软件工程 (2025 秋) 期末测试

Lecturer: 张天、李宣东、王豫、王林章; Time: 2025/12/26 14:00-16:30; 开卷;

一、简答题 (25 分)

斜体标注的部分需要检查。

1. (5 分) 螺旋模型的*设计目标*是什么？和其它过程模型的区别？螺旋模型的四个象限？
2. (5 分) 什么是功能需求和非功能需求？以一个图书管理系统为例，举出一个功能需求和非功能需求。
3. (5 分) 敏捷开发和 DevOps 的目的是什么？二者之间有什么区别？*DevOps 的哪些特性可以用于敏捷开发？*
4. (5 分) 什么是内聚？什么是耦合？内聚有什么类型（举出两种即可）？你举出的两种内聚哪个更好？为什么？耦合有什么类型（举出两种即可）？你举出的两种耦合哪个更好？为什么？
5. (5 分) 人工审查是什么？人工审查在软件设计过程中为什么很重要？

二、设计题 (45 分)

1. 现设计一个购买车票的系统 v1.0，支持投币支付和扫码支付，过程如下：

- 通电后，系统进入 “等待 (Idle)” 状态。
- 用户点击 “购票” 按钮后，系统进入 “选择 (Selecting)” 状态，允许用户选择乘车目的地。
- 用户选择完目的地后，系统进入 “准备支付 (Paying)” 状态，计算车票价格。
- 当用户投入的金额大于等于票价时，系统进入 “吐票 (Dispensing)” 状态，打印车票并返还找零。
- 若用户在 “选择模式”、“准备支付” 状态中选择 “取消”，则回到 “等待模式”。
- 车票打印完成后，立刻回到 “等待模式”。
- 在任何状态下，如果系统检测到硬件故障，立刻进入 “停止运营 (Out of Service)” 状态并作红灯警告，须等待工作人员执行 “重置” 操作才能回到 “等待” 状态。

(1) 画出 v1.0 系统的状态图。

现在对这个系统进行如下升级：

- 用户选择完目的地后，先进入 “账单确认” 状态：
  - 此时，若用户选择 “确认”，系统进入 “准备支付 (Paying)” 状态。
  - 否则，若用户选择 ”修改“，系统重回 “选择 (Selecting)” 状态（保留用户选择）。
  - 否则，若用户选择 ”取消“，系统进入 “等待 (Idle)” 状态。
- 若用户选择扫码支付，那么系统先进入 ”正在处理“ 状态：
  - 如果可以正常支付，进入 ”吐票“ 状态。
  - 否则（例如余额不够等情况），返回 “准备支付 (Paying)” 状态。

(2) 画出 v2.0 系统的状态图，并使用文字或者注释醒目标记出新添加的转移等。

2. 某家超市有会员和消费点优惠活动，所有可能的优惠倍率是 1, 0.9, 0.85, 0.8, 0.7。已知：当且仅当顾客的消费点不足 100 时，顾客无法享受优惠。如下代码展示了这家超市优惠活动的逻辑：

```
float get_rate(int points, bool isVip) {
    float rate = 1.0;
    if(points >= 100) {
        if(points < 500) {
            if(isVip) {
                rate = 0.85;
            } else {
                rate = 0.9;
            }
        } else {
            if(isVip) {
                rate = 0.7;
            }
        }
    }

    return rate;
}
```

(1) 画出这个代码的控制流图。这份代码的圈复杂度是多少?

(2) 这份代码有什么错误?这错误可能会产生什么影响?

(3) 运用基本路径测试法，设计一组最小的测试数据集。每一组测试数据应包括 `<输入: points, isVip>` 和 `<期望输出: rate>`。

3. 如下代码能够在 n 满足一定数量条件且未被处理过时，计算 n 的约数和。但是这份代码的可读性较低:

```
int main() {
    int n;
    std::vector<int> processed;

    ...

    if(n > 1 && n % 3 == 0 && std::find(processed.begin(), processed.end(), n) ==
processed.end()) {
        int sum = 0;
        for(int i = 1; i <= n; i++) {
            if(n % i == 0) {
                sum += i;
            }
        }
        std::cout << "n = " << n << std::endl;
        std::cout << "sum of divisors = " << sum << std::endl;
        processed.push_back(n);
    }

    ...
}
```

(1) 在这段代码的合适处添加注释，使得这份代码更加清晰易懂。

(2) 这份代码的结构让人不易理解其整体逻辑。重构这份代码，使其清晰易懂。

三、开放题 (30 分)

1. (20 分) 假设你现在是一个金融软件组的成员，希望使用 LLM 生成代码。

(1) 简述 LLM 生成的代码会有怎样的质量问题?

(2) 为了解决这些质量问题，有什么应对策略?

(3) 简述 LLM 会对软件项目管理产生什么新的风险?

(4) 参考常见的开发过程模型，画出人机协同模式下，软件开发的过程模型。

2. (10 分) 以一个具体的软件工程知识点为素材，设计一道简答题。这道简答题应当体现你对这个知识点的理解，而不应该停留在概念层面。给出完整的试题描述和参考答案，并标注这个知识点在书上的位置。