

Comp Sci-262 Fall 2008

Object-Oriented Design and Programming II (4 units)

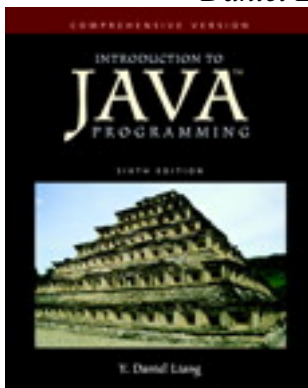
Course: Object-Oriented Design and Programming II (4 units)

Description: A second course in problem solving, software design, and computer programming with the Java language. Problem solving/software design topics will be drawn from: abstract data types, universal modeling language (UML), and finite-state machines (FSM). Data structures and algorithms include: recursive sorting, implementation of linked lists, stacks, queues, and use of other collection classes. Programming topics include: recursion, inheritance, polymorphism, templates, and graphical user interfaces.

Instructor: Wing Huen, HS221, (920)424-1324 **email:** huen@uwosh.edu

Office Hours: Tuesdays Thursdays: 9:30 a.m. to 11:00 a.m. and 1:00 p.m. to 2:30 p.m.
or by appointment..

Text: *Introduction to Java Programming Comprehensive Version, sixth Edition* by Y. Daniel Liang, Prentice Hall



Lectures: (HS202) Tuesdays Thursdays 11:30 am – 12:30 pm

Labs: (HS101C) Wednesdays 11:30 am – 12:30 pm

References:

Java APIs and Documentation on Sun Development Network

<http://java.sun.com/j2se/1.5.0/docs/api/index.html>

Instructor's Lecture Notes

BlueJ

jGRASP

Course Outcomes:

Topic areas and corresponding Learning Outcomes

1. Finite State Machine – You will be expected to
 - a. apply the finite state machine concept to design systems by drawing a state diagram based on input actions, transition of state, and output actions

- b. model states of a system as objects
 - c. apply the State Pattern to accommodate changes
2. Debugging with BlueJ – You will be expected to
- a. analyze a program on its correctness
 - b. identify software bugs with the debugger in BlueJ
3. Objects and Classes – You will be expected to
- a. specify a class with the UML graphical notation
 - b. use the UML graphical notation to describe classes
 - c. distinguish between object reference and primitive data type variables
 - d. apply classes in the Java API (Application Programming Interface)
 - e. differentiate between instance and static variables
 - f. develop methods in classes
 - g. store and process objects in arrays
 - h. apply class abstraction to develop software
4. Inheritance and Polymorphism – You will be expected to
- a. develop a subclass from a superclass through inheritance
 - b. apply the polymorphism concept to handle different data types using a uniform interface
5. Abstract Classes and Interface – You will be expected to
- a. identify the similarities and differences between an abstract class and an interface
 - b. model weak inheritance relationships with interfaces
 - c. specify a natural order using the Comparable interface
 - d. to wrap primitive data values into objects
 - e. create a generic sort method
 - f. simplify programming using JDK 1.5 automatic conversion between primitive types and wrapper class types
6. Design Patterns – You will be expected to
- a. identify the weaknesses of inheritance and interface in software maintenance when requirements change
 - b. identify the aspects of your application that vary and separate them from what stays the same
 - c. program to a supertype, not an implementation

- d. favor composition (HAS-A) relationship over (IS-A) relationship in design of classes
 - e. apply the Strategy Pattern that defines a family of algorithms, encapsulates each one, and makes them interchangeable so that strategy lets the algorithm vary independently from clients that use it
7. Exceptions and Assertions – You will be expected to
- a. distinguish exception types: Error versus Exception in Java
 - b. throw exception in a method
 - c. write an exception handler using a try-catch-finally block
 - d. explain the propagation of an exception
 - e. apply assertions to help ensure program correctness
8. Text I/O – You will be expected to
- a. read and write characters using the InputStreamReader, FileReader, BufferedReader, OutputStreamWriter, FileWriter, PrintWriter, BufferedWriter classes
 - b. be able to apply the appropriate class in text I/O operations based on the requirements and performance needs
 - c. apply the Decorator Pattern in the use of text I/O Reader and Writer classes
9. Binary I/O – You will be expected to
- a. distinguish between text I/O and binary I/O
 - b. read and write bytes using FileInputStream and FileOutputStream
 - c. read and write primitive values and strings using DataInputStream and DataOutputStream
 - d. use the Serializable interface to enable objects to be serializable
 - e. store and restore objects using ObjectOutputStream and ObjectInputStream
 - f. apply the Decorator Pattern in the use of binary I/O Stream classes
10. Object-Oriented Design – You will be expected to
- a. become familiar with the software development process
 - b. model a system with the appropriate relationships: association, aggregation, composition, dependency, strong inheritance, and weak inheritance
 - c. declare classes to represent the relationships among them.
 - d. design systems by identifying the classes and discovering the relationships among these classes

11. Unit testing with JUnit – You will be expected to
 - a. create test classes, test methods, and run tests with JUnit
 - b. create and use test fixtures in JUnit
 - c. interpret test results with JUnit
 - d. correlate the test fixtures with assertions
 - e. verify a software project performs as specified
12. GUI and Graphics – You will be expected to
 - a. describe the Java GUI hierarchy
 - b. create user interfaces using frames, panels, and simple GUI components
 - c. apply layout managers
 - d. use JPanel as subcontainers
 - e. draw figures using the methods in the Graphics class
 - f. override the paintComponent method to draw figures on a GUI component
 - g. create tables, trees, and split panes in graphical user interface
13. Event Driven Programming
 - a. declare listener classes and write event handlers to handle events
 - b. apply the Observer Pattern to decoupled programs
 - c. register listener objects in the source object
 - d. create inner classes and anonymous inner classes
 - e. write programs to handle ActionEvent, MouseEvent, KeyEvent, and Timer event
14. Recursion – You will be expected to
 - a. solve problems with recursion
 - b. write program using recursion
 - c. explain the difference between iteration and recursion
15. Data Structures: Lists, Stacks, Queues, Binary Search Trees – You will be expected to
 - a. debug a dynamic data structure with jGRASP
 - b. describe what a data structure is
 - c. explain the limitation of arrays
 - d. design and implement a dynamic list using an array
 - e. design and implement a dynamic list using a linked structure of nodes

- f. design and implement a stack using an array list
- g. design and implement a queue using a linked list
- h. design and implement a binary search tree
- i. traverse a binary search tree
- j. insert nodes into a binary search tree
- k. delete a node from a binary search tree
- l. develop API (Application Programming Interface)

16. Generics – You will be expected to

- a. improve reliability and readability of Java programs by using generic types

17. Java Collections Framework – You will be expected to

- a. describe the Java Collections Framework hierarchy
- b. utilize the common methods in the Collection interface for operating sets and lists
- c. use the Iterator interface to traverse a collection
- d. examine the Set interface and be capable of deciding when to use HashSet, LinkedHashSet, or TreeSet to store elements
- e. compare elements using the Comparator interface
- f. examine the List interface, and be capable of deciding how and when to use ArrayList or LinkedList to store elements
- g. examine the Collection and Map and be capable of deciding how and when to use HashMap, LinkedHashMap, and TreeMap to store values associated with keys.

Planned Class Schedule (may vary depending on student progress)

	Dates	Tuesday	Wednesday	Thursday	Notables
1	-, 9/4	-----	Overview, BlueJ, jGRASP Debugging	Finite State Machine	
2	9/9, 9/10, 9/11	Ch. 19 Recursion	Lab 1 Debugging;	Ch. 7 Objects and Classes	Assignment 1 on FSM; Quiz Ch. 19
3	9/16, 9/17, 9/18	Ch. 9 Inheritance & Polymorphism	Lab2 Recursion	Ch. 9 Inheritance & Polymorphism	Quiz Ch. 7
4	9/23, 9/24, 9/25	Ch. 10 Abstract Classes and	Lab 3 Objects/Classes	Ch. 10 Abstract Classes and	Assignment 2 on Recursion; Quiz Ch. 9

		Interfaces		Interfaces	
5	9/30, 10/1, 10/2	Ch. 17 Exceptions and Assertions Review	Lab 4 Inheritance & Polymorphism	Ch. 17 Exceptions and Assertions Exam 1 in evening 10/2	Quiz Ch. 10
6	10/7, 10/8, 10/9	Ch. 8 Text I/O; Ch. 18 Binary I/O;	Lab 5 Text I/O FSM	Ch. 18 Binary I/O	Assignment 3 on Inheritance & Polymorphism; Quiz Ch. 17
7	10/14, 10/15, 10/16	Ch. 11 Object- Oriented Design; JUnit	Lab 6 Binary I/O	Java Patterns	Quiz Ch. 8, 18
8	10/21, 10/22 10/23	Java Patterns	Lab 7 Interface – sort objects of any type	Conference Presentation; No Class	Assignment 4 Integrating Exceptions with Polymorphism; Quiz Ch. 11
9	10/28, 10/29, 10/30	Ch. 14 Event- Driven Programming	Lab 8 Unit testing	Ch. 12-13 GUI, Graphics	Quiz Ch. 12-13
10	11/4, 11/5, 11/6	Ch. 12-13 GUI, Graphics; Review	Lab 9 Event	Advanced GUI JTable, JTree, JList, JSplitPane; Exam 2 in evening 11/6	Assignment 5 Unit Testing or Java Patterns; Quiz Ch. 14
11	11/11, 11/12, 11/13	Advanced GUI JTable, JTree, JList, JSplitPane;	Lab 10 Data Structure	Ch. 20: Lists, Stacks, Queues, Trees, and Heaps	Quiz on Advanced GUI
12	11/18, 11/19, 11/20	Ch. 20: Lists, Stacks, Queues, Trees, and Heaps	Lab 11 Data Structure	Ch. 20: Lists, Stacks, Queues, Trees, and Heaps	Assignment 6 Data Structure; Quiz Ch. 20
13	11/25 11/26 11/27	Ch. 21 Generics	Thanksgiving; No class	Thanksgiving; No class	
14	12/2, 12/3, 12/4	Ch. 22 Java Collections Framework	Lab 12 Collections/Generics	Ch. 22 Java Collections Framework	Advanced GUI Project due; Quiz Ch. 21
15	12/9, 12/10,	Ch. 22 Java Collections	Review	Exam 3 in class	

	12/11	Framework			
--	-------	-----------	--	--	--

Courses Slides & Other Material: \Soft on 'Student' (Q:\Shared\Huen\262)

Attendance

Attendance is expected at all class and lab sessions. Students are responsible for all material presented in the course whether or not they attend the class, including announcements about course procedures. Quizzes or class exercises will be given regularly during class or labs. It is difficult to envision a student missing and/or arriving unprepared to a number of the class sessions and still succeeding in the course.

Up-to-date information about the course:

You **must** check the class web page regularly, especially when a programming assignment is due. I may need to send out new/modified information; it is *your* responsibility to obtain updates in a timely manner.

Course Grading:

The course grade is based on the total points earned on exams, programming assignments, and quizzes with the following weighting factors:

- Three(3) exams, weighted equally, for a total of 45%
- About 6 program assignments, weighted equally, worth 25%
- One project on advanced Graphical User Interface, worth 10%.
- Class participation: unannounced written quizzes and oral quizzes and class exercises, worth 10%
- Weekly labs, weighted equally, worth 10%

The course grade will be determined by the following percentage cutoffs:

Score	Grade
92-100	A
89-91	AB
82-88	B
79-81	BC
72-78	C
69-71	CD
57-68	D
< 57	F

Exam Dates

Exam 1: Thursday October 2, 2008 at 6 p.m.
 Exam 2: Thursday November 6, 2008 at 6 p.m.
 Exam 3: Thursday December 11 in class

Part of the class period before each exam will be a review session. (Changes to these exam dates will be notified approximately two weeks in advance). If you have a conflict with these dates, please inform the instructor as soon as possible to schedule a makeup exam.

Make-up Exam:

If you are unable to take a scheduled exam, you may take a make-up exam provided that you do **BOTH** of the following steps, which are subject to the instructor's approval:

1. Make arrangements *prior* to the scheduled exam (for last minute emergencies, telephone me or leave a message at 424-1324). No after-the-fact notifications will be accepted.
2. Have a written medical excuse signed by the attending physician or a note from the Dean of Students Office.

Only one make-up exam will be given. It will be a comprehensive exam given in the last week of semester.

Programming Assignments: Programming assignments are to be submitted electronically and a hard copy handed in on or before the beginning of the class period on the due date announced. Assignments turned in after this time will be counted as one day late. The grade for the assignment will be reduced by 20% for each day the assignment is late. Saturdays, Sundays, and holidays count when computing penalties. Assignments that are more than 5 days overdue will not be accepted, and will result in a grade reduction. A very late but accepted program will receive a zero grade for that assignment but will satisfy the requirement for all assignments being handed in.

The time of the electronic submission of an assignment will determine the submission date.

It is usually better to turn in a good effort on time rather than be penalized for lateness in order to add finishing touches. However, a non-working program must never be handed in on time just to avoid the late schedule. Do not miss a lecture in order to complete an assignment.

Quality of Submitted Assignments:

1. Adequate and appropriate programming comments, and good programming style.
2. No syntax errors – an assignment submitted with syntax errors will receive a score of zero.
3. Additional written documentation as required.
4. Appropriate and documented test data and test results.
5. Overall professional presentation.

Cooperation and Collusion:

All programming assignments, unless specified as team projects, are to be the sole work of the individual student. It is acceptable (and encouraged) for students to discuss the

assignment, but all programming and other written work must not involve collusion of any form i.e. sharing, borrowing, or stealing of code (code segments or entire programs). **Program coding must be 100% your own work!** Sharing or stealing code is academic dishonesty and will result in a zero grade for that assignment. In order to protect yourself, make sure your account is password protected. Don't share your account with anyone. **Don't leave computer listings in the trash or in the computer labs.** When I find two essentially identical listings, I will be forced to apply the penalty to both persons involved.

Laboratory Assignments: Each week you will be given either exercises or programming assignments for the teaching lab. The Lab Assistant and I will be available to answer your questions in the teaching lab. Often you will be asked to do some preliminary work before coming to the lab. It is your responsibility to ensure that you complete this work prior to the lab. The labs are worth 10% of the total grade.

Quizzes: There will be a quiz each week except the weeks of the exams and the first week. The quizzes will be worth 10% of your grade. There will be no make-ups for missed quizzes for any reason. Please see the instructor if you miss a quiz for a valid reason.

Frequently Asked Questions (FAQ)

What do I have to hand in for these programming assignments and what if it's late?

Handing in a programming assignment requires that you (1) submit an electronic copy of your program to the shared directory and (2) hand in a neatly stapled report, which will include a hard copy of your program, as well as notes (including pseudo-code or comments) on how you solved the problem. Each assignment will carry with it a due date. The electronic copy of your assignment is due "at the bewitching hour" on the due date. The hard copy is due at or before the beginning of the first class meeting following the due date. The hard copy must completely match the electronic copy. Failure to submit either component on time means that your assignment is late. Late programming assignments will be accepted but will be penalized at the rate shown above.

Is there any way I can carelessly lose points in the course?

Be late in handing in your work on assignments.

Don't "participate" in the class or lab and do poorly on the quizzes.

Come to your lab session late and unprepared.

What is this class participation stuff? How does one "participate" in a subject like this?

Do well in lab sessions. Prepare for the labs by completing the required reading.

Do well on the class exercises, written quizzes and oral quizzes.

"Research has demonstrated that after a lecture, students recall 62% of the information. However, only 45% is recalled by students after 3-4 days and in 8 weeks only 24% of the information is recalled. If a quiz or exam was administered after the lecture, recall was doubled at the 8-week period. It is interesting that many faculty members appear to ignore the potential impact which quizzes and tests can have upon learning." -- Bonwell C.C., Eison J.A.: Active Learning: Creating Excitement in the Classroom. Washington, DC: George Washington University, 1991.

Can I get an extension on work that is due on a specified date?

Only if you're gravely ill. Be sure that you have signed documents from a medical professional to verify the illness.

If I miss an exam, can I make it up?

Only if you were gravely ill at the time of the exam. Be sure that you have signed documents from a medical professional to verify the illness. You cannot make up a quiz.

Can I work with others on programming assignments?

There are no team assignments in this course. All the lab and programming assignments are to be the sole work of the individual student. You may help each other by discussing your approach or difficulties with other students but you should not share your program with others or copy the programs of other students. You may discuss the programming assignments at a high level (i.e., work on flow charts) with others, but the lower level work (i.e., all the work on a computer) must be done by you. You are encouraged to seek the help of the instructor and the assigned tutors of this course.

Submitting an assignment that was not entirely done by you is considered academic dishonesty and will result in appropriate disciplinary action.

Can I do programming assignments on my own computer instead of using the computer systems in a university lab?

Sure, if you have your own Windows computer system installed with Java 5.0 and BlueJ. Note that your submitted programming assignments will be graded and tested with the standard versions of Java (currently Java 5.0) and BlueJ (currently 2.2.1). It is your responsibility to ensure that your programming assignments can "Build" and "Start" (i.e. compile and run) on the lab machines.

How can I get a good grade?

Scan the class material before class. Ask the instructor and/or lab assistants on the finer points you don't understand. Work on exercises and the quizzes in the textbook to check your understanding. **Be prepared to spend at least 3 hours for each hour of contact time in class.**