MPP Midterm Review Points

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 (see Lecture code of Lecture 2).
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. Know the elements of a Use Case Diagram.
8. Be familiar with applications of Template methods and Factory methods.
9. Be able to determine when properties of a UML class should be modeled as *attributes* and when they should instead be modeled by *associations.*
10. Given a problem statement, be able to draw a class diagram to model the key abstractions you extract and be able to draw a sequence diagram for a main flow.
11. Be able to draw a little class diagram that models reflexive relationships. Examples from class: Course Prerequisites, Position Hierarchy (see Lab 2, Problem 3).
12. Be able to translate class diagrams and sequence diagrams into Java code

The SCI 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.

Exam Policy:
There is no tolerance policy for exams. **You will be asked to leave the exam room immediately without a warning** once you do the following things which mean you'll get **NC**.

1. You are caught cheating or trying to cheat.
2. All mobile phones should be turned off and submitted along with your luggage at the beginning of the exam.
3. You're not allowed to ask/get extra papers from other students or proctors. All your notes must be written on the exam paper provided. Use the back side if you need to draft.
4. Please write down your answer clearly. If I cannot read your answer, you'll not get credit.
5. Please turn in your answer and question sheets both.

You can ask for clarification of the question but do not ask me to evaluate your answer during the exam.

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

a. Rules for inheriting/overriding static methods

b. Order of execution

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 (Duck App – person/personRole), in a design and in code

Skills

- Solve a problem using inheritance and polymorphism.

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. The reason why static, private, and final methods have early binding
19. Open-Closed Principle

Skills

- Create a sequence diagram based on a use case description.
- Solve a problem using polymorphism.
- Converting Java code to a sequence diagram

Lesson 5

20. Differences between abstract class and interface (in Java 7)
21. UML notation for abstract classes and interfaces
22. The Object Creation Factory pattern (know the diagram and what it means)
23. The simple factory method pattern

24. Know the benefits of these patterns and some examples of when to use them
25. The “Diamond Problem” for languages with multiple inheritance
26. Benefits of using interfaces
27. What is the Evolving API Problem?

Skills

- Create instances using Object Creation Factory pattern

Short answer questions – 10 points (8 questions: True/False, Multiple coice, Explain)

Create a class diagram with attributes, operations, associations, based on a problem statement – 15

Translate a class diagram into Java code – 15

Solve a problem using inheritance and polymorphism – 30

Create a sequence diagram based on a use case description – 10

Converting Java code to a sequence diagram – 10

Implement the Object Creation Factory pattern to create instances for given problem – 10

SCI – 3 => 103

Duration: 2.5 hours

Calculate total salary()=?

Centralize control

In Company:

-> Department,

-> position,

-> employee

Distributed control

In Company:

➔ Department

-> Position

-> Employee