UML Class Diagrams

Design Patterns
1. Visitor
2. Chain of Responsibility
3. Adapter/Decorator/Proxy
4. Strategy
5. Template Method
6. Abstract Factory
7. Plugins
8. Command
9. Adapter
10. Factory Method
11. Plugins
12. State Pattern
13. Problem, Abstract Design, Concrete Design, Concrete Application (write the code)

Minimizing Dependency
Dependency Inversion – Problem that solves, benefits, how to write code
Dependency Injection – Problem that it solves, benefits, how to write code for it

Design By Contract
1. Class Semantics
   a. Domain Definition
      1. Class Invariant
   b. Method/Constructor Semantics
      1. Pre-conditions
      2. Post-conditions
2. Inheritance/Specialization Semantics
   a. Mathematical formulation using pre and post-conditions
   b. Mathematical formulation of behavior specialization (not violating superclass’s semantics)
   c. What constraints guarantee b.
      1. Weaken Preconditions written as implications
      2. Strengthen Postcondition written as implications

Inheritance, Composition, and Specialization
1. “has-a”, “uses”, “is-a”
2. How can composition implement specialization
3. What is and how to implement “Dynamic” specialization using composition
4. What is and how to implement multi-valued specialization using composition

OO to relational mapping
Horizontal, Vertical, Unification

Reuse
1. Forces against/Problems
2. Forms of reuse
3. Patterns: Adapter, Decorator, Strategy
Refactoring

1. Bad Smells
   a. Problem/Violated Principles
   b. How do you detect it.
   c. Refactorings that can be used to fix the problem.
   d. How does the bad smell relate to other principles taught in class.