# CS G231: PYTHON PROGRAMMING 2

| Item | Value |
|---|---|
| Curriculum Committee Approval Date | 11/05/2024 |
| Top Code | 070710 - Computer Programming |
| Units | 3 Total Units |
| Hours | 90 Total Hours (Lecture Hours 36; Lab Hours 54) |
| Total Outside of Class Hours | 0 |
| Course Credit Status | Credit: Degree Applicable (D) |
| Material Fee | No |
| Basic Skills | Not Basic Skills (N) |
| Repeatable | No |
| Open Entry/Open Exit | No |
| Grading Policy | Standard Letter (S),<br>• Pass/No Pass (B) |

## Course Description

This course covers data structures and object-oriented programming (OOP) concepts using the Python language. Arrays, queues, stacks, linked-lists, trees, hashing, graphs, recursion, sorting, searching, optimization, classes, objects, inheritance, polymorphism, and algorithm complexity will be discussed and practiced. PREREQUISITE: CS G131, CS G153, or CS G175. Transfer Credit: CSU; UC. C-ID: COMP 132.**C-ID:** COMP 132.

## Course Level Student Learning Outcome(s)

1. Course Outcomes
2. Implement object-oriented class hierarchy and inheritance.
3. Apply complex data storage mechanisms and manipulation algorithms.
4. Develop programs that use abstract data structures.

## Course Objectives

- 1. Summarize advanced object-oriented programming concepts in Python.
- 2. Utilize modules including containers from the Python built-in library.
- 3. Create iterators and recursive algorithms to find, remove, reverse, and edit elements in containers.
- 4. Apply exception handling.
- 5. Apply software development methodologies and debugging techniques.
- 6. Apply algorithm optimization for improved efficiency.
- 7. Design abstract data structures using classes and objects.
- 8. Apply inheritance, polymorphism, searching and sorting.
- 9. Utilize stacks, queues, trees and linked-lists.

## Lecture Content

Python Fundamentals Python Overview Objects in Python Expressions, Operators, and Precedence Control Flow Functions Simple Input and Output Exception Handling Iterators and Generators Scopes and Namespaces Modules and the Import Statement Object-Oriented Programming Goals, Principles, and Patterns Software Development Class Definitions Inheritance Namespaces and Object-Orientation Shallow and Deep Copying Algorithm Analysis Experimental Studies Moving Beyond Experimental Analysis The Seven Functions Used in This Book Asymptotic Analysis Simple Justification Techniques Recursion Illustrative Examples Analyzing Recursive Algorithms Designing Recursive Algorithms Eliminating Tail Recursion Array-Based Sequences Python s Sequence Types Low-Level Arrays Dynamic Arrays and Amortization Efficiency of Python s Sequence Types Using Array-Based Sequences Multidimensional Data Sets Stacks, Queues, and Deques Stacks Queues Double-Ended Queues Linked Lists Singly Linked Lists Circularly Linked Lists Doubly Linked Lists The Positional List ADT Sorting a Positional List Link-Based vs Array-Based Sequences Trees General Trees Binary Trees Implementing Trees Tree Traversal Algorithms Priority Queues The Priority Queue Abstract Data Type Implementing a Priority Queue Heaps Sorting with a Priority Queue</ li> Adaptable Priority Queues Maps, Hash Tables, and Skip Lists Maps and Dictionaries Hash Tables Sorted Maps Skip Lists Sets, Multisets, and Multimaps Search Trees Binary Search Trees Balanced Search Trees Python Framework for Balancing Search Trees AVL Trees Splay Trees (2,4) Trees Red-Black Trees Sorting and Selection Sorting Algorithms Analysis Merge-Sort Quick-Sort Comparing Sorting Algorithms Python s Built-In Sorting Functions Selection Text Processing Digitized Text Pattern-Matching Algorithms Dynamic Programming Text Compression and the Greedy Method Tries Graph Algorithms Graphs Data Structures for Graphs Graph Traversals Transitive Closure Directed Acyclic Graphs Shortest Paths Minimum Spanning Trees Memory Management and B-Trees Memory Management Memory Hierarchies and Caching External Searching and B-Trees External-Memory Sorting

## Lab Content

Code all the necessary expressions, branches, loops, functions, and classes Add the appropriate error handling routines Classes and object-oriented programming Break the programs into appropriate classes Inheritance and polymorphism Linked lists and iterators Recursive techniques Stacks using arrays or linked lists Queues using arrays or linked lists Binary trees Hashing techniques Maps and sets Searching and sorting Graphs and data modeling Processing large text files External Python modules

## Method(s) of Instruction

- Lecture (02)
- DE Live Online Lecture (02S)
- DE Online Lecture (02X)
- Lab (04)
- DE Live Online Lab (04S)
- DE Online Lab (04X)

## Reading Assignments

Textbook, online resources, and instructor prepared materials.

## Writing Assignments

Students will be required to complete software development projects presented to them in the form of business automation or scientific

problems requiring Python solution implementation.  Students will be required to write documentation on their projects.

## Out-of-class Assignments

An optional library research paper will promote further study and research in current Python programming or other related topics selected by the student and approved by the instructor.

## Demonstration of Critical Thinking

Students will analyze requirements and select data structures for efficient Python solution implementation. Testing and debugging will require students to perform data tracing and problem isolation during program execution.

## Required Writing, Problem Solving, Skills Demonstration

Students will be required to complete software development projects presented to them in the form of business automation problems requiring solution implementation. Students will be required to write documentation on their projects.

## Eligible Disciplines

Computer science: Master's degree in computer science or computer engineering OR bachelor's degree in either of the above AND master's degree in mathematics, cybernetics, business administration, accounting or engineering OR bachelor's degree in engineering AND master's degree in cybernetics, engineering mathematics, or business administration OR bachelor's degree in mathematics AND master's degree in cybernetics, engineering mathematics, or business administration OR bachelor's degree in any of the above AND a master's degree in information science, computer information systems, or information systems OR the equivalent. Note: Courses in the use of computer programs for application to a particular discipline may be classified, for the minimum qualification purposes, under the discipline of the application. Master's degree required.

## Textbooks Resources

1. Required Liang, Daniel Y.. Introduction to Python Programming and Data Structures, 2nd ed. Pearson, 2020