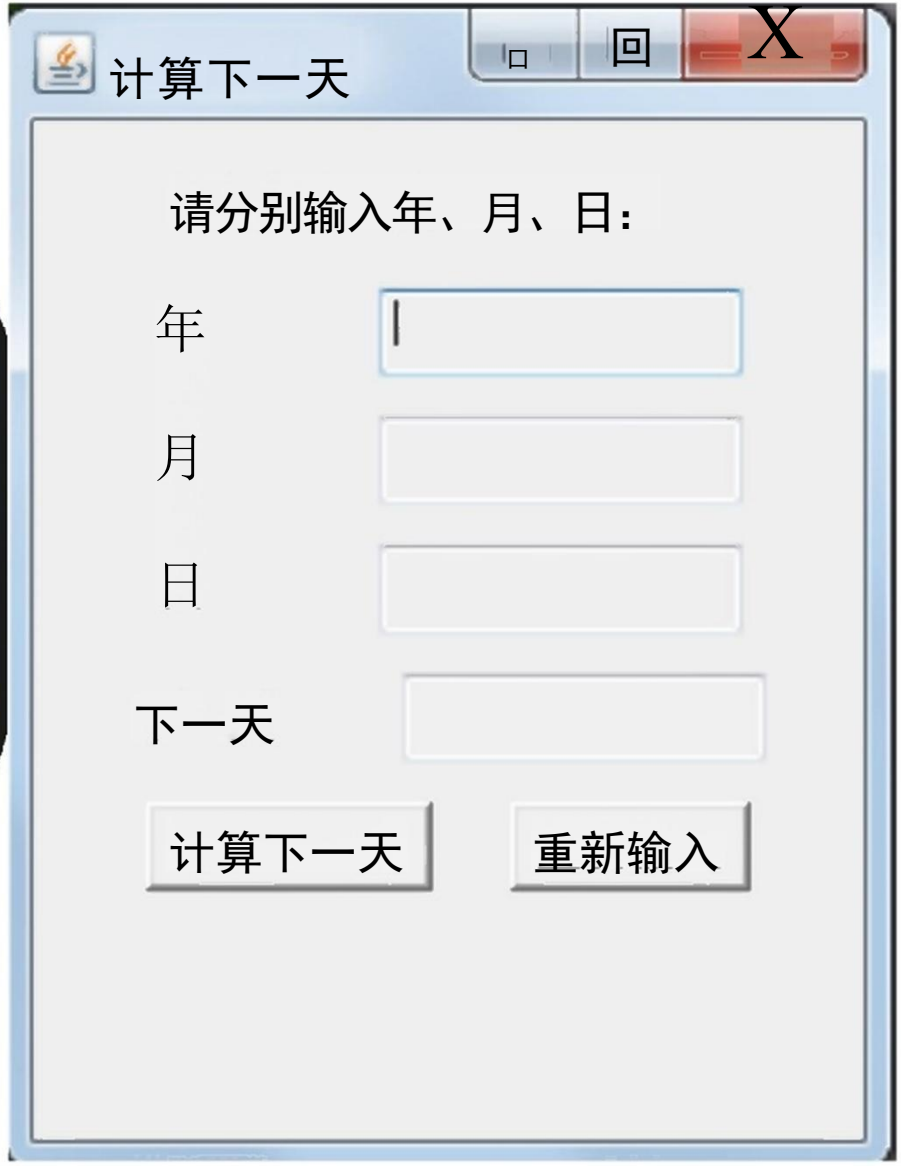
用　户pass/pass　为有效登录用户；

用　户past/pass　为无效登录用户；

用　户pass'jean/password　为有效登录用户；

练习

◆有一个“计算下一天”的程序，请设计测试输入数据。

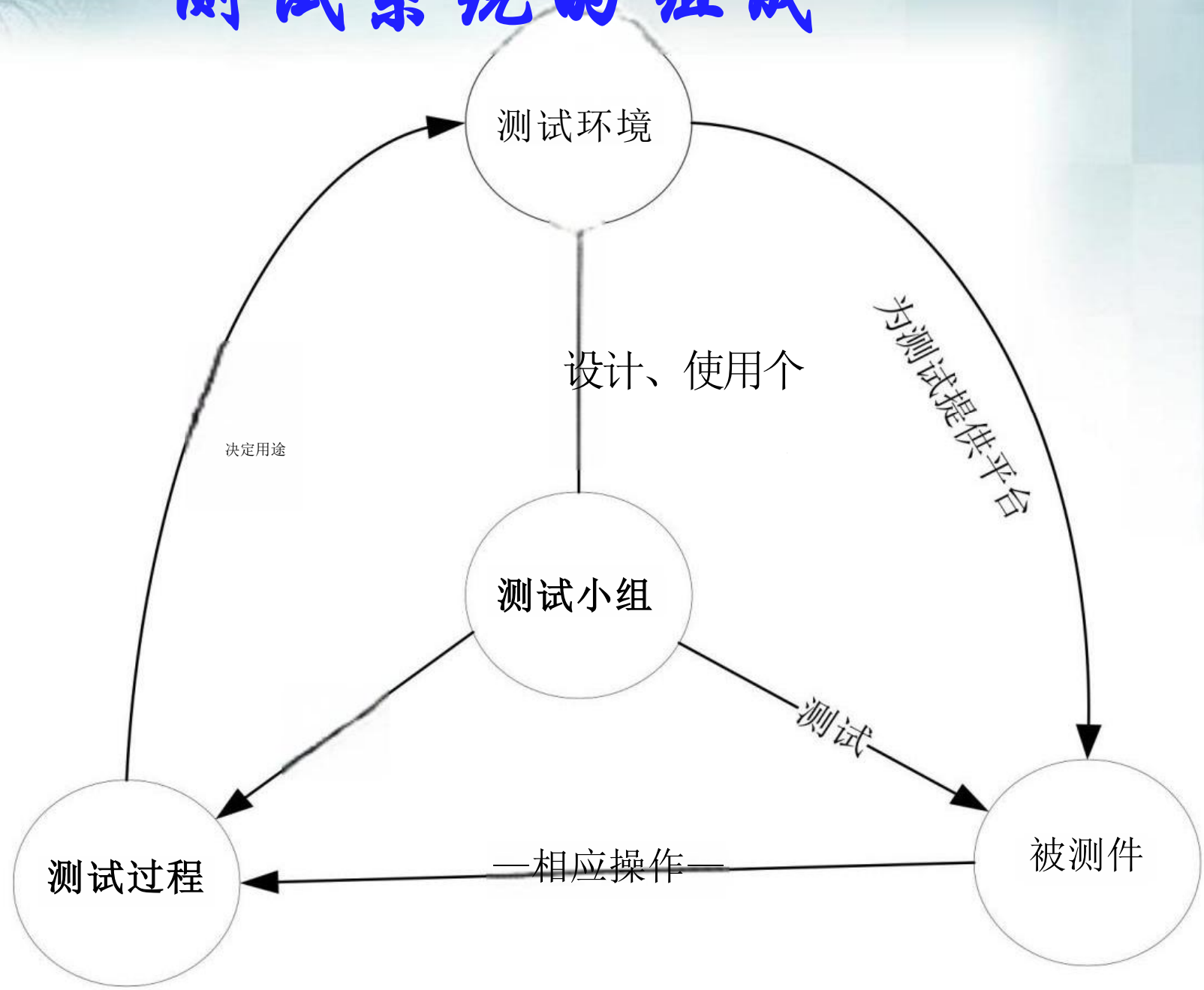


软件测试管理

本章内容

- 测试团队建设与管理
- 软件测试计划
- 缺陷管理

测试系统的组成



测试管理

- 测试管理的内容包括：
  - 测试团队建设
  - 测试过程和方法建设
  - 测试管理
  - 测试执行

测试活动

- 测试活动包括：
  - 静 态 评 审
    - 需求评审、概要设计评审、详细设计评审
  - 动 态 测 试
    - 单元测试、集成测试、系统测试、验收测试
  - 版 本 发 布
  - 维护

3.1测试团队建设与管理

- 相传释迦牟尼曾问他的弟子：“一滴水怎样才能不干涸?”
- 弟子们面面相觑，无法回答。
- 释迦牟尼说：“把它放到大海里去。”

个人再完美，也就是一滴水；一个优秀的团队就是大海。

3.1测试团队建设与管理

■ 团队概念

团队是指一群互助互利、团结一致为统一目标和标准而坚毅奋斗到底的一群人。

团队不仅强调个人的业务成果，更强调团队的整体业绩。

1+1>2

古人云：人心齐，泰山移。

3.1.1测试团队建设

招募测试人员；

测试团队制度建设；

测试流程建设；

- 测试人员职责分工；
- 团队成员能力的提高；

测试团队的建设

有效的软件测试团队的特征：

对软件项目的测试目标有清晰的理解；

- 对每位测试工程师的角色和职责有明确的期望；
- 以目标为导向；

高度的互助合作；

测试团队的基本职责：

- 尽早地发现尽可能多的缺陷；
- 协助开发人员尽快解决产品中的缺陷；
- 协助项目管理人员制定合理的开发计划和项目测试计划；
- 对缺陷进行跟踪、分析和总结；
- 评估软件产品的当前质量状态；
- 培养测试工程师的测试技能；

测试团队常见问题：

- 目标不明确；
- 角色和职责不明确；
  项目结构不健全；
- 工作缺乏投入；
- 沟通不够：没有沟通就没有效率

唐僧团队



3.1.2测试小组分类

- 按测试小组的独立性来划分可分为：
  - 非独立测试小组；
  - 相对独立测试小组；
  - 独立测试小组；
- 各种测试组织的优点和缺点

3.1.3 测试团队成员的合适人选

1. 计算机专业技能

+ 测试专业技能

测 试 理 论

测 试 技 术

测 试 工 具

测 试 管 理



1. 计算机专业技能(续)

+ 软件编程技术

- 程序编写和调式
- 测试脚本编写和调试

计算机基础知识

- 网络
- 操作系统： Windows,Linux,Unix,DOS 等；
- 数 据 库 ： SQLserver、Orical、MySQL
- 应用服务器

2. 行业领域知识

软件企业涉及的领域

- 金 融 行 业
- 证 券 行 业
- 电子商务
- 医 药 行 业
- 教育行业
- 电信行业
- .. ...

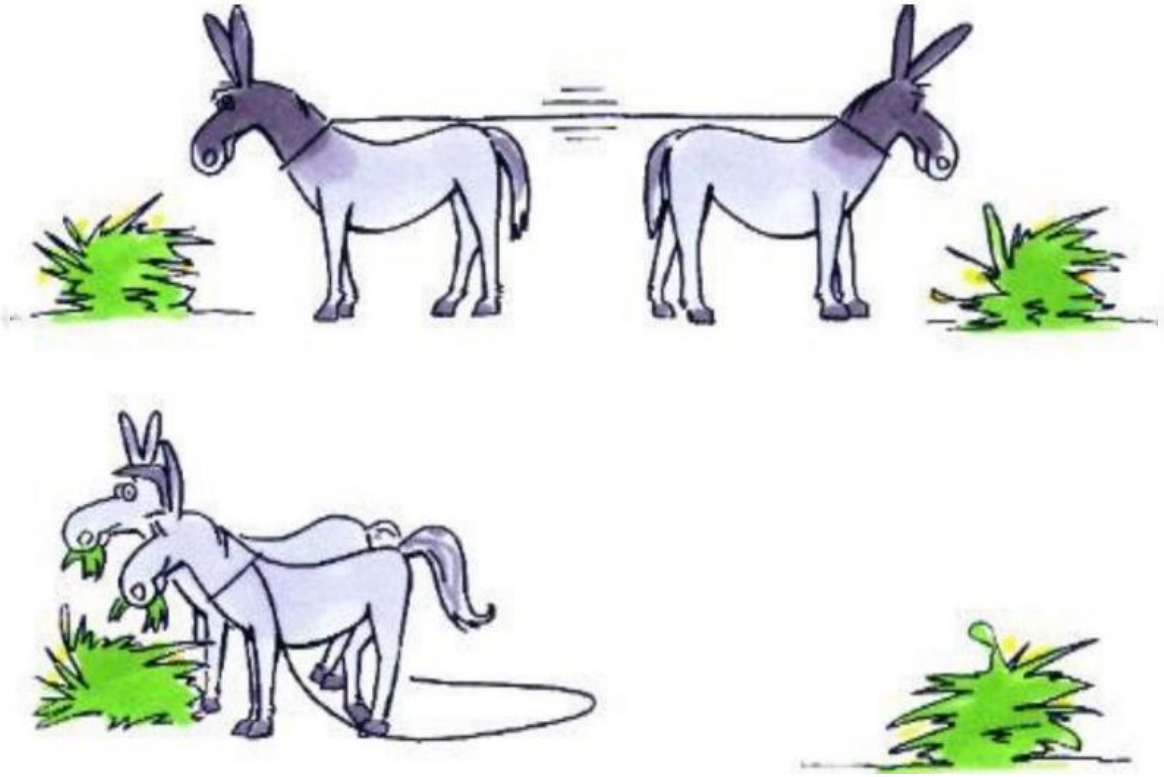
3. 个人素养

- 自信心
- 责任心
- 专注力
- 细心
- 耐心
- 洞察力
- 怀疑精神、探索精神
- 沟通能力



4. 团队协作

- 团队协作是一种为达到既定目标所显现出来的自愿合作和协同劳力的精神。



软件测试职业发展

-
  - 管理路线
  - 技术路线
  - 管理技术路线
-

双V模 型


咨询域
专家域
测试总监
产品经理 业务专家
质量保证经理
配置管理经理
测试经理
安全测试工程师
资深性能测试工程师
资深白盒测试工程师
资深自动化测试工程师
高级
业务测试工程师
质量保证工程师
配置管理工程师
测试主管
性能测试工程师
白盒测试工程师
自动化测试工程师
中级
测试工程师
初级
管理
技术

软件测试人才紧缺：

- 中国软件产业正在快速增长，需要大量软件相关人才；
- 软件企业的发展要求软件测试人才达到一个合适的比例。
- 软件企业开始重视软件测试，但由于历史的原因，找不到合适的软件测试人员。

测试人员职位

例：

◆ 工作职责

1、负责软件系统测试验证工作，包括测试需求分析、测试方案设计、测试用例设计、脚本以及测试代码开发、测试执行等。

2、编写软件系统帮助文档，维护系统百科网站；

3、对其他部门人员进行软件系统使用培训与功能答疑

◆ 任职要求

1、良好的软件基础知识，了解软件工程。

2、掌握软件测试理论和方法。

3、具有较强的学习能力，能在短时间内理解软件的设计思想并掌握操作流程。

4、工作细致，严谨，具有良好的分析、沟通和表达能力，以及良好的文档写作能力。

5、1年以上软件测试经验。

3.2软件测试计划

测试与捞鱼

- 有人把开发比作打靶，目标明确，即按照需求说明书去实现系统的功能。
- 把测试比作捞鱼，目标不明确，自己判断哪些地方鱼多，就去哪些地方捞；如果只捞大鱼(严重缺陷)，网眼就可以大些、撒网区域相对比较集中(测试点集中在主要功能)。如果想把大大小小的鱼捞上来，网眼就要小、普遍撒网，不放过任何一块区域(测试点遍及所有功能)。

3.2软件测试计划

- 在测试计划活动中，要确认测试目标、范围和需求，然后制定测试策略、并对测试任务、时间、资源、成本和风险等进行估算或者评估。

- 在测试计划中，需要解决的问题：
  - 测试的目标和范围；
  - 测试工期估算、 进度安排和资源配置；
  - 测试风险评估；
  - 确定不同测试阶段的过渡条件；
  - 测试版本的管理；

测试计划模版

- 测 试 计 划 (GB8567——88).doc
- SOVO网站（第二版）系统测试计划书 .doc

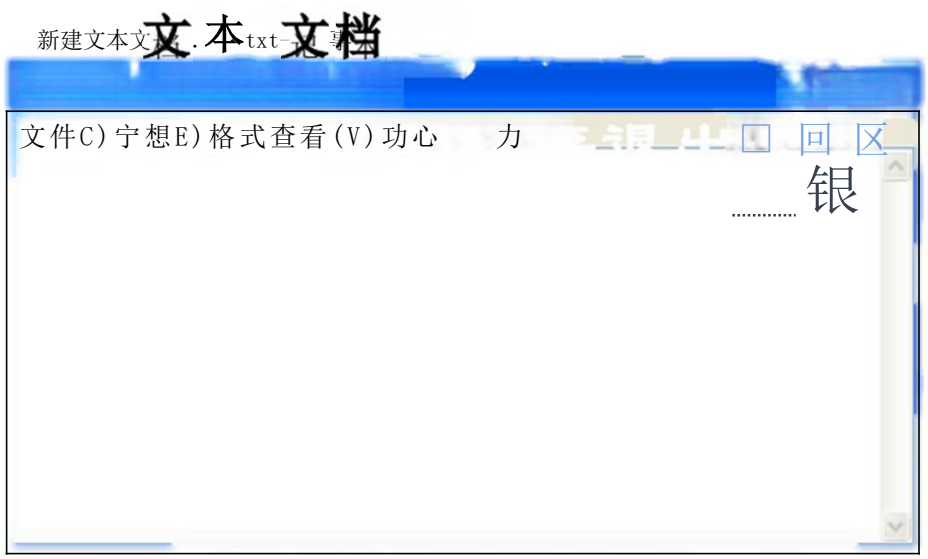
3.3缺陷管理

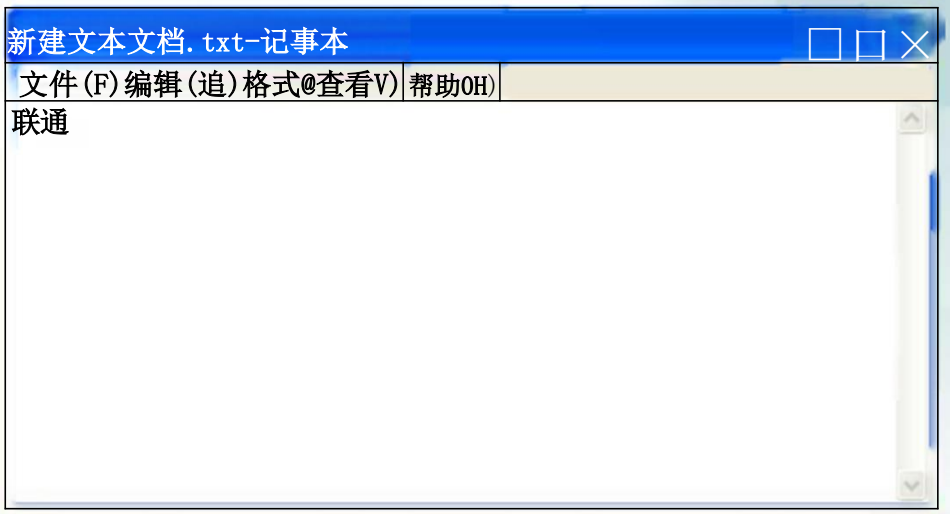
- 什么是缺陷
- 缺 陷 分 类
- 缺陷管理流程
- 缺陷数据分析
- 测试有效性度量
- 缺陷管理工具

1.什么是缺陷?

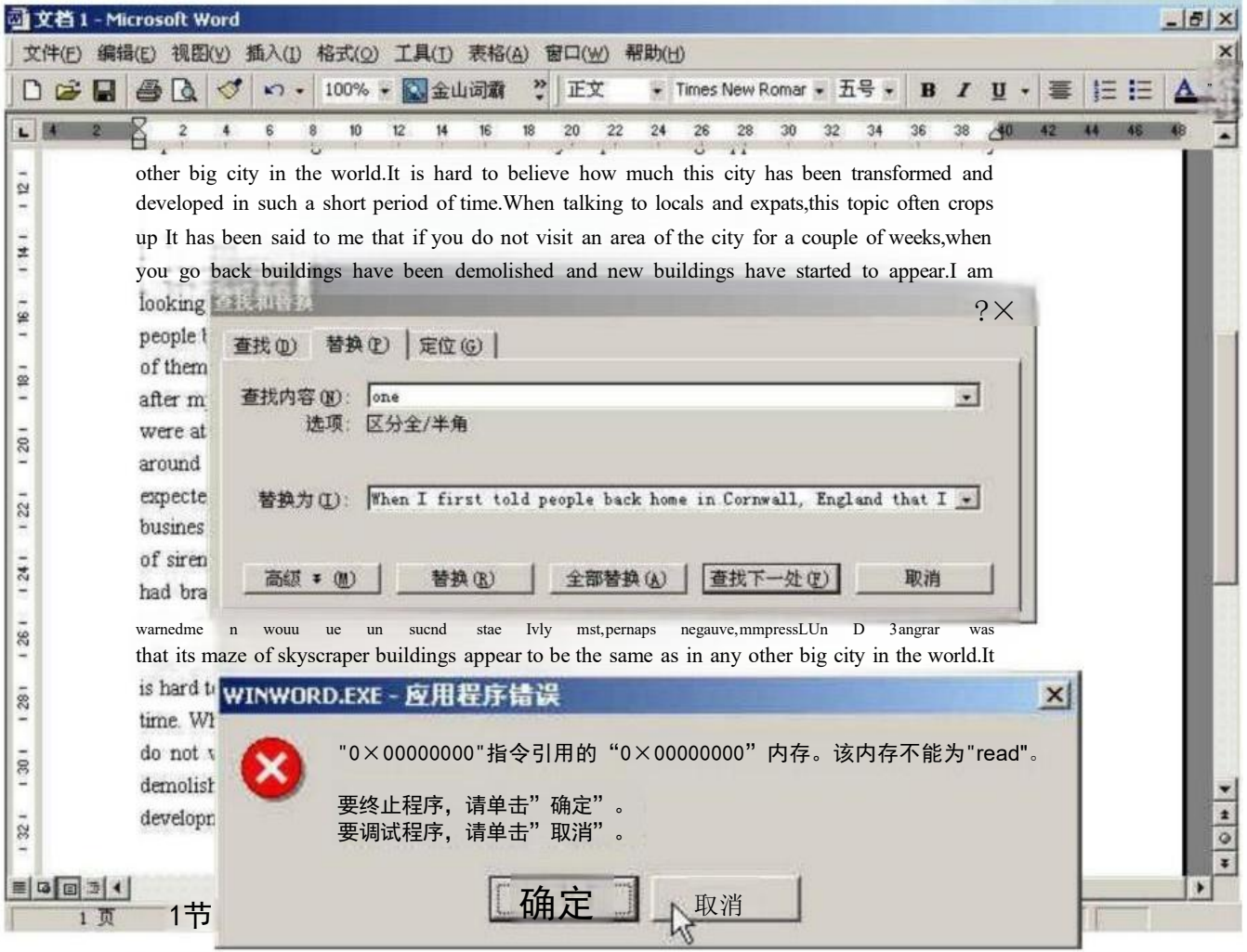
- 功能没有实现或与规格说明不一致;
- 不能工作(死机、没反应);
- 不兼容;
- 边界条件未做处理;
- 界面、消息、提示、帮助不够准确;
- 屏幕显示、打印结果不正确;
- 响应速度慢;
- …… ……

举例


新建文本文档.txt-记事本
文件C)宁想E)格式查看(V)功心 力
银


新建文本文档.txt-记事本
文件(F)编辑(追)格式@查看V)
帮助OH)
联通

举　　例　　2


文档 1 - Microsoft Word
文件(F) 编辑(E) 视图(V) 插入(I) 格式(O) 工具(T) 表格(A) 窗口(W) 帮助(H)
100% 金山词霸 正文 Times New Romar 五号
other big city in the world.It is hard to believe how much this city has been transformed and developed in such a short period of time.When talking to locals and expats,this topic often crops up It has been said to me that if you do not visit an area of the city for a couple of weeks,when you go back buildings have been demolished and new buildings have started to appear.I am
looking
people t
of them
after m
were at
around
expecte
busines
of siren
had bra
查找和替换
查找(D) 替换(P) 定位(G)
查找内容(N): one
选项: 区分全/半角
替换为(I): When I first told people back home in Cornwall, England that I
高级 * (M) 替换(R) 全部替换(A) 查找下一处(F) 取消
warnedme n wouu ue un sucnd stae Ivly mst,pernaps negauve,mmpressLUn D 3angrar was
that its maze of skyscraper buildings appear to be the same as in any other big city in the world.It
is hard t
time. Wh
do not v
demolish
developn
WINWORD.EXE - 应用程序错误
"0×00000000"指令引用的“0×00000000”内存。该内存不能为"read"。
要终止程序，请单击”确定”。
要调试程序，请单击”取消”。
确定
取消
1 页
1节

举 例3


Tools 属性
常规
共享
自定义
本地共享和安全
共享
该共享名包含无效的字符。
确 定
网络共享
共享名。
在网络上共享这个文件夹S)
共享名(H)：
:文件名超长文件名超长文件名超长
允许网络用户更改我的文件W)
了解共享和安全的更多信息。
Windows 防火墙配置为允许此文件夹与网络上其他计算机共享。
查看Windows防火墙设置
确定
取消
应用(A)

缺陷的处理方式

郎中治病的故事

在中国古代，有一家三兄弟全是郎中。其中老三是名医，人们问他：“你们兄弟三人谁的医术最高？”

他回答说：“我常用猛药给病危者医治，偶尔有些病危者被我救活，于是我的医术远近闻名并成了名医。我二哥通常在人们刚刚生病的时候马上就治愈他们，临近村庄的人说他是好郎中。我大哥不外出治病，他深知人们生病的原因，所以能够预防家里人生病.他的医术只有我们家里才知道。”

老大治病的方式最高明

- 提高软件质量最好的办法是：在开发过程中有效地防止工作成果产生缺陷，将高质量内建于开发过程之中。
- 提高软件质量措施：
  - 提高技术水平
  - 规范化管理

老二治病的方式就是医院的模式

- 当工作成果刚刚产生时马上进行质量检查，及时找出并消除工作成果中的缺陷。
- 最常用的方法
  - 技术评审
  - 软件测试
  - 过程检查

老三治病的方式代价最高

- 可是在现实之中，大多数软件企业采用老三的方式来对付质量问题。
- 典型现象是：在软件交付之前，没有及时消除缺陷。

2. 缺陷分类

- 为了更好地分析缺陷，需要对缺陷在严重程度、优先级以及状态上加以区分。
  - 缺陷优先级；
  - 缺陷状态；
  - 缺陷起源；
  - 缺陷来源；
  - 缺陷严重级别

缺陷优先级

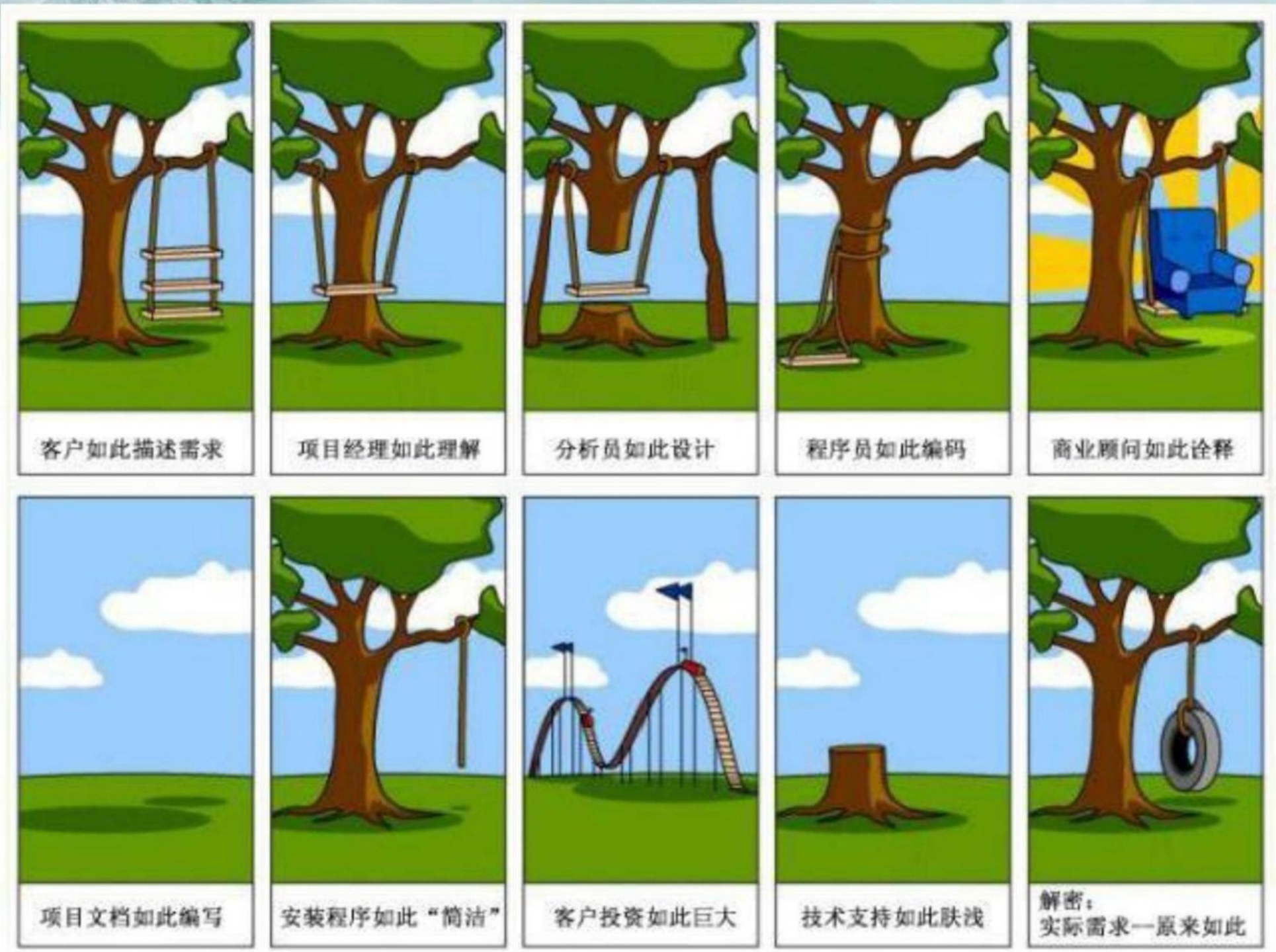
级别	描述
立即解决 (Immediately)	缺陷必须被立即解决。
正常排队 (Normal Queue)	缺陷需要正常排队等待修复或列入软件发布清单。
不紧急 (Not Urgent)	缺陷可以在方便时被纠正。

缺陷状态

缺陷状态	描述
已提交 (Submitted)	已提交的缺陷
确认 (Open)	确认“提交的缺陷”，等待处理
已拒绝 (Rejected)	拒绝“提交的缺陷”，不需要修复或不是缺陷
已解决 (Resolved)	缺陷被修复
已关闭 (Closed)	确认被修复的缺陷，将其关闭

缺陷起源

起源	描述
需求 (Requirement)	在需求阶段发现的缺陷
架构 (Architecture)	在架构阶段发现的缺陷
设计 (Design)	在设计阶段发现的缺陷
代码 (Code)	在编码阶段发现的缺陷
测试 (Test)	在测试阶段发现的缺陷


客户如此描述需求
项目经理如此理解
分析员如此设计
程序员如此编码
商业顾问如此诠释
项目文档如此编写
安装程序如此“简洁”
客户投资如此巨大
技术支持如此肤浅
解密：
实际需求—原来如此

缺陷严重级别

缺陷严重等级	描述
严重缺陷 (Critical)	不能执行正常工作功能或重要功能。或者危及人身安全。
较严重缺陷 (Major)	严重地影响系统要求或基本功能的实现，且没有办法更正。
一般缺陷 (Average Serverity)	严重地影响系统要求或基本功能的实现，但存在合理的更正办法。
次要缺陷 (Minor)	使操作者不方便或遇到麻烦，但它不影响执行工作功能或重要功能。
改进型缺陷 (Enhancement)	其它错误

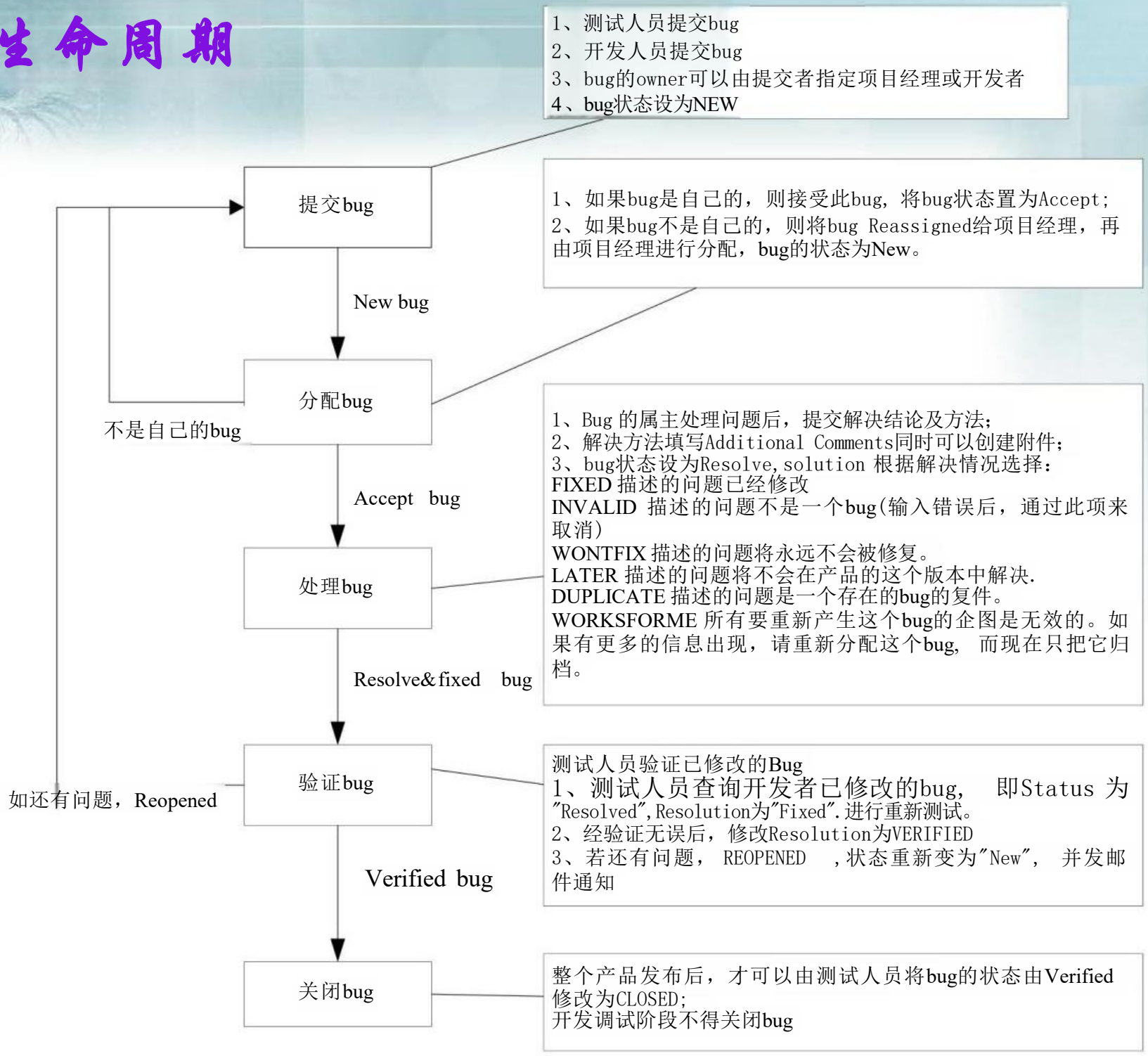
3. 缺陷管理流程

- 在缺陷管理系统中，通常将缺陷的状态分为：
  - 新缺陷(New): 测试中新发现的缺陷；
  - 打开(0pen): 被确认并分配给开发人员处理拉；
  - 修正(Fixed): 开发人员已经完成，等待测试人员验证；
  - 拒绝(Declined): 拒绝修改缺陷；
  - 延期(Deferred): 不在当前版本修复，在下一版本修复；
  - 关闭(Closed): 缺陷已被修复；
  - 重新打开(Reopen): 缺陷重新出现，需开发人员重新处理；

缺陷管理流程（续）

- 缺陷管理的一般流程：
  - 测试人员发现缺陷，提交新缺陷入库，缺陷状态为New；
  - 测试经理审阅。确为缺陷，分配给相应的开发人员，并设置为Open状态；若不是缺陷，则拒绝，设置为Declined 状态。
  - 开发人员对标记为Open状态的缺陷进行确认，若不是缺陷，状态修改为Declined; 如果是缺陷则修复，并置状态为Fixed 。不能解决的缺陷，留下文字说明并保持Open状态。
  - 缺陷修复后由测试人员验证后，确认已修复，可关闭，状态改为Closed。 如果仍有问题，状态改为Reopen。

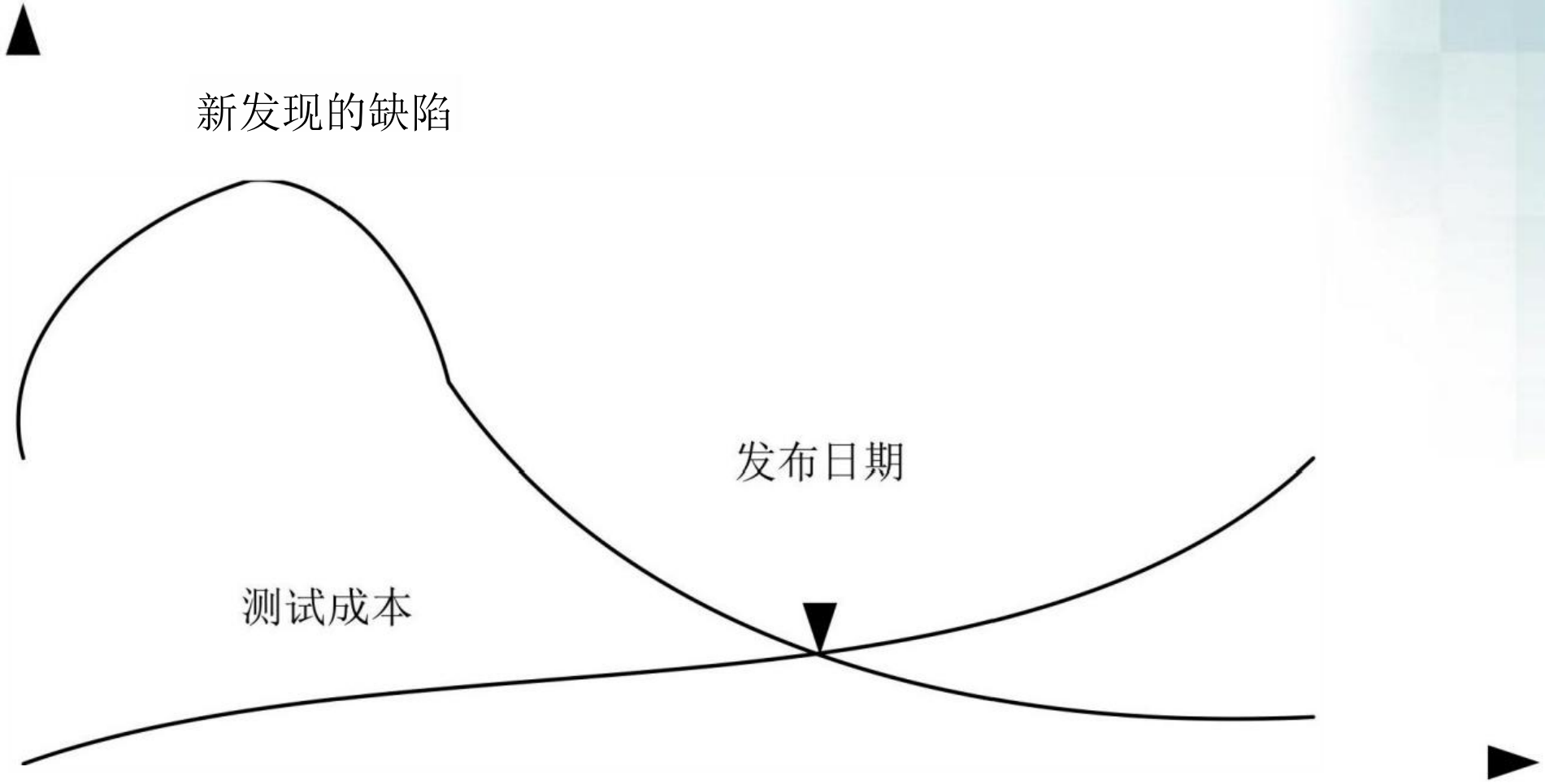
Bug 的生命周期


1、测试人员提交bug
2、开发人员提交bug
3、bug的owner可以由提交者指定项目经理或开发者
4、bug状态设为NEW
提交bug
1、如果bug是自己的，则接受此bug，将bug状态置为Accept；
2、如果bug不是自己的，则将bug Reassigned给项目经理，再由项目经理进行分配，bug的状态为New。
New bug
分配bug
不是自己的bug
Accept bug
1、Bug 的属主处理问题后，提交解决结论及方法；
2、解决方法填写Additional Comments同时可以创建附件；
3、bug状态设为Resolve,solution 根据解决情况选择：
FIXED 描述的问题已经修改
INVALID 描述的问题不是一个bug(输入错误后，通过此项来取消)
WONTFIX 描述的问题将永远不会被修复。
LATER 描述的问题将不会在产品的这个版本中解决.
DUPLICATE 描述的问题是一个存在的bug的复件。
WORKSFORME 所有要重新产生这个bug的企图是无效的。如果有更多的信息出现，请重新分配这个bug, 而现在只把它归档。
处理bug
Resolve&fixed bug
如还有问题，Reopened
验证bug
测试人员验证已修改的Bug
1、测试人员查询开发者已修改的bug, 即Status 为"Resolved",Resolution为"Fixed".进行重新测试。
2、经验证无误后，修改Resolution为VERIFIED
3、若还有问题， REOPENED ，状态重新变为"New", 并发邮件通知
Verified bug
关闭bug
整个产品发布后，才可以由测试人员将bug的状态由Verified修改为CLOSED;
开发调试阶段不得关闭bug

4. 缺陷数据分析

- 缺陷趋势分析
  - 缺陷的趋势分析是缺陷在时间轴上对缺陷进行分析，有助于进度控制和测试过程的管理。
  - 在一个成熟的软件开发过程中，缺陷趋势一般会遵循一种和预测比较接近的模式向前发展。

缺陷数据分析（续）



缺陷数据分析（续）

- 缺陷分布分析
  - 缺陷分布分析是缺陷的横向分布，即空间上的分布。
  - 可以针对一个或多个缺陷参数（如项目、功能点、来源、严重级别等）进行缺陷分析 。

5. 测试有效性度量

- 缺陷去除率 (DRE)

$$DRE = \frac{\text{测试期间发现的缺陷数量}}{\text{测试期间发现的缺陷数量+未发现的缺陷数量}}$$

测试有效性度量(续)

- 缺陷损耗
- 缺陷损耗可定义为：

$$\text{损耗} = \frac{\text{缺陷数量} \times \text{发现的阶段潜伏期尺度}}{\text{缺陷总量}}$$

6. 测试管理工具

- 测试管理工具对测试需求、测试计划、测试用例、测试实施进行管理，并且测试管理工具还包括对缺陷的跟踪管理。



问题与讨论

黑盒测试(1)

本章内容

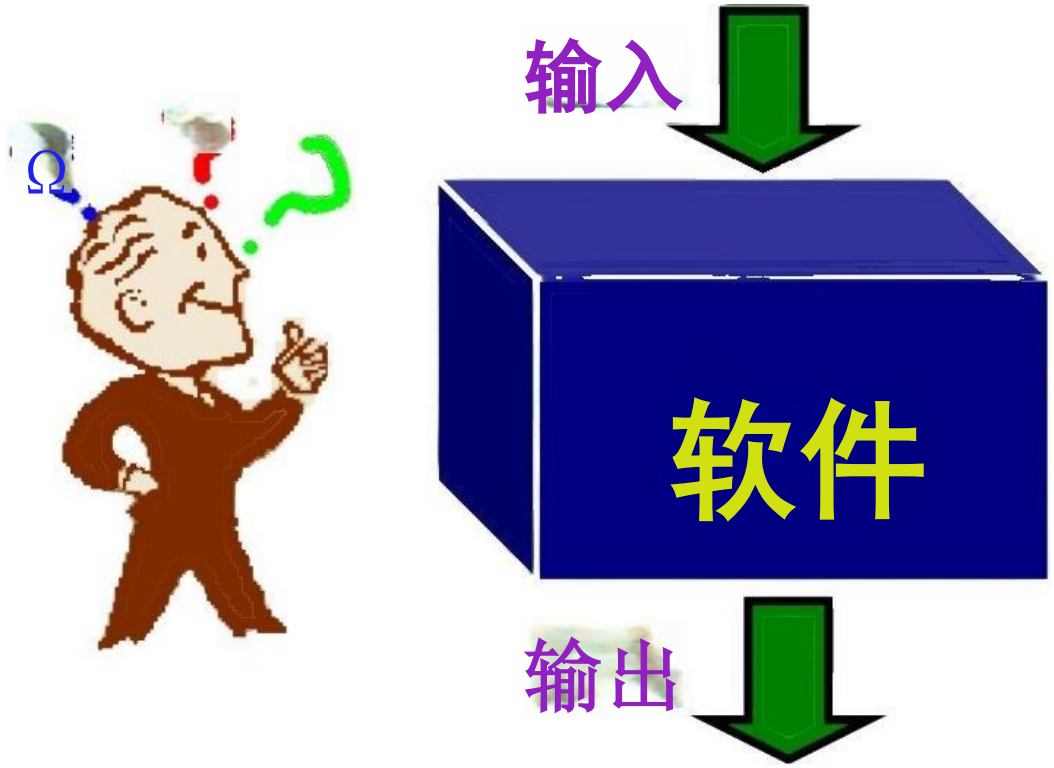
- 软件测试方法
- 边界值测试
- 等价类测试
- 基于决策表的测试
- 因果图

软件测试方法

- 软件测试方法一般分为两种：
  - 白盒测试(White Box Testing)
  - 黑盒测试 (Black-box Testing)

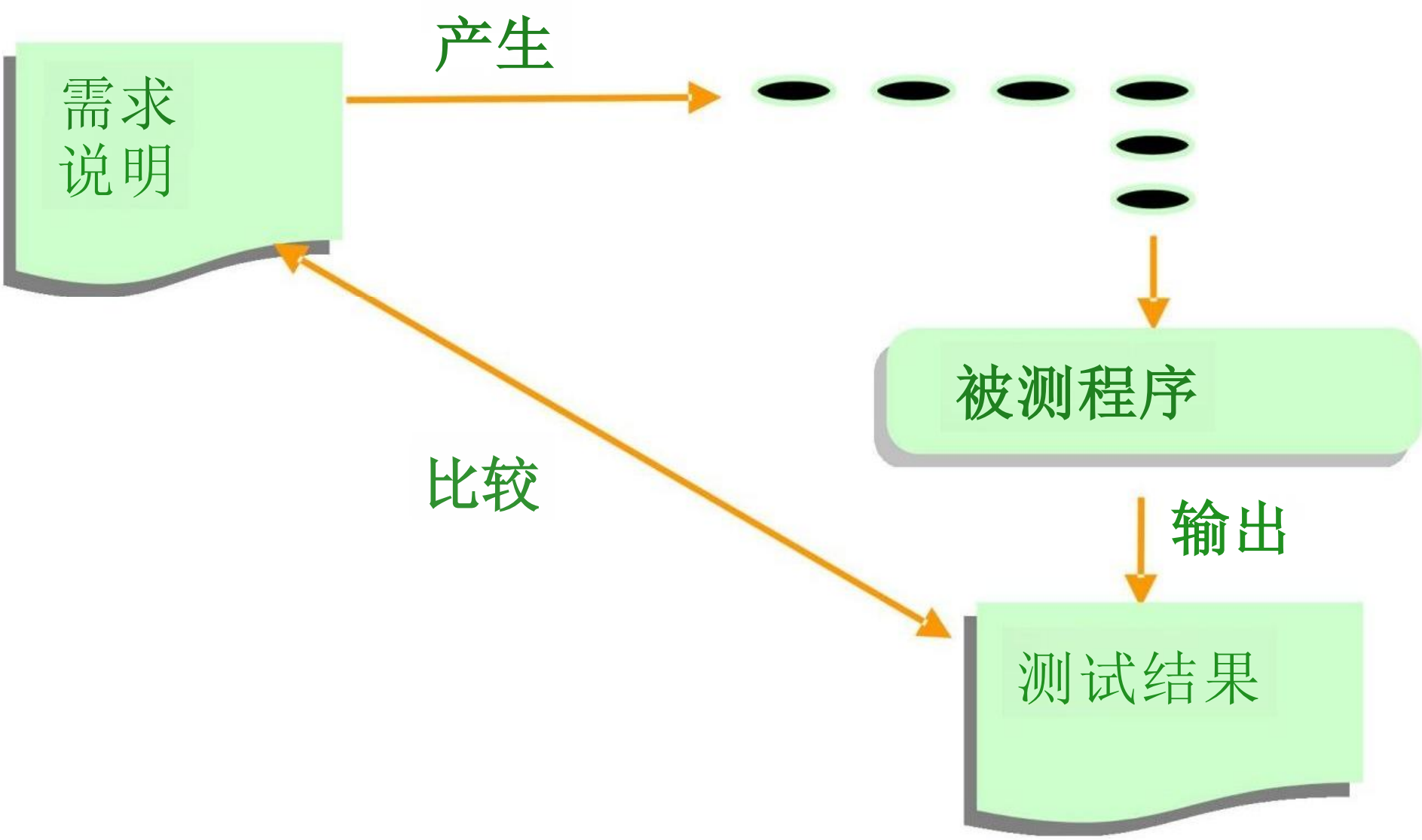
黑盒测试

- 黑 盒 测 试 (Black-box Testing)



- 又 叫功 能 测 试，数 据 驱 动 测 试 或 基 于 规 格 说 明 的 测 试。

黑盒测试过程

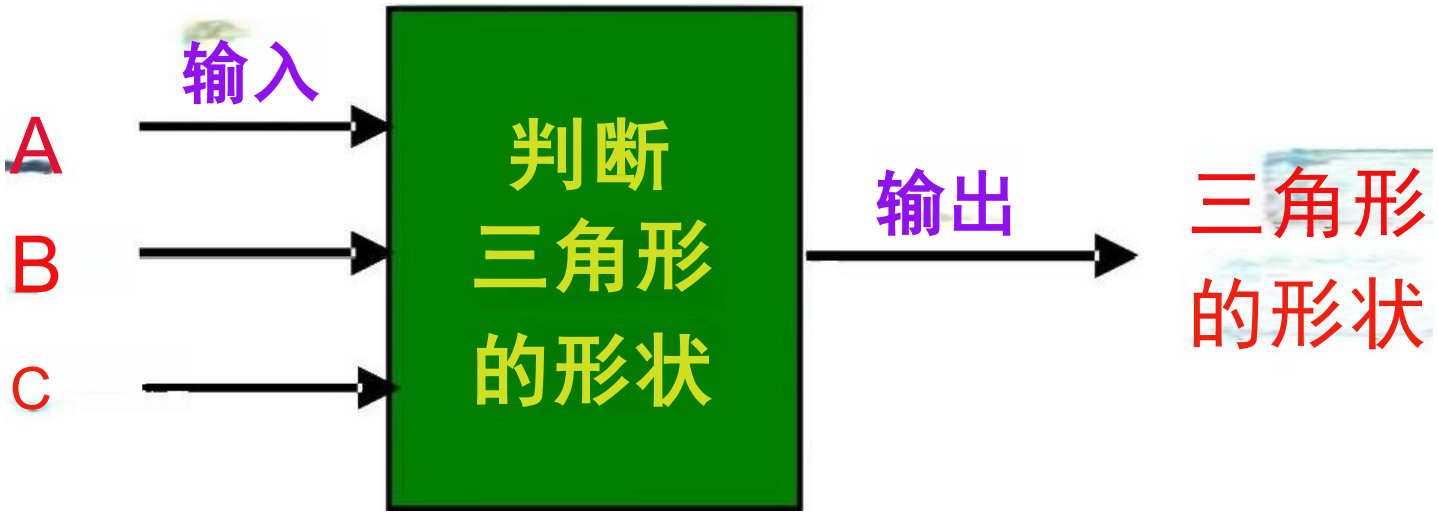


黑盒测试的目标

- 黑盒测试试图发现以下类型的错误：
  - 功能错误或遗漏；
  - 界面错误，不美观；
  - 外部信息访问错误；
  - 性能错误；
  - 初始化和终止错误；
  - 在接口上，能否正确地接受输入数据，能否产生正确地输出信息；

彻底的黑盒测试

- 例：输入三角形的三条边长，判断三角形的形状。



黑盒测试的难点

- 黑盒测试的难点在于如何构造有效的输入。

  由于输入空间通常是无限的，穷举测试显然行不通。

  寻找最小最重要的用例集合以精简测试复杂性。

黑盒测试方法

- 黑盒测试的测试用例设计方法
  - 边界值分析
  - 等价类划分
  - 基于决策表的测试
  - 因果图
  - 错误推测

4.1 边界值测试

- 边界条件
- 边界值分析
- 健壮性测试
- 最坏情况测试
- 特殊值测试
- 随机测试

1. 边界条件

- 数值的边界值
- 字符的边界值
- 其他边界条件

◆ 任何值得测试的范围的临界点，可分为：

边界值：在规格说明书中明确定义；

次边界：隐含在软件中必须经过分析才能获得；

例如：

+ 屏幕上光标在最左上、最右下位置；

+ 报表的第一行和最后一行；

  数组元素的第一个和最后一个；

  - C++ 语言中，int A[10]:A[0]….A[9]

    循环的第0次、第1次和倒数第2次、最后一次；

2. 边界值分析

- 边界值分析的原理
- 边界值分析的思想
- 边界值分析测试用例设计方法

（1）边界值分析的基本原理

- 边界值分析关注**输入空间**的边界，并从中标识测试用例。

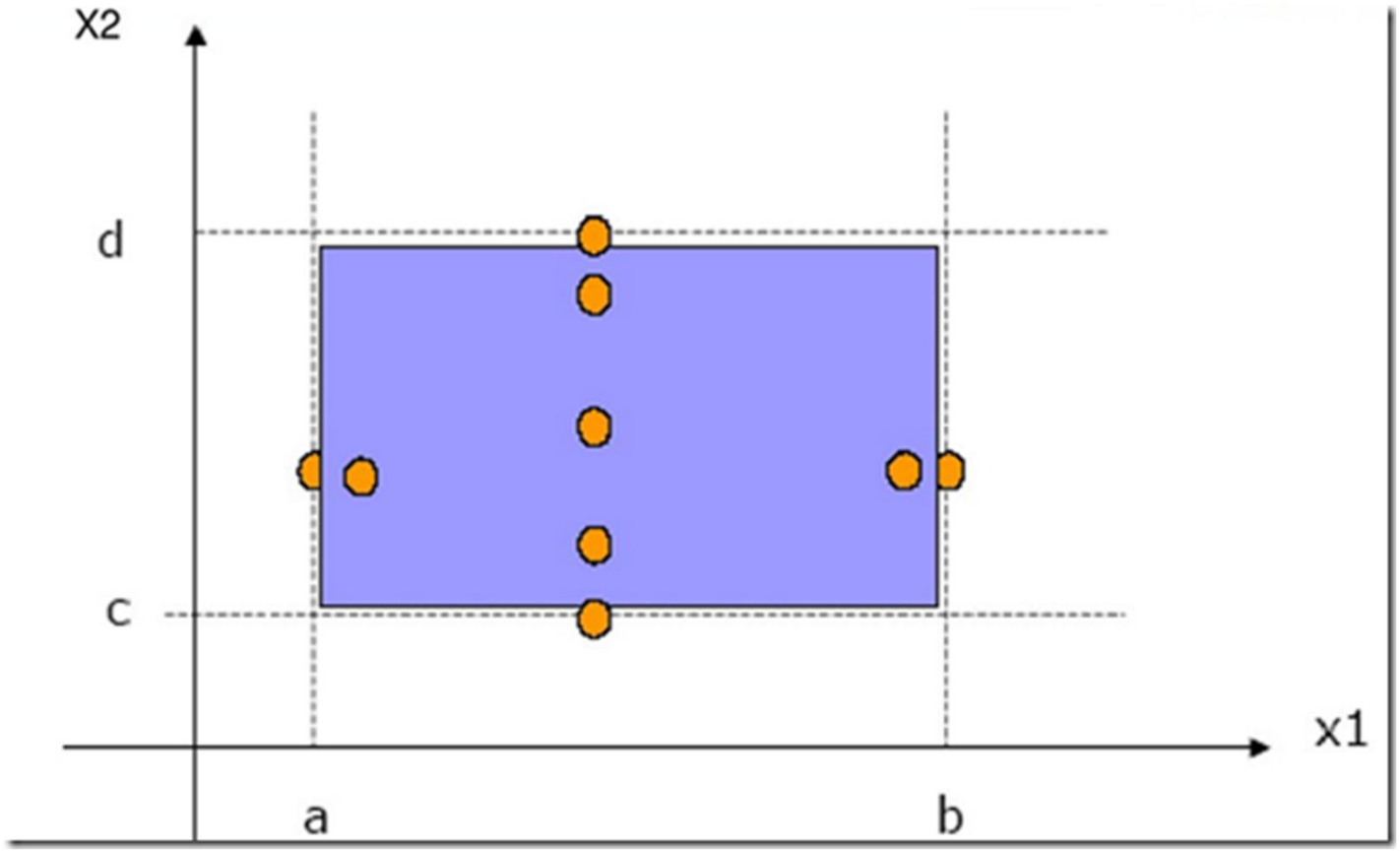
- 基本原理

  错误更可能出现在输入变量的极值附近。

⑵边界值分析的基本思想

- 在最小值、略高于最小值、正常值、略低于最大值和最大值处取输入变量值。
- 表示方法min、min+、nom、max-、max
- 例如：涉及两个变量的函数X1,X2
  - X1 的 取 值X1min,X1min+,X1nom,X1max-,X1max
  - X2 的 取 值X2min,X2min+,X2nom,X2max-,X2max

边界值分析的取值



“单缺陷”假设

- “单缺陷”假设

  失效极少是由两个或多个缺陷的同时发生引起的

- “单缺陷”假设是边界值分析的关键假设

(3)边界值分析测试用例

- 边界值分析获得测试用例的方法：
  - 使所有变量取正常值，只使一个变量取极值；

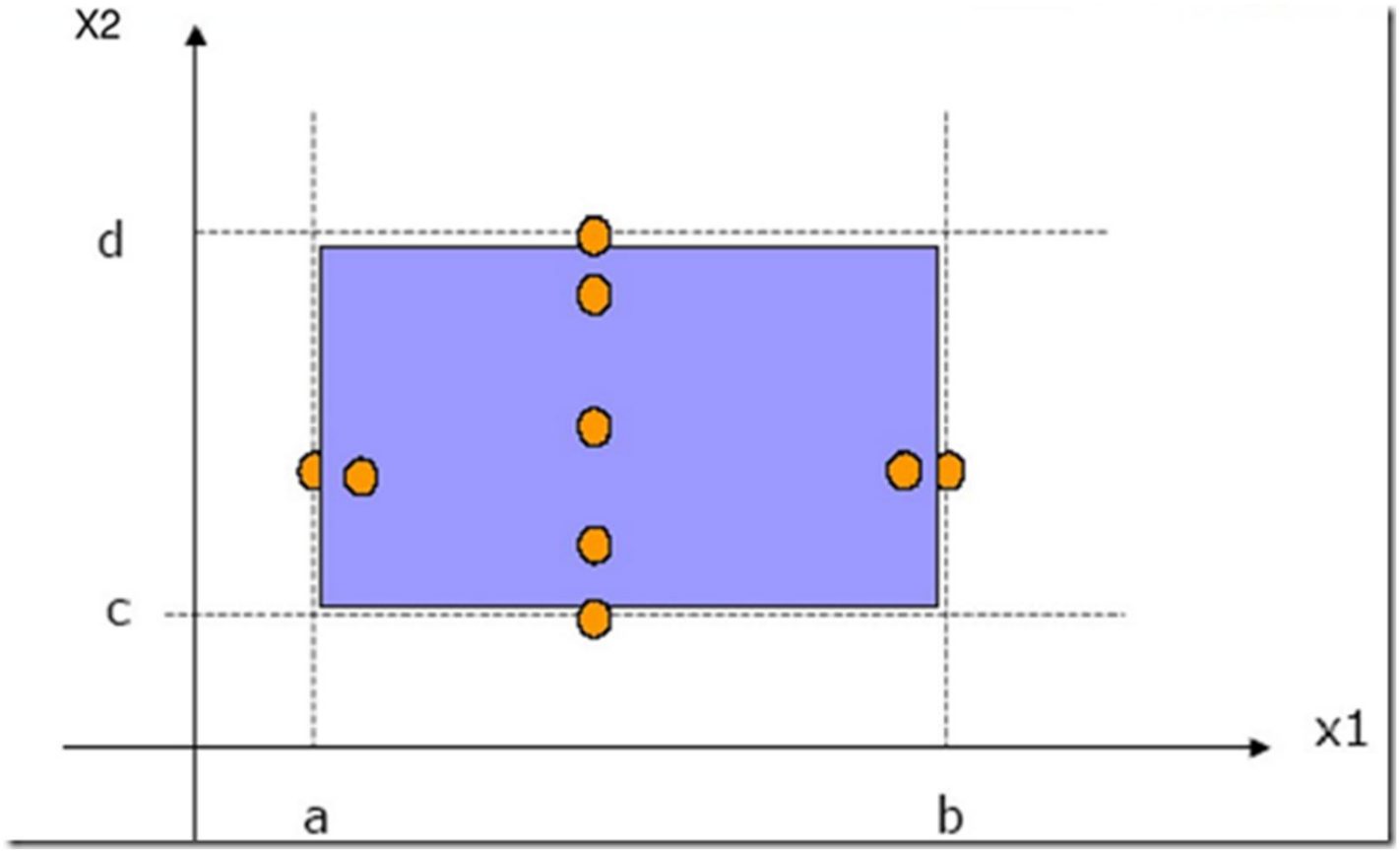
  对于一个n变量函数，边界值分析会产生4n+1个测试用例。

两个变量函数边界值分析测试用例

- X1 取 值 ： X1min, X1min+, X1nom, X1max−, X1max
- X2 取 值 ： X2min, X2min+, X2nom, X2max−, X2max
- 两个变量函数的边界值分析测试用例

{ <X1nom,X2min>,<X1nom,X2min+>,
<X1nom,X2nom>,<X1nom,X2max->,
<X1nom,X2max>,
<X1min,X2nom>,<X1min+,X2nom>,
<X1max-,X2nom>,<X1max,X2nom>}

两个变量函数边界值分析测试用例



举例：三角形问题

- 三角形问题描述

  三条边a,b,C取整数值，且各边的取值范围是：

  [1,200]

  边界值分析设计测试用例

  每条边的取值：

  ■1,2,100,199,200

  测试用例数目：4n+1

  □三角形问题(n=3)：　用例数为13

(4)边界值分析优缺点

- 边界值测试分析采用了可靠性理论的单 缺陷 假设。
- 优点：简便易行；生成测试数据的成本很低；
- 局限性：

  测试用例不充分；不能发现测试变量之间的依赖关系；不考虑含义和性质；
- 结论：只能作为初步测试用例使用

3. 健壮性测试

- 健壮性是指在异常情况下，软件还能正常运行的能力。
- 健壮性有两层含义：
  - + 容错能力
  - 恢复能力

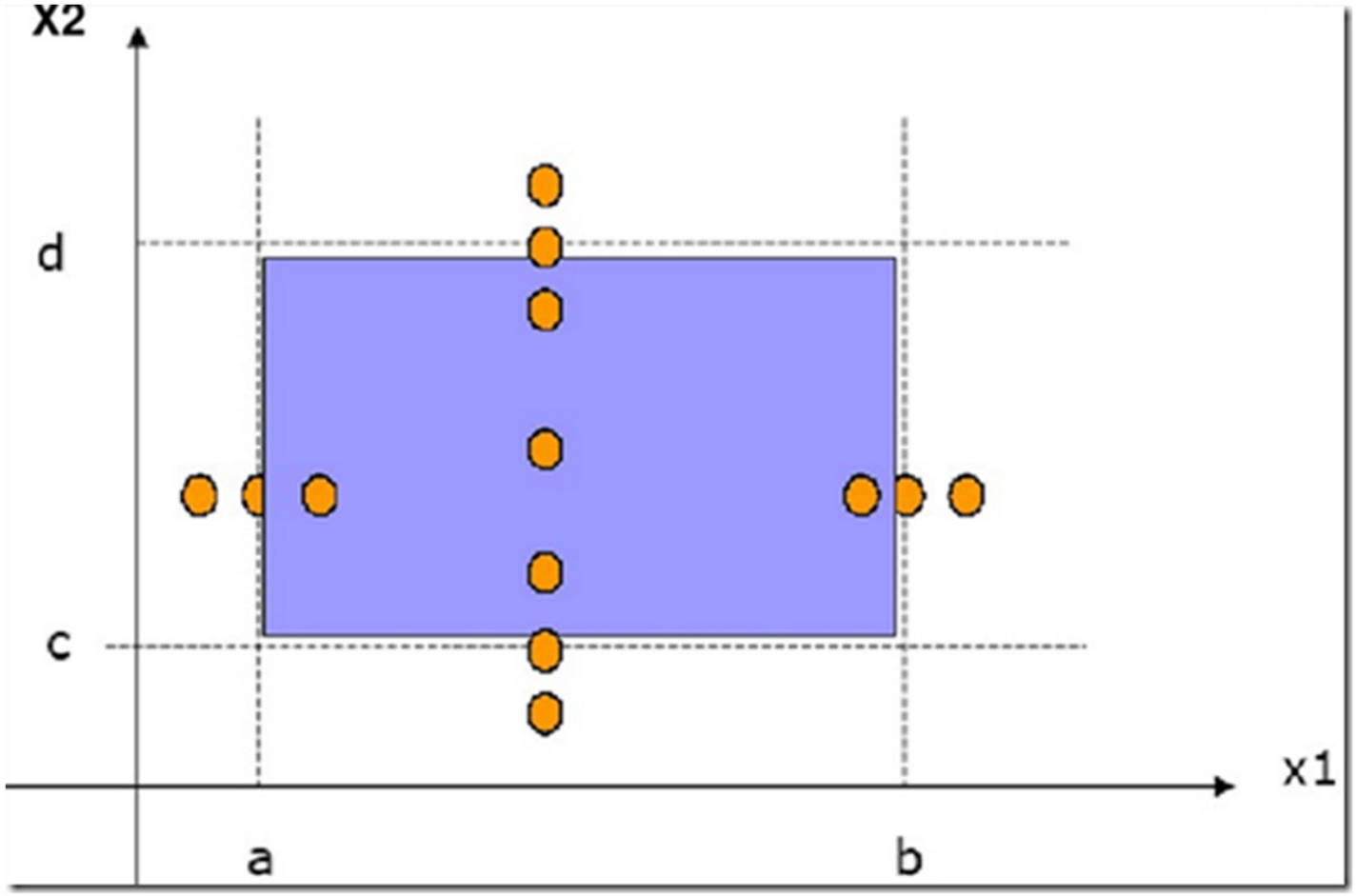
(1)健壮性测试的基本思想

健壮性测试是边界值分析的一种简单扩展，除了使用五个边界值分析取值，还要过采用：

十一个略超过最大值(max+) 的取值

十一个略小于最小值(min-)的取值

两个变量函数的健壮性测试用例



(2) 健壮性测试的讨论

- 健壮性测试最有意思的部分不是输入而是预期的输出
- 健壮性测试的主要价值是观察异常情况的处理
  - 软件质量要素的衡量标准：软件的容错性
  - 软件容错性的度量：
    - 从非法输入中恢复

健壮性测试的测试用例个数



思考：一个变量个数为n的函数的健壮性测试会产生多少个测试用例？

练习：NextDate函数

- 程序输入条件

  Month 1≤月份≤12

  + Day. 1≤日期≤31
  + Year:1812≤年≤2012
  + 输入日期，计算其下一天。

- 请用健壮性边界值分析设计测试用例。

◆请有兴趣的同学在课后编写NextDate函数或其他程序，在后续课程中进行更深入的测试。

5.5.2 NextDate 函数的测试用例

回顾以下第2章中关于NextDate的**问题描述：**NextDate是一个有三个变量（月份、日期和年）的函数。函数返回输入日期后面的那个日期。变量月份、日期和年都是整数，且满足以下条件：

C1：1≤月份≤12

C2：1≤日期≤31

C3：1812≤年≤2012

对无效的输入日期，比如6月31日，或c1、c2、c3中的任意一个条件失败，则NextDate，都会产生一个输出，指示其为无效日期，或给出具体的无效原因，或笼统的给出"无效日期输入。"提示。

注：以下是依据方法产生测试用例，未考虑输入之间的逻辑关系，比如6月31日，出现的常识性错误并不是疏落。

用例标识	月份	日期	年	预期输出
1	6	15	1812	1812年6月16日
2	6	15	1813	1912年6月16日
3	6	15	1912	1912年6月16日
4	6	15	2011	2011年6月16日
5	6	15	2012	2012年6月16日
6	6	1	1912	1912年6月2日
7	6	2	1912	1912年6月3日
8	6	15	1912	1912年6月16日
9	6	30	1912	1912年7月1日
10	6	31	1912	错误
11	1	15	1912	1912年1月16日
12	2	15	1912	1912年2月16日
13	6	15	1912	1912年6月16日
14	11	15	1912	1912年11月16日
15	12	15	1912	1912年12月16日

4. 最坏情况测试

最坏情况测试的基本思想

- 边界值测试分析采用了可靠性理论的单缺陷假设

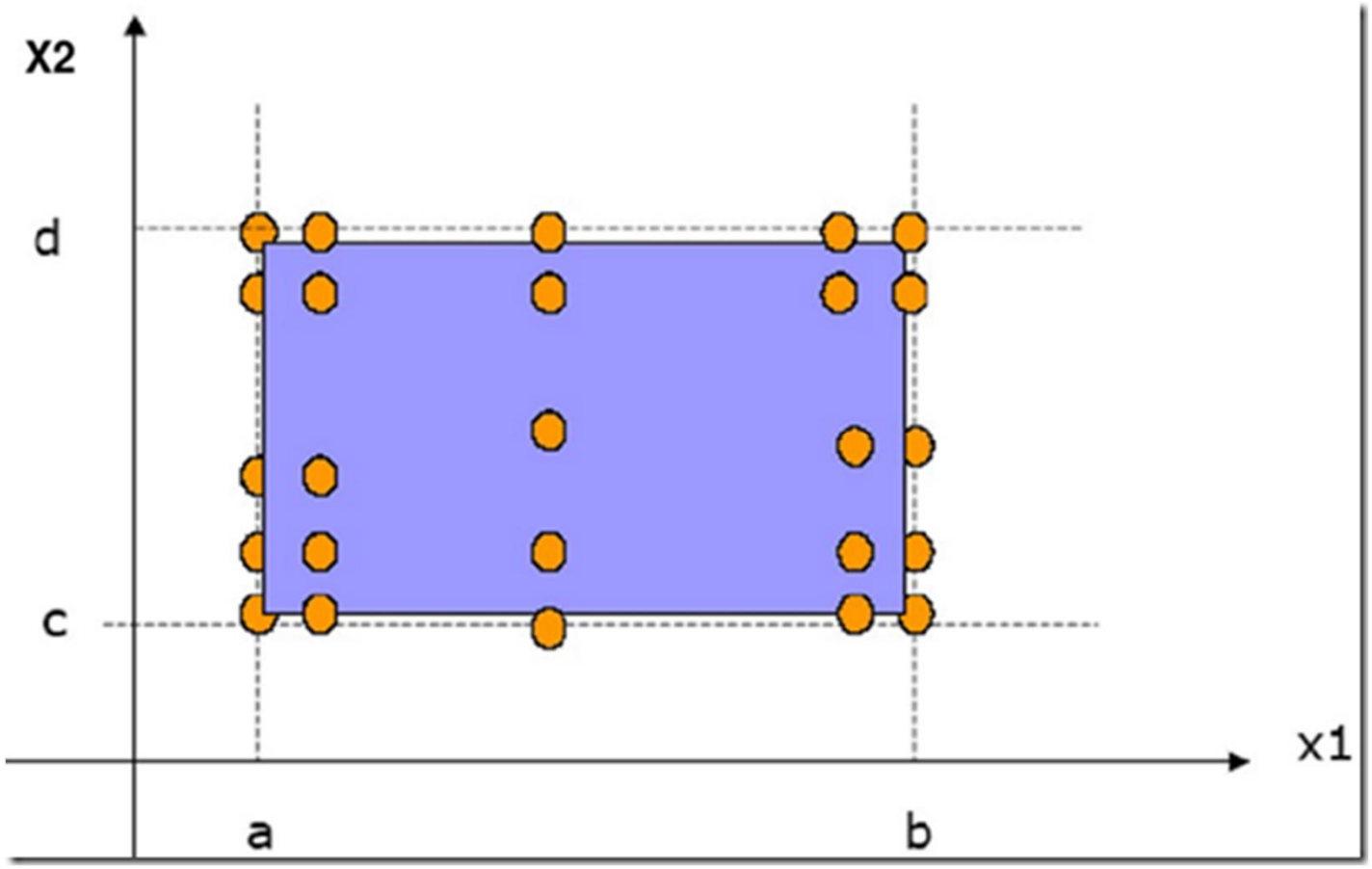
  最坏情况测试拒绝这种假设，关心当多个变量取极值时会出现什么情况？

(1)最坏情况测试用例设计方法

- 对每一个变量首先进行包含最小值、略高于最小值、正常值、略低于最大值、最大值五个元素集合的测试，然后对这些集合进行笛卡尔积计算，以生成测试用例。

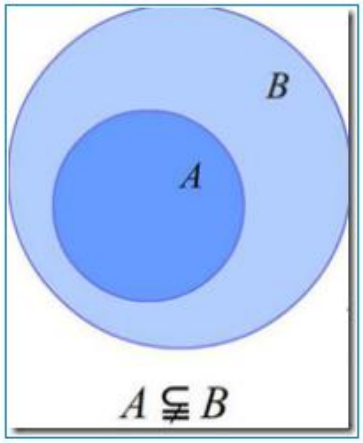
- 一个变量个数为n的函数的最坏情况测试会产生5个测试用例。

两变量函数的最坏情况测试用例



（2）最坏情况与边界值分析的比较

- 基本边界值分析测试用例是最坏情况测试用例 的真子集。



- 最坏情况测试显然更彻底；
- 最坏情况测试工作量大得多；

+n 变量函数的最坏情况测试会产生5 的n次 方个 测试用例 ，

边界值分析只产生4 n+1 个测试用例

最坏情况测试举例(1)

- 三角形问题

  三条边a, b, C取整数值，且各边的取值范围是：[1, 200]

- 最坏情况测试用例
  - 每条边的取值：
    - 1, 2, 100, 199, 200
  - 测试用例数目：5
    - 三角形问题(n=3)：　用例数为125
    - 测试用例

最坏情况测试举例(2)

- NextDate 的测试用例

  输入条件

  - 1≤月份≤12
  - 1≤日期≤31
  - 1812≤年≤2012

  最坏情况测试用例设计

  - 月份取值： 1,2,6,11,12
  - 日期取值：1,2,15,30,31
  - 年取值：1812,1813,1912,2011,2012

最坏情况测试举例(2)

- NextDate 的测试用例
  - 测试用例数目
    - n=3, 5^3 =125

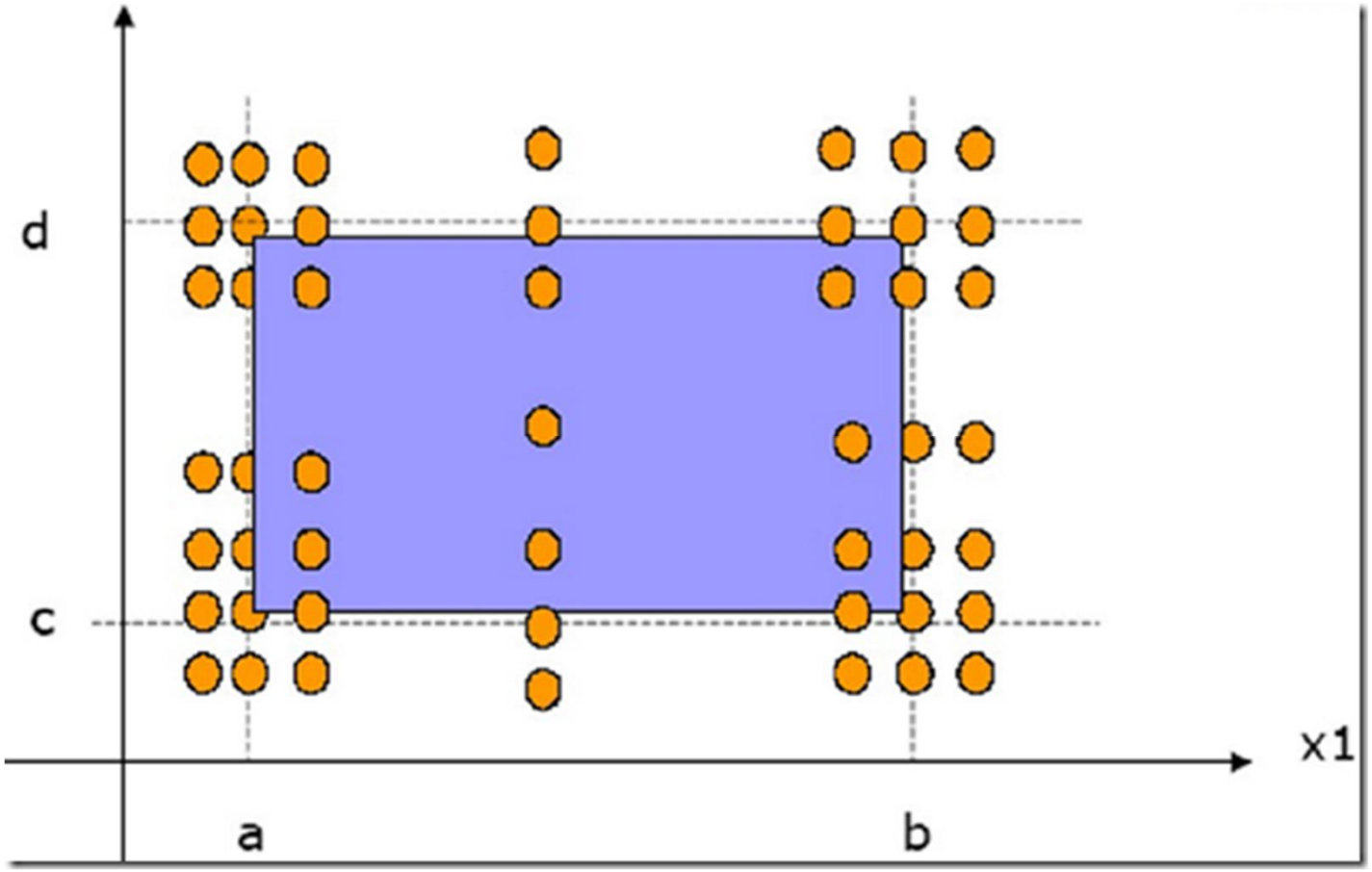
  错误的测试用例
    - 1812年2月30日

  遗漏重要的测试用例
    - 日期取28或29
    - 闰年：2000年

5. 健壮最坏情况测试

- 对每一个变量，首先进行包含最小值、略高于最小值、正常值、略低于最大值、最大值五个元素集合的测试，还要采用一个略超过最大值的取值，以及一个略小于最小值的取值。然后对这些集合进行笛卡尔积计算以生成测试用例。

两变量函数的健壮最坏情况测试用例



- 边界值测试比较适合运用在有下列情况：
  - 数　值
    - 字　母
    - 位　置
  - 数　量
  - 速　度
  - 大　小
  - ……

6. 特殊值测试

- 边界值分析假定n个变量是相互独立的，没有考虑这些变量之间的相互依赖关系；

- 特殊值测试使用领域知识、使用类似程序的经验设计测试用例的特殊值。

特殊值测试

- 特点：最直观、最不一致、具有高度主观性。
- 特殊值测试特别依赖测试人员的能力。
- 虽然特殊值测试是高度主观性的，但是能更有效地发现缺陷。

7. 随机测试

随机测试的基本思想

不是永远选取有界变量的最小值、略高于最小值、正常值、略低于最大值和最大值，而是 *使用随机数生成器选出测试用例值。*

随机测试的一种实现方法

- 有界变量a≤x≤b值的一个Visual Basic应用程序生成的x,满足下式：
  - x=Int(b-a+1)*Rnd+a
  - 函数Int返回浮点数的整数部分
  - 函数Rnd生成区间[0,1]内的随机数

思 考 ：

需要考虑哪些边界问题?

商家资料

店铺名称： *

Logo: 上 传

分 类： 网上订餐 电话订餐

外送费用： 元

送餐时间： 小时

起送价格： 元

上午营业时间： 点

下午营业时间： 点

送餐电话： *

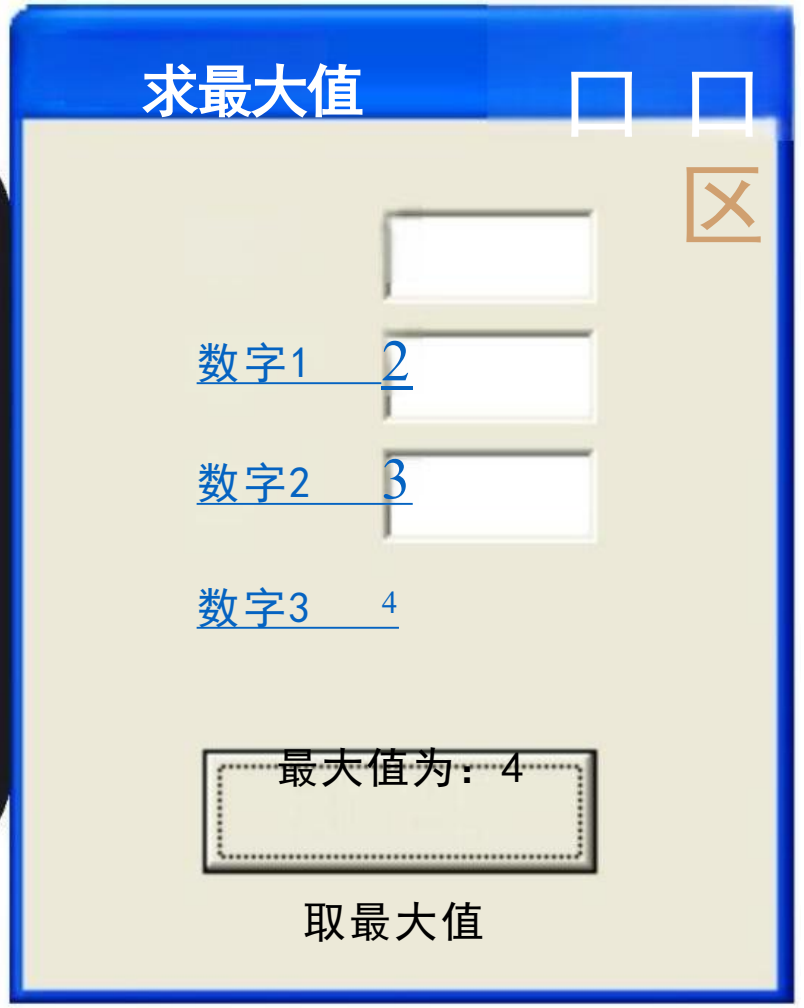
经营地址： *

主营产品： 快餐 □小炒 西餐 **套餐** **早餐**

更新

课堂练习

有一个小程序，能够求出三个在0到9999间整数中的最大者，请分别用边界值分析和健壮性测试方法设计测试用例。



有一个小程序，能够求出三个在0到9999间整数中的最大者，请分别用边界值分析和健壮性测试方法设计测试用例。
要求：1.列出边界分析方法。2.设计测试用例。
提交：提交文件为excel。

a,b,c极值分析						
max+	max	max-	nom	min+	min	min-
10000	9999	9998	9997~2中间任意正整数，一般取中间值	1	0	-1
边界值测试方法：如果仅有单变量，那么就取max,max-,min,min+,nom。如果有多变量则在单变量的方法上基于"单缺陷"假设进行数据组合。						
健壮性测试方法：考虑正常值之外的值，将max+和min-考虑在内。多值情况下基于"单缺陷"假设进行数据组合。						

序号	第一个数a	第二个数b	第三个数c	预期结果	测试结果	备注
1	0	3000	6000	最大值为6000		a取极值
2	1	3000	6000	最大值为6000		
3	9999	3000	6000	最大值为9999		
4	9998	3000	6000	最大值为9998		
5	3000	0	6000	最大值为6000		b取极值
6	3000	1	6000	最大值为6000		
7	3000	9999	6000	最大值为9999		
8	3000	9998	6000	最大值为9998		
9	3000	6000	0	最大值为6000		c取极值
10	3000	6000	1	最大值为6000		
11	3000	6000	9999	最大值为9999		
12	3000	6000	9998	最大值为9998		
13	3000	6000	5000	最大值为6000		a,b,c皆为正常值
14	-1	3000	6000	输入数据非法(无效)		考虑a值健壮性
15	10000	3000	6000			
16	3000	-1	6000			考虑b值健壮性
17	3000	10000	6000			
18	3000	6000	-1			考虑c值健壮性
19	3000	6000	10000			

觉得还不错？一键收藏



问题与讨论

黑盒测试 (2)

本节内容

- 等价类
- 等价类测试类型
- 用等价类设计测试用例
- 案例

问题引入

- **测试的矛盾**
  - 为保证软件产品的质量，需要进行完备(彻底)的测试；但这是不现实的，因为：
    - 输入量太大
    - 输出结果太多；
    - 软件实现途径太多

如果决定不去测试所有的情况，就选择了风险。

问题引入

从经济的角度来说，我们希望测试没有冗余；

- 一个好的测试用例在于能发现至今未发现的错 误；
- 一个成功的测试是发现了至今未发现的错误的测试 。

软件质量和投入的成本如何平衡?



4.2等价类测试

- 等价类方法可令测试事半功倍。

  - 举 例：

    设计测试用例，实现一个对所有实数进行开方运算（f=sqrt(x)） 的程序的测试。

4.2.1等价类

- 等价类的划分
- 划分等价类的方法
- 等价类测试类型
- 用等价类设计测试用例

1. 等价类划分

- **划分**

  划分是指**互不相交的**一组子集，这些子集的并是整个集合。

  划分的定义

  - 给定集合B，以及B的一组子集A1、A2……、An，这些子集是B的一个划分，当且仅当：

  A1 U A2 U...U An=B，且

  i≠j有Ai∩Aj=Φ

划分在软件测试中的作用

边界值分析方法的固有弱点是漏洞和冗余性。

划分对于测试的作用：

+ 完备性

+ 无冗余性

等价类测试的主要任务之一就是：

找出合适的划分 。

- **等价类划分**

  将输入定义域进行一个划分，并且划分的各个子集是由**等价关系**决定的。

  此处的等价关系是指：　在子集合中，各个输入数据对于揭露程序中的错误都是等效的。

- **等价类的分类**
  - **有效等价类**
    - 对于程序的规格说明来说，是合理的、有意义的输入数据所构成的集合；
  - **无效等价类**
    - 对于程序的规格说明来说，是不合理的、没有意义的输入数据所构成的集合；

2. 等价类测试

- 等价类测试的思想：

  把全部的输入数据**划分成若干个等价类，在每一个等价类中取一个数据**来进行测试。

  在子集合中，各个输入数据对于揭露程序中的错误是**等 效**的 。

- 等价类测试的假设：

  测试某等价类的代表值，就等效于对这个等价类中其他值的测试。

- 等价类的特点

  完备性
  无冗余性
  等价性

注意

- 设计测试用例时，要同时考虑有效等价类和无效等价类。

3. 划分等价类的方法

- (1) 按区间划分

  如果输入条件规定了取值范围或值的个数就可确定一个有效等价类和两个无效等价类。

  例如：

  - 输入学生成绩，范围是0到100；
  - 实验课一个班选课人数20至40人；

- （2）按照数值划分

  如果规定了输入数据的**一组值**，且程序要对每一个输入值分别进行处理，要对每一个规定的输入值确立一个有效等价类，而对于这组值之外的所有值确立一个无效等价类。

  例：输入条件说明学历可为：专科、本科、硕士、博士四种之一。

- （3）按照数值集合划分

  +输入条件规定了输入值的**集合**或是规定了“**必须如何**”的条件，则可确定一个有效等价类和一个无效等价类

● 例如：

“文件名称必须以字母开头，……”

◆(4)如果输入条件是一个**布尔量**，则可以确立一个有效等价类和一个无效等价类。

- 例如：用验证码登录网站。

◆(5) 进一步细分等价类

如已划分的等价类各元素在程序中的处理方式不同，则应将此等价类进一步划分成更小的等价类。

◆例如：**输入三条边，判断三角形的形状。**

（首先判断是否能构成三角形，如果可以构成三角形，再进一步判断三角形的形状。）

4.2.2 等价类测试类型

- 单缺陷与多缺陷假设产生弱等价类与强等价类测试之分；
- 是否进行无效数据的处理产生健壮与一般等价类测试之分；

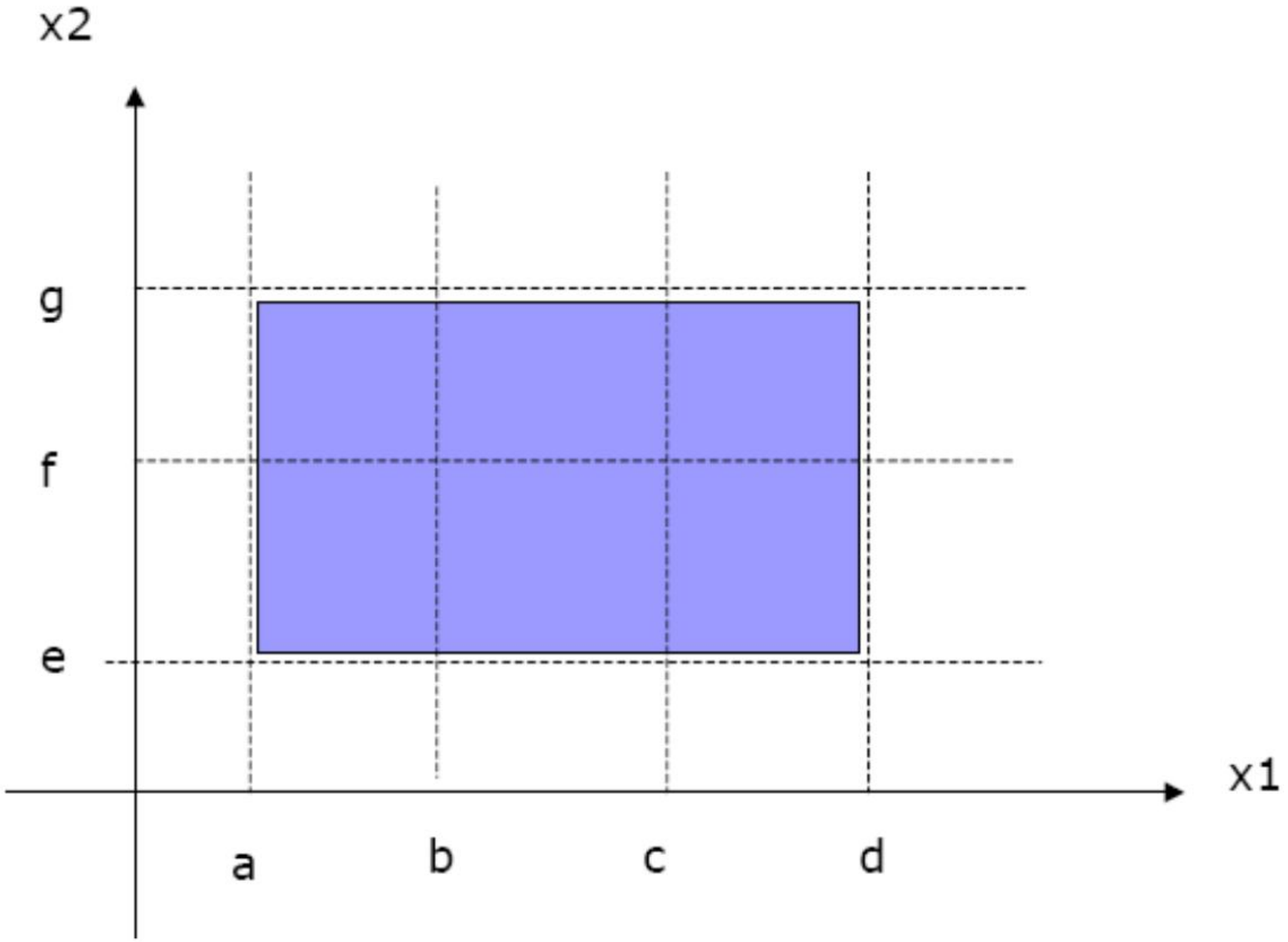
- 等价类测试的分类：
  - 弱一般等价类测试
  - 强一般等价类测试
  - 弱健壮等价类测试
  - 强健壮等价类测试

- 讨论对象
  - 一个具有两个自变量x1和x2的函数F

    F实现为一个程序且输入变量x1和x2的边界以及边界内的区间

    - a≤x1≤d,　区间为[a, b], (b, c), [c, d]
    - e≤x2≤g,　区间为[e, f], [f, g]

- 讨论对象(续)



1. 弱一般等价类测试

- 弱一般等价类测试的假设：“单缺陷”

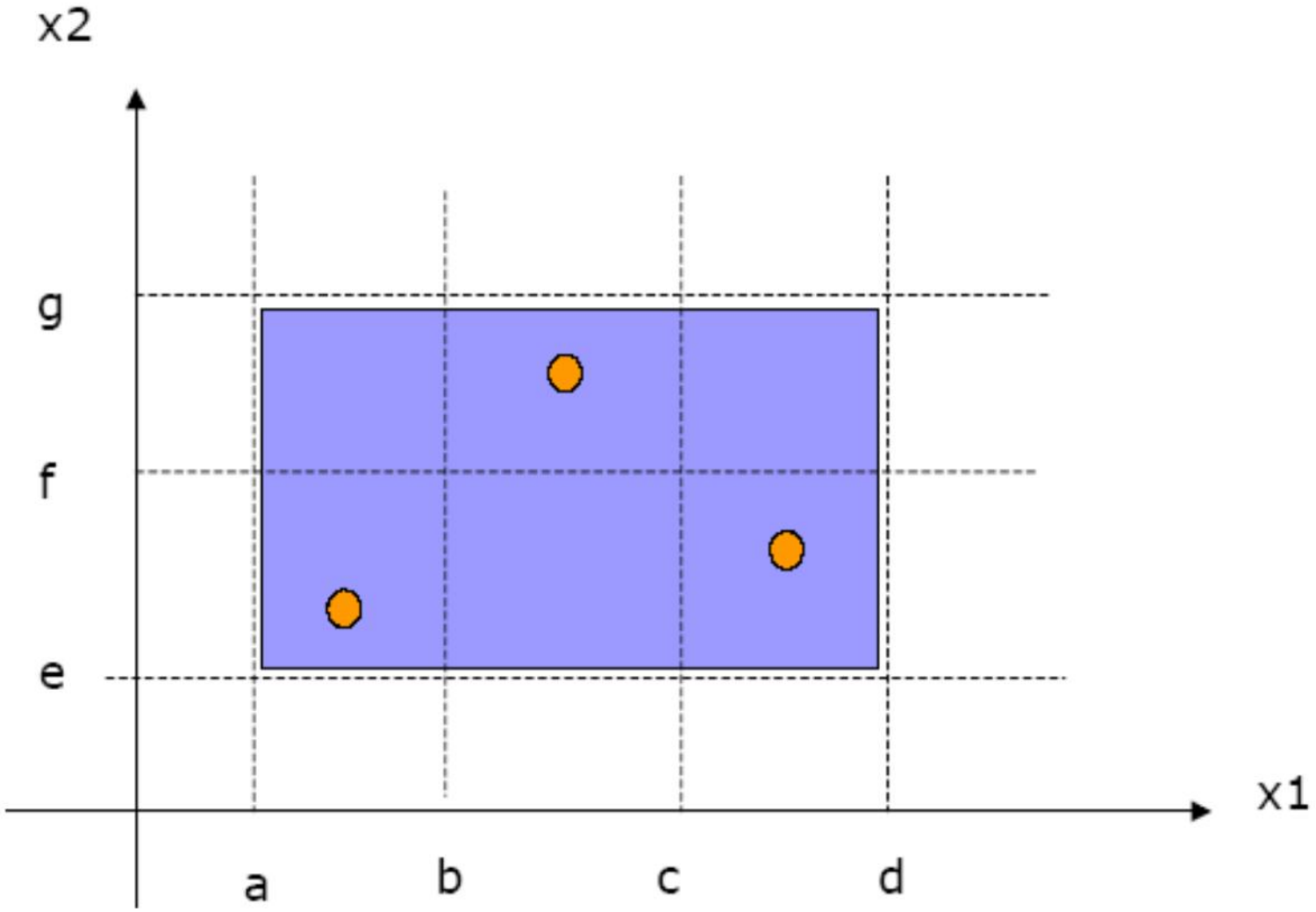
  假设

  失效极少是由两个或多个缺陷的同时发生引起的

- 测试用例设计：

  一选取的测试用例覆盖所有的有效等价类。

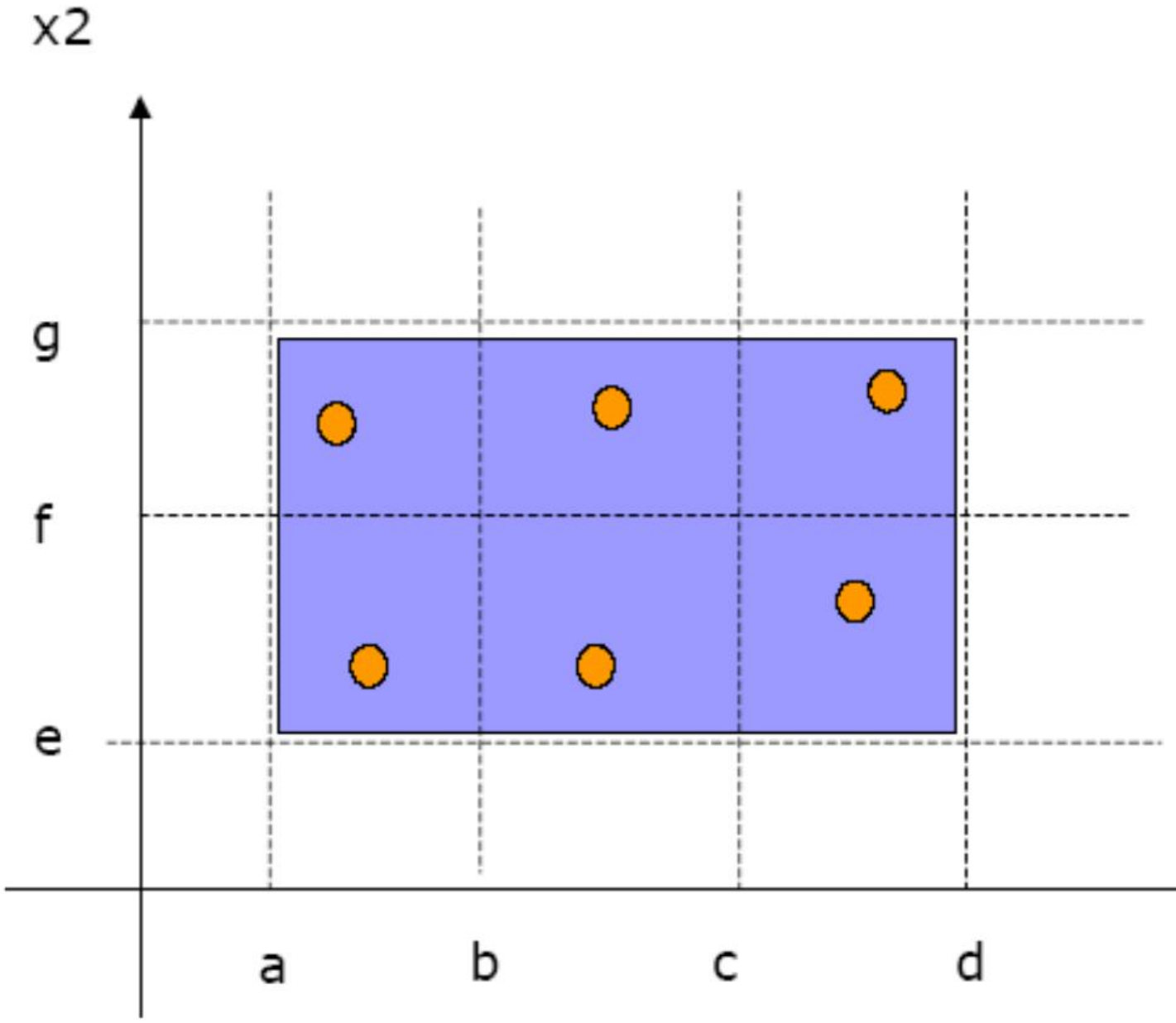
弱一般等价类测试用例



2. 强一般等价类测试

- 基于多缺陷假设；
- 测试用例设计：

  -将每个变量的有效等价类做笛卡尔积，设计测试用例覆盖笛卡尔积的每个元素。





思考：

有n个变量的函数，强一般等价类测试会产生多少个测试用例？

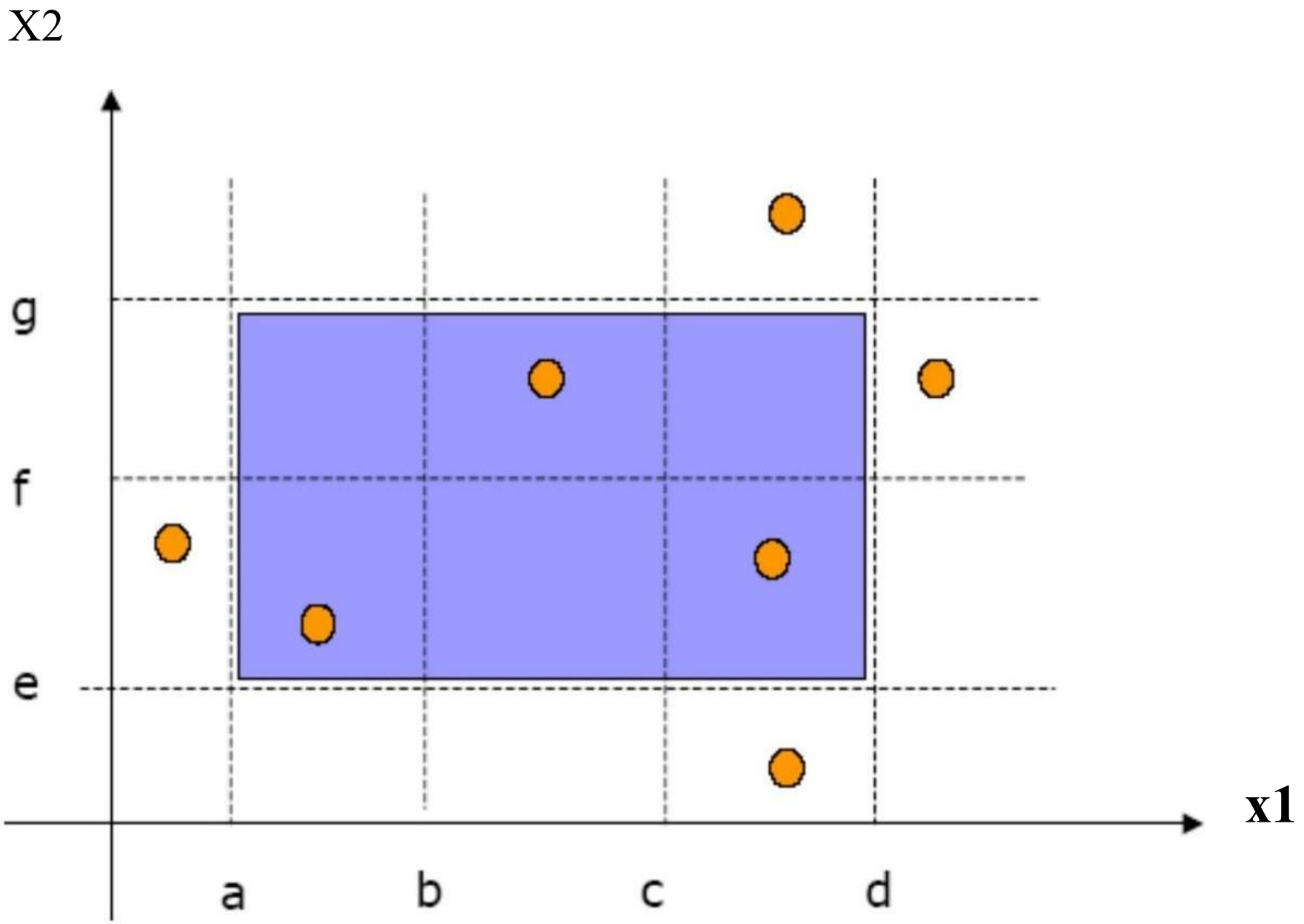
X1

a b c d

3. 弱健壮等价类测试

- 弱：基于单缺陷假设
- 健壮：考虑无效值
- 测试用例设计：
  - 对有效输入，覆盖每个有效等价类的一个值
  - 对无效输入，测试用例将拥有一个无效值，并保持其余的值都是有效的

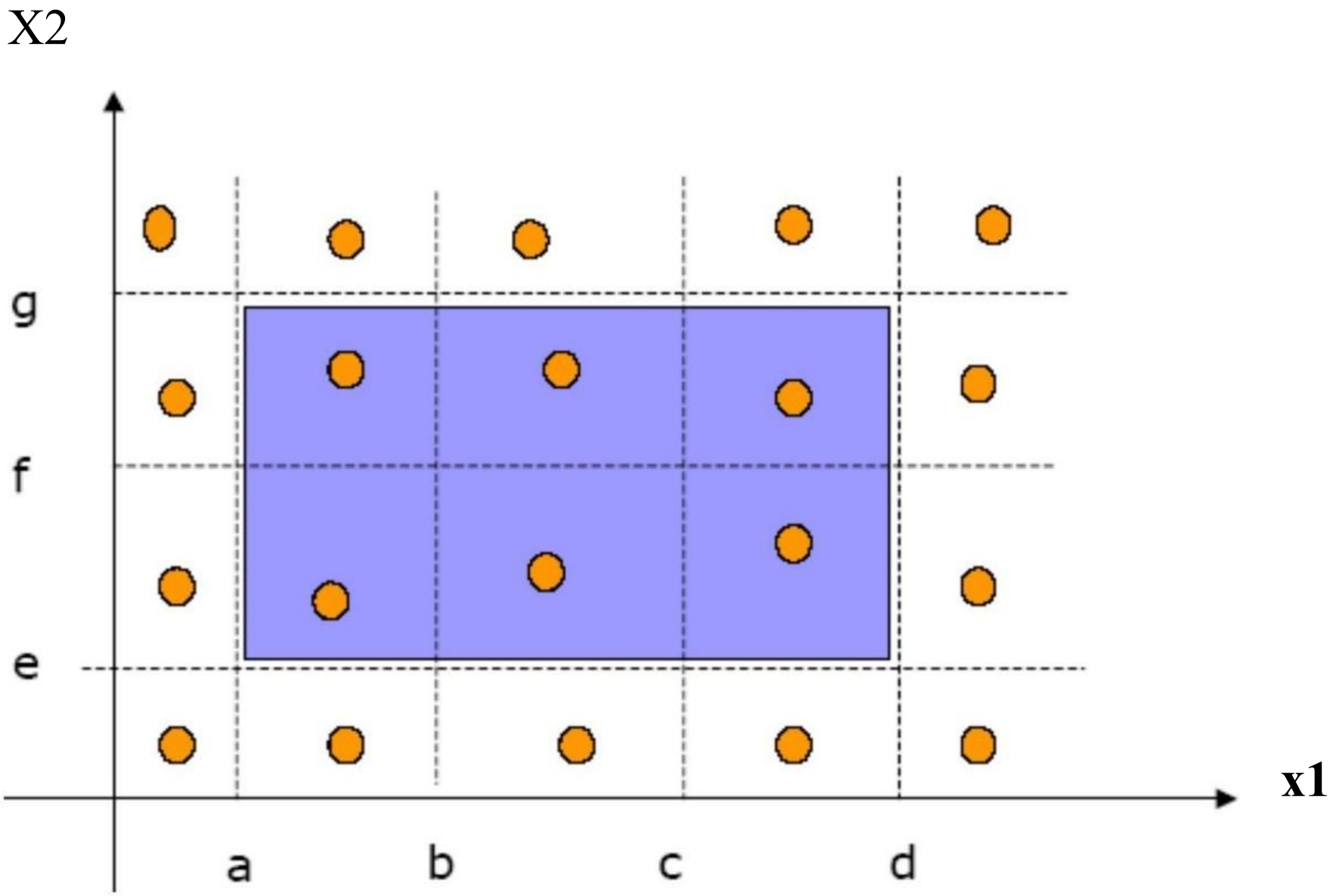
弱健壮等价类测试用例



4. 强健壮等价类测试

- 强健壮等价类测试：基于多缺陷假设，从所有等价类的笛卡尔积的每一个元素中获得测试用例
  - 强：多缺陷假设
  - 健壮：考虑无效值

强 健 壮 等 价 类 测 试 用 例



5. 用等价类设计测试用例

- (1)划分等价类，形成等价类表
- (2)设计一新的测试用例，使其尽可能多地覆盖尚未覆盖的有效等价类，重复这一步骤，直到所有的有效等价类都被覆盖为止；
- (3)设计一新测试用例，使其只覆盖一个无效等价类，重复这一步骤直到所有无效等价类均被覆盖；

4.2.3案例

- 三角形问题

  以不同的角度划分等价类：

  输出值域等价类

  输入条件(数据范围)的等价类

  以“整数相等”关系划分的等价类

  以两边之和与第三边之间的关系

1. 输出值域等价类

- 输出值域等价类
  - R1={<a, b, c>:有三条边a、b和c的等边三角形}
  - R2={<a,b,c>: 有三条边a、b和c的等腰三角形}
  - R3={<a, b, c>:有三条边a、b和c的一般三角形}
  - R4={<a, b, c>:三条边a、b和c不能构成三角形}

2. 输入条件的等价类

- 有效等价类：

  正整数；

  – 三个数；

- 无效等价类：

  零或负数；

  – 少于三个数；

3. “整数相等”关系划分等价类

- $D1=\{<a,b,c>:a=b=c\}$
- $D2=\{<a,b,c>:a=b,a\neq c\}$
- $D3=\{<a,b,c>:a=c,a\neq b\}$
- $D4=\{<a,b,c>:b=c,a\neq b\}$
- $D5=\{<a,b,c>:a\neq b,a\neq c,b\neq c\}$

4. 以两边之和与第三边之间的关系划分等价类

·D6={<a,b,c>:a≥b+c}

·D7={<a,b,c>:b≥a+c}

·D8={<a,b,c>:c≥a+b}

● 等价类D6 的进一步划分

-D6‘={<abc>:a=b+c}

-D6"={<abc>:a>b+c}

三角形的等价类

输入条件	有效等价类	编号	无效等价类		编号
三个正整数	**正整数**	1	一边为非正整数	a为非正整数	10
				b为非正整数	11
				c为非正整数	12
			两边为非正整数	a, b为非正整数	13
				a, c为非正整数	14
				b, c为非正整数	15
			三边均为非正整数		16
	三个数	2	只输入一个数	只给a	17
				只给b	18
				只给c	19
			只输入两个数	只给a, b	20
				只给a, c	21
				只给b, c	22
			未输入数		23

三角形的等价类

输入条件	有效等价类	编号	无效等价类	编号
构成一般三角形	a+b>C, 且a≠b≠c	3	a+b<C	24
			a+b=C	25
	a+c>b, 且a≠b≠c	4	a+c<b	26
			a+c=b	27
	b+c>a, 且a≠b≠c	5	b+C<a	28
			b+c=a	29
构成等腰三角形	a=b, a≠c, 且两边之和大于第三边	6		
	a=c, a≠b, 且两边之和大于第三边	7		
	b=c, a≠b, 且两边之和大于第三边	8		
构成等边三角形	a=b=C	9		



● 电话号码问题

某城市电话号码由三部分组成，分别是：

地区码——空白或4位数字；

前缀——为三位数字，但不能为“0”，“1”；

后缀————4位数字。

假定被测程序能接受一切符合上述规定的电话号码，拒绝所有不符合规定的电话号码。请用强一般等价类方法进行测试，设计测试用例。



NextDate问题

- NextDate 函数包含三个变量month、day 和 year， 函数的输出为输入日期后面一天的日期 。 (1800≤ year≤2050)

用弱健壮等价类测试方法设计测试用例。

课堂练习

C SDN 会员一登录



请设计测试用例，测试登录功能。

等价类测试指导方针和观察

- 1. 等价类测试的弱形式不如对应的强形式的测试全面。
- 2. 如果实现语言是强类型，则没有必要使用健壮形式的测试。
- 3. 如果错误条件非常重要，则进行健壮形式的测试是合适的。

- 4. 如果输入数据以离散值区间和集合定义，则等价类测试是合适的。当然也适用于如果变量值越界系统就会出现故障的系统。
- 5. 通过结合边界值测试，等价类测试可得到加强。
- 6. 如果程序函数很复杂，则等价类测试是被指示的。在这种情况下，函数的复杂性可以帮助标识有用的等价类。

- 7. 强等价类测试假设变量是独立的，相应的测试用例相乘会引起冗余问题。如果存在依赖关系，则常常会生成错误测试用例。
- 8. 在发现合适的等价关系之前，可能需要进行多次尝试。

小　　结

- ◇ 等价类
- ◆ 等价类分类
  - + 弱一般等价类测试
  - + 强一般等价类测试
  - + 弱健壮等价类测试
  - + 强健壮等价类测试

黑盒测试(3)

4.3 基于判定表的测试

- 判定表的组成
- 基于判定表的测试
- 案 例

4. 3. 1判定表

判定表的原理

- 判定表的组成
- 判定表建立的步骤

1. 判定表的原理

判 定 表

- 判 定 表 (Decision Table)，又叫决策表
- 判定表是分析和表达多逻辑条件下执行不同操 作的判定表的工具。

  在程序设计发展的初期，决策表就已被当作编 写 程序的辅助工具了。

2. 判定表的组成

■判定表的组成

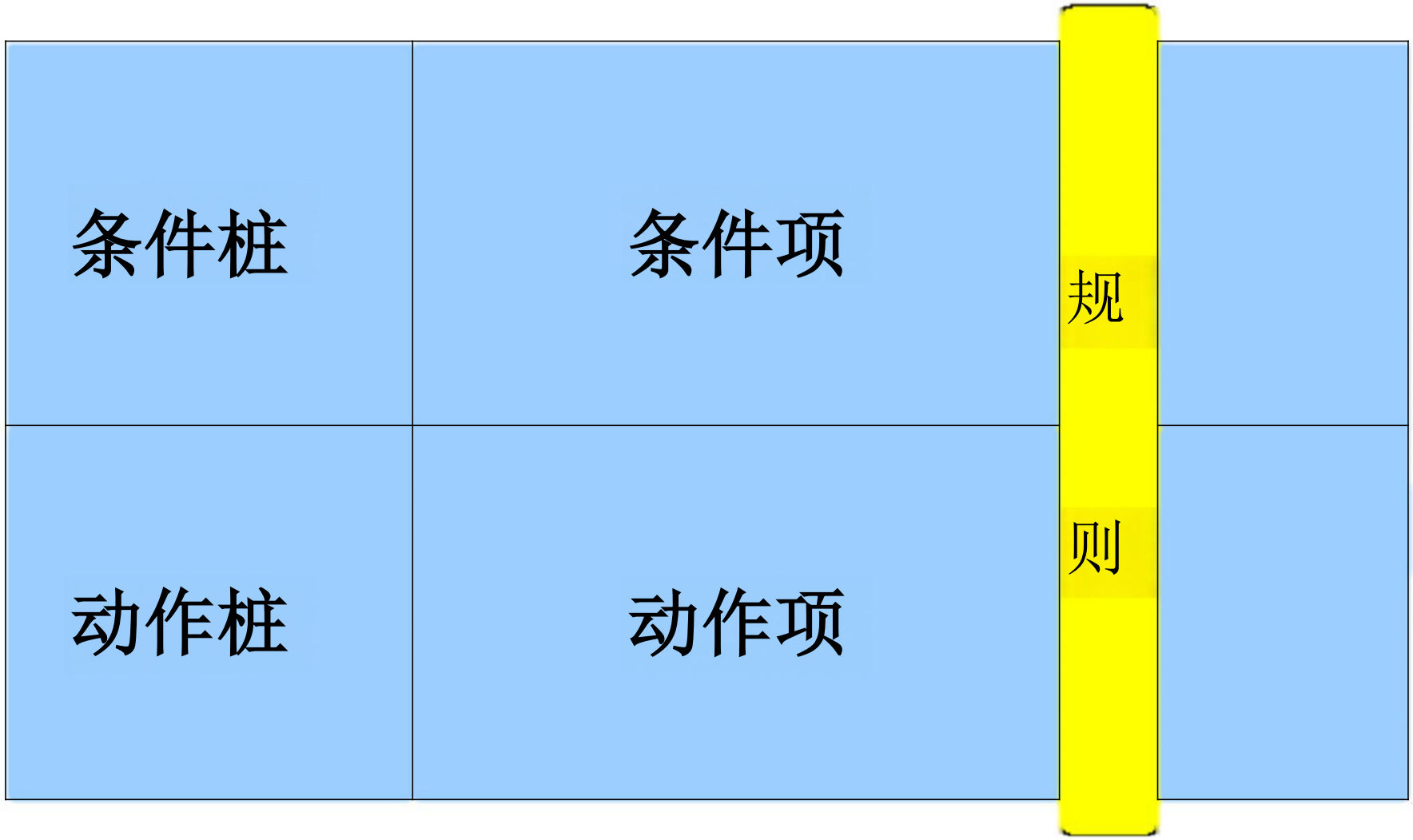
十 条件桩 (Condition Stub)

十 条件项 (Condition Entity)

动作桩 (Action Stub)

动作项 (Action Entity)

判定表的组成



规则

- 任何一个条件组合的特定取值及其相应要执行的操作称为规则；
- 在判定表中贯穿条件项和动作项的一列就是一条规则；
- 判定表中列出多少组条件取值，也就有多少条规则，即条件项和动作项有多少列。

规则示例：三角形问题

a, b, c构成三角形?	N	Y	Y	Y	Y	Y	Y	Y	Y
a=b?	–	Y	Y	Y	Y	N	N	N	N
a=c?		Y	Y	N	N	Y	Y	N	N
b=c?		Y	N	Y	N	Y	N	Y	N
非三角形	X								
不等边三角形									X
等腰三角形					X		X	X	
等边三角形		X							
不可能			X	X		X			

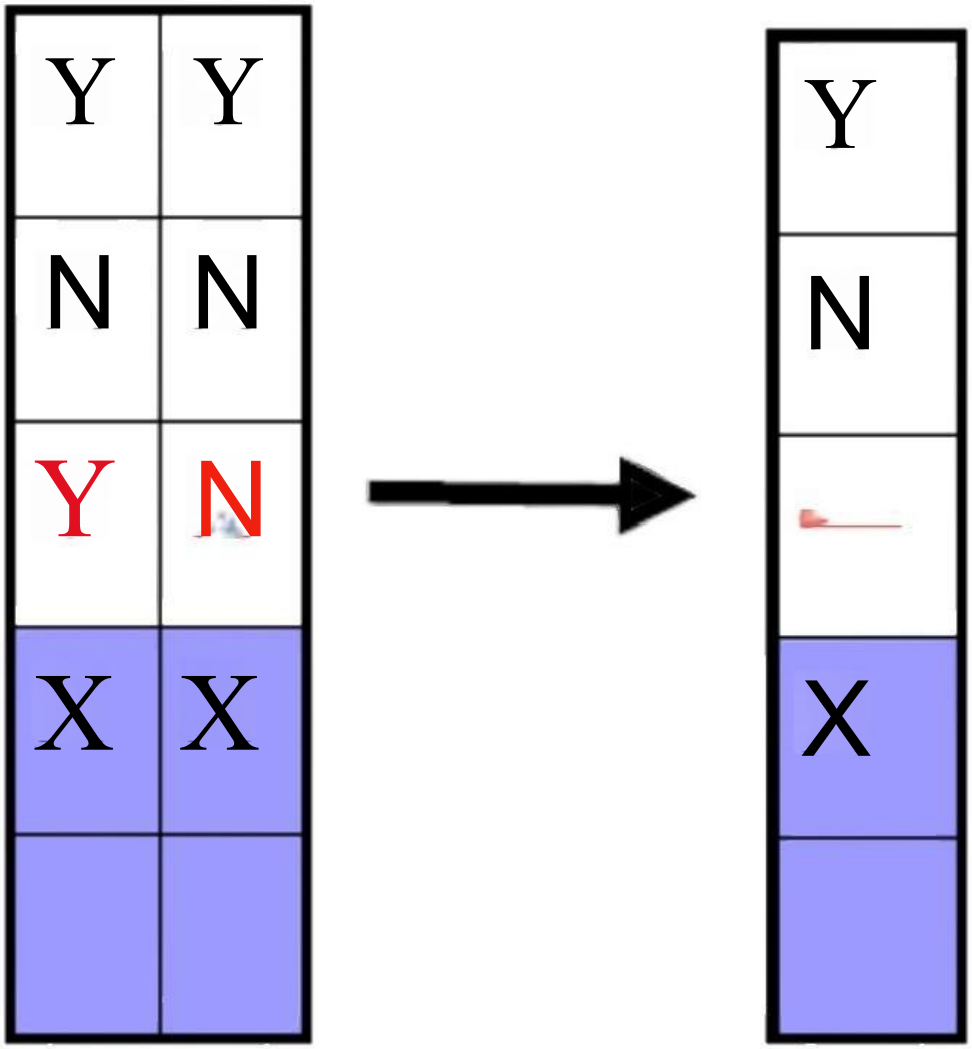
规则合并

- 规则合并

十规则合并：就是判定表的简化。

有两条或多条规则具有相同的动作，并且其条件项之间存在着极为相似的关系，就可以将规则合并。

两条规则合并成一条



■ 条件项“—”表示与取值无关

两条规则的进一步合并

Y	Y
-	N
N	N
X	X



Y
-
N
X

- 条件项“—”在逻辑上包含其它的条件

3. 基于判定表的测试

■ 判定表建立步骤

①确定规则的个数；

■假如有n个条件，每个条件有两个取值(0,1)，则有2 种规则；

②列出所有的条件桩和动作桩；

③填入条件项；

④填入动作项，得到初始决策表；

⑤简化，合并相似规则(相同动作)。

基于判定表的测试

■ 基于判定表的测试

根据输入输出绘制判定表

设计测试用例覆盖判定表中各条规则

4.3.3 案例1:考生录取

问题描述

某程序规定：“对总成绩大于450分，且各科成绩均高于85分或者是优秀毕业生，应优先录取，其余情况作其他处理”。请建立判定表。

- (1) 列出所有的条件桩和动作桩
  - ① 总成绩大于450分吗?
  - ② 各科成绩均高于85分吗?
  - ③ 优秀毕业生吗?
- 动作桩有两项:
  - ① 优先录取;
  - ② 作其他处理;
- (3) 规则的个数:

(4) 绘制判定表

		1	2	3	4	5	6	7	8
条件	总成绩大于450分吗	Y	Y	Y	Y	N	N	N	N
	各科成绩均高于85分吗?	Y	Y	N	N	Y	Y	N	N
	优秀毕业生吗?	Y	N	Y	N	Y	N	Y	N
动作	优先录取;	√	√	√					
	作其他处理				√	√	√	√	√

(5)简化后的判定表

		工	2	3	4	5
条	总成绩大于450分吗?	Y	Y	Y	N	N
	各科成绩均高于85分吗?	Y	N	N	Y	N
	优秀毕业生吗?	—	Y	N	—	—
动	优先录取;	√	√			
	作其他处理			√	√	√

(6)进一步简化

		1	2	3	4
条	总成绩大于450分吗?	Y	Y	Y	N
	各科成绩均高于85分吗?	Y	N	N	—
	优秀毕业生吗?	—	Y	N	—
动	优先录取;	√	√		
	作其他处理			√	√

4.3.3案例：隔一日问题

■ 隔一日问题

十程序有三个输入变量month、day、year (month、day和year均为整数值，并且满足：1≤month≤12和1≤day≤31)，分别作为输入日期的月份、日、年份，通过程序可以输出该输入日期在日历上隔一天的日期。例如，输入为2005年11月29日，则该程序的输出为2005年12月1日。请用判定表测试法进行测试。

- (1)分析各种输入情况，列出为输入变量month、day、year　划分的有效等价类。
  - + month 变量的有效等价类：
  - + day 变量的有效等价类：
    year 变量的有效等价类



思考：如何划分等价类?

- (2)分析程序规格说明，结合以上等价类划分的情况给出问题规定的可能采取的操作(即列出所有的动作桩)。

  a1:day+2

  a2:day=1

  a3:day=2

  a4:month+1

  +a5:month=1

  a6:year+1

  a7: 不可能

(3)绘制判定表

- (4) 设计测试用例

一个游戏

主持人对甲、乙、丙三人说：“这里有三顶红帽子，两顶白帽子。现在用布蒙上你们的眼睛，给你们每人戴上一顶帽子，然后请你们依次睁开眼睛，能正确说出自己所戴帽子的颜色者有奖。”

带完帽子后，甲拿下布后看了其他两人的帽子说：“我不知道。”然后，乙解开布看了其他两人的帽子后说：“我不知道。”轮到丙时，他没有拿下布就正确地说出了自己所戴帽子的颜色。

- 试问：丙戴的是什么帽子?他是怎样得出结论的?

一个游戏







04

04



4

	1	2	3	4	5	6	7
甲	R	R	R	R	W	W	W
乙	R	R	W	W	R	R	W
西	R	W	R	W	R	W	R
甲不知							
乙不知							
西知道							

课堂练习

- 货运收费策略：铁路运费时，若收货地点在本省以内，快件每公斤1.5元，慢件每公斤1元。若收货地点在外省，重量小于或等于20公斤，快件每公斤2元，慢件每公斤1元，若重量大于20公斤，超重部分每公斤0.2元。

  请绘制判定表。



指导方针和观察

判定表技术适用的应用程序的特征：

- if-then-else　逻辑很突出
- 输入变量之间存在逻辑关系
- 涉及输入变量子集的计算
- 输入与输出之间存在因果关系

很高的McCabe　圈复杂度

黑盒测试(4)

问题引入

- 等价类划分方法和边界值分析方法，都是着重考虑输入条件，但未考虑输入条件之间的联系，相互组合等.

- 如果考虑输入条件之间的相互组合，组合的情况可能会相当多. 因此必须考虑采用一种适合于描述对于多种条件的组合，相应产生多个动作的形式来考虑设计测试用例. 这就需要利用因果图。

因果图

本节内容

- 因果图分析法
- 因果图介绍
- 利用因果图导出测试用例
- 举例

1. 因果图分析法

- 因果图（**Cause-Effect　Graphics**）分析法

  因果图分析法最终生成的是判定表，适合于检查程序输入条件的各种组合情况。

- 因果图分析依据

  需求规格说明中的因果关系。

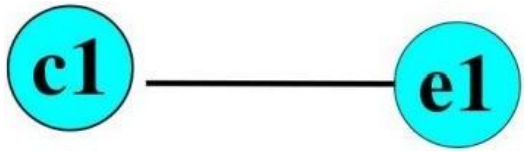
- 作用

  能够按一定步骤，高效率地选择测试用例；

  指出程序规约描述中存在着的问题；

2. 因果图的基本符号

恒等



c1=1 ⇨ e1=1

c1=0 ⇨ e1=0

非

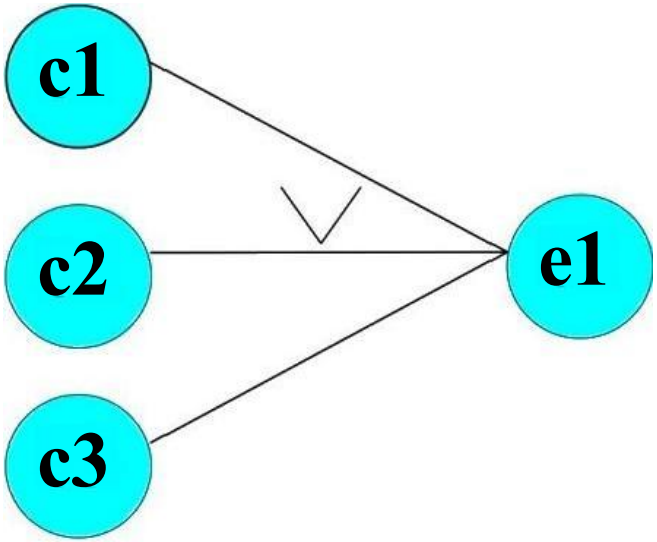


c1=1 — e1=0

c1=0 e1=1

2. 因果图的基本符号

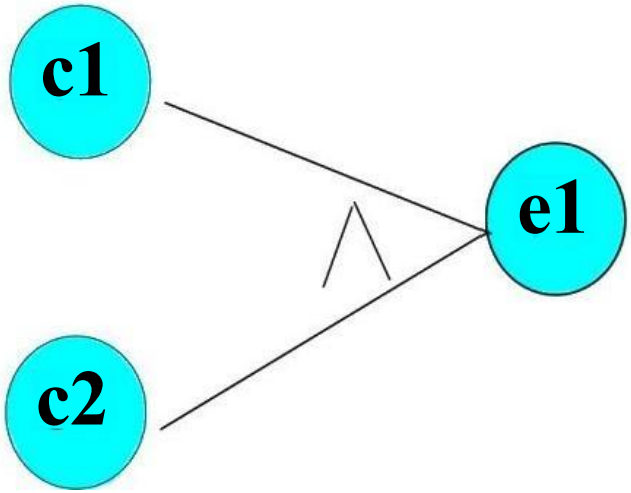
或



c1=1 或 c2=1 或 c3=1 则 e1=1

否则 e1=0

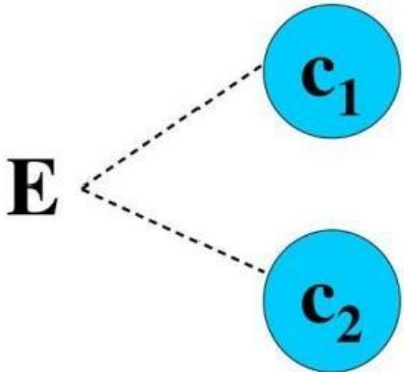
与



c1=1 且c2=1 e1=1

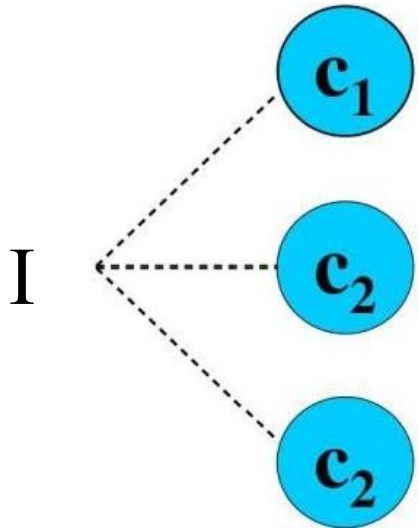
否则 e1=0

输入条件的约束



E(异)

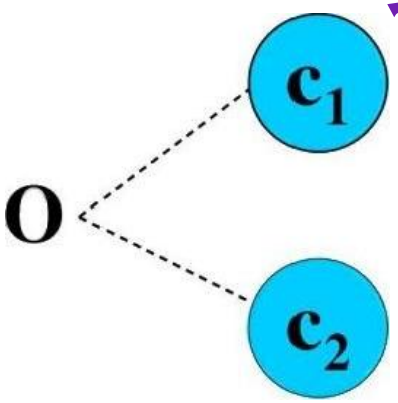
E (异)：表 示c1,c2　中至多有一个可能为1, 即c1和c2不能同时为1。



I(或)

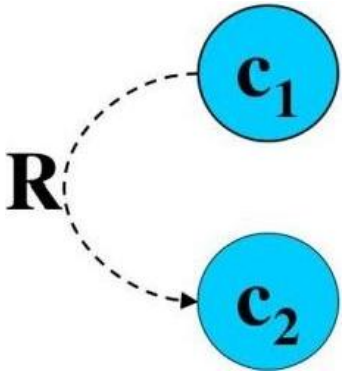
I (或)：表 示c1,c2,c3　　中至少有一个是1, 即c1,c2,c3　不能同时为0。

输入条件的约束



O（唯一）：表 示c1,c2　中必须有一个且仅有一个为1。

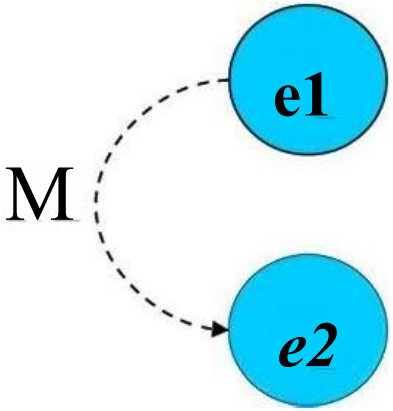
O（唯一）



R（要求）：表示c1是1时，c2必须是1，即不可能c1是1时c2是0。

R（要求）

输出条件的约束



M（强制）

M（强制）：表示如果结果e1是1时，则结果e2强制为0。

3. 利用因果图设计测试用例

- 1. 分析程序规格说明的描述中，哪些是原因，哪些是结果?

  原因常常是**输入条件**或是输入条件的等价类;

  结果是程序的**输出**;

- 2. 分析程序规约的描述中语义的内容，并将其表示成连接各个原因与各个结果的“因果图”;

3. 利用因果图设计测试用例

- 3. 在因果图上使用若干个特殊的符号

  标明约束条件；

- 4. 把因果图转换成判定表；

- 5. 把判定表中每一列表示的情况写成

  测试用例；

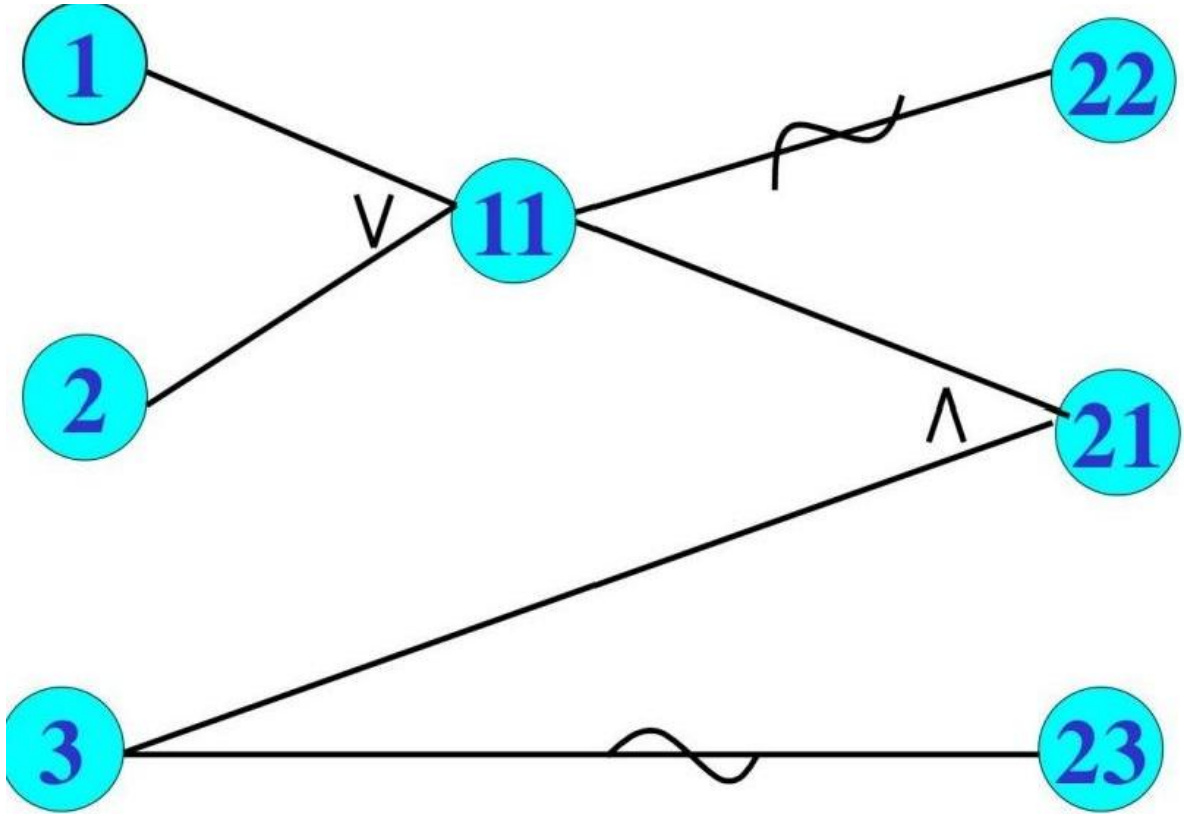
案例1：软件规格说明书

- **第一列字符必须是A或B，第二列字符必须是一个数字**，在此情况下进行文件的修改；
- 但如果第一列字符不正确，则给出信息**L;**
- 如果第二列字符不是数字，则给出信息**M;**

原因和结果

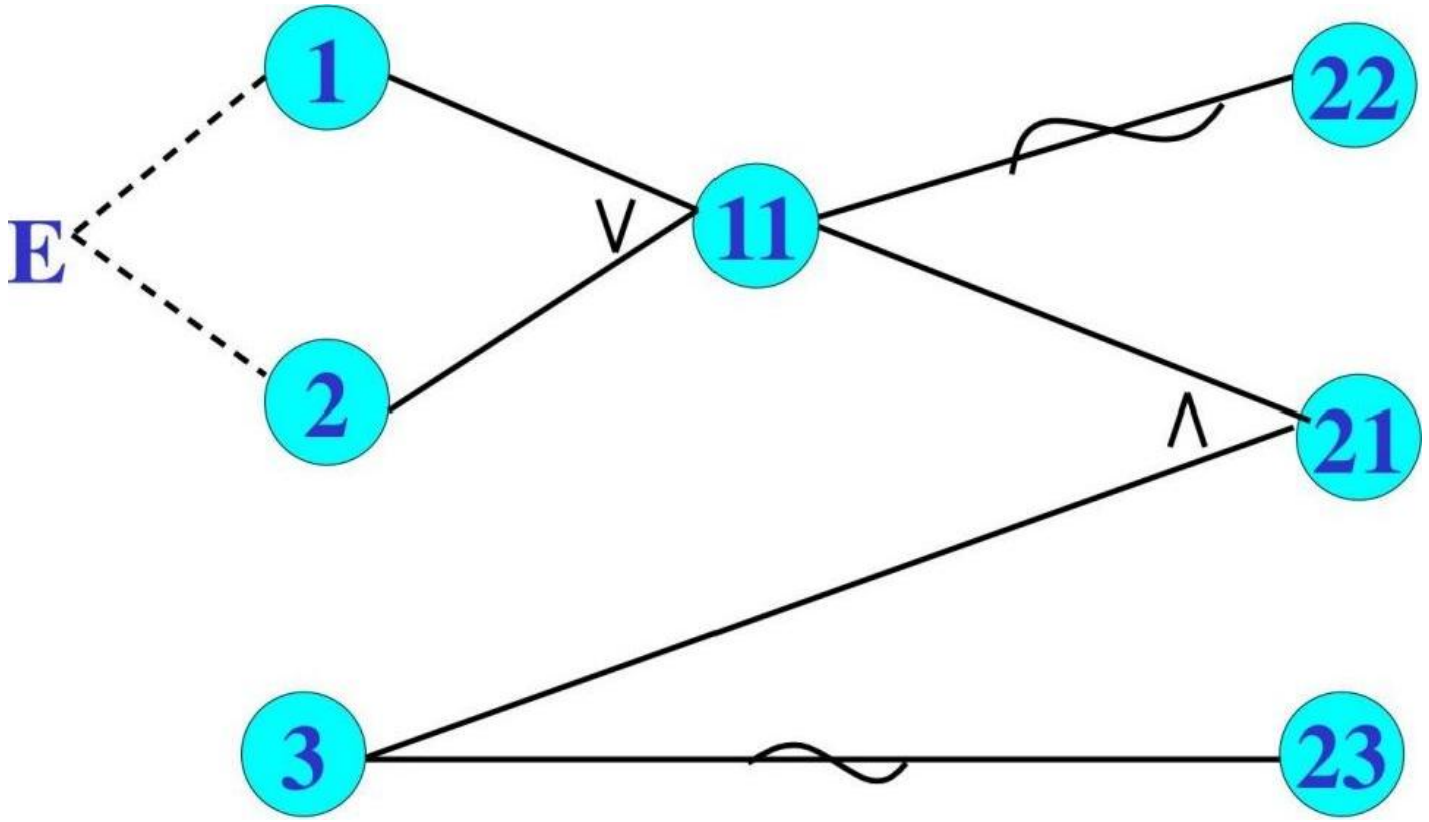
- 原因
  - 1------第一列字符是A;
  - 2------第一列字符是B;
  - 3------第二列字符是一数字
- 结果
  - 21------修改文件;
  - 22------给出信息L;
  - 23 ------给出信息M;

因 果 图



- 注：11是中间节点

具有约束的因果图



- 考虑到原因1和原因2不可能同时为1，因此在因果图上施加E约束

根据因果图所建立的判定表

		1	2	3	4	5	6	7	8	
条件	1	1	1	1	1	0	0	0	0	原因
	2	1	1	0	0	1	1	0	0	
	3	1	0	1	0	1	0	1	0	
	11	/////	//	1	1	1	1	0	0	
动作	22	//	//	0	0	0	0	1	1	结果
	21	//	//	1	0	1	0	0	0	
	23	//	//	0	1	0	1	0	1	
测试用例		// /////	// ///	A3 A8	AM A?	B5 B4	BN B!	C2 X6	DY P;	

设计测试用例

- 最左边两列，条件1和2同时为1是不可能的，排除掉，根据表可设计出6个测试用例。

Test1: 输入数据一A3 预期输出：修改文件；

Test2: 输入数据一AM 预期输出：给出信息M;

Test3: 输入数据一B5 预期输出：修改文件；

Test4: 输入数据一BN 预期输出：给出信息M;

Test5: 输入数据一C2 预期输出：给出信息L;

Test6: 输入数据一DY 预期输出：给出信息L和M;

案例2：电力收费

- 问题描述

某电力公司有A、B、C、D　四类收费标准，并规定：

居民用电：<100度/月，按A 类收费

≥100度/月，按B 类收费

动力用电：<10000度/月，非高峰，　B 类收费

≥10000度/月，非高峰，C 类收费

<10000度/月，高峰，C 类收费

≥10000度/月，高峰，D类收费

请用因果图法设计测试用例。

(1)列出原因和结果

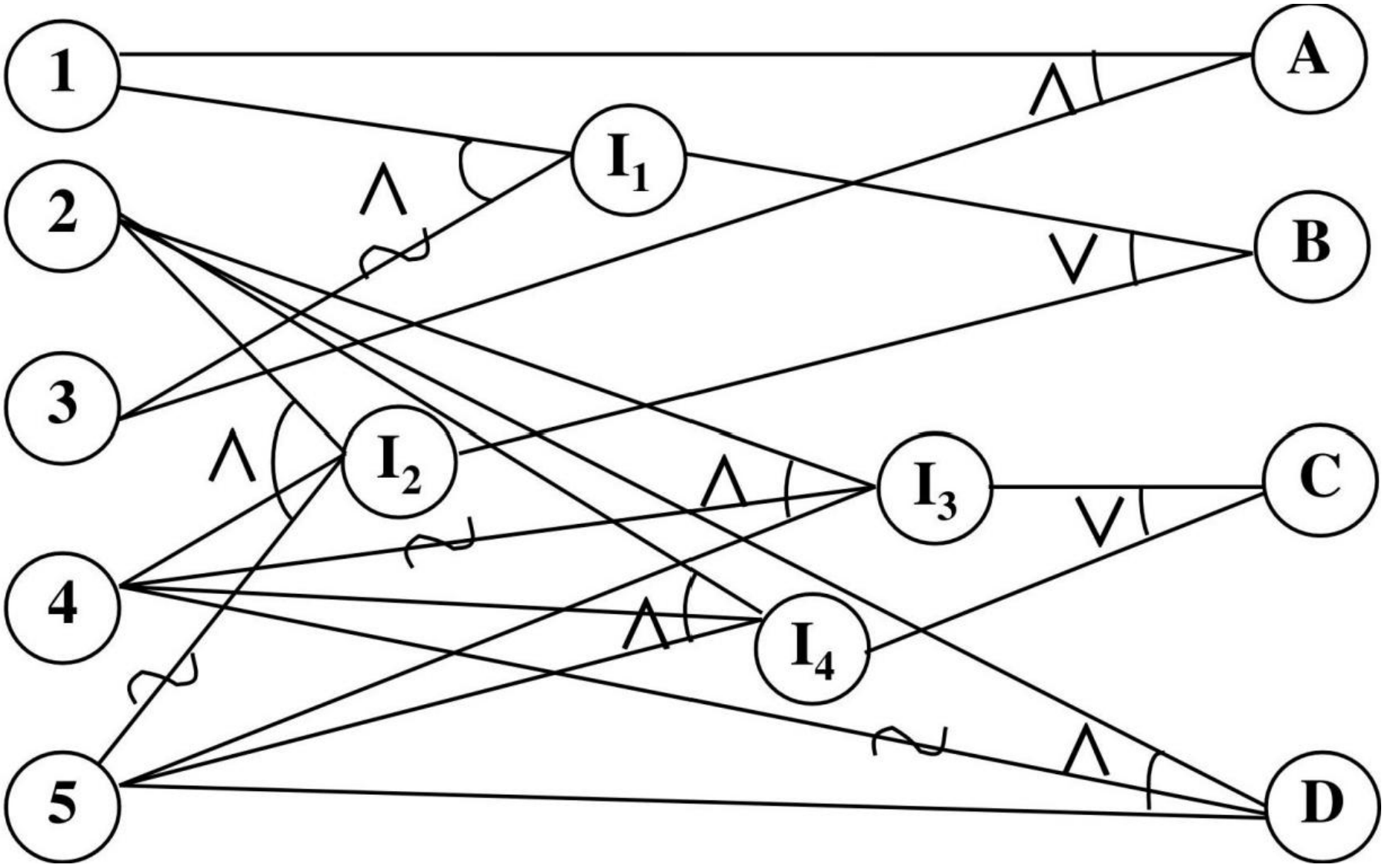
原 因：

- 1-———居民用电
- 2——动力用电
- 3———<100度/月　3’———≥100度/月
- 4———非高峰　　4’———高峰
- 5———<10000度/月5’———≥10000度/月

结果：

- A——— 按A 类收费
- B- ———按B 类收费
- C ———按C 类收费
- D——— 按D 类收费

● (2) 绘制因果图



(3) 绘制判定表

组合条件		1	2	3	4	5	6
条件（原因）	1	1	1	0	0	0	0
	2	0	0	1	1	1	1
	3	1	0	—	—	—	—
	4	—	—	1	0	1	0
	5	—	—	0	0	1	1
动作（结果）	A	1	0	0	0	0	0
	B	0	1	1	0	0	0
	C	0	0	0	1	1	0
	D	0	0	0	0	0	1

注：表中“—”表示本条规则与此条件无关，或不适用。

(4) 设计测试用例

测试用例编号	输入数据	预期结果	覆盖组合条件
1	居民用电，99度/月	A类收费	1
2	居民用电，101度/月	B类收费	2
3	动力电，非高峰，9000度/月	B类收费	3
4	动力电，非高峰，1.01万度/月	C类收费	4
5	动力电，高峰，0.98万度/月	C类收费	5
6	动力电，高峰，1.1万度/月	D类收费	6

课堂练习

分析中国象棋中走马的实际情况

1、如果落点在棋盘外，则不移动棋子；

2、如果落点与起点不构成日字型，则不移动棋子；

3、如果落点处有自己方棋子，则不移动棋子；

4、如果在落点方向的邻近交叉点有棋子(绊马腿)，则不移动棋子；

- 5、如果不属于1-4条，且落点处无棋子，则移动棋子；

6、如果不属于1-4条，且落点处为对方棋子(非老将),则移动棋子并除去对方棋子；

7如果不属于1-4条，且落点处为对方老将，则移动棋子，并提示战胜对方，游戏结束。

二、根据分析明确原因和结果

- **原因：**

1、 落点在棋盘上；

2、 落点与起点构成日字；

3、 落点处不为自己方棋子；

4、 落点方向的邻近交叉点有棋子；

5、 落点处无棋子；

6、 落点处为对方棋子（非老将）；

7、落点处为对方老将。

二、根据分析明确原因和结果

- 结 果：

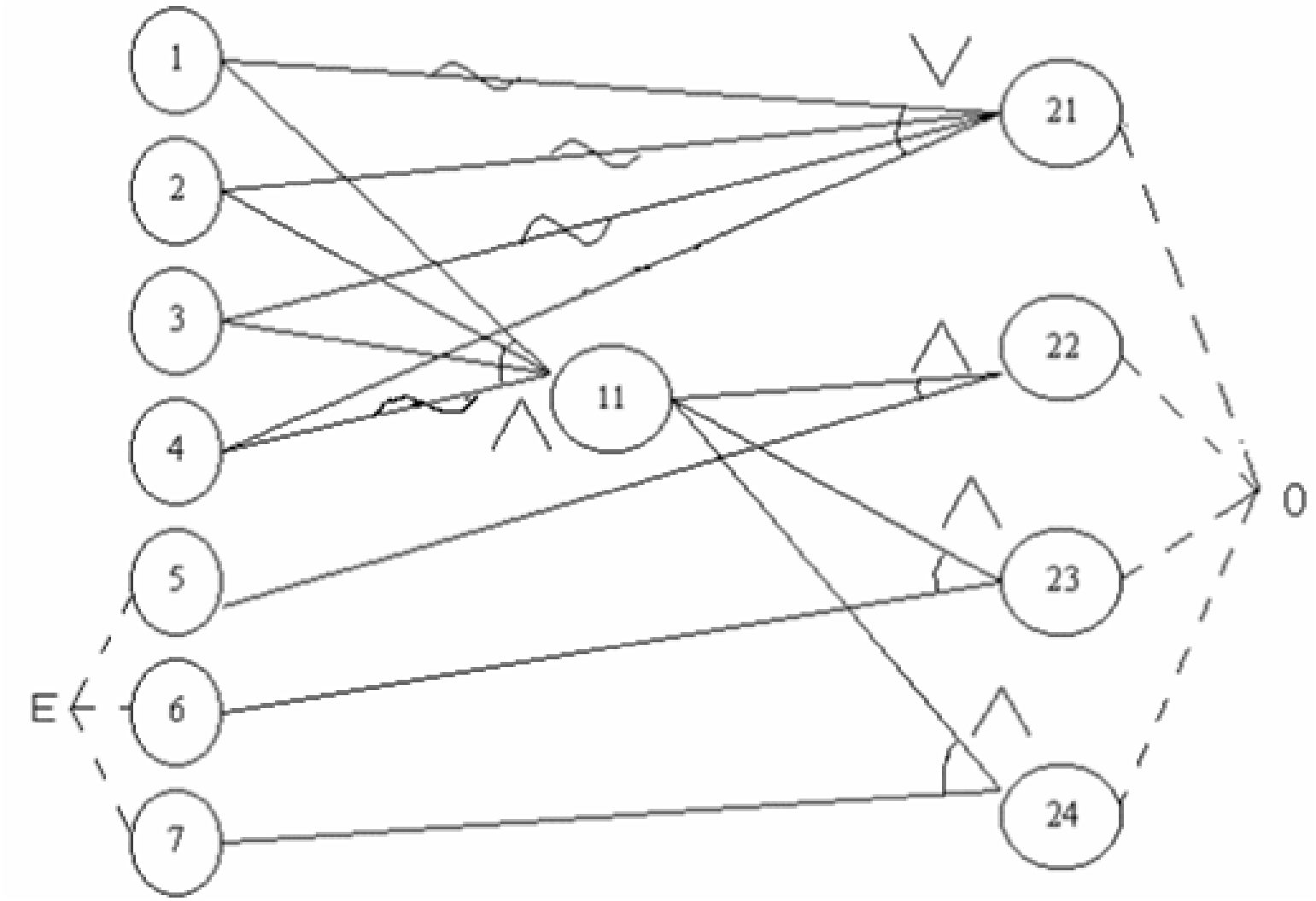
  21、不移动棋子；

  22、移动棋子；

  23、移动棋子，并除去对方棋子；

  24、移动棋子，并提示战胜对方，结束游戏。

- 添加**中间节点**11，目的是作为导出结果的进一步原因，简化因果图导出的判定表


1
2
3
4
5
6
7
11
21
22
23
24
E
O

三、建立判定表

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
原因	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
	2	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
	3	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
	4	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
结果	11	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
	21	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1

三、建立判定表

		1	2	3	4	5	6	7	8	9	`0	11	12	13	14	15	16
原因	11	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
	5	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
	6	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
	7	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
结果	22	0		0	1	0	0			0	0						
	23	0		0	0	0	1			0	0						
	24	0		0	0	0	0			0	1						

- 表中部分列被合并表示不可能发生的现象；
- 通过中间节点将用例的判定表简化为两个小表。减少工作量

课堂练习

● 自动售货机

处理单价为5角钱的饮料自动售货机：

投入5角或1元的硬币，压下[橙汁]或[啤酒]的按钮，送出饮料 。

如果投入1元硬币并压下按钮后，若售货机没有零钱找，则[零钱找完]红灯亮，饮料不送出并将1元硬币退出来。若有零钱找，则[零钱找完]红灯灭，送出饮料同时还退还5角硬币。

(1)分析原因、结果

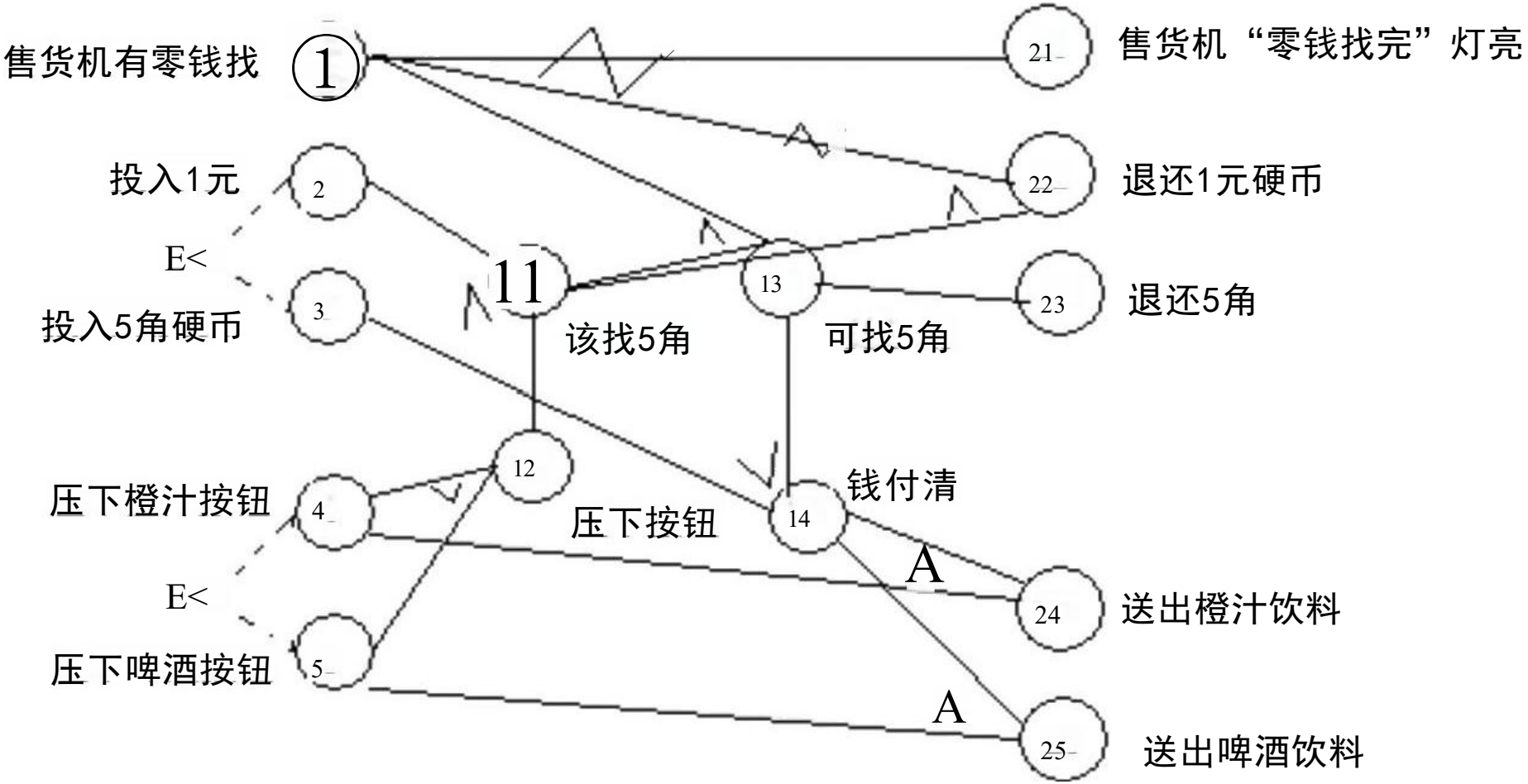
● 原 因 ：

1. 售货机有零钱找
2. 投入1元
3. 投入5角
4. 压下橙汁按钮
5. 压下啤酒按钮

□结 果 ：

21. 售货机[零钱找完]灯亮
22. 退还1元
23. 退还5角
24. 送出橙汁饮料
25. 送出啤酒

(2) 画出因果图


售货机有零钱找
1
投入1元
2
E<
投入5角硬币
3
11
该找5角
13
可找5角
21
售货机“零钱找完”灯亮
22
退还1元硬币
23
退还5角
12
压下橙汁按钮
4
压下按钮
14
钱付清
A
24
送出橙汁饮料
E<
压下啤酒按钮
5
A
25
送出啤酒饮料

(3) 转换成判定表

序号		1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1	2
条件	①	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	②	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
	③	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
	④	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
	⑤	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
中间结果	⑪						1	1	0		0	0	0		0	0	0						1	1	0		0	0	0		0	0	0
	⑫						1	1	0		1	1	0		1	1	0						1	1	0		1	1	0		1	1	0
	⑬						1	1	0		0	0	0		0	0	0						0	0	0		0	0	0		0	0	0
	⑭						1	1	0		1	1	1		0	0	0						0	0	0		1	1	1		0	0	0
结果	㉑						0	0	0		0	0	0		0	0	0						1	1	1		1	1	1		1	1	1
	㉒						0	0	0		0	0	0		0	0	0						1	1	0		0	0	0		0	0	0
	㉓						1	1	0		0	0	0		0	0	0						0	0	0		0	0	0		0	0	0
	㉔						1	0	0		1	0	0		0	0	0						0	0	0		1	0	0		0	0	0
	㉕						0	1	0		0	1	0		0	0	0						0	0	0		0	1	0		0	0	0
测试用例							Y	Y	Y		Y	Y	Y		Y	Y							Y	Y	Y		Y	Y	Y		Y	Y	

黑盒测试(5)

其它黑盒测试方法

- 场景测试
  - 正交试验
- 错误推测法

1. 场景测试

- **场景的定义**

场景从用户的角度描述系统的运行行为。

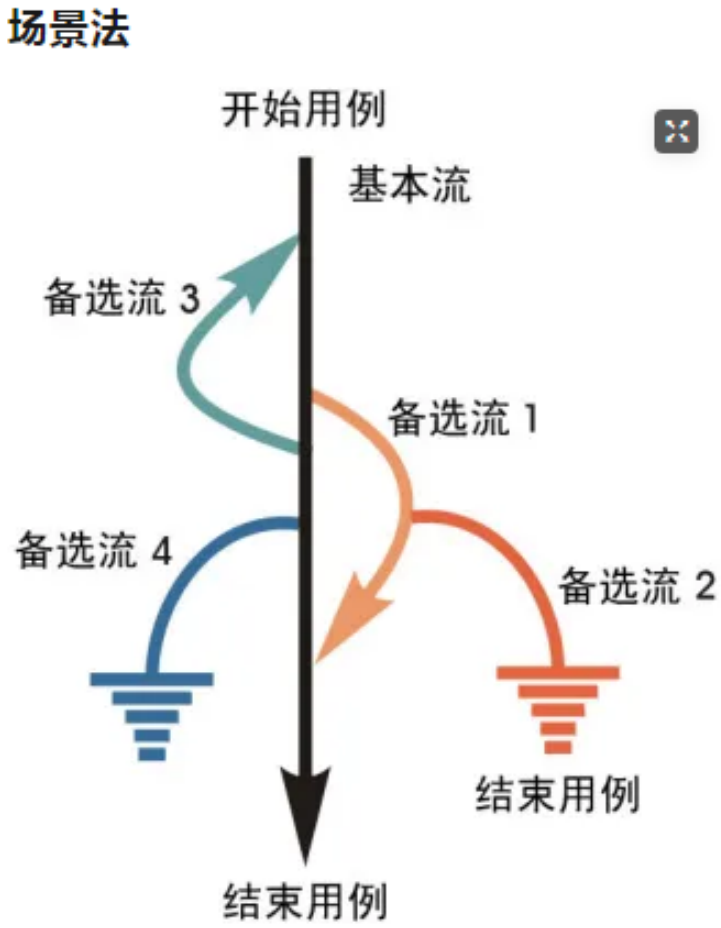
场景：是由一系列相关的活动组成的，而且场景中的活动还可以由一系列的场景构成。

现在的软件几乎都是用事件触发来控制流程的，事件触发时的情景便形成了场景，而同一事件不同的触发顺序和处理结果就形成事件流。

- **场景的构成**

  基本流

  备用流



场景设计

场景1:基本流

场景2:基本流备选流1

场景3:基本流备选流1备选流2

场景4:基本流备选流3

场景5:基本流备选流3备选流1

场景6:基本流备选流3备选流1备选流2

场景7:基本流备选流4

场景8:基本流备选流3备选流4

●场景测试

-(1)根据程序说明，找出程序的基本流及备选流；

(2)根据基本流和各项备选流生成不同的场景；

-(3)对每一个场景生成相应的测试用例，对每一个测试用例确定测试数据值。

- 分析ATM　自动取款机的场景流程，并设计测试用例和测试数据。

■基本流：　1. 插入磁卡；

2.ATM机验证帐户正确；

3. 输入密码正确，通过验证；

4. 输入取款金额；

5. 取出金额；

6. 取卡；

■备选流一：帐户不存在或受限制；

■备选流二：密码不正确，还有输入机会；

■备选流三：密码不正确，没有输入机会；

■备选流四：卡中余额不足；

■备选流五：ATM 机中余额不足；

■备选流六：超过每日最大提款限额；

■备选流七：输入金额非50倍数；

场景描述	基本流	备选流
场景1——成功的提款	基本流	
场景2——帐户不存在/帐户受限	基本流	备选流1
场景3——密码不正确(还有输入机会)	基本流	备选流2
场景4——密码不正确(不再有输入机会)	基本流	备选流3
场景5——卡中余额不足	基本流	备选流4
场景6——机中余额不足	基本流	备选流5
场景7——超过每日提款上限	基本流	备选流6
场景8——输入金额非100倍数	基本流	**备选流7**

序号	场景/条件	PIN	帐号	输入金额	帐面金额	ATM内的金额	预期结果
CW1	场景1:成功的提款	4987	809-498	100	500	2000	成功的提款。帐户余额被更新为400
CW2	场景2:帐户不存在/帐户受限	n/a	809-497	n/a	500	2000	提款选项不可用，用例结束
CW3	场景3:密码不正确(还有输入机会)	4987	809-498	n/a	500	700	警告消息，返回基本流步骤3输入密码
CW4	场景4:密码不正确(不再有输入机会)	4987	809-498	n/a	500	2000	警告消息，吞卡

序号	场景/条件	PIN	帐号	输入金额	帐面金额	ATM内的金额	预期结果
CW5	场景5:卡中余额不足	4987	809-498	600	500	2000	警告消息，返回基本流步骤4,输入金额
CW6	场景6:机中余额不足	4987	809-498	400	500	300	提示消息，返回基本流步骤4,输入金额
CW7	场景7:超过每日提款上限	4987	809-498	（24小时内已取款19000） 500	2000	3000	警告消息，返回基本流步骤4,输入金额
CW8	场景8:输入金额非50倍数	4987	809-498	110	500	2000	警告消息，返回基本流步骤4,输入金额

课堂练习

分析的网络购物的场景，并设计测试用例。

- 浏览商品；
- 立即购买或加入购物车
- 登录支付宝；
- 确认收货地址；
- 确认订单信息；
- 支付货款(使用支付宝或者网银)

2. 正交试验

- 正交试验设计法(Orthogonal experimental design)
- 正交试验法是研究多因素多水平的一种设计方法，它是根据正交性从全面试验中挑选出部分有代表性的点进行试验。这些有代表性的点具备了“均匀分散，齐整可比”的特点，正交试验设计是一种基于正交表的、高效率、快速、经济的试验。

3. 错误推测法

- **错误推测法：**靠经验和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例子。**

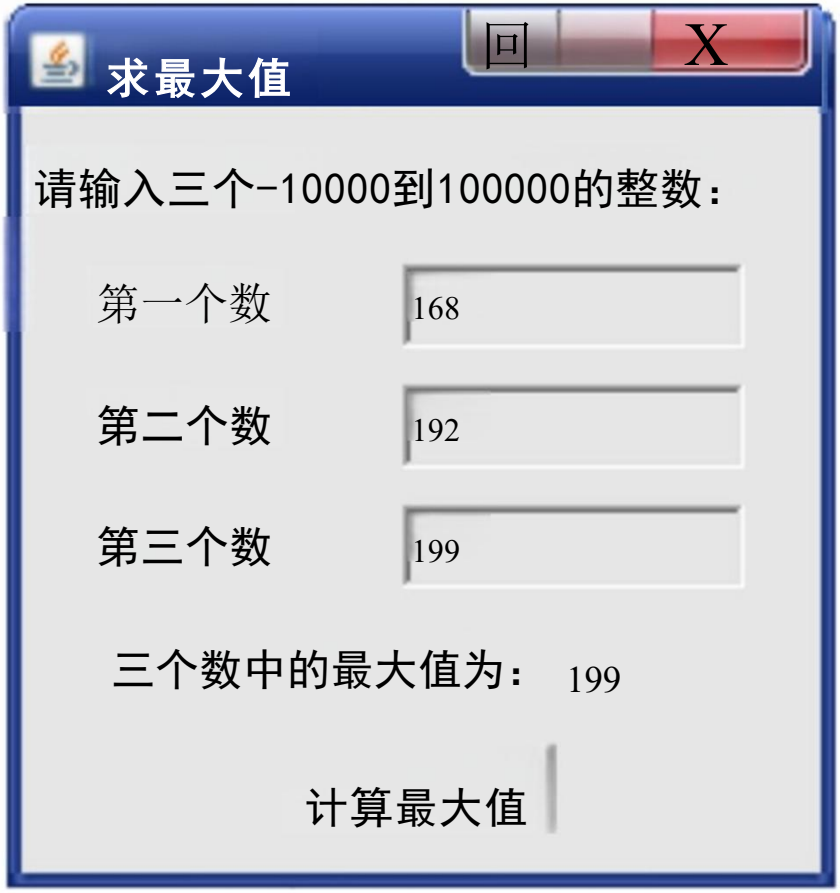
4.6综合案例

●求三个数中的最大值

现有一个小程序，能够求出三个在-10000到+10000间整数中的最大7-1。

用黑盒测试法

用 例 。



1. 现有一个小程序，能够求出三个在-10000 到+10000 间整数中的最大者，程序界面如图所示，用等价类划分法设计测试用例。



答案：（1）进行等价类划分，如下：



（2）根据已经划分好的等价类建立等价类表，如下表所示。

条件	有效等价类	编号	无效等价类	编号
输入	整数	1	小数	12
			字符	13
			空白	14

专心---专注---专业

精选优质文档-----倾情为你奉上

	三个有效数	-10000≤a≤10000	2	a<-10000	15
				a>10000	16
		-10000≤b≤10000	3	b<-10000	17
				b>10000	18
		-10000≤c≤10000	4	c<-10000	19
				c>10000	20
输出	最大值是一个数	a 最大	5		
		b 最大	6		
		c 最大	7		
	最大值是两个数	a=b>c	8		
		b=c>a	9		
		a=c>b	10		
	最大值是三个数	a=b=c	11		

（3）根据等价类表设计测试用例，如下表所示。

用例编号	测试用例	覆盖等价类	预期输出
1	（5000，0，－5000）	1、2、3、4、5	a 最大
2	（0，5000，－5000）	1、2、3、4、6	b 最大
3	（0，－5000，5000）	1、2、3、4、7	c 最大
4	（2000，2000，0）	1、2、3、4、8	a、b 最大
5	（0，2000，2000）	1、2、3、4、9	b、c 最大
6	（2000，0，2000）	1、2、3、4、10	a、c 最大
7	（2000，2000，2000）	1、2、3、4、11	a、b、c 最大
8	（2.6， 5.5， 8）	12	输入有小数，错误
9	（三，3，3）	13	输入有字符，错误
10	（3， ，6）	14	输入有空白，错误
11	（－20000，10，100）	15	数字 a 超出范围
12	（20000，10，100）	16	数字 a 超出范围
13	（10，－20000，100）	17	数字 b 超出范围
14	（10，20000，100）	18	数字 b 超出范围
15	（10，100，－20000）	19	数字 c 超出范围
16	（10，100，20000）	20	数字 c 超出范围

黑盒测试总结

- 黑盒测试方法：
  - 边界值分析
  - 等价类测试
  - 基于判定表的测试
  - 因果图测试法
  - 场景测试法

- 黑盒测试的优缺点

- 判定表的优点
  - 能把复杂的问题按各种可能的情况一一列举出来
  - 简明而易于理解
  - 可避免遗漏

判定表的缺点

- 不能表达重复执行的动作，例如循环结构
- 判定表不能很好地伸缩
  - 有n个条件的决策表有2^n个规则。

解决方法

- 使用扩展条目判定表、代数简化表，将大表“分解”为小表，查找条件项的判定表式

课后练习

- 某公司为本科以上学历的人重新分配工作，分配原则如下：1.如果年龄不满20岁，学历是本科，男性要求报考研究生，女性担任行政工作；2.如果年龄满20岁不满50岁，学历本科，不分男女，任中层领导职务，学历是硕士不分男女，任课题组组长，3.如果年龄满50岁，学历本科，男性任科研人员，女性则担任资料员，学历是硕士不分男女，任课题组组长。



问题与讨论