The University of Georgia

Department of Computer Science

# *Course Information Sheet*

# CSCI 2720

## Data Structures

**Brief Course Description**
(50-words or less)

The design, analysis, implementation, and evaluation of the fundamental structures for representing and manipulating data: lists, arrays, trees, tables, heaps, graphs, and their memory management.

**Extended Course Description / Comments**

Use this section to put additional information that's relevant to whom this course is targeting

The topics listed below are covered:
Abstract Data Structures, Arrays and Array Mapping Functions (AMFs), Linked Lists, Composite Structures, Chronologically ordered lists, Frequency ordered lists, Rings, general tree structure and terminologies, Abstract Binary trees, Expression/Parse trees, Binary Search Trees, AVL trees, Heap data structures, Priority queues (HPIFO), Heap implementation of HPIFO, Pyramidal data structures, Quadtrees, Octress, Graphs and networks, Sorting and merging algorithms, Searching algorithms, pattern matching, Efficiency issues (order of complexity), B-Trees, Data structures for distributed/parallel systems, Hashing functions, and a number of illustrative applications.

**Pre-Requisites and/or Co-Requisites**

CSCI 1730
Systems Programming

CSCI 2610
Discrete Mathematics for Computer Science

**Approved Textbooks**
(if more than one listed, the textbook used is up to the instructor's discretion)

Author(s): Harry R. Lewis & Larry Denenberg
Title: Data Structures & Their Algorithms
Edition: only one edition exists.
ISBN-13: 978-0673397362

**Specific Learning Outcomes (Performance Indicators)**

This course presents a survey of topics in computer data structures most relevant to students studying computer science. At the end of the semester, all students will be able to do the following:

1. Have a solid foundation on Abstract Data Structures.
2. Use recursion as a powerful problem solving technique in design and development of data structures.
3. Analyze the efficiency of data structures.
4. Select the most appropriate data structure for applications and being able to defend the selection.
5. Build data structures and use them as building blocks to form more complex and advanced data structures in a hierarchical manner.
6. Implement various types of lists.
7. Develop binary trees; including: abstract binary tree, expression/parse tree, AVL trees, …
8. Develop Heap data structures for important applications.
9. Design and implement various methods in creating Priority Queues.
10. Implement Quad-trees and Oct-trees.
11. Develop various searching algorithms.
12. Develop B-Trees of different orders.
13. Design and develop various Hashing functions.
14. Develop Graph-based data structures (adjacency and path matrices, …)
15. Appreciate the importance and significance of data structures in building real applications.

**Relationship Between Program Outcomes and Learning Outcomes**

| | | *Student Outcomes* | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | a | b | c | d | e | f | g | h | i | j | k |
| *Learning Outcomes* | 1 | X | X | | | | | | | X | X | |
| | 2 | X | X | | | | | | | X | X | |
| | 3 | X | X | | | | | | | X | X | |
| | 4 | X | X | | | | | | | X | X | |
| | 5 | X | X | | | | | | | X | X | |
| | 6 | X | X | | | | | | | X | X | |
| | 7 | X | X | | | | | | | X | X | |
| | 8 | X | X | | | | | | | X | X | |
| | 9 | X | X | | | | | | | X | X | |
| | 10 | X | X | | | | | | | X | X | |
| | 11 | X | X | | | | | | | X | X | |
| | 12 | X | X | | | | | | | X | X | |
| | 13 | X | X | | | | | | | X | X | |
| | 14 | X | X | | | | | | | X | X | |
| | 15 | X | X | | | | | | | X | X | |

**Student Outcomes**

a. An ability to apply knowledge of computing and mathematics appropriate to the discipline.
b. An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution.
c. An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs.
d. An ability to function effectively on teams to accomplish a common goal.
e. An understanding of professional, ethical, legal, security and social issues and responsibilities.
f. An ability to communicate effectively with a range of audiences.
g. An ability to analyze the local and global impact of computing on individuals, organizations, and society.
h. Recognition of the need for and an ability to engage in continuing professional development.
i. An ability to use current techniques, skills, and tools necessary for computing practice.
j. An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices.
k. An ability to apply design and development principles in the construction of software systems of varying complexity.

**Major Topics Covered**
(Approximate Course Hours)

3 credit hours = 37.5 contact hours
4 credit hours = 50 contact hours

Note: Exams count as a major topic covered

An Introduction to Data Structures (2 hours)
Abstract Data Structures (2 hours)
Arrays and Array Mapping Functions, AMFs (2 hours)
Various types of Lists and Applications (3 hours)
Composite Structures (2 hours)
General tree structure and terminologies (1 hour)
Abstract Binary trees (1 hour)
Expression/Parse trees (1 hour)

Binary Search Trees (3 hours)
AVL trees (2 hours)
Heap data structures (3 hours)
Priority queues, HPIFO (1 hour)
Heap implementation of HPIFO (2 hours)
Pyramidal data structures (1 hour)
Quad-trees and Oct-trees (2 hours)
Graphs and networks (3 hours)
Sorting & merging algorithms in the context of data structures (2 hours)
Searching algorithms (2 hours)
Pattern matching (1 hour)
Efficiency issues and analysis (4 ours)
B-Trees (2 hours)
Data structures for distributed/parallel systems (2 hours)
Hashing functions (2 hours)
A number of illustrative applications (4 hours)

**Assessment Plan for this Course**

Each time this course is offered, the class is initially informed of the Course Outcomes listed in this document, and they are included in the syllabus. At the end of the semester, an anonymous survey is administered to the class where each student is asked to rate how well the outcome was achieved. The choices provided use a 5-point Likert scale containing the following options: Strongly agree, Agree, Neither agree or disagree, disagree, and strongly disagree. The results of the anonymous survey are tabulated and results returned to the instructor of the course.

The course instructor takes the results of the survey, combined with sample student responses to homework and final exam questions corresponding to course outcomes, and reports these results to the ABET committee. If necessary, the instructor also writes a recommendation to the ABET committee for better achieving the course outcomes the next time the course is offered.

**How Data is Used to Assess Program Outcomes**

Each course Learning Outcome, listed above, directly supports one or more of the Student Outcomes, as is listed in "Relationships between Learning Outcomes and Student Outcomes". For CSCI 2720, Student Outcomes (a), (b) , (i), and (j) are supported.

**Course Master**

Dr. Hamid Arabnia

**Course History**

05/2002 Course Approved in CAPA
02/2012 Course Information Sheet Prepared
04/2012 Pre-requisite change to CSCI 1730 AND (CSCI 2610 or 2611))