Contact Information

Instructor(s):	Dr. Chung-Wen (Albert) Tsao
Office Location r	

Prerequisite(s): CS46Aï¿½or CS46AXï¿½(with a grade of "C-" or better). (If CS46A was not in Java, then CS46AW also required.) Math Enrollment Category M-I or M-II and satisfactory score on the Precalculus Proficiency Assessment (70 or higher), or MATH 19ï¿½with a C- or better, or MATH 18Aï¿½and MATH 18Bï¿½with C- or better; Allowed Majors: Computer Science, Data Science, Stats, Applied/Computational Math, Software Engineering or Forensic Science: Digital Evidence.

Letter Graded

Classroom Protocols

- Students may be dropped from the class by the instructor for either one of the following reasons:
  - absence for 1st day of class without informing you before 2nd day of class
  - lack of prerequisites.

Diversity Statement - At SJSU, it is important to create a safe learning environment where we can explore, learn, and grow together. We strive to build a diverse, equitable, inclusive culture that values, encourages, and supports students from all backgrounds and experiences.

Course Goals

C D

Intermediate concepts of Java: Classes, Inheritance, Polymorphism, Memory management, Exceptions

Introductory concepts of Data Structures: Stacks and queues, recursion, lists, dynamic arrays, binary search trees. Iteration over collections. Hashing. Searching, elementary sorting. Big-O notation. Standard collection classes. Weekly hands-on activity.

Course Learning Outcomes (CLOs)

Upon successful completion of this course, students will be able to:

1. Use and work with basic structures such as linked lists, stacks, queues, binary search trees, and iterators.
2. Implement Java classes that embody data structures.
3. Use pre-existing implementations such as the Java Collections framework.
4. Make relative estimates of the running times of alternative algorithms using Big-O analysis.
5. Formulate and test for pre-and post-conditions.
6. Distinguish between different types of program defects and understand how testing and debugging are used to correct them.
7. Implement simple sorting algorithms such as Insertion Sort and Selection Sort.
8. Implement the Sequential Search and Binary Search algorithms.
9. Implement simple recursive algorithms such as binary tree traversal.
10. Work competently with commonly used tools for software development.
11. Create custom data structures when appropriate pre-existing classes are not available

Course Materials

Big Java: Early Objects, 7e

- A : Ca Sse

c
