COURSE OUTLINE

COS210 Computer Science II - Data Structures
Course Number Course Title

4 3 Lecture Hours/Week and 2 Lab Hours/Week
Credits Hours: lecture/laboratory/other (specify)

Catalog Description:
A study of advanced programming topics focused on logical structures of data, their physical representation, design and analysis of algorithms operating on the structures, and techniques for program development and debugging. Emphasis is placed on the appropriate use and choice of standard data structures.

Latest Review: Fall, 2015

Prerequisites: COS102 or permission of department
Corequisites: None


Information resources: Instructors website: www.mccc.edu/~reichman/CS210SCD.htm

Course Coordinator: Donald O. Reichman
Course Objectives.

1. Familiarize the student with good programming design methods, particularly Top-Down design.
2. Develop algorithms for manipulating stacks, queues, linked lists, trees, graphs.
3. Develop the data structures for implementing the above algorithms.
4. Develop recursive algorithms as they apply to trees and graphs.
5. Familiarize the student with the issues of Time complexity and examine various algorithms from this perspective.

Course-specific General Education goals and objectives.

Critical thinking, problem solving and information literacy: Students will use critical thinking and problem solving skills in analyzing information gathered through different media and from a variety of sources.

Students will identify a problem and analyze it in terms of its significant parts and the information needed to solve it. Students will use computers to access, analyze or present information, solve problems, and communicate with others. Students will formulate and evaluate possible solutions to problems, and select and defend the chosen solutions. Students will recognize weaknesses in arguments, such as the use of false or disputable premises, suppression of contrary evidence, faulty reasoning, and emotional loading.

EVALUATION
To ensure academic freedom for the individual faculty member the following is only suggested:

5 programming projects including a top-down design for each

| for each | 25% |
| 1 midterm | 15% |
| 1 final | 15% |
| 45% | 3 in-class exams |
Week 1 - Chapter 1
At the conclusion of week 1 the student should be able to discuss the following:

1. What is computer science?
2. The elements of good programming style and good project design.
3. Top-Down project design.

Week 2 -
At the conclusion of week 2 the student should be able to describe the following in detail:

1. The algorithms for manipulating singly, doubly, and circular Linked Lists.
2. The Implementation of Linked Lists using an array and pointer variables.

Lab: Students will begin coding first project.

Week 3 - Chapter 3
At the conclusion of week 3 the student should be able to describe the following in detail:

1. The algorithms for manipulating strings and character manipulation will be discussed.
2. The implementation of these algorithms with Linked Lists.

Lab: Students will begin designing their second project.

Week 4 - Chapter 4
At the conclusion of week 4 the student should be able to describe the following in detail:

1. The algorithms for manipulating stacks and queues.
2. The Implementation of the above using an array and Linked Lists.
Week 5 - Chapter 5

At the conclusion of week 5 the student should be able to:

1. Apply stacks to parsing and recursion problems.

Lab: Begin to design and code third project

Week 6 - Chapter 5

At the conclusion of week 6 the student should be able to:

1. Discuss recursion by examining several examples.
2. Unfold the recursive program by coding it non recursively.
3. Discuss the computer’s use of stacks to support recursion.
4. Create the stack frames for a recursive program

Lab: Continue coding project #3.

Week 7 - Mid Term Exam

Week 8 - Chapter 6

At the conclusion of week 8 the student should be able to describe the following in detail:

1. Tree definitions.
3. The Implementation of trees using pointer variables and arrays.

Lab: Continue coding project #3.
Week 9 - Chapter 7

At the conclusion of week 9 the student should be able to describe the following in detail:

1. Algorithms for creating complete Binary trees and almost complete Binary trees.
3. The Implementation of the above.

Lab: Begin to design and code fourth project

Week 10 - Chapter Appendix B

At the conclusion of week 10 the student should be able to:

1. Describe Time-complexity issues - definitions of Big-OH, Big Omega, Running-time.
2. Analyze several previously defined algorithms to determine their running time and the order of their running time.

Lab: Continue work on project #4.

Week 11 - Chapters 8, 9

At the conclusion of week 11 the student should be able to:

1. Discuss Sparse matrices and generalized dope vectors.
2. Implement the above with Linked Lists.
3. Understand graph terminology.

Lab: Continue work on project #4.

Week 12 - Chapter 9

At the conclusion of week 12 the student should be able to:

1. Implement graphs as adjacency matrix, adjacency list.
2. Implement Searching technique - Breadth First Search and Depth First Search
3. Describe various Graph algorithms

Lab: Continue work on project #4.
Week 13 - Chapter 10

At the conclusion of week 13 the student should be able to:

1. Understand Algorithms for simple sorts and for best sorts.

Lab: Continue work on project #4.

Week 14 - Chapter 11, 12

At the conclusion of week 14 the student should be able to:

1. Discuss algorithms for searching-hashing algorithm, binary and linear search.
2. Describe Data Management issues - secondary key searches, types of DBMS.

Lab: Complete work on final project.

Week 15 - Review for final exam.

Hand in Final Project

NOTE:

Each project should consist of the following:

1. program listing - including liberal use of comments and contiguously, a run of the project.

2. project design - the top-down structure of the project with brief pseudo code describing the logic used in the program.

3. all the above submitted in a folder.