CS G179: C++ PROGRAMMING, ADVANCED

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

This course covers advanced features of software development using the C++ language. Topics covered will include input/output streams, file input and output, exception handling, Standard Template Library (STL) including String class, sequential, and associative containers. Understanding function objects, STL algorithms, adaptive containers, bitset class, and smart pointers will also be discussed in lectures and practiced through lab projects. ADVISORY: CS G175. Transfer Credit: CSU; UC.

Course Level Student Learning Outcome(s)

1. Course Outcomes
2. Create data manipulation programs using STL containers.
3. Produce programs that implement principles of Object Oriented Programming (OOP) of inheritance and polymorphism.
4. Employ error isolation using both built-in and custom exception classes

Course Objectives

- 1. Apply advanced OOP concepts in C++.
- 2. Create modules including containers using STL.
- 3. Use iterators and recursive algorithms to find, remove, reverse, and update elements in data containers.
- 4. Create methods to work with bit flags to organize and manipulate bitwise information.
- 5. Employ exception handling.
- 6. Use software development methodologies and debugging techniques.

Lecture Content

Input and output streams File Input/Output (I/O) Exception handling Introduction to the STL Iterators and algorithms STL String class STL dynamic sequence container classes Array Vector Deque Forward list List Singly and doubly linked lists (STL List class). Efficient searches using STL associative containers Set Multiset Map Multimap Supplying custom sort predicates (unary and binary) Concept and usage of function objects Adaptive STL container classes Stack Queue Priority queue STL bitset and vector classes Bitwise operations Data manipulation Smart pointers

Lab Content

Analyze problem requirements and design classes Create projects with the correct file structure Design classes and class hierarchies using OOP principles Design user interface to satisfy the user interactions Implement solutions using class hierarchies Program dynamic structures using smart pointers Design solutions using function objects Select appropriate STL sequential containers to manipulate data Create programs using STL associative containers Use iterators and adapters for data searches and updates Implement exception handling using existing exception classes Create custom exception classes

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 advanced C++ solution implementation. Students will be required to write documentation with 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 advanced C++ 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 for 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 Deitel, P., Deitel, H.. C++20 for Programmers: An Objects-Natural Approach, 3rd ed. Pearson, 2022