Object-Oriented Design and Programming II

Comp Sci 262 – Fall 2015

Instructor: Scott Summers **Office:** Halsey 220
Email: summerss@uwosh.edu **Phone:** 920-424-1324

Office hours: MWF 9:10 AM – 10:20 AM, 11:30 AM – 12:40 PM

Lectures:	MWR	Arts & Comm Center S212	10:20 AM – 11:20 AM
Lab:	F	Halsey 101C	10:20 AM – 11:20 AM

Prerequisites: Math-108 or equivalent with a grade of C of better, or qualifying for a higher level mathematics via the Mathematics Placement Test, and CompSci-221 or equivalent with a grade of C or better.

Course website: http://www.uwosh.edu/d2l. You should check d2l on a regular basis as it will contain any lecture notes, handouts, assignments, announcements, and grades. I'll do my best to let you know when something new and important comes up, but it is your responsibility to check the web site frequently for information that you might not get otherwise.

Recommended textbook: Liang, Introduction to JAVA Programming Comprehensive, Version 8/E, Prentice Hall, 8th Ed., ISBN=9780132130806

Note: If you have special needs, please come and talk to me at the end of the first class.

Course description: A second course in problem solving, software design, and computer programming using an object-oriented language. Problem solving/software design topics include: abstract data types, universal modeling language (UML), simple recursion, unit testing, event-handling, simple concurrency. Data structures and algorithms include: binary search, simple sorting algorithms, use of collection classes and their iteration protocols, sequential file processing. Additional topics include: inheritance, polymorphism, graphical user interfaces, simple use of threads.

Course grade: Your final course grade will be based on the following components:

100 pts	**Homework**: There will be several programming assignments. Programming assignments will be worth between 15 and 25 points.
300 pts	**Exams**: There will be three exams – two midterm exams and a final exam, each worth 60 points.
50 pts	**Labs**: There will be 11 lab assignments, each worth 5 points and the lowest of which will be dropped before your final grade is computed.
50 pts	**Quizzes**: There will be six quizzes, each worth 10 points. Your lowest quiz score will be dropped before your final grade is computed. Each quiz will be taken in about 20-25 minutes of a normal class period, usually at the end of class. At least two quizzes will be given before each exam.

Grading will be on a plus/minus system. Grading *may* be done on a curve depending on the overall performance of the class. If no curve is used, your grade will be computed based on the following:

Percentage	Grade	Percentage	Grade	Percentage	Grade
>91	A	>79 and ≤ 81	B-	>67 and ≤ 69	D+
>89 and ≤ 91	A-	>77 and ≤ 79	C+	>61 and ≤ 67	D
>87 and ≤ 89	B+	>71 and ≤ 77	C	>55 and ≤ 61	D-
>81 and ≤ 87	B	>69 and ≤ 71	C-	≤55	F

Exams: Exam material will come from the lecture notes, quizzes, textbook, programming assignments and labs. There will be more information about each exam as it approaches. The actual exam dates will be announced in class at least one week before the exam. All exams will be taken during the regular class period.

If you are unable to take a scheduled exam, it may be possible to take a make-up exam provided that you do both of the following, which are then subject to my approval:

1. Make arrangements prior to the scheduled exam (for last minute emergencies, telephone me at 920-424-1324 or leave a message at the Computer Science office, 920-424-2068 **No after-the-fact notifications will be accepted.**
2. Have a written medical excuse signed by the attending physician OR have a note of justification from the Dean of Students Office.

If allowed, only one make-up exam will be given. It will be a comprehensive exam given at an arranged time during the last week of the semester.

Assignments and Labs: Most assignments and labs will consist of short to medium-length programming projects. One of your goals (during this class and beyond, in Java or any programming language) should be to write understandable, readable code. You should be making every effort to comment anything that might be confusing to a reader unfamiliar with your program, to name variables intelligently, to use indentation that reflects the code's organization, and so on. All of this will be taken into account during grading: poorly organized or written code may have a negative impact on your grade, even if the resulting program works fine. For more specific information regarding proper formatting of your programs, refer to the **Programming Ground Rules for Comp Sci 262** document. All programming assignments must strictly adhere to the aforementioned programming ground rules. Labs DO NOT need to follow the programming ground rules but some labs may require you to follow certain programming ground rules.

One of the goals of this class is to teach you to write functioning programs in Java. Thus, your code must compile and run correctly in order for you to receive full credit. **Code that does not compile will receive substantially less than full credit**. Keep this in mind when writing programs: write your code in small pieces, making sure each piece works before moving on to the next one. It is much better to turn in a project that is not finished but has many working pieces than to turn in one that doesn't work at all, even though most of the code is written.

All assignments and labs must be submitted electronically via d2l (each lab and assignment will contain specific instructions). It is your responsibility to ensure that your assignment or lab was submitted correctly. You must double check to ensure your assignment or lab was uploaded correctly.

Late homework assignments will NOT be accepted. Extensions on assignments and labs may be granted at the discretion of the instructor if you provide a valid justification (in the form of a written excuse from a medical doctor or the Dean of Students Office) before the due date.

Late assignments and labs are worth 0 points.

If you believe an assignment or exam was graded incorrectly or unfairly and would like to have it re-graded, please let me know about it *within one week* of having the assignment or exam graded. I will re-grade the entire assignment and you may gain or lose points.

Academic Dishonesty: Academic dishonesty of any kind will not be tolerated. All assignments (except for those designated as group assignments), labs, quizzes and exams are to be completed individually. While discussion of ideas and problems with fellow students is encouraged, all projects and labs must be done individually. In certain circumstances, code fragments from the instructor may be provided to eliminate tedious coding or to provide a common framework for all students. All other code must be original. Online resources may be used to help you understand the material, but you may not copy online code nor can you "borrow" code from other students, past or present. For group assignments, each group must submit original work.

Any suspected academic dishonesty will be dealt with on a case-by-case basis. Any clarification of what does or does not constitute academic dishonesty must take place *before* you turn in questionable work. For clarification on what constitutes academic dishonesty, contact me or consult the printed policy in the UWO Student Discipline Code, Chapter UWS 14.

Course Outcomes:

Topic areas and corresponding Learning Outcomes:

1. Debugging Java programs with BlueJ – You will be expected to:
    a. analyze a program on its correctness
    b. identify software bugs with the debugger in BlueJ
2. 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
3. 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
4. Abstract Classes and Interfaces – 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
5. Exceptions and Assertions – You will be expected to:
    a. distinguish exception types: Error versus Exception in Java
    b. throw an 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
6. 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. distinguish between text I/O and binary I/O

7. 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
8. 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 that a software unit performs as specified
9. 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. Introduction to Threads – creation, simple usage
10. Event Driven Programming – You will be expected to:
    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
11. Recursion – You will be expected to:
    a. solve problems with recursion
    b. write program using recursion
    c. explain the difference between iteration and recursion
12. Generic Types – You will be expected to:
    a. improve reliability and readability of Java programs by using generic types
13. 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 suing 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.