CSC/CPE 203: Project-based Object-Oriented Programming and Design

Professor: Bill Foote
Lecture: MWF 3:10-4 room 186-C302; lab 4:10-5 room 014-0302
Office: 14-240
Department Phone: 805-756-2824
My Cell Phone: 310 980 1044
Office Hours: TR 3-4:30, WF 2-3
Email: wffoote@calpoly.edu

Course Description: Object-oriented programming and design with applications to project construction. Introduction to class design, interfaces, inheritance, generics, exceptions, streams, and testing. 3 lectures, 1 laboratory. Prerequisite: CPE/CSC 202 with a grade of C- or better or consent of instructor.

Course Objectives: By the end of the quarter students will be able to:

- Explain key object-oriented concepts including: classes, objects, methods, instantiation invocation, interaction between objects, composition, encapsulation, and use of class libraries.
- Use object-oriented concepts and design to implement moderately sophisticated "large" programs
- Describe the philosophy and mechanics of interfaces including abstraction and specification independence of specification and implementation, contractual requirements in interface implementation, subtypes and type casting, polymorphism, interface hierarchies, and implementation of multiple interfaces.
- Define interfaces in programs to support abstractions according to the principles of interface segregation and dependency inversion where design weaknesses are identified.
- Create a unit test plan for a set of methods in a class.
- Implement program pieces (classes and methods) that use generic types.
- Discuss the differences among generics, subtyping, and overloading.
- Describe the philosophy and mechanics of inheritance including generalization, specialization via extension, difference between extension and composition, subtypes and type casting, polymorphism, and inheritance hierarchies.
- Use inheritance in the implementation of program components such as in the application of a refactoring process to an existing software implementation to improve some aspect of its design.
- Explain the relationship between object-oriented inheritance (code-sharing and overriding) and subtyping
- Articulate design principles including separation of concerns, information hiding, coupling and cohesion, and encapsulation.

- Apply class design principles to the development of a design document for a moderately sophisticated "large" program.

(Some of the) Computer Science and University learning objectives

- Ability to apply knowledge of computing and mathematics
- Ability to analyze a problem and identify and define the computing requirements appropriate for its solution
- An ability to communicate effectively with a range of audiences
- An ability to use current techniques, skills, and tools necessary for computing practice
- Recognition of the need for, and an ability to engage in continuing professional development

Assignments/Grade breakdown:

- 1 final exam (30% of final grade)
- 2 mid-term exams (16% of final grade)
- ~10 Lab exercises (10% of final grade) (1-2% each)
- 5 stages of one larger projects (5-10% each of final grade, 40% total)
- Participation (4% of final grade)
  - In class group exercises
  - attend class/ talk in class or office hours interaction (I have the right to fail you if you miss more than 3 classes) / class questionnaires
- Please see each program description for final grading/rubric details. **There is a strict late policy for all assignments** – no late projects are allowed.

Recommended Text: Core Java – Volume 1 - Fundamentals" by Cay Horstmann – available as an ebook for online reading at https://bit.ly/2q1DlXP.

Class style and logistics

I expect you to participate in class and engage with the class material. Studies suggest that taking longhand notes is one of the better ways to guarantee your engagement with the material in class)[1] I also expect you to form a community of scholars for the duration of the quarter (and hopefully longer). My teaching style is very interactive – if you want to know more about why see ([2]).

Laptops have been shown to be distracting in lecture[3] and are not allowed unless specified (or a specific exception is negotiated) -- same for cell phones.

Lecture and Lab Attendance: Attendance and participation in the course is mandatory. Participation includes responding to questions in class, lab or office

---

1 http://www.theatlantic.com/technology/archive/2014/05/to-remember-a-lecture-better-take-notes-by-hand/361478/

2 Applying the Seven Principles for Good Practice in Undergraduate Education" (1991) Chickering and Gamson

3 http://www.yorku.ca/ncepeda/laptopFAQ.html

hours and making observations or discussing course material in class, lab or office hours. Bi-weekly group exercises are also required.

Lab and Lab Exercises: Regular and frequent labs will be assigned and collected each week and, together; will comprise 10% of your course grade. The three hours of scheduled lab time each week is the primary time your instructor will be available for questions and assistance – ***make wise use of this resource!*** You are expected to work on the lab exercises during your scheduled lab time plus as much additional time as necessary to complete them. The lab exercises are designed to familiarize you with some of the concepts necessary to complete your projects and to help you do well on quizzes and exams. You may work on your projects in lab ***after*** completing all currently assigned labs. **IMPORTANT:** No late labs will be accepted.

Honesty: Although I encourage you to have lively discussions with one another, **all work you hand in must be your own work, unless otherwise specified**. If your program or parts of your program are plagiarized from another student or unapproved source, you will fail the course and a letter will be put in your file with Cal Poly Judicial Affairs.

The following schedule for the lectures and assignments is tentative.

Week 1	4/2/18	Introduction & orientation Core Java chapters 1, 2, 3	**Due date (midnight)**
	Lab 0:	*Python to Java types exercise*	due 4/9
	4/4/18	Overview of objects	
	4/6/18	Language support for classes	
	Lab 1:	*Implement classes, instantiate and use classes*	due 4/12
Week 2	4/9/18	Testing objects and classes	
	Design Doc I:	*Class identification: given code base partition data and functionality into classes*	due 4/16
	4/11/18	Class design: single responsibility	
	4/13/18	Static methods vs. instance methods	
	Lab 2:	*Static methods vs. instance methods*	due 4/19
Week 3	4/16/18	Design: information hiding; exceptions	
	Program 1	*Rewrite code based on UML design document*	due 4/23
	4/18/18	File I/O, exceptions, design	
	4/20/18	File I/O, exceptions, design	
	Lab 3:	*File I/O and class design for data*	due 4/26
Week 4	4/23/18	Design: cohesion, coupling	
	4/25/18	Interfaces: Mechanics, Polymorphism	
	4/27/18	Using interfaces to reduce coupling	
	Lab 4:	Interfaces and instanceof	due 5/3
Week 5	**Design Doc II:**	*Cohesion: use interfaces to improve project design*	due 5/4
	4/30/18	Interfaces: Lambda expressions	
	Program 2	*Rewrite code based on UML design document*	due 5/7

	5/2/18	Inheritance: open/closed principle and testing	
	5/4/18	Inheritance: open/closed principle and testing	
	Lab 5	*Mechanics of lambda expressions*	due 5/10
Week 6	5/7/18	**Midterm 1**	
	Program 3	*Generalizations: id classes with commonalities and reduce/eliminate, id related behaviors (inheritance)*	due 5/18
	5/9/18	Inheritance: open/closed principle and testing	
	5/11/18	Inheritance: open/close principle and testing	
	Lab 6:	*inheritance*	due 5/17
Week 7	5/14/18	inheritance	
	5/16/18	Object-Oriented Design: SOLID/GRASP	
	5/18/18	Object-Oriented Design: Interface segregation, Dependency inversion	
	Lab 7:	*Implement graph data structure and traversal*	due 5/24
Week 8	5/21/18	Pathing algorithms, A*	
	Program 4	*Implement project specific extension: A**	due 6/4
	5/23/18	Generic programming: Parametric polymorphism	
	5/25/18	Generic programming: design	
	Lab 8:	*More with lambda (filter)*	due 5/31
Week 9	5/28/18	Generic programming: bounded polymorphism	
	5/30/18	Bounded polymorphism, Stream processing	
	6/1/18	Stream processing	
	Lab 9:	*More exceptions*	due 6/7
Week 10	6/4/18	Exceptions & design case studies	
	Program 5	*Implement project specific extension: entity change*	due 6/8
	6/6/18	Introduction to design patterns	
	6/8/18	Review, what's next	
	Lab 10:	*wildcard*	
Final	6/11/18	Friday, 4:10 PM – 7 PM	