复习：进制和进制转换

1. 进制基础
2. 系数：

0-9 （十）

0,1（二）

0-7（八）

0-9，A-F（十六）

1. 进位规则

逢基数进1

基数：几进制基数就是几

1. 权：基数的次幂

规律：右侧第一位是基数的0次幂，每向左移动一位次幂数+1

1. 进制的表示

方式1：下角标

方式2：后缀B(2),O(8),H(16),D(10)

二、进制的转换

1、十进制🡪任意进制

方法：除基取余逆读法

1. 任意进制🡪十进制

方法：按权展开求和

3、二进制🡪十六进制（八）

方法：4（3）合1

4、十六进制（八）🡪二进制

方法：1分4（3）

5、八进制与十六进制之间的相互转换

在中间加入二进制（十进制）完成转换。

八进制🡪二进制🡪十六进制

第三讲 缺陷报告

1. 软件项目的测试流程：（常见面试题）
2. 熟悉需求，分析需求，整理业务。
3. 制定《测试计划》

一般是由测试组长或测试经理制定《测试计划》

测试人员要阅读和按要求执行测试计划

1. 设计测试（分析，编写《测试用例》）
2. 执行测试，记录测试结果
3. 记录缺陷，跟踪和管理缺陷（《缺陷报告》）
4. 测试总结（《测试总结报告》）

主要就是对测试数据的统计和分析

1. 缺陷报告
2. 什么是缺陷报告？

发现缺陷，通过缺陷报告记录缺陷—>将缺陷告知给开发方，并对缺陷进行跟踪和管理，是测试人员和开发人员之间重要的沟通方式。

1. 缺陷报告如何编写？

1）在企业中缺陷通常使用测试管理工具或者缺陷管理工具进行管理，例如：禅道，qc，bugzilla等

说明：不同的缺陷管理工具中的缺陷组成部门模板会有差异，但是主要组成部分大同小异。

2)缺陷报告的主要组成：

案例：除数为0时，程序异常退出

（1）缺陷编号（defect id）

（a）发现bug的顺序号

（b）缺陷编号应该记录整个项目的bug编号，而不是个人的。

（c）在测试工具中，缺陷编号一般是自动生成的。

（2）缺陷的标题（summary）

用简明扼要的语言，将缺陷概括出来。

（3）发现者（detected by）

就是测试人员自己（账号）

（4）提交缺陷的日期（detected on date）

注意：测试人员要及时提交缺陷

在测试管理工具中，常常会自动将系统日期填写在该位置

（5）缺陷指派给谁处理（assigned to）

首次指派：测试人员🡪开发方的负责人（开发经理、项目经理等）

接下来：开发方负责人再将bug指派给具体的开发人员

总结：测试人员🡪开发方负责人🡪具体的开发人员

（6）功能模块（subject）

在哪个功能中发现的该缺陷

（7）版本（detected in release/version）

说明：在测试中版本不仅包含产品发布的最终版本，还包含在研发过程中，出现的很多临时版本（物资管理系统，5个临时版本）

扩展：

回归测试：在测试当前版本时，将上一个版本测过的功能重新测试一遍。

回归测试的原因（必要性）？

1. 添加的新功能可能会对原有功能产生影响，出现bug
2. 修复的bug，有可能会带来新的问题。

补充：由于回归测试中存在测试执行的重复，所以在满足条件的前提下，很多企业会采取自动化的方式来完成回归测试。

(8)状态（status）

表示缺陷处于什么情况。

New—新的缺陷

Open—以激活的缺陷

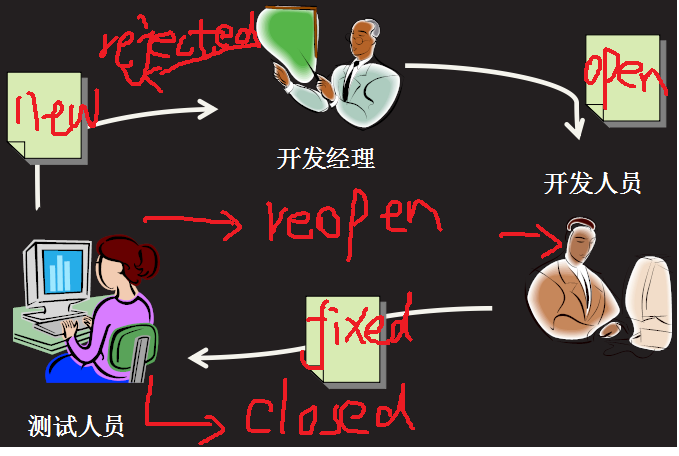
Fixed—已修复（待返测的bug）

Closed—关闭（结束，bug可以归档）

Rejected—被拒绝的bug

Reopen—重新激活的bug

常见的面试题：缺陷报告的跟踪处理过程（步骤，生命周期）？



步骤1：测试人员将《缺陷报告》提交给开发经理，此时缺陷报告的状态是：new（新的缺陷）

步骤2：开发经理要验证提交的缺陷：

情况1：如果验证是缺陷，开发经理将激活（open）缺陷，并将缺陷指派给相应的开发人员

情况2：如果验证不是缺陷，开发经理将拒绝（rejected）该缺陷。（缺陷被拒绝后，测试人员要自检确认，或者与产品经理、开发人员或测试组长等沟通，讨论等确定是否是缺陷，如果是缺陷，谁拒绝的谁负责激活bug，如果不是缺陷，测试组长或测试人员关闭bug ）

步骤3：开发人员负责修复指派给他的缺陷，修复好后将缺陷状态设置为：fixed（已修复）

步骤4：测试人员返测已修复的缺陷，

情况1：返测通过，测试人员将缺陷关闭（closed）

情况2：返测未通过，测试人员将缺陷重新激活（reopen），开发人员继续修复缺陷，直到缺陷通过返测，关闭为止

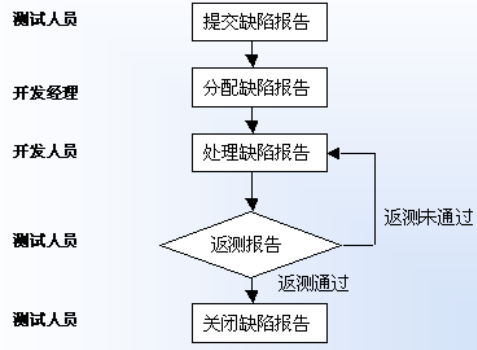
问题：

Q1：常规的bug处理过程

New🡪open🡪fixed🡪closed

Q2：带有返测失败的bug处理过程（返测失败1次）

New🡪open🡪fixed🡪reopen🡪fixed🡪closed



Q3：被拒绝的bug处理流程

情况1：是bug

New 🡪rejected🡪open🡪fixed🡪closed

情况2：假bug

New🡪rejected🡪closed

9)缺陷的严重程度（severity）

表示缺陷有多糟糕，影响有多坏

* 严重级别划分：

致命的—urgent

非常严重（有可能没有该级别） --very high

严重的—high

一般的（中等）--medium( 比较常见)

建议性的小问题—low

重要提示：严重级别的定义比较笼统，在实际工作中容易引发争议，公司为了解决该问题，在实际项目中制定了详细的严重级别的标准，在实际工作中要注意查看，参考。（不同公司，不同项目组，定义可能会有不同）

（10）缺陷的优先级（priority）

希望（建议）开发方在什么时间或什么版本来解决缺陷。

优先级的级别划分：

Urgent—立即修复bug

Very high—本版本内修复（有可能没有该级别）

High—下一个版本修复（比较常见）

Medium—在软件发布（上线）之前修复

Low—尽量在软件发布（上线）之前修复

说明：不同公司，不同项目组，对于优先级的具体定义可能不同。

常见面试问题：

Q1:影响bug优先级制定的因素？

1. 缺陷的严重程度，一般bug越严重，优先级越高
2. 缺陷影响的范围，一般影响的范围越大，优先级越高
3. 开发人员的开发压力，开发压力越小，优先级越高
4. 解决缺陷的成本（时间），解决缺陷的成本越低，优先级越高。

Q2：缺陷的严重级别和优先级一定是严格的正比关系吗？

缺陷的严重级别和优先级不一定是严格成正比关系，有时候严重级别低，但是优先级却高。例如：界面文字拼写错误，严重级别低，但是优先级高。

Q3：缺陷的严重级别和优先级确定后，还能修改吗？

严重级别一般确定后不修改，优先级开发方常会修改，而且常常是向后拖延（delay）

Q4：发布的软件中，是否可能存在发现但是没有解决的bug？

有可能存在发现但是没有解决的bug，此类缺陷需要专门的bug讨论会，权衡解决bug的成本，和不解决是否会对用户造成严重影响或法律纠纷，才能最终决定。而且这类缺陷在软件发布后软件企业要通过升级版本或者打补丁的方式解决。

（11）缺陷描述（description）

将发现缺陷的过程、数据记录下来，让开发人员可以重现该缺陷（要让开人员能够明白）