CIC Approval: 04/14/2022
BOT APPROVAL:
STATE APPROVAL:
EFFECTIVE TERM: Fall 2022

SAN DIEGO COMMUNITY COLLEGE DISTRICT
CITY, MESA, AND MIRAMAR COLLEGES
ASSOCIATE DEGREE COURSE OUTLINE

SECTION I

SUBJECT AREA AND COURSE NUMBER: Computer and Information Sciences 187

COURSE TITLE:	**Units:**
Data Structures in C++	4
	Grade Only

CATALOG COURSE DESCRIPTION:

This course introduces students to data structures and object-oriented software engineering. Emphasis is placed on implementing basic data structures, including collections and linked structures (stacks, queues, lists, arrays, trees, and hash tables) from the perspective of object-oriented programming. Topics include algorithms, object-oriented analysis, and the design and implementation of data structures in C++. This course is designed for students majoring in computer information systems and professionals in the field who want to update their programming skills.

REQUISITES:

Prerequisite:
CISC 192 with a grade of "C" or better, or equivalent

FIELD TRIP REQUIREMENTS:
May be required

TRANSFER APPLICABILITY:
Associate Degree Credit & transfer to CSU UC Transfer Course List

CID:
COMP 132

TOTAL LECTURE HOURS:
48 - 54

TOTAL LAB HOURS:
48 - 54

TOTAL CONTACT HOURS:
96 - 108

OUTSIDE-OF-CLASS HOURS:
96 - 108

TOTAL STUDENT LEARNING HOURS:
192 - 216

STUDENT LEARNING OBJECTIVES:

Upon successful completion of the course the student will be able to:

1. Apply modularity, basic C++ data structures, pointers, function templates, and associated data processing algorithms to develop software programs.
2. Apply computational complexity analysis to explain growth rate and algorithm running time within a variety of data structures.
3. Employ Object-Oriented Programming (OOP) principles to design and represent classes and Abstract Data Types (ADTs).
4. Use classes to implement data structures.
5. Define and explain the linked list ADT and associated operations.
6. Create and implement a linked data structure.
7. Design, implement, and test stacks and queues.
8. Use recursion to create, traverse, search, and sort binary trees.
9. Define and explain binary tree ADTs and associated operations.
10. Define and explain hash table ADTs and associated operations.
11. Create and implement hash tables.
12. Collaborate to design, code, debug, and test robust C++ programming projects.

SECTION II

1. COURSE OUTLINE AND SCOPE:

A. **Outline Of Topics:**
The following topics are included in the framework of the course but are not intended as limits on content. The order of presentation and relative emphasis will vary with each instructor.

- I. Overview of C++ programming
  - A. The C++ type system
  - B. Arrays
  - C. Pointers and References
  - D. Functions and function templates
- II. Overview of computational complexity
  - A. Computational and asymptotic analysis
  - B. Big-O notation
  - C. Growth rates
  - D. Best, worst, and average case running time
- III. Introduction to Object-Oriented Programming (OOP)
  - A. Structs, classes, and objects
  - B. Abstract Data Types (ADTs)
  - C. Inheritance, encapsulation, abstraction, and polymorphism in OOP
  - D. Inheritance versus Composition
  - E. Overloaded constructors and the "Rule of 5"
  - F. Single responsibility, Open-closed, Liskov on substitution, Interface design (SOLID) principles
- IV. Linked data structures
  - A. Linked list ADT
  - B. Doubly linked list ADT
  - C. Linked list operations
  - D. Implementation
- V. Stacks and Queues
  - A. Stack and Queue ADTs
  - B. Stack and Queue implementation
  - C. Standard Template Library (STL) stacks and queues and the Adapter pattern
- VI. Binary trees
  - A. Recursive binary tree ADTs
  - B. Binary tree applications
  - C. Binary tree traversal
  - D. Recursive tree implementations
  - E. Binary Search Tree ADT
  - F. Array-based binary tree ADTs

VII. Hash tables
  A. Hash table ADT
  B. Has functions and overriding std::hash<>
  C. Collision handling
    1. Separate chaining
    2. Linear probing
    3. Quadratic probing
  D. Hash table implementations

VIII. Laboratory activities
  A. Programming projects
  B. Group problem solving
  C. Debugging
  D. Testing

B. **Reading Assignments:**
Reading assignments are required and may include, but are not limited to, the following:

I. Assigned textbook.
II. Articles in professional journals or trade magazines about C++ programming.
III. Websites related to C++.
IV. Language documentation.

C. **Writing Assignments:**
Writing assignments are required and may include, but are not limited to, the following:

I. Summaries of code design, problems, and solutions.
II. Internal and external code documentation.
III. Short papers related to topics, such as object-oriented software development methodology, data structure syntax, and semantics.
IV. Weekly discussion posts involving the current assignment, including program details, questions about code, solutions to coding issues, and/or coding best practices.

D. **Appropriate Outside Assignments:**
Outside assignments may include, but are not limited to, the following:

I. C++ user-interactive programming projects.
II. Program design, debugging, and evaluation.
III. Data structure implementation using C++ language.
IV. Algorithm implementation using C++ language.

E. **Appropriate Assignments that Demonstrate Critical Thinking:**
Critical thinking assignments are required and may include, but are not limited to, the following:

I. Analysis and critique of C++ programming packages and constructs.
II. Program design sequence creation using a variety of design tools, such as pseudocode.
III. Program design and implementation using C++.
IV. Implementation of techniques to debug and correct syntactical and logical errors.

2. METHODS OF EVALUATION:

A student's grade will be based on multiple measures of performance unless the course requires no grade. Multiple measures may include, but are not limited to, the following:

I. Performance on applied outside-class assignments.
II. Written responses to in-class assignments.
III. Responses to in-class objective and/or essay question quizzes and/or examinations.
IV. Development of programs and algorithms in C++.
V. Demonstration of interactive one-on-one algorithm analysis and program testing and operations.

VI. Participation in classroom discussions.
VII. Development of program documentation.

3. METHODS OF INSTRUCTION:

Methods of instruction may include, but are not limited to, the following:

* Audio-Visual
* Collaborative Learning
* Computer Assisted Instruction
* Distance Education (Fully online)
* Lecture-Lab Combination

4. REQUIRED TEXTS AND SUPPLIES:
Textbooks may include, but are not limited to:

TEXTBOOKS:
1. Drozdek, Adam. Data Structures and Algorithms in C++, 4th ed. Cengage Learning, 2012, ISBN: 9781133608424
2. Gaddis, Tony. Starting Out With C++ From Control Structures through Objects, 9th ed. Pearson, 2018, ISBN: 9780134498379
3. Gaddis, Tony et al. Starting Out with C++: Early Objects, 10th ed. Pearson, 2019, ISBN: 9780135862391
4. Stroustrup, Bjarne. The C++ Programming Language, 4th ed. Addison-Wesley, 2013, ISBN: 9780321563842

MANUALS:

PERIODICALS:

SOFTWARE:

SUPPLIES:
1. A removable storage device or access to Cloud storage.

ORIGINATOR: Walter Wesley

CO-CONTRIBUTOR(S)
DATE: 03/04/2022