Midterm Review

CS 3354
Spring 2017

Jill Seaman

Midterm Exam

- Wednesday, March 8
- In class, closed book, closed notes, clean desk

- 30% of your final grade
- 80 minutes

- Bring your ID card
- Bring a number 2 pencil and eraser
- I will bring extra paper and stapler, in case they are needed.

Exam Format

- 100 points total
  - ✦48 pts: 16 Multiple Choice questions (scantron), may include:
    - Tracing code (what is the output)
    - Reading diagrams (what does it mean)
  - ✦52 pts: Coding and Design documentation:
    - Writing Use Cases, Writing CRC cards
    - Drawing UML diagrams (class, sequence)
    - Writing programs/classes/code in Java
- Each question will indicate how many points it is worth (out of 100)

Content

- Java (Chapter 1 + 4.1-5 + 6.1)
  - ✦Unit 1 Java Basics
  - ✦Unit 2 Java Interfaces, Inheritance, Polymorphism, Collections, and Exceptions
- Object-Oriented Design (Chapter 2)
  - ✦Unit 3 Object-Oriented Design & UML
  - ✦Use Cases, CRC cards, Class diagrams, Sequence Diagrams, State Diagrams

Unit 1: Java Basics

- Compilation, execution (byte code)
- Features
  - ✦Object-oriented, inheritance, polymorphism, garbage collection
  - ✦Exception handling, concurrency, persistence, platform independence
- Primitive types, control flow, operators, assignment (like C++)
- Classes, fields, methods
- Objects are references (pointers underneath)
- Parameter passing (pass by value, but objects can be mutated)
- Packages, directories, import statement
- Java library, API

Unit 1: Java Basics

- String, toString, substring, equals
- ArrayList
- arrays

- Javadoc, how to document the elements of a program
- Access specifiers: public, private, protected, [package]

Unit 1: Java Basics

- Input using a Scanner
- Output using System.out.println()
- Wrapper classes (Integer, Float, Double, etc)
- Formatting using the DecimalFormatter and/or String.format

- Object serialization
  - ✦ObjectInputStream, ObjectOutputStream
  - ✦readObject, writeObject
  - ✦Understand how it works (when to cast)
  - ✦Don't memorize the exceptions

Unit 2: Java Inheritance/Polymorphism/Interface

- Interfaces
  - ✦Using, Defining and implementing Interfaces
  - ✦Sorting: implementing Comparable<T> or Comparator<T>
- Inheritance
  - ✦class hierarchy: superclass, subclass, (extends keyword)
  - ✦overriding methods
  - ✦constructors
- Polymorphism
  - ✦upcasting, polymorphic functions, dynamic binding
- Abstract methods and classes

Unit 2: Java Collections and Exceptions

- Collections
  - ✦LinkedList<T>
  - ✦Iterator<T> (next(), hasNext(), remove())
  - ✦iterator() method
- Exceptions
  - ✦Semantics (how exceptions are thrown/caught) and syntax
  - ✦Catch or specify requirement (how to satisfy)
  - ✦Runtime exceptions (unchecked)
  - ✦Create your own exception classes (and instances)

Unit 3 (Ch 2): The Object-Oriented Design Process

- Analysis, Design, Implementation
- Objects and Classes
- Identifying Classes and Responsibilities
- Identifying Relationships
  - ✦Dependency
  - ✦Aggregation
  - ✦Inheritance
- Use Cases
  - ✦Actor
  - ✦textual descriptions (set of steps), with variations
  - ✦single interaction between actor and system.

Unit 3 (Ch 2): The Object-Oriented Design Process

- CRC cards
  - ✦Classes, Responsibilities, Collaborators
  - ✦Index cards describing each class
  - ✦Walkthrough use cases to generate/develop the cards
- Class Diagrams
  - ✦Classes, attributes, operations, relationships
  - ✦unidirectional, bidirectional relationships
  - ✦Dependency, Aggregation (or association), Inheritance, Interface type implementation
  - ✦Multiplicity {1, 0..1, 0..n, 1..n}

Unit 3 (Ch 2): The Object-Oriented Design Process

- Sequence Diagrams
  - ✦Describes interactions between objects
  - ✦Objects, lifelines, activation boxes
  - ✦Method calls from one object to another (must be methods defined on the receiving object), method calls run in sequence top to bottom.
- State Machine Diagrams
  - ✦States an object can go through in response to external events,
  - ✦State is a node
  - ✦Transition is a directed edge labeled with the event that causes it
- For all of the types of diagrams:
  - ✦Be able to draw simple diagrams, class + sequence, like in A3.
  - ✦Be able to read (understand, interpret) diagrams.

Sample Questions: Multiple Choice

- You want to draw a diagram to show the different screens that your mobile phone app can display, including the events and conditions that cause the app to move from one screen to another. What kind of UML diagram is best for that?

  (a) Use Case diagram

  (b) Class diagram

  (c) Activity diagram

  (d) Sequence diagram

  (e) State machine diagram

- _____ is an implicit type conversion performed in Java that permits an object of a subclass type to be used where an object of its superclass type is expected.

  (a) Upcasting

  (b) Polymorphism

  (c) Dynamic binding

  (d) Extensibility

Sample Questions: UML Diagrams

- In a given company, each department has a name and owns at least one employee. Each employee has a name, street address, city, state, zip, and an id number. The employees may be full time, in which case they have an annual salary, or they may be part time, in which case they have an hourly pay rate. Each employee can be in only one department.

- Draw a **class diagram** to represent the above scenario. Include relationships and multiplicity. (Attributes are optional, but you will use them in the next question).

Sample Questions: Java programming

Implement the Company system from the question above using classes in Java.

- The Employee class should have a polymorphic function called weeklyPay. For Full time employees, their weekly pay is their salary divided by 52. For part time employees their salary is their hourly pay rate time 40.
- Add a method called totalPay to the Department class that will return the amount of total weekly pay owed to all of the Employees currently assigned to the Department (sum up the weeklyPay for each Employee).
- Use an abstract class and polymorphic functions to implement your solution!
- Use public and private appropriately!