COLLEGE OF THE DESERT

Course Code CS-009

Course Outline of Record

1. Course Code: CS-009
2. a. Long Course Title: Data Structures and Algorithms
   b. Short Course Title: DATA STRUCTURES
3. a. Catalog Course Description:
   This course provides an introduction to data structures, algorithms, and software engineering techniques. Topics include recursion, fundamental data structures (including stacks, queues, linked lists, hash tables, trees, and graphs) and the basics of algorithmic analysis (including hashing, sorting, heaps, searches and algorithm efficiency using Big-O notation). Topics will also include the development of large programs including definition, implementation, and analysis. Focus will be on object-oriented programming and its principles of objects, classes, encapsulation, inheritance and object-oriented design of abstract data types. Students will implement these concepts by writing numerous programs in an object-oriented language such as C++.
   b. Class Schedule Course Description:
   This course will use an object-oriented language such as C++ to study more advanced programming principles and techniques. Students will learn about important data structures, how to create and evaluate important sorting and searching algorithms, and basic software engineering principles.
   c. Semester Cycle (*if applicable*): *N/A*
   d. Name of Approved Program(s):
   - COMPUTER SCIENCE AS Degree and Transfer Preparation
4. Total Units: 4.00 Total Semester Hrs: 108.00
   Lecture Units: 3 Semester Lecture Hrs: 54.00
   Lab Units: 1 Semester Lab Hrs: 54.00
   Class Size Maximum: 28 Allow Audit: No
   Repeatability No Repeats Allowed
   Justification 0
5. Prerequisite or Corequisite Courses or Advisories:
   Course with requisite(s) and/or advisory is required to complete Content Review Matrix (CCForm1-A)
   Prerequisite: CS 007B
   Prerequisite: MATH 015
6. Textbooks, Required Reading or Software: *(List in APA or MLA format.)*
   a. Carrano, F. (2012). *Data abstraction and problem solving with C++: wall and mirrors* (6/e). San Francisco Addison Wesley. ISBN: 978-01329237
   College Level: Yes
   Flesch-Kincaid reading level: 12
   b. Adam Drozdek (2012). *Data Structures and Algorithms in C++* (4/e). Course Technology. ISBN: 978-113360842
   College Level: Yes
   Flesch-Kincaid reading level: *N/A*
7. Entrance Skills: *Before entering the course students must be able:*
   a. Design and develop computer software utilizing an object-oriented language, packages, modules and libraries.
   - CS 007B - Design, implement, test, and debug simple programs in an object-oriented programming language
   - CS 007B - Compare and contrast object-oriented analysis and design with structured analysis and design

   b. Understand and use techniques of inheritance and polymorphism.
   - CS 007B - Design, implement, test, and debug simple programs in an object-oriented programming language
   - CS 007B - Compare and contrast object-oriented analysis and design with structured analysis and design

c. Select appropriate data structures from the Standard Template Library such as vectors, linked lists, stacks and queues in the design of computer programs to solve complex problems from math and science.

- CS 007B - Write programs that use each of the following data structures: arrays, records, strings, linked lists, stacks, queues, and hash tables
- CS 007B - Explain how abstraction mechanisms support the creation of reusable software components

d. Create classes which implement dynamic memory allocation techniques appropriate to the design of computer programs.

- CS 007B - Write programs that use each of the following data structures: arrays, records, strings, linked lists, stacks, queues, and hash tables

e. Implement advanced file reading and writing methods.

- CS 007B - Explain how abstraction mechanisms support the creation of reusable software components
- CS 007B - Design, implement, test, and debug simple programs in an object-oriented programming language
- CS 007B - Compare and contrast object-oriented analysis and design with structured analysis and design

f. Provide recursive, iterative and explicit solutions to classic discrete mathematical problems.

- MATH 015 - Provide recursive, iterative and explicit solutions to classic discrete mathematical problems.

g.
Create and manipulate trees and specifically spanning trees to find their minimized forms.

- MATH 015 - Create and manipulate trees and specifically spanning trees to find their minimized forms.

8. Course Content and Scope:

Lecture:

1 Principles of Programmingand Software Engineering
1.1 Software Engineering and Object-Oriented Design
1.2 Achieving a Better Solution
1.3 Key Issues in Programming

2 Recursion: The Mirrors
2.1 Recursive Solutions
2.2 Counting Things
2.3 Searching an Array
2.4 Organizing Data
2.5 Recursion and Efficiency

3 Data Abstraction: The Walls
3.1 Abstract Data Types
3.2 Specifying ADTs
3.3 Implementing ADTs

4 Linked Lists
4.1 Preliminaries
4.2 Programming with Linked Lists
4.3 Variations of the Linked List
4.4 Application: Maintaining an Inventory
4.5 The C++ Standard Template Library

5 Recursion as a Problem-Solving Technique
5.1 Backtracking
5.2 Defining Languages
5.3 The Relationship Between Recursionand Mathematical Induction

6 Stacks
6.1 The Abstract Data Type Stack
6.2 Simple Applications of the ADT Stack
6.3 Implementations of the ADT Stack
6.4 Application: Algebraic Expressions

6.5 Application: A Search Problem
6.6 The Relationship Between Stacks and Recursion

7 Queues
7.1 The Abstract Data Type Queue
7.2 Simple Applications of the ADT Queue
7.3 Implementations of the ADT Queue
7.4 A Summary of Position-Oriented ADTs
7.5 Application: Simulation

8 Advanced C++ Topics
8.1 Inheritance Revisited
8.2 Virtual Methods and Late Binding
8.3 Friends
8.4 The ADTs List and Sorted List Revisited
8.5 Class Templates
8.6 Overloaded Operators
8.7 Iterators

9 Algorithm Efficiency and Sorting
9.1 Measuring the Efficiency of Algorithms
9.2 Sorting Algorithms and Their Efficiency

10 Trees
10.1 Terminology
10.2 The ADT Binary Tree
10.3 The ADT Binary Search Tree
10.4 General Trees

11 Tables and Priority Queues
11.1 The ADT Table
11.2 The ADT Priority Queue: A Variation of the ADT Table
11.3 Tables and Priority Queues in the STL

12 Advanced Implementations of Tables
12.1 Balanced Search Trees
12.2 Hashing
12.3 Data with Multiple Organizations

13 Graphs
13.1 Terminology
13.2 Graphs as ADTs
13.3 Graph Traversals
13.4 Applications of Graphs

Lab: *(if the "Lab Hours" is greater than zero this is required)*

Complete programming assignments incorporating design elements and code.

9. Course Student Learning Outcomes:
   1. Apply a systematic approach to the design, construction and management of computer programs, emphasizing programming style, documentation, and debugging techniques.
   2. Demonstrate knowledge of data structures such as stacks, lists, trees, graphs, and queues. Implement (program) these structures in appropriate applications
   3. Analyze and implement (program) sorting and searching algorithms
   4. Utilize design principles of object-oriented programming, including encapsulation, inheritance

10. Course Objectives: *Upon completion of this course, students will be able to:*

a. Compare iterative and recursive solutions for elementary problems such as factorial
b. Implement, test, and debug simple recursive functions and procedures
c. Determine when a recursive solution is appropriate for a problem
d. Implement the user-defined data structures in a high-level language.
e. Choose the appropriate data structure for modeling a given problem
f. Compare alternative implementations of data structures with respect to performance
g. Write programs that use each of the following data structures: arrays, strings, linked lists, stacks, queues, and hash tables
h. Compare and contrast the costs and benefits of dynamic and static data structure implementations
i. Discuss the computational efficiency of the principal algorithms for sorting, searching, and hashing
j. Discuss the computational efficiency of the principal algorithms for sorting, searching, and hashing

11. Methods of Instruction: *(Integration: Elements should validate parallel course outline elements)*
    a. Collaborative/Team
    b. Individualized Study
    c. Laboratory
    d. Lecture
    e. Supplemental/External Activity
    f. Technology-based instruction
    g. Tutorial

12. Assignments: *(List samples of specific activities/assignments students are expected to complete both in and outside of class.)*
    In Class Hours: 108.00
    Outside Class Hours: 108.00
    a. In-class Assignments
        1. Take quizzes.
        2. Take tests.
        3. Participate in discussion.
        4. Develop original programs to solve given problems
    b. Out-of-class Assignments
        1. Read the text.
        2. Write descriptions of programs in pseudocode.
        3. Complete unfinished lab work.
        4. Take quizzes.

13. Methods of Evaluating Student Progress: *The student will demonstrate proficiency by:*
    - Laboratory projects
      Computer programs
    - Group activity participation/observation
      Team programming.
    - Mid-term and final evaluations
      short answer essays and fill in the blank type midterms and cumulative final exam

14. Methods of Evaluating: Additional Assessment Information:

15. Need/Purpose/Rationale -- *All courses must meet one or more CCC missions.*
    PO-GE C4.b - Language & Rationality (Communication & Analytical Thinking)
    Raise questions and problems, formulating them clearly and precisely.
    Gather, assess, and interpret relevant information.
    Compare and contrast ideas from conclusions and solutions based on relevant criteria and standards
    Recognize and assess assumptions, implications, and practical consequences of alternative systems of thought.

Express solutions to complex problems using language and logic.
Apply logical and critical thinking to solve problems; explain conclusions; and evaluate, support, or critique the thinking of others.

IO - Critical Thinking and Communication
Apply principles of logic to problem solve and reason with a fair and open mind.
Apply standard conventions in grammar, mechanics, usage and punctuation.
Conduct research, gather and evaluate appropriate information, organize evidence into oral and written presentation, using proper MLA, APA, and other discipline-specific formats to cite sources.
Utilizing various communication modalities, display creative expression, original thinking, and symbolic discourse.

16. Comparable Transfer Course

University System	Campus	Course Number	Course Title	Catalog Year
CSU	CSU San Bernardino	Comp 182	Data Structures	2010-2011
UC	UC Riverside	CS 14	Data Structures	2010-2011

17. Special Materials and/or Equipment Required of Students:

18. Materials Fees: ☐ Required Material?

Material or Item	Cost Per Unit	Total Cost

19. Provide Reasons for the Substantial Modifications or New Course:

Periodic course review.

20. a. Cross-Listed Course *(Enter Course Code)*: *N/A*
    b. Replacement Course *(Enter original Course Code)*: *N/A*

21. Grading Method *(choose one)*: Letter Grade Only

22. MIS Course Data Elements
    a. Course Control Number [CB00]: CCC000525441
    b. T.O.P. Code [CB03]: 70710.00 - Computer Programming
    c. Credit Status [CB04]: D - Credit - Degree Applicable
    d. Course Transfer Status [CB05]: A = Transfer to UC, CSU
    e. Basic Skills Status [CB08]: 2N = Not basic skills course
    f. Vocational Status [CB09]: Advanced Occupational
    g. Course Classification [CB11]: Y - Credit Course
    h. Special Class Status [CB13]: N - Not Special
    i. Course CAN Code [CB14]: *N/A*
    j. Course Prior to College Level [CB21]: Y = Not Applicable
    k. Course Noncredit Category [CB22]: Y - Not Applicable
    l. Funding Agency Category [CB23]: Y = Not Applicable
    m. Program Status [CB24]: 1 = Program Applicable

    Name of Approved Program *(if program-applicable)*: *N/A*
    Attach listings of Degree and/or Certificate Programs showing this course as a required or a restricted elective.)

23. Enrollment - Estimate Enrollment
    First Year: 15
    Third Year: 15

24. Resources - Faculty - Discipline and Other Qualifications:
    a. Sufficient Faculty Resources: Yes
    b. If No, list number of FTE needed to offer this course: *N/A*
25. Additional Equipment and/or Supplies Needed and Source of Funding.
    N/A
26. Additional Construction or Modification of Existing Classroom Space Needed. *(Explain:)*
    N/A
27. FOR NEW OR SUBSTANTIALLY MODIFIED COURSES
    Library and/or Learning Resources Present in the Collection are Sufficient to Meet the Need of the Students Enrolled in the Course: Yes
28. Originator Geoffrey Hagopian Origination Date 04/06/16