Data Structure and Algorithm

EG2202CT

Year: II **Total: 7 hours /week**

Part: II **Lecture: 3 hours/week**

Tutorial: 1 hour/week

Practical: hours/week

Lab: 3 hours/week

Course Description:

The purpose of this course is to provide the students with the basic concepts of data structures and algorithms. The main objective of the course is to teach the students how to select and design data structures and algorithms that are appropriate for problems that might occur. This course offers the students a mixture of theoretical knowledge and practical experience.

Course Objectives:

On completion of this course the students will be enabled to:

1. Introduce data abstraction and data representation in memory.
2. Discuss, design and use elementary data structures such as stack, queue, linked list, tree and graph.
3. Decompose complex programming problems into manageable sub-problems.
4. Introduce theory of algorithms and their complexity.

Course Contents:

Theory

Unit 1. Introduction **[2 Hrs.]**

1.1. Algorithm and its types
1.2. Data structure and its types
1.3. Tools for algorithm analysis (Big O Notation)
  1.3.1. Type of analysis: Time and space complexity
  1.3.2. Asymptotic Notations: Big- O, Big-Ω and Big-θ

Unit 2. Stack and Queue **[6 Hrs.]**

2.1. Stack and Operation
  2.1.1. Continuous implementation of Stack with varying and fixed TOS
2.2. Application of Stack
  2.2.1. Converting Infix to Post fix expression
  2.2.2. Evaluating Post Fix expression
2.3. Queue and Operation
  2.3.1. Definition
  2.3.2. Algorithm of Enqueue and dequeue
  2.3.3. Linear Queue
  2.3.4. Circular Queue
  2.3.5. Priority Queue
  2.3.6. Applications of Queue

Unit 3. List **[8 Hrs.]**

3.1. Definition and Structure of link list
3.2. Advantage and disadvantages of link list

3.3. Operations in Singly Linked list
  3.3.1. Insertion at the beginning and end, after the node, before the node
  3.3.2. Deletion at the beginning and end, after the node, before the node
3.4. Doubly linked list
  3.4.1. Definition
  3.4.2. Structure of doubly linked list
  3.4.3. Insertion at the beginning and end, after the node, before the node
  3.4.4. Deletion at the beginning and end, after the node, before the node
  3.4.5. Advantages and disadvantages

Unit 4. Recursion **[3 Hrs.]**

4.1. Properties of recursion
4.2. Recursion vs Iteration
4.3. TOH and its solution
4.4. Solution of Fibonacci sequence and factorial

Unit 5. Trees **[6 Hrs.]**

5.1. Tree concepts
5.2. Binary tree
5.3. Application of binary tree
5.4. Node representation
5.5. Operation in Binary Tree
  5.5.1. Insertion
  5.5.2. Deletion
5.6. Algorithm of tree search
5.7. Tree traversals
  5.7.1. Pre order
  5.7.2. In order
  5.7.3. Post order
5.8. Height, level and depth of tree and its importance
5.9. AVL balance tree
  5.9.1. Definition
  5.9.2. Detection of unbalance
  5.9.3. Single and double rotation in balancing

Unit 6. Sorting **[6 Hrs.]**

6.1. Definition
6.2. Types of sorting (Internal and external)
6.3. Algorithm of Bubble sort
6.4. Algorithm of Insertion sort
6.5. Algorithm of Selection sort
6.6. Algorithm for Quick sort
6.7. Algorithm for Merge sort
6.8. Algorithm for Heap sort

Unit 7. Search **[7 Hrs.]**

7.1. Sequential search
7.2. Binary search
7.3. Tree search algorithm
7.4. Hashing

7.4.1. Definition
7.4.2. Hash function and Hash table
7.4.3. Collision in Hashing
7.4.4. Collision Resolution Techniques (Open and Closed)

Unit 8. Graph **[7 Hrs.]**

8.1. Components of Graph
8.2. Directed and Undirected
8.3. Connected and Unconnected
8.4. Path and Cycle
8.5. Adjacency sets and tables
8.6. Array based representation
8.7. Linked based and mixed implementation
8.8. Minimum Spanning Trees:
8.8.1. Kruskal's Algorithms and prim's algorithm
8.8.2. Algorithm of graph traversal (Depth First traversal, Breadth First traversal)
8.8.3. Shortest path algorithm

Practical: **[45 Hrs.]**

1. Implement stack using array
2. Implement linear and circular queue
3. Solve TOH & Fibonacci sequence using recursion
4. Implement linked list: singly and doubly
5. Perform basic operations on a binary tree data structure.
6. Implement binary search using function and without function.
7. Implement Hashing for handling the collision.

Final written exam evaluation scheme			
Unit	**Title**	**Hours**	**Marks Distribution***
1.	Introduction	2	4
2.	Stack and Queue	6	11
3.	List	8	14
4.	Recursion	3	5
5.	Trees	6	11
6.	Sorting	6	11
7.	Search	7	12
8.	Graph	7	12
	Total	**45**	**80**

* There may be minor deviation in marks distribution.

References:

1. Agarwal, U. (2012). Data Structure Using C. (3rd ed.). : S K Katari & Sons.
2. Tenenbaum, A.M, Langsam, Y & Augustein, M.J. (1996). Data Structure Using C and C++. (2nd ed.). India: Prentice Hall India.
3. Sahni, S. (2002). Data Structures, Algorithms and Applications in C++. (2nd ed.). India: University Press