Large-Scale Refactoring Kata

Week 10: Practical Session

Advanced Software Engineering

Learning Objectives

- Apply large-scale refactoring strategies
- Manage technical debt in practice
- Ensure business continuity during change

Kata Instructions

Work in pairs to refactor the following codebase excerpt using large-scale strategies.

C++ Starting Point (Excerpt)

```cpp
class CustomerManager {
public:
    double calculateDiscount(const Customer& customer) {
        if (customer.getType() == "VIP") {
            return 0.2;
        } else if (customer.getType() == "Regular") {
            return 0.1;
        } else {
            return 0.0;
        }
    }
    // ... many more methods ...
};

class OrderManager {
public:
    double calculateOrderTotal(const Order& order) {
        CustomerManager cm;
        double discount = cm.calculateDiscount(order.getCustomer());
        return order.getSubtotal() * (1.0 - discount);
```

Tasks

1. **Identify Refactoring Targets (20 min)**
   - List code smells and technical debt
   - Prioritize areas for improvement
2. **Apply Large-Scale Patterns (30 min)**
   - Use Strangler Fig or Parallel Change to introduce new logic
   - Ensure old and new code can coexist
3. **Manage Risk (20 min)**
   - Plan for business continuity
   - Write tests to validate both old and new behavior
4. **Review and Discuss (20 min)**
   - Present your approach to another pair
   - Discuss lessons learned and remaining risks

Success Criteria

- Large-scale refactoring applied safely
- Technical debt identified and addressed
- Business continuity maintained
- Lessons and risks documented

Notes for Facilitators

- Encourage incremental, reversible changes
- Focus on communication and planning
- Discuss real-world applications and outcomes

Follow-up

- Review the full curriculum
- Plan for ongoing improvement and learning