CS G189: C++ 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), • Pass/No Pass (B)

Course Description

Formerly: Data Structures with C++. This course covers data structures and object-oriented programming (OOP) concepts using the C++ 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 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. Apply complex data storage mechanisms and manipulation algorithms.
3. Implement object-oriented class hierarchy and inheritance.
4. Develop programs that use abstract data structures.

Course Objectives

- 1. Summarize advanced object-oriented programming concepts in C++.
- 2. Utilize modules including containers from the Standard Template Library (STL).
- 3. Create iterators and recursive algorithms to find, remove, reverse, and edit elements in containers.
- 4. Manipulate bitwise algorithms.
- 5. Apply exception handling.
- 6. Apply software development methodologies and debugging techniques.
- 7. Improve algorithm optimization and efficiency.
- 8. Design abstract data structures using classes and objects.
- 9. Apply inheritance, polymorphism, searching and sorting.
- 10. Utilize stacks, queues, trees and linked-lists.

Lecture Content

C++ and object oriented programming overview Primitive types, arrays, records, and string processing Declaration models, type-checking Classes and instances of class data types Member and non-member functions Encapsulation and data hiding Aggregation and composition Inheritance and class hierarchies Polymorphism and virtual functions Exception handling Files and streams String stream Garbage collection Data representation in memory Static, stack, and heap allocation Runtime storage management Pointers and references Abstract Data Structure (ADT) Linked lists using array implementation Linked lists using classes Doubly linked lists Stacks and queues using arrays Stacks and queues using linked-lists Trees Binary search trees Balanced trees Traversal algorithms Data modeling and manipulation using graphs Containers, collections, and iterators Maps and hash tables Map ADT Hash tables and hash functions Collision-handling schemes Recursive mathematical functions Simple recursive procedures Divide-and-conquer strategies Recursive backtracking Standard Template Library (STL) Type parameters and parameterized types-templates or generics STL string class Dynamic array classes (vector and deque classes) Doubly linked list (STL list class) Efficient searches using STL associative containers (set, multiset, map and multi-map) Supplying custom sort predicates (unary and binary) Concept and usage of function objects Adaptive containers (STL stack, que ue and priority queue classes) Organize and manipulate bitwise information using STL bit set and vector classes Smart pointers Searching and sorting algorithms Algorithm design and optimization Using inheritance to implement priority queues Using polymorphism principles to store objects of varying types in a collection class Software development Design strategy and debugging techniques Development models

Lab Content

Pointers and references Classes and object-oriented programming Inheritance and polymorphism Templates and parameterized types Linked lists and iterators Recursive techniques Stacks using arrays or linked lists Queues using arrays or linked lists Binary trees Hashing techniques Searching and sorting Graphs Create a project with the correct file structure Break the programs into appropriate classes Design a simple user interface to satisfy the user interactions Code all the necessary expressions, branches, loops, functions, and classes Add the appropriate error handling routines

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 problems requiring 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 Windows Programming or other related topics selected by the student and approved by the instructor.

Demonstration of Critical Thinking

Students will analyze requirements and select the appropriate data structures for an efficient 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 Malik, D.S. . Data Structures Using C++ , 2nd ed. Cengage (Latest), 2010 , ISBN: 032119716X. Rationale: - 2. Required Liang, Daniel Y.. Introduction to C++ Programming and Data Structures, 5th ed. Georgia Southern University: Pearson, 2022

Other Resources

1. Instructor prepared materials.