GAVILAN COLLEGE
5055 Santa Teresa Blvd
Gilroy, CA 95023

Course Outline

COURSE: CSIS 27 DIVISION: 50 ALSO LISTED AS:

TERM EFFECTIVE: Summer 2026 CURRICULUM APPROVAL DATE: 05/13/2025

SHORT TITLE: JAVA PROGRAMMING II

LONG TITLE: Java Programming II

Units	Number of Weeks	Type	Contact Hours/Week	Total Contact Hours
3	18	Lecture:	3	54
		Lab:	0	0
		Other:	0	0
		Total:	3	54

Out of Class Hrs:	108.00
Total Learning Hrs:	162.00

COURSE DESCRIPTION:

This course is a continuation of Java Programming I, intended for students majoring in programming and/or planning to transfer to a 4-year college or university. This course will cover topics discussed in Java Programming I in more detail. Emphasis will be placed on implementation and analysis of algorithms and abstract data types: lists, queues, stacks, arrays, trees, priority queues, heaps, tables, hashing, balanced trees, graphs, searching and sorting, and recursion. (C-ID: COMP 132) PREREQUISITE: CSIS 24 Java Programming I, or CSIS 45, or equivalent experience.

PREREQUISITES:
CSIS 24 Java Programming I, or CSIS 45, or equivalent experience.

CREDIT STATUS: D - Credit - Degree Applicable

GRADING MODES
L - Standard Letter Grade

REPEATABILITY: N - Course may not be repeated

SCHEDULE TYPES:

02 - Lecture and/or discussion

05 - Hybrid

71 - Dist. Ed Internet Simultaneous

72 - Dist. Ed Internet Delayed

STUDENT LEARNING OUTCOMES:

By the end of this course, a student should:

1. Explain the representation and use of primitive data types and built in data structures
2. Describe and demonstrate how the various data structures are allocated and used in memory.
3. Describe and utilize common applications for a variety of data structures.

COURSE OBJECTIVES:

By the end of this course, a student should:

1. Write programs that use hash tables.
2. Describe how iterators access the elements of a container.
3. Justify the philosophy of object-oriented design and the concepts of encapsulation, abstraction, inheritance, and polymorphism.
4. Design, implement, test, and debug simple programs in an object-oriented programming language.
5. Describe the concept of recursion and give examples of its use.
6. Determine when a recursive solution is appropriate for a problem.

COURSE CONTENT:

Curriculum Approval Date: 05/13/2025

Java Review

Program Design: Pseudocode, Coding, Documentation and Style, Testing and Debugging Object-Oriented Design: Goals, Principles, and Patterns, Object-Oriented Design Goals, Object-Oriented Design Principles, Design Patterns, Inheritance, Polymorphism and Dynamic Dispatch, Inheritance Hierarchies, Interfaces and Abstract Classes

Fundamental Data Structures: Using Arrays, Sorting an Array, javautil Methods for Arrays and Random Numbers, Simple Cryptography with Character Arrays, Two-Dimensional Arrays, Singly Linked Lists, Circularly Linked Lists

Doubly Linked Lists: Implementing a Doubly Linked List Class, Equivalence Testing with Arrays, Equivalence Testing with Linked Lists, Cloning Data Structures, Cloning Arrays, Cloning Linked Lists

Recursion: Illustrative Examples, The Factorial Function, Drawing an English Ruler, Binary Search

File Systems: Analyzing Recursive Algorithms, Further Examples of Recursion, Linear Recursion, Binary Recursion, Multiple Recursion, Designing Recursive Algorithms, Maximum Recursive Depth in Java, Eliminating Tail Recursion

Stacks, Queues, and Deques

Stacks: The Stack Abstract Data Type, A Simple Array-Based Stack Implementation, Implementing a Stack with a Singly Linked List, Reversing an Array Using a Stack, Matching Parentheses and HTML Tags

Queues: The Queue Abstract Data Type, Array-Based Queue Implementation, Implementing a Queue with a Singly Linked List, A Circular Queue, Double-Ended Queues

Deques: The Deque Abstract Data Type, Implementing a Deque, Deques in the Java Collections Framework

List and Iterator ADTs: The List ADT, Array Lists, Dynamic Arrays, Java's StringBuilder class

Positional Lists: Positions, The Positional List Abstract Data Type, Doubly Linked List Implementation, Iterators, The Iterable Interface and Java's For-Each Loop, Implementing Iterators, The Java Collections Framework

COURSE CONTENT (CONTINUED):

Trees: General Trees, Tree Definitions and Properties, The Tree Abstract Data Type, Computing Depth and Height, Binary Trees, The Binary Tree Abstract Data Type, Properties of Binary Trees

Implementing Trees: Linked Structure for Binary Trees, Array-Based Representation of a Binary Tree, Linked Structure for General Trees, Tree Traversal, Algorithms, Preorder and Postorder Traversals of General Trees, Breadth-First Tree Traversal, Inorder Traversal of a Binary Tree, Implementing Tree Traversals in Java

Priority Queues: The Priority Queue Abstract Data Type, Implementing a Priority Queue, The Abstract Priority Queue Base Class, Implementing a Priority Queue with an Unsorted List, Implementing a Priority Queue with a Sorted List

Hash Tables: Hash Functions, Collision-Handling Schemes, Load Factors, Rehashing, and Efficiency, Java Hash Table Implementation

Search Trees: Binary Search Trees, Searching Within a Binary Search Tree, Insertions and Deletions, Java Implementation, Performance of a Binary Search Tree, Balanced Search Trees, Java Framework for Balancing Search Trees, Red-Black Trees

Sorting and Selection: Merge-Sort, Divide-and-Conquer, Quick-Sort, Randomized Quick-Sort, Additional Optimizations for Quick-Sort, Studying Sorting through an Algorithmic Lens, Lower Bound for Sorting, Linear-Time Sorting: Bucket-Sort and Radix-Sort

Memory Management: Stacks in the Java Virtual Machine, Allocating Space in the Memory Heap, Garbage Collection, Memory Hierarchies and Caching

Final Exam

METHODS OF INSTRUCTION:

Lecture, computer demonstration, hands on exercises and practices.

OUT OF CLASS ASSIGNMENTS:

Required Outside Hours 108

Assignment Description

Read textbook and support documents

Work on sample programs, homework programs, and projects

Read and implement program specifications

Provide accurate design solutions to programming assignments

Read and modify pre-existing code following rigid specifications for program design

Create, enter, compile, run, and test programming assignments including a final program.

METHODS OF EVALUATION:

Problem-solving assignments
Evaluation Percent 40
Evaluation Description
Homework problems
Programming projects
Quizzes

Skill demonstrations
Evaluation Percent 10
Evaluation Description
Presentations

Objective examinations
Evaluation Percent 50
Evaluation Description
Objective examinations

REPRESENTATIVE TEXTBOOKS:

Data Structures and Abstractions with Java (5th Edition, 2021 update), Frank M. Carrano and Timothy M. Henry, Pearson, 2021 or a comparable textbook/material.
ISBN: 9780137515134
12+ Grade Verified by: MS Word

ARTICULATION and CERTIFICATE INFORMATION

CSU GE:

CSU B3

CSU B8

CSU E1

CSU TRANSFER:

Transferable CSU

UC TRANSFER:

Transferable UC

SUPPLEMENTAL DATA:

Basic Skills: N
Classification: Y
Noncredit Category: Y
Cooperative Education:
Program Status: 1 Program Applicable
Special Class Status: N
C-ID: (COMP 132)
Prior to College Level: Y
Non Credit Enhanced Funding: N
Funding Agency Code: Y
In-Service: N
Occupational Course: B
Course Control Number: CCC000655024
Sports/Physical Education Course: N
Taxonomy of Program: 070600