Approval: 9th Senate Meeting

Course Name: Data Structures and Algorithms
Course Number: CS 202
Credits: 3-1-0-4
Prerequisites: IC-250 Programming and Data Structure Practicum
Intended for: UG
Distribution: Compulsory for CSE; CS elective for EE and ME
Semester: 4th

Preamble: The proposed new curriculum for CSE includes six discipline core courses:

1. Mathematical Foundations of Computer Science
2. Advanced Data Structures and Algorithms
3. Paradigms of Programming
4. Computer Organization
5. Information Systems
6. Communicating Distributed Processes

The proposed course follows the new CS curriculum design approach that strives to cover in the above-mentioned six core course all the fundamental concepts that a CS undergraduate student must know of.
The course proposal attempts to include the fundamental topics in data structures and algorithms. The topics form the essential core of this course that must be covered comprehensively, with lots of examples, and practice exercises, and weekly tutorials. Advance topics in algorithm design and analysis are going to be covered in discipline electives.

Objective: After the students have gone through a course on discrete structures, where they learn the formal and abstract representations of data and its manipulation, a course on data structures and algorithms should teach the students concrete implementations and manipulation of such discrete structures and their use in design and analysis of non-trivial algorithms for a given computational task. On completion of such a course, students should be able to

- analyse the asymptotic performance of algorithms
- demonstrate their familiarity with major data structures, rule to manipulate those, and their canonical applications
- construct efficient algorithms for some common computer engineering design problems

Further, as programming is an integral part of the CS education, in this course students should implement the data structures and algorithms they learn, compute the corresponding achievable performance (computation time, memory requirement, etc), and if possible compare the achievable performance with alternative designs.

Syllabus:

1. Complexity Analysis: Time and Space complexity of algorithms, asymptotic analysis, average and worst case analysis, asymptotic notation, importance of efficient algorithms, program performance measurement, data structures and algorithms. (2 hours)
2. Stacks and Queues: Abstract data types, sequential and linked implementations, representative applications such as towers of Hanoi, parenthesis matching, finding path in a maze. (4 hours)
3. Lists: Abstract data type, sequential and linked representations, comparison of insertion, deletion and search operations for sequential and linked lists, list and chain classes, doubly linked lists, circular lists, skip lists, applications of lists in bin sort, radix sort, sparse tables. (6 hours)
4. Dictionary: Abstract data type, array and tree based implementations. (1 hour)
5. Hashing: Search efficiency in lists and skip lists, hashing as a search structure, hash table, collision resolution, universal hashing, linear open addressing, chains, hash tables in data-compression, LZW algorithm. (4 hours)
6. Trees: Abstract data type, sequential and linked implementations, tree traversal methods and algorithms, Binary trees and their properties, threaded binary trees – differentiation, leftist trees, tournament trees, use of winner trees in mergesort as an external sorting algorithm, bin packing. (8 hours)
7. Search Trees: Binary search trees, search efficiency, insertion and deletion operations, importance of balancing, AVL trees, searching, insertion and deletions in AVL trees, Tries, 2-3 tree, B-tree. (4 hours)
8. Heaps: Heaps as priority queues, heap implementation, insertion and deletion operations, binary heaps, binomial and Fibonacci heaps, heapsort, heaps in Huffman coding. (3 hours)
9. Graphs: Definition, terminology, directed and undirected graphs, properties, implementation - adjacency matrix and linked adjacency chains, connectivity in graphs, graph traversal - breadth first and depth first, spanning trees. (4 hours)
10. Basic algorithmic techniques: Greedy algorithms, divide & conquer, dynamic programming. Search techniques – backtracking, Sorting algorithms with analysis, integer sorting, selection sort. Graph algorithms: DFS and BFS with applications, MST and shortest paths. (6 hours)

Reference Books:

1. S. Sahni, Data Structures, Algorithms, and Applications in C++, Silicon Press, 2/e, 2005.
2. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, Introduction to Algorithms, MIT Press, 3/e, 2009.
3. A. M. Tenenbaum, Y. Langsam, and M. J. Augenstein, Data Structures Using C and C++, Prentice Hall, 2/e, 1995.