# CS2013 Course Syllabus
# Fall 2017
## Section 1: Lecture: Tue/Thu 6:00 P.M. – 6:50 P.M. E&T 309
## Lab: Tue/Thu 6:50 P.M. – 8:05 P.M. E&T 309
## Section 2: Lecture Mon/Wed 3:05 P.M. – 3:55 P.M. E&T 210
## Lab: Mon/Wed 4:00 P.M. – 5:15 P.M. E&T 210

**Instructor** Jungsoo (Sue) Lim, Harminder Singh

Email: jlim34@calstatela.edu, minder91singh@gmail.com

Office hours: 1. Monday - Thursday 9:00 AM – 10:30 AM at E&T A310
2. Tuesday: 4:00 PM – 5:30 PM at E&T A310
3. By Appointment

**Course name** Programming with Data Structures

**Credits** 3 units

**Contact hours** 4 hours/week

**Text book** Data Structures & Algorithms
by Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser
in Java, 6th Edition (ISBN: 978-1-118-77133-4)

**Course Information**

a) Advanced programming techniques; elementary data structures such as dynamic arrays, linked lists, stacks, queues, and trees, sorting and searching algorithms. Laboratory activities on problem analysis and software development.
b) Prerequisites: CS2012
c) Recommended Prerequisite: Math 207, Math 248
d) This course is required in the BS program

**Course Goals** The Student Learning Outcomes that are addressed by the course are:

*SLO #2: Students will be able to demonstrate fluency in at least one programming language and acquaintance with at least three more.*
*SLO #3: Students will have a strong foundation in the design, analysis, and application of many types of algorithms.*
*SLO #5. Students will have the training to analyze problems and identify and define the computing requirements appropriate to their solutions.*
*SLO #6. Students will have the training to design, implement, and evaluate large software systems working both individually and collaboratively.*
*SLO #7. Students will be able to communicate effectively orally and in writing.*

Other outcomes of instruction: At the end of the course students are able to:

- Use recursion as a tool to solve some specific problems.

- Know the standard Abstract Data Types, and their implementations.
- Study and use different available JAVA Data Structures.
- Know the standard searching and sorting algorithms and their efficiency.
- Understand the complexity analysis for some simple software.

**Brief list of topics to be covered**

- Recursion
- Algorithm Efficiency (brief)
- Sorting
- Lists, Stacks, Queues, and Priority Queues
- Binary Search Trees
- Graphs and Applications (Optional)

**Laboratory Projects**

Each week students will complete a 3-hour lab projects on selected topics, except the exam weeks. At the end of each lab students will turn in lab programming assignment.

**Out of class Assignments**

Each week students will have an assignment due on the following week, except the exam weeks. For these assignments students may required to complete an unfinished implementation, design/implement a system, produce system document (pseudo code, UML, etc), and prepare a user's manual.

**Quizzes**

Each class will have in-class quiz.

**Grading Policy**

Quizzes: 10%

Laboratory assignments: 20 %

Laboratory Project: 10%

2 Midterms: 30%

Final: 30%

| | |
|---|---|
| A-, A | 90 – 100 |
| B-, B, B+ | 80 – 89 |
| C-,C,C+ | 65 – 79 |
| D,F | <65 |

**Academic Integrity**

Cheating will not be tolerated. Cheating on any assignment or exam will be taken seriously. All parties involved will receive a grade of F for the course and are reported to the proper authorities.

**ADA Statement**

Reasonable accommodation will be provided to any student who is registered With the Office of Students with Disabilities and requests needed accommodation.

# CS2013 Course Weekly Schedule

| Week | Lecture topics | Homework |
|---|---|---|
| 1 | **Review**<br>• Encapsulation – data protection<br>• Inheritance – super class/sub class, constructor chaining, overriding vs. overloading<br>• Polymorphism – dynamic binding, declared type, actual type, casting (implicit casting/explicit casting, up casting/down casting)<br>• Generics – creating generic class and method, benefits of generics, and restrictions<br>• Java FX – creating application, inserting components, adding event handlers, and adding UI controls. | R-2.1 – R-2.15 |
| 2 | **Chapter 3 – Fundamental Data Structure**<br>• Array List<br>• Linked List<br>○ Singly linked list<br>○ Circularly linked list<br>○ Doubly linked list<br>• Array List vs. Linked List | R-3.1- R-3.16<br>C-3.17- C-3.28 |
| 3 | **Chapter 4 – Algorithm Analysis**<br>• Compare growth rates<br>• Estimate algorithm efficiency using the Big O notation<br>• Examples of algorithm analysis | R-4.1- R-4.28<br>C-4.35 - C-4.46 |
| 4 | **Chapter 5 – Recursion**<br>• Review what a recursion method is and its benefits<br>• Examples of recursions<br>• Analyze recursive algorithms<br>• Design recursive algorithms<br>• Eliminating tail recursion | R-5.1- R-5.10<br>C-5.11 - C-5.19 |
| 5 | **Chapter 6 – Stacks, Queues, and Double-Ended Queues (Deques)**<br>• Stacks – ADT and Implementation<br>• Queues – ADT and Implementation | R-6.1- R-6.15<br>C-6.16 - C-6.22 |

| | | |
|---|---|---|
| | • Deques – ADT and implementation | |
| 6 | **Chapter 7 – List and Iterator ADTs**<br>• Array Lists<br>• Positional Lists<br>• Iterators<br>• Java Collection Framework<br>○ Overview of Lists Iterators<br>○ Overview of List-based Algorithms<br>**Midterm 1 (Chapter 3 – Chapter 7)** | R-7.1-<br>R-7.24 |
| 7 | **Chapter 8 – Trees**<br>• General Trees<br>• Tree ADT<br>• Binary Trees<br>• Implementing Trees<br>• Tree Traversal (in-order, pre-order, post-order, breath first traversal) | R-8.1-<br>R-8.20<br>C-8.27 -<br>C-8.30<br>C-8.52 |
| 8 | **Chapter 9 – Priority Queues**<br>• Priority Queue ADT<br>• Implementing Priority Queue<br>• Heaps<br>• Sorting with a Priority Queue<br>○ Selection sort<br>○ Insertion sort<br>○ Heap sort | R-9.1-<br>R-9.24<br>C-9.25 -<br>C-9.34<br>C-9.43 |
| 9 | **Chapter 10 – Maps, Hash tables, and Skip Lists**<br>• Maps ADT<br>• Hash tables – hash function, collision-handling, load factors, rehashing, and efficiency<br>• Sorted Maps<br>• Skip Lists<br>• Sets, Multisets, and Multimaps | R-10.1-<br>R-10.20<br>C-10.34 -<br>C-10.35 |
| 10 | **Chapter 11 – Search Trees**<br>• Binary Search Trees (BST) | R-11.1-<br>R-11.24 |

| | | |
|---|---|---|
| | • Balanced search trees<br>○ AVL Tree<br>○ Splay trees<br>○ (2,4) Trees<br>○ Red-black Trees<br>**Midterm 2 (Chapter 8 – Chapter 11)** | C-11.28 - C-11.48 |
| 11 | **Chapter 12 – Sorting and Selection**<br>• Merge Sort<br>• Quick Sort<br>• Selection<br>○ Prune-and-search<br>○ Randomized quick-select | R-12.1- R-12.18<br>C-12.26 - C-12.29 |
| 12 | **Chapter 14 – Graphs Algorithm - I**<br>• Data Structure for Graphs<br>• Graph Traversals<br>• Shortest Paths<br>○ Weighted Graphs<br>○ Dijkstra’s Algorithm | R-14.1- R-14.36 |
| 13 | **Chapter 14 – Graphs Algorithm - II**<br>• Minimum Spanning Tree<br>○ Prim’s Algorithm<br>○ Kruskal’s Algorithm | R-14.1- R-14.36 |
| 14 | **Chapter 15 – Memory Management and B-Tree**<br>• Memory Management<br>• Memory Hierarchies and Caching<br>• (a, b) Trees<br>• B-Tree | R-15.1- R-15.8 |
| 15 | Review for final exam | |
| 16 | Final Exam | |