**讨论一 代码评审，变异测试**

**前提：各组设定一个程序（函数），需要多个函数调用；【简单的程序会拉低总体分数（参考代码评审的速度一般为每小时125行代码，本次讨论课评审过程每组对20分钟左右），此次讨论总分15分】**

**讨论主题：**代码审查(静态分析)；

**目的：**培养学生熟悉代码审查总体流程，让其亲身体验代码审查过程的各项实践；

**要求：**

**（1）静态分析该程序，熟悉静态分析（代码审查）的流程。**

AB组配对：

A组在设定的程序代码中设置不少于6个bug，原则上A组每位同学负责1个或2个，事先需要读懂程序，并提供低层次的设计规范（单元的详细设计）或其他对于程序需求的合适描述；A组成员（记录人员）也需要记录B组成员提问及A组回答的详细问答记录；

B组成员对该程序进行代码审查检测，严格执行代码审查程序（**预备-准备-检查-重做-验证-退出**），讨论报告上详细写明步骤及记录，说明各个小组成员的分工；

**（2）AB两组双方针对程序提出变异测试:** 各组给出2条以上变异测试示例，计算变异分数；

**课上：演示审查程序，变异测试；**

**课后：**每组提交一份讨论报告（代码评审：A组所选程序，所设置的bug，需求文档和设计文档；B组详细代码审查步骤及记录，说明各个小组成员的分工，**要求“参考1：代码评审步骤”中标红色字部分讨论报告中必须体现**。变异测试：每组四条以上变异测试，计算变异分数）

提交电子版及打印版各一份（A4纸打印）。

**参考1：代码评审步骤**

**1）预备**

①单元的作者确保被测单元已经可以进行检查**（源代码）**。**（A组提供不少于6个bug的程序代码，****需求文档和设计文档）**

确保的标准：

a.所有与单元检查有关的代码均是可用/完成的；

b.最小功能性：代码需要能够通过编译和链接，必须已经经过了一定程度的测试以保证它能完成基本的功能；

c.可读性：代码规范；

d.复杂性：指代码中的条件语句的数量，单元中数据元素的数量、单元生成的输出数据元素的数量、代码的实时过程、代码与其单元交互的数量；

e.**需求文档和设计文档**：低层次的设计规范或其他对于程序需求的合适描述，其最后被批准的版本应该时可用的，这些文档帮助检查人员验证待检查的代码是否实现了预期的功能。

②构造检查小组（角色分工）：**（请AB组标明小组成员所承担的角色）**

主持人：负责检查会议的主持工作（项目无关人员，保持客观性）；

作者：待检查的代码的编写人员；

展示人员：预先阅读并理解代码的非工作人员，或作者本身；

记录人员：记录问题和建议采取的行动；

检察人员：代码评审领域的专家小组。3-7人构成；

观察人员：想要学习待检查的代码，不参与检查过程。

③**提前两三天通知小组成员，提供待检查工作包副本以便小组提前阅读**。

**2）准备**

会议开始之前，检查人员应做到：**列出问题，**想要向作者或其他组员提出的问题（参考文后表3.2）；**提出****潜在变更请求（CR），**即错误报告，但不会被包含在与产品相关的错误统计之中；**建议改进的机会**，建议如何修复被检查代码中的问题。（**B组列出问题、提出潜在变更请求和建议改进的机会**）

**3）检查**

a.作者讲解代码使用的逻辑，主要计算的路径，以及被检查单元与其他单元的依赖关系；

b.展示人员逐行阅读代码，此时检查人员可以提出问题即修复错误的大概建议，由作者在会后决定采取何种修正措施；

c**.记录人员记录变更请求和修复问题的建议**，其中**变更请求需包含如下细节**：

**简要描述问题和动作项；**

**为该变更请求赋予一个优先级（主要或次要）**

**指定一个人员（提出该变更请求的检查人员）跟进这个问题**

**设定一个解决该变更请求的最后期限**

d.主持人确保会议始终围绕检查过程以实现会议的目标。

e.会议末尾，**决定是否需要进行另一次会议继续检查代码**。如无新会议，则进入步骤4重做。

**4）重做**

a.记录人员生成**会议总结**，包含如下信息：**一个所有变更请求的列表，包括将被修复的日期，以及负责验证这些变更请求的人员名单；一个改进机会列表；**

b.分发总结副本给所有小组成员;

c.会议结束后，**作者争取在指定时间内解决变更请求，记录对代码的改进**;

**5）验证**

由主持人或者之前指定人员独立进行验证。过程包括**检查变更请求中记录的修改后的代码，确保建议的改进已被正确实现**。

**6）退出**

总结检查过程，并执行下述步骤：

a.确保单元中的每行代码都被检查了。

b.如果错误过多（超过5%的代码行数），则在作者修正错误之后需要二次检查。

c.作者和检查人员达成一致。

d.所有的变更请求都已存档，并由主持人或其他人

员验证。

e.分发会议总结给所有检查小组人员。

**参考2：变异测试**

**（1）变异测试的概念**

变异测试是一种考察测试数据（测试用例）完备性的技术，是衡量测试用例质量的一个手段（附带实现更多的单元测试），通常用于传统单元测试技术的补充。

**程序的一次变异：对程序源代码一次单一、微小、符合语法的更改。**修改后的程序称为一个变异体。当测试用例执行引起变异体失败时，则称为变异被杀死(killed)，该变异体称为“死亡了”。如果一个变异体总是产生和原始程序相同的输出，则称它与原始程序等价，反之则为不等价。如果现有测试用例没有杀死一个变异体，则称这个变异体为“可杀死的”或“顽强的”。

**（2）变异测试的一般步骤**

1）测试套件是否能区分原始程序和变异体决定了它的充分程度。一个测试套件可能无法杀死所有的非等价变异体。

2）向测试套件中添加新的测试用例，从而杀死这些顽强变异体。直到达到期望的变异分数为止。（迭代增强测试套件的测试能力）。

**（3）使用变异测试获取健壮测试套件的步骤**

步骤1：为程序P建议一组正确的测试用例集合T;

步骤2：在程序P上执行T中的每一个测试用例，如有失败，则修改程序，

再执行测试用例，直到程序不出现失败为止;

步骤3：生成一组变异体{Pi}。

步骤4：在每个Pi上执行T中的测试用例。则会出现如下情况：

a.Pi被杀死（现有测试用例是足够的，但不确定）；

b.Pi没有被杀死，则要么Pi与P是等价的，这时不需做任何处理；要么Pi是顽强的，这时可能需要添加新的测试用例。

步骤5：**计算T的变异分数，=100\*D/(N-E)，其中D是被杀死的变异体，N是变异体的总数，E是等价的变异体；**

步骤6：如果T的变异分数不够高，则需要设计新的测试用例添加到T中，回到步骤2，重复直到变异分数超过某个适当的阈值。

