Winter 2015
Mon / Wed 8:00PM

Computer Information Science 28
Object Oriented Analysis and Design

Ron Kleinman
kleinmanronald@fhda.edu

1: Course Description

This course defines and illustrates the Object Oriented paradigm for analyzing, designing and implementing computer applications. Topics include techniques for classifying objects in terms of both attributes and behavior, and designing systems of interrelated objects leveraging techniques such as inheritance, polymorphism, encapsulation and the reuse of both interface and implementation.

Prerequisites: Prior programming experience in an object oriented language
Office Hour: Monday 7:00-8:00PM, Computer Lab

2: Course Goals

To obtain a proficiency in the Object Oriented approach which will enable the student to examine a wide range of system requirements and:

- Use Object Oriented Analysis (OOA) methodology to identify and "flesh out" major abstractions within the problem space.
- Use Object Oriented Design methodology to refine these abstractions into concrete classes, and determine the relationships between these classes and the operations they support. Several key design patterns will be introduced as building blocks for this effort.
- Understand and use various Universal Modeling Language (UML) diagrams including Class Lists, State Transition diagrams, Class Relationship diagrams, Sequence Flows, Class Responsibility and Collaboration cards, and Use Case diagrams.

Trade-offs between various OO techniques will be illustrated with a series of real world applications to allow the student to optimize his / her solutions for robustness and reuse.

3: Schedule

Three and one half hours of classroom lecture each week, plus forty five minutes of lab. Plan to spend **at least** an additional five to six hours each week in outside study. Of necessity, this effort will be somewhat concentrated in the second half of the quarter.

Lectures will be used primarily to introduce new OO concepts. The homework will be in the practical application of the classroom material.

4: Materials Required and Recommended

- Various online references as noted during class lectures
- The official course notes for CIS 28, also avialable in the De Anza Bookstore. Purchase of these notes is required, as all of the lectures will make reference to them.

5: Attendance

This is not a self-study course. The necessary material will be introduced through classroom lecture. Regular and prompt attendance is therefore essential for a good learning experience.

If you reach a total of three (3) unexcused absences, and are behind in your work, you **may** be dropped from the section (but do not count on that). Homework is due on announced dates and late work will not receive full credit.

6: Academic Integrity

Homework assignments will be considered as individual efforts. If it is determined that outside collaboration occurred between two or more turned in assignments, all will be assigned a grade of zero.

Both the midterm and final are open book. As the best object analysis and design is often a team effort, joint outside study and discussion before these tests is encouraged, but all such collaborative efforts stop at the door of the testing room. A grade of zero will be assigned to any student discovered cheating on an exam – no exceptions. In addition, I must also reserve the right to ask for the ID of anyone taking either the midterm or the final exam to confirm their registration in this section.

7: Grading

Both the midterm and the final examination will be graded on a 100 point scale. A minimum passing grade is 60%. Your grade for this section will be computed by weighing the following factors in roughly the ratio shown:

A: Midterm: 30%
B: Homeworks: 30%
C: Final Examination: 40%

DE ANZA COLLEGE
BUSINESS/COMPUTER SYSTEMS DIVISION
COURSE OUTLINE

Degree Applicable

COMPUTER INFORMATION SYSTEMS 28 Effective Quarter Winter 2015

I. Catalog Information

CIS 28 Object Oriented Analysis and Design 5 Units

Prerequisites: OO Programming experience

Four and one half hours lecture and laboratory

This course defines and demonstrates the Object Oriented paradigm for analyzing, designing and implementing computer applications. C++ will be the "reference" language used to illustrate basic OO concepts although Java examples will also be given.

Trade-offs between various OO techniques will be explored with a series of real world applications to allow the student to optimize his / her solutions for robustness and reuse.

II. Course Objectives

The student will obtain a proficiency in the Object Oriented approach which will include the ability to:

- Use Object Oriented Analysis (OOA) methodology to identify and "flesh out" major abstractions within the problem space.
- Use Object Oriented Design methodology to refine these abstractions into concrete classes, and determine the relationships between these classes and the operations they support. Several key design patterns will be introduced as building blocks for this effort.
- Understand and use various Universal Modeling Language (UML) diagrams including Class Lists, State Transition diagrams, Class Relationship diagrams, Sequence Flows, Class Responsibility and Collaboration cards, and Use Case diagrams.

III. Student Materials

Course Notes - required
UML references: Recommended but not required
http://www.uml-diagrams.org/ (a lot more diagrams than we'll use)
http://www.agilemodeling.com/artifacts/ (diagrams and examples)
http://www.wikipedia.org/ (for the terminology definitions)

IV. Essential College Facilities

None

V .Expanded Description: Content and Form

A. Introduction
    1: What is analysis, what is design and what is an object?
    2: Why OOP isn't enough
    3: Basic concepts of object oriented methodology
        a. encapsulation
        b. abstraction
        c. inheritance
        d. reusability
        e. polymorphism

B. Fundamentals of object oriented analysis and design
    1. Setting design objectives
        a. portability
        b. reliability
        c. maintainability
        d. efficiency
    2. Notations for objects
    3. Client / Server Model
    4. Where analysis stops and design begins

C. Object Oriented Analysis
    1. Use Cases
    2. Object Classification
    3. Object Granularity
    4. Separation of Interface from implementation
    5. Object Identity
    6. Object Attributes
    7. Class Hierarchy Diagrams

D. Object methods
    1. Primary methods (Constructor, Destructor, Copy)
    2. Secondary Methods (get / put / identify)
    3. Worker Method design
    4. Dynamic object behavior and the Event / State Diagram
    5. Object Persistence

E. Object Oriented Design
    1. Object Relationships
        a. Inherits (single, multiple and mix-in)
        b. Uses
        c. Has
        d. Associates
    2. Polymorphism, dynamic binding and overloaded operators
    3. Collection Classes: Types and Tradeoffs
        a. Stacks (LIFO, FIFO)
        b. Tables (Sets, Bags)
        c. Linked Lists (Single, Double) and Iterators
    4. Object Service Invocation alternatives (the client / server connection)
        a. Local (object library)
        b. Synchronous Remote Procedure Call
        c. Asynchronous XML Document exchange
    5. Object Design Patterns & Design Reusability
        a. Purpose
        b. Examples (Factory, Stateless, Publisher,)

F. Putting it all together
    1. Class Responsibility Collaboration Cards (CRCs)
    2. Sequence Flow Diagrams
    3. Addressing Security Concerns (Authentication and Authorization)
    4. Multi-threading and idempotent interfaces
    5. More on object persistence
    6. Aspect Programming (cross object “snippets”)
    7. Method Signatures cross-function dependencies and ... (finally) code

VI. Assignments

A. Object oriented analysis and design of complex systems, and production of relevant UML diagrams to document the work.
B. Object oriented programming (in Java, C# or C++) for extra credit in at least one of these assignments

VII. Methods of Evaluating Objectives

A. Successful completion of assigned problems
B. A written midterm exam
C. A comprehensive, written final exam