# MPP Midterm Review Points

Location**: V29, Verill Hall** (same classroom)

Duration: **2.5 hrs**

Date: **10/10/2022** (next Monday)

Time: **2 ~ 4:30 pm**

The midterm will covering Lesson 1 to 5. The midterm will consist of approximately 10% short answer questions (including possibly some true/false questions) and 90% skill questions. Skill questions require you to write code or draw diagrams. The exam will be **a paper exam**; you will not have access to laptops, internet, phones, books, or notes. There will be some coding on the exam, so you should be prepared to write code without the help of Eclipse.

1. Be prepared to answer questions about and write code for 1-1 associations (either unidirectional or bidirectional).
2. Be familiar with the rules concerning 1-many associations discussed in class.
3. Know the difference between associations and dependencies, both conceptually and in the way they are implemented in code
4. What is the purpose of a class diagram? What is the purpose of a sequence diagram? Be able to answer these questions.
5. Be familiar with how polymorphism is implemented in Java.
6. Be familiar with the implementation of late binding in Java code contrasted with early binding when static methods are used.
7. Be familiar with applications of factory methods and object creation factory to replace the use of constructors, as discussed in Lesson 5.
8. Be able to determine when properties of a UML class should be modeled as *attributes* and when they should instead be modeled by *associations.*
9. Given a problem statement, be able to draw a class diagram and be able to draw a sequence diagram for a main flow.
10. Be able to translate class diagrams and sequence diagrams into Java code.

**The SCI/STC Question**: (3 points) You will be given an insight/principle from SCI and you will be asked to explain what it means and give an example of how it is exemplified or illustrated by a Computer Science concept. This is a short essay; richer content will be awarded more credit.

# Review Recommendations:

**Lessons 1 and 2**

1. Discovering classes from a problem statement
2. Differences between association, dependency, inheritance
3. Associations, dependencies in code
4. Difference between one-way, two-way associations
5. Properties as attributes, properties as associations
6. Association adornments: name, role, multiplicity,
7. Significance of different multiplicities in code
8. Reflexive associations

*Skills:*

- Create a class diagram with attributes, operations, associations, based on a problem statement
- Translate a class diagram into Java code

**Lesson 3**

9. Inheritance rules
10. IS-A and LS principles (subclass must be able to be substituted for the superclass)
11. Benefits of inheritance
12. Principle: Design for inheritance or prevent it – class final or private constructor
13. How to replace inheritance by composition, or supplement inheritance with composition in a design and in code

*Skills*

- Solve a problem using inheritance and polymorphism. (lab3 and Practice 1 - 3)

**Lesson 4**

14. Syntax of sequence diagrams – use of activation bars; how to show looping; how to show message passing and self-calls; iteration marker; return arrows

15. Sequence diagrams as a way to model a use case
16. Centralized control vs distributed control in sequence diagrams
17. The meaning of polymorphism and late binding
18. static, private, and final methods have early binding
19. Open-Closed Principle

*Skills*

- Create a sequence diagram based on a use case description.
- Converting Java code to a sequence diagram (see: exercise 4.1 and 4.2 of lecture4)
- See the polymorphism example (Bank) of lecture4

**Lesson 5**

20. Differences between abstract class and interface (pre-Java 8)
21. UML notation for abstract classes and interfaces
22. The Object Creation Factory pattern
23. The simple factory method pattern
24. The "Diamond Problem" for languages with multiple inheritance
25. Benefits of using interfaces

*Skills*

- Be familiar with applications of factory methods and object creation factory to replace the use of constructors, as discussed in Lesson 5 (See lecture code's corresponding examples)

**Questions:**

**Short answer questions – 10 points**

5-7 questions: True/False, Multiple choice, and Short Answer

**Draw diagrams – 20 points**

1. Create a class diagram with attributes, operations, associations, multiplicity based on a given problem statement. [15 pts]
2. Create a sequence diagram for a given use case with your created class diagram and problem statement. [5 pts]

**Translate diagrams into Java code – 35 points**

1. Implement classes, its instance variables, getters, setters, and constructors based on the given problem description, the class diagram and a Main class (it will be provided). [15 pts]
2. With above your implementations, implement instance methods using inheritance and polymorphism according to the given sequence diagram. [20 pts]

**STC – 3 points**

**Total => 68 points**