Final Exam Review

CS 4354
Fall 2012
Jill Seaman

Final Exam

• Friday, December 14, 11AM
• Closed book, closed notes, clean desk
• Content:
  ✦ Textbook: Chapters 1, 2, 4-10
  ✦ Java Lectures, GRASP + JUnit
• 35% of your final grade
• I recommend using a pencil (and eraser)
• I will bring extra paper and stapler, in case they are needed.

Exam Format

• 150 points total
  ✦ Multiple choice questions
  ✦ Drawing UML diagrams
  ✦ Writing programs/functions/code
  ✦ Tracing code (what is the output)
  ✦ Short answer (like Assignments 1 and 4)

• Each question will indicate how many points it is worth

Ch 1: Introduction:
Object-oriented analysis, design, implementation

• Object-oriented analysis: finding+describing domain objects
  ✦ concepts
• Object-oriented design: design software objects to fulfill system requirements
  ✦ class diagram
• Object-oriented programming/implementaiton
  ✦ Java classes
Ch 2: Modeling with UML: Modeling concepts

- **System model**: set of all models built during development
- Three models of a software system:
  - **Functional Model**: functionality from users' point of view (use case diagrams)
  - **Object Model**: structure of the system (class diagrams)
  - **Dynamic Model**: behavior of the system (sequence diagrams, state diagrams, activity diagrams)
- **Application domain**: all aspects of customer's "problem"
  - object-oriented analysis: models this domain
- **Solution domain**: modeling space of all possible solutions
  - object-oriented design: models this domain

Ch 2: Modeling with UML: UML diagrams

- **Use Case Diagrams**
  - Actors, relationships: communication, inclusion, extension, inheritance
- **Class Diagrams**
  - Classes, attributes, operations, objects, links/associations
  - unidirectional, bidirectional associations, roles, multiplicity
  - Aggregation, composition, qualification, inheritance
- **Interaction Diagrams**
  - Sequence diagrams (and communication/collaboration diagrams)
- **Activity Diagrams**
  - Activities, control flow, decisions, forks and joins, swimlanes
- **State Machine Diagrams**
  - State is a node, event is a directed edge labeled: Event[Guard] / Action

Java: Introduction

- Compilation, execution (byte code)
- Features
  - Object-oriented, inheritance, polymorphism, garbage collection
  - Exception handling, concurrency, Persistence, platform independence
- Objects are references (pointers)
- Types:
  - Primitive types
  - arrays
  - classes, methods
- Operators, assignment, control flow
  - Similar to C++

Java: Input/Output

- Reading from the keyboard
  - use EasyIn or scanner
- Writing to the screen (formatting)
- Object serialization
  - ObjectInputStream, ObjectOutputStream
  - readObject, writeObject
Java: Inheritance

• Composition
• Inheritance
  ➤ hierarchy, superclass, subclass,
  ➤ overriding methods, upcasting, constructors
• Polymorphism
  ➤ upcasting, extensibility
• Abstract methods and classes
• Interfaces
  ➤ Multiple inheritance
  ➤ Sorting: implementing Comparable
  ➤ Extending an interface

Java: Exceptions and Threads

• Exceptions
  ➤ Semantics (how exceptions are thrown/caught), syntax
  ➤ Catch or specify requirement
  ➤ finally block
  ➤ Runtime exceptions
• Threads
  ➤ Thread class, Runnable interface
  ➤ Using the above to implement multi-threading
  ➤ Thread methods

Ch 4-5: OO Software Development: Requirements elicitation and analysis

• Requirements Elicitation
  ➤ Functional vs Nonfunctional requirements, quality
  ➤ Activities: Identifying actors, scenarios, use cases, relationships
• Analysis Activities (from use cases to objects)
  ➤ Identifying Entity Objects, Boundary Objects, Control Objects
  ➤ Mapping Use Cases to Objects with Sequence Diagrams
  ➤ Identifying Associations, Aggregations, Attributes
  ➤ Modeling Inheritance Relationships
  ➤ Modeling State-Dependent Behavior of Individual Objects
  ➤ Reviewing the Analysis Model

Ch 6: System design: Decomposing the system

• Concepts
  ➤ Subsystems, subsystem interfaces
  ➤ Coupling and cohesion, layers+partitions (no architectural styles!)
• System Design Activities
  ➤ Identifying Design Goals: five groups of criteria:
  - Performance
  - Dependability
  - Cost
  - Maintenance
  - End user criteria.
  ➤ Identifying Subsystems
## Ch 7: System design: Addressing design goals

- Concepts
  - Deployment diagrams
- System Design Activities
  - Mapping subsystems to processors and components
  - Identifying and storing persistent data
  - Providing access control
  - Designing global control flow
  - Identifying boundary conditions
  - Reviewing system design

## Ch 8: Object design: Reusing pattern solutions

- Concepts
  - Specification Inheritance vs Implementation Inheritance, Delegation
- Design Patterns
  - Bridge Pattern
  - Adapter Pattern
  - Strategy Pattern
  - Abstract Factory Pattern
  - Command Pattern
  - Composite Pattern
  - Observer Pattern
  - Proxy Pattern
  - Facade Pattern

## Ch 9: Object design: Specifying Interfaces

- Concepts
  - Class implementor, user, extender (developer roles)
  - Invariants, preconditions, postconditions (contracts)
  - Object Constraint Language (OCL)
- Activities
  - Specifying pre and post-conditions
  - Specifying invariants
  - Inheriting contracts: when you can weaken or strengthen the conditions in the subclasses.

## Ch 10: Mapping models to code

- Concepts
  - Four types of transformations:
  - Model transformations, refactoring, forward engineering, reverse engineering
- Activities
  - Mapping associations to collections
    - Unidirectional one-to-one associations
    - Bidirectional one-to-one associations
    - Bidirectional one-to-many associations
    - Bidirectional many-to-many associations
  - Mapping contracts to exceptions (implementing pre-post-conditions, invariants)
Extra topics

• GRASP
  ✦ Deciding which classes should perform which operations
  ✦ Information Expert, Creator, Low Coupling, High Cohesion, Controller

• JUnit
  ✦ Framework for writing and running unit tests
  ✦ Provides automation
  ✦ Be able to write a simple test case, using assertEquals() or assertTrue(), etc.

Office Hours

<table>
<thead>
<tr>
<th>Day</th>
<th>Date</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>12/10</td>
<td>None (or by appt)</td>
</tr>
<tr>
<td>T</td>
<td>12/11</td>
<td>2:30-4pm</td>
</tr>
<tr>
<td>W</td>
<td>12/12</td>
<td>1-2:30pm</td>
</tr>
<tr>
<td>Th</td>
<td>12/13</td>
<td>1-2:30pm</td>
</tr>
<tr>
<td>F</td>
<td>12/14</td>
<td>None (exams)</td>
</tr>
</tbody>
</table>