

CMP 331.3 Data Structure and Algorithm (3-1-3)

Evaluation:

	Theory	Practical	Total
Sessional	30	20	50
Final	50	-	50
Total	80	20	100

Course Objectives:

The purpose of the course is to provide fundamental knowledge on data structure designing and implementation for storing information. Moreover, it provides the knowledge of various algorithms used in computer science.

Course Contents:

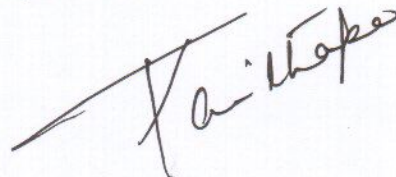
- 1. Introduction to Data Structure and algorithms (3hrs)**
 - 1.1. Review of Array, Structure, Union, Class, Pointer
 - 1.2. Abstract data type
 - 1.3. Data Structure Concept

- 2. The Stack (4hrs)**
 - 2.1. Definition and Primitive Operations
 - 2.2. Stack as an ADT, Stack operations
 - 2.3. Stack application
 - 2.4. Evaluation of Infix Postfix and prefix expressions.
 - 2.5. Expression Conversion

- 3. Queue (3hrs)**
 - 3.1. Definition, Queue as an ADT and Primitive operations in queue
 - 3.2. Linear and circular queue and their application
 - 3.3. Double Ended Queue
 - 3.4. Priority queue

- 4. Static and Dynamic List (8hrs)**
 - 4.1. Definition and Array implementation of lists
 - 4.2. Queues as a list
 - 4.3. Link List Definition and link list as an ADT
 - 4.4. Dynamic implementation
 - 4.5. Basic operations in linked list
 - 4.6. Doubly linked lists and its advantages
 - 4.7. Implementation of Doubly Linked List
 - 4.8. Linked Implementation of stacks and Queues,

- 5. Recursion (2hrs)**
 - 5.1. Principle of recursion and Comparison between recursion and iteration
 - 5.2. Factorial, TOH and Fibonacci sequence
 - 5.3. Applications of recursion and Validity of an Expression



- 6. Trees** (7hrs)
- 6.1. Concept and definitions
 - 6.2. Basic operation in binary tree
 - 6.3. Binary search tree and insertion /deletions
 - 6.4. Binary tree traversals (preorder, post order and in order) tree height level and depth
 - 6.5. Balanced trees
 - 6.6. AVL balanced trees
 - 6.7. Balancing algorithm
 - 6.8. The Huffman algorithm
 - 6.9. Game tree
 - 6.10. B- Tree.
- 7. Sorting** (5hrs)
- 7.1. Internal and external sort
 - 7.2. Insertion and selection sort
 - 7.3. Exchange sort
 - 7.4. Bubble and quick sort
 - 7.5. Merge and Radix sort
 - 7.6. Shell sort
 - 7.7. Heap sort as priority queue
 - 7.8. Efficiency of sorting.
- 8. Searching** (3hrs)
- 8.1. Search technique essential of search
 - 8.2. Sequential search
 - 8.3. Binary search
 - 8.4. Hashing :
 - 8.5. Hash function and hash tables ,
 - 8.6. Collision resolution technique ,
 - 8.7. Efficiency comparisons of different search technique.
- 9. Graphs** (8hrs)
- 9.1. Representation and applications
 - 9.2. Graphs as an ADT
 - 9.3. Transitive closure and Wars hall's algorithm
 - 9.4. Graphs types
 - 9.5. Graphs traversal and spanning forests
 - 9.6. Kruskal 's and Round Robin algorithms
 - 9.7. Shortest-path algorithm
 - 9.8. Greedy algorithm
 - 9.9. Dijkstra's Algorithm
- 10. Algorithms** (2hrs)
- 10.1. Deterministic and no-deterministic algorithm
 - 10.2. Divide and conquer algorithm
 - 10.3. Series and Parallel algorithm

K. S. S. S.

10.4. Heuristic and Approximate algorithm.

10.5. Big O Notation

Laboratory:

There shall be lab exercises based on C or C++

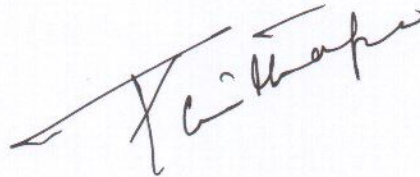
1. Implementations of stack
2. Implementations of linear and circular queues
3. Solutions of TOH and Fibonacci Recursion
4. Implementation of linked list: singly and double linked
5. Implementation of trees; AVL tree Balancing of ALV
6. Implementation of merge sort
7. Implementation of search: sequential tree and binary
8. Implementation of Graphs: Graph traversals
9. Implementation of hashing
10. Implementation of heap

Text Books:

1. Y Langsam, MJ, Augenstein and A.M , Tenenbaum Data Structures using C and C++ , Prentice Hall India
2. G.W Rowe, Introduction to Data Structure and Algorithms with C and C++ , prentice Hall India

Reference Books:

1. R.L Kruse, B.P. Leung, C.L. Tondo, data structure and program Design in C Prentice hall India
2. G. Brassard and P. Bratley fundamentals of Algorithms, prentice hall India

A handwritten signature in black ink, appearing to read 'K. S. Thakur', is written across the lower right portion of the page.