

CSC 241 - DATA STRUCTURES

CREDIT HOURS: 3
PREREQUISITES: CSC 202; CSC 211 recommended
GRADE REMINDER: Must have a grade of C or better in each prerequisite course.

CATALOG DESCRIPTION

Advanced programming techniques including indirection and recursion. Conceptual development and implementation of data structures including arrays, records, linear lists, stacks, queues, trees, tables, and graphs. Applications involving strings, sorting, searching, and file operations.

PURPOSE OF COURSE

The purpose of this course is to familiarize the student with advanced programming techniques and to introduce the student to the most commonly used methods of data organization. Emphasis is placed on advanced programming concepts and use of information structures in applications from both physical and logical views.

EDUCATIONAL OBJECTIVES

Upon successful completion of the course, students should be able to:

1. Demonstrate knowledge of the software life cycle and the program development process.
2. Analyze problems and develop program designs with a variety of data structures including stacks, queues, lists, strings, tables, trees and graphs involving both definition and implementation issues.
3. Apply analysis techniques to problems involving iteration and recursion.
4. Build small program systems from carefully specified requirements using software engineering design and reuse principles, appropriate data structure designs, and algorithmic and program performance measures.
5. Describe well known problems and solutions in computation including searching, sorting, arithmetic evaluation, backtracking, programming languages, and string manipulation.
6. Develop and implement abstract data type specifications.
7. Apply comprehensive language features including indirection.
8. Develop both structured procedural and object oriented solutions.
9. Demonstrate an understanding of machine memory organization and operation.

CONTENT

Hours

Programming Concepts Review	6
Specification, design and implementation issues	
Abstract data type (ADT) concept	
Scalars, arrays, records, sets, files	
Correctness and analysis	
Advanced Programming Techniques	6
Indirection	
Recursion	
Examples: strings, linked lists	
Linear Data Structures	8

- Linear lists (array, bit, and linked representations)
- Stacks
- Queues
- Examples: infix to postfix, postfix evaluation, memory allocation

- Trees 6
 - Binary trees
 - General trees
 - Examples: traversal algorithms

- Sorting 3
 - Selection sorts, insertion sorts, exchange sorts including both N^2 and $N \log N$ sorts.
 - Examples: Shell, quick, radix, bubble, tree, heap, merge

- Searching 3
 - Sequential, binary, hashing
 - Example: traversal

- Files and Directories 6
 - Sequential, indexed, direct, inverted, chained
 - Example: file merge

- Advanced Topics 4
 - Graphs and digraphs
 - Matrix and list representations
 - Examples: traversal, minimum spanning tree
 - Special trees
 - B, B*, B+, AVL
 - Examples: traversal, dynamic data organization

- Exams 3

- TOTAL 45

REFERENCES

Carrano, F. M., Data Abstraction and Problem Solving with C++, 4th Ed., Addison-Wesley, 2005.

Dale, N. and Teague, D., C++ plus Data Structures, 2nd Ed., Jones and Bartlett, 2001.

Knuth, D. E., The Art of Computer Programming, Vol. I & III, Addison-Wesley, 1974.

Kruse, R. L., and Ryba, A. J., Data Structures and Program Design in C, Prentice Hall, 1999.

Main, M. And Savitch, W., Data Structures & Other Objects Using C++, 3rd Ed., Addison-Wesley, 2005.

Malik, D. S., Data Structures Using C++, Course Technology Publishing, 2003.

Tenenbaum, A. M., Langsam, Augenstein, M. J., Data Structures Using C++, Prentice Hall, 1996.

Weiss, M. A., Data Structures and Algorithm Analysis in C++, 2nd Ed., Addison-Wesley, 1999.