CIS 113: Data Structures
Semester: Spring 2012

Instructor: Dr. Adam Lee
E-mail: avc.dr.lee@gmail.com
Phone: N/A
Office/Hours: N/A – correspondence will be handled through e-mail

Textbook: Object Oriented Data Structures Using Java 3rd Edition (ISBN: 978-1-4496-1354-9)

Prerequisites: Completion of the following – CIS 111 / CIS 161 / (CIS 121 is also advised)
You must be eligible for – ENGL 099, READ 099, and MATH 130
You will need a flash drive with at least 4GB (Gigabytes) of memory.

*Reasonable Accommodation: If you have a legally protected disability under the Americans with Disabilities Act (ADA) or California discrimination law, and you believe you need reasonable accommodation to participate fully in this class, please contact me during to discuss your need.

Description: This course continues the introduction to programming and algorithms begun in CIS 111, with a particular focus on the ideas of data abstraction and object-oriented programming. Topics include object oriented programming, fundamental data structures, design and implementation of abstract data types, common types of collections (such as stacks, queues, lists, graphs, trees and sets), algorithm analysis and complexity, search and sort algorithms, and the use of recursion. Students plan and create programs using data structures and collection types to solve problems frequently encountered by professional computer scientists. This course is intended for students majoring in CIS. (Engineering and science majors consult counselors) (CSU, UC, AVC)

Assignments: During the course of the semester, you will be given one assignment per class. Copied work will be graded only once and the grade divided among those students who turned them in. You will be expected to turn in assignments at the beginning of the next class after the assignment was made.

To receive credit: All assignments are due by the **BEGINNING** of class on the due date
- You will not be given time to print out a program on the day of its deadline.
- **LATE WORK WILL RECEIVE NO CREDIT (period… no exceptions)**
- All assignments must contain proper headers!
- Paper assignments: Name, page # (e.g. page 2 of 5), and chapter # on each page
- Programming assignments: header must including your name & chapter #
  All programming assignments must be submitted via e-mail prior to class time.

If you are going to be absent the day an assignment is due or a test is given…
- Contact me prior to class to discuss your situation or schedule a makeup exam.
- You MUST contact me no later than one day after the date or you will receive a 0 (ZERO), NO EXCEPTIONS.

EXAMS: Only two exams will be given during the semester. You will not be allowed to make up a missed exam unless I receive prior notice that you are going to be absent. You will be expected to be prepared to take the exam on your return. A final project will be given in lieu of a final exam. The project will be due on May 30th, so… feel free to turn this one in early!

GRADING:

Class participation (& quizzes)		80 points
Assignments	12 @ 10 points each	120 points
Exams	2 @ 50 points each	100 points
Final Project	1 @ 100 points	100 points
Total		400 points

A (360-400), B (320-359), C (280-319), D (240-279), F (<240)

LAB RULES:

1. Food and drinks other than bottled water are not allowed on the third floor of this building. There will be **no exceptions** to this rule.
2. Computers must be turned off during the lecture unless otherwise specified.

ATTENDANCE: **Bottom line – BE HERE and BE ON TIME**

1. If you are registered for this class and miss the first day, **you will be dropped**.
2. Unexcused tardies will result in a **0 for class participation**.
3. **If you have more than two unexcused absences, you will be dropped.** Exceptions may be made, if I am given advance notice, there is a legitimate reason and you complete all the assigned work.

It is recommended that you become friends with another student in class and exchange contact information.

COURSE OBJECTIVES: (to complete chapters 1-13, more if possible)

1. What is/are…
    a. object oriented programming, encapsulation, inheritance, and polymorphism?
    b. the big-O notation, as it applies to the time and space complexity of simple algorithms?
    c. Application Programming Interfaces (APIs)?
2. Be able to …
    a. design, implement, test, debug and document user defined data structures.
    b. describe computational efficiency in terms of sorting, searching and hashing.
    c. compare the performance of alternative implementations of abstract data types.
    d. evaluate the tradeoffs of an algorithm in terms of time and space efficiencies.
    e. choose the appropriate type of collection for modeling a given problem.
    f. describe common applications for each type of collection.
    g. programmatically solve problems using data structures and common collection types.
    h. explain the value of APIs in software development.
    i. describe the use of divide-and-conquer and backtracking approaches in problem solving.
    j. demonstrate different traversal methods for trees and graphs.
    k. describe how iterators access the elements of a collection.

SCHEDULE: (tentative)

02/07	Intro, Review Topics, Eclipse
02/14	Chapter 1
02/21	Chapter 2.1-2.4
03/28	Chapter 2.5-2.8
03/07	Chapter 3
03/14	Chapter 4
03/21	Exam 1, Chapter 5.1-5.5
03/28	Chapter 5.6-5.8
04/04	SPRING BREAK (No Class)
04/11	Chapter 6.1-6.5
04/18	Chapter 6.6-6.8
04/25	Chapter 7
04/29	Exam 2, Chapter 8.1-8.6
05/02	Chapter 8.7-8.10
05/09	Chapter 9
05/16	Chapter 10.1-10.3
05/23	Chapter 10.4-10.6
05/30	Final Project Due!!!