UE22CS252A : Data Structures and its Applications (4-0-2-5-5)

This course introduces abstract concepts, shows how the concepts are useful for problem solving and then shows how the abstractions can be made concrete by using a programming language. Equal emphasis is placed on both the abstract and the concrete versions of a concept so that the student learns about the concept itself, its implementation and its application.

Course Objectives:

- Basic approaches and mindsets for analyzing and designing data structures and construct essential skills of data structures to store and retrieve data quickly and usefully (efficiently).
- Usage the of different data structures that support different set of operations which are suitable for different type of tasks.
- Implement how to insert, delete, search and modify data in any given data structures- Stack, Queue, List, Tree, heap, Graphs.
- Implement a given application using the available data structure.

Course Outcomes:

At the end of this course, the student will be able to,

- Choose relevant data structures for any given application appl
- Apply the required to implement any data structure.
- Appropriate data structure in competitive programming.
- Design and develop efficient software systems with good knowledge of data structures.

Course Content:

Unit 1: Linked List and Stacks

Review of C , Static and Dynamic Memory Allocation. Linked List: Doubly Linked List, Circular Linked List – Single and Double, Multilist: Introduction to sparse matrix (structure). Skip list Case study: Dictionary implementation using skip list Stacks: Basic structure of a Stack, Implementation of a Stack using Arrays & Linked list. Applications of Stack: Function execution, Nested functions, Recursion: Tower of Hanoi. Conversion & Evaluation of an expression: Infix to postfix, Infix to prefix, Evaluation of an Expression, Matching of Parenthesis.

15 Hours

Unit 2: Queues and Trees

.Queues & Dequeue: Basic Structure of a Simple Queue, Circular Queue, Priority Queue, Dequeue and its implementation using Arrays and Linked List. Applications of Queue: Case Study – Josephus problem, CPU scheduling- Implementation using queue (simple /circular). General: N-ary trees, Binary Trees, Binary Search Trees and Forest: definition, properties, conversion of an N-ary tree and a Forest to a binary tree. Implementation of BST using arrays and dynamic allocation : Insertion and deletion operations, Traversal of trees: Preorder, Inorder and Postorder.

13 Hours

Unit 3: Application of Trees and Introduction to Graphs

Implementation of binary expression tree., Threaded binary search tree and its implementation. Heap: Implementation using arrays. Implementation of Priority Queue using heap - min and max heap. Applications of Trees and Heaps: Implementation of a dictionary / decision tree (Words with their meanings). Balanced Trees: definition, AVL Trees, Rotation, Splay Tree, Graphs: Introduction, Properties, Representation of graphs: Adjacency matrix, Adjacency list. Implementation of graphs using adjacency matrix and lists. Graph traversal methods: Depth first search, Breadth first search techniques. Application: Graph representation: Representation of computer network topology.

14 Hours

Unit 4: Applications of Graphs , B-Trees, Suffix Tree and Hashing

Application of BFS and DFS: Connectivity of graph, finding path in a network. Case Study –Indexing in databases (B Tree: K-way tree)- Insertion and deletion operations with examples. Suffix Trees: Definition, Introduction of Trie Trees, Suffix trees. Implementations of TRIE trees, insert, delete and search operations. Hashing: Simple mapping / Hashing: hash function, hash table, Collision Handling: Separate Chaining & Open Addressing, Double Hashing, and Rehashing. Applications: URLs decoding, Word prediction using TRIE trees / Suffix Trees.

14 Hours

Lab / Hands-on : 14 Hours

1:Implementation of singly linked list and advanced operations.

2: Implementation of circular linked list and an application based on it.
3: Implementation of stack and its application for prefix and postfix expression evaluation.
4: Implementation of Binary expression tree from given prefix expression and evaluate it.
5: Implementation of Graph Data structure and application based on it.
6: Implementation of Hashing Techniques.

Tool/ Languages: C Programming Language

Text Book(s):

1: "Data Structures using C / C++" , Langsum Yedidyah, Moshe J Augenstein, Aaron M Tenenbaum Pearson Education Inc, 2nd edition, 2015.

Reference Book(s):

1: "Data Structures and Program Design in C", Robert Kruse, Bruce Leung, C.L Tondo, Shashi Mogalla, Pearson, 2nd Edition, 2019.

UE22CS241B: Design and Analysis of Algorithms (4-0-0-4-4)

Algorithms play a key role in science and practice of computing. Learning algorithm design technique is a valuable endeavour from practical standpoint and algorithm design techniques have considerable utility as general problem solving strategies, applicable to problems beyond computing. This course includes classic problems of computer science, application of design techniques and analysis in terms of time and space.

Course Objectives:

- Learn to design and analyze algorithms with an emphasis on the resource utilization in terms of time and space.
- Learn various techniques in the development of algorithms so that the effect of problem size and architecture
- Design on the efficiency of the algorithm is appreciated.
- Learn to apply appropriate algorithmic design techniques for specific problems.
- Learn to trade space for time in algorithmic design using input enhancement and per-structuring.
- Learn to improve the limitations of algorithmic power.

Course Outcomes:

At the end of the course, the student will be able to:

- Identify the design technique used in an algorithm.
- Analyse algorithms using quantitative evaluation.
- Design and implement efficient algorithms for practical and unseen problems.
- Analyse time efficiency over trading space.
- Understand the limits of algorithms and the ways to cope with the limitations.

Course Content:

Unit 1: Introduction and Brute Force

Algorithms, Fundamentals of Algorithmic Problem Solving, Important Problem Types. Analysis of Algorithm Efficiency: Analysis Framework, Asymptotic Notations and Basic Efficiency Classes, Mathematical Analysis of Non Recursive and Recursive Algorithms. Brute Force: Selection Sort, Bubble Sort, Sequential Search, Brute Force String Matching, Exhaustive Search.

14 Hours

Unit 2: Decrease – and – Conquer & Divide-and-Conquer

Decrease-and-Conquer: Insertion Sort, Topological Sorting, Algorithms for Generating Combinatorial Objects, Decrease-by-a-Constant-Factor Algorithms. Divide-and-Conquer: Master Theorem, Merge Sort, Quick Sort, Binary Search, Binary Tree Traversals, Complexity analysis for finding the height of BST, Multiplication of Large Integers, Strassen's Matrix Multiplication.

14 Hours

Unit 3: Transform-and-Conquer Space and Time Tradeoffs & Greedy Technique

Transform and- Conquer**:** Pre-sorting, Heap Sort, Red-Black Trees, 2-3 Trees and Analysis of B Trees. Problems on - Decrease by a constant factor /constant number. Space and Time Tradeoffs: Sorting by Counting, Input Enhancement in String Matching - Horspool's and Boyer-Moore Algorithms. Greedy Technique: Prim's Algorithm, Kruskal's Algorithm and union-find algorithm, Dijkstra's Algorithm, Huffman Trees

16 Hours

Unit 4: Limitations, Coping with the Limitations of Algorithm Power & Dynamic Programming,

Limitations of Algorithm Power**:** Lower-Bound Arguments, Decision Trees, P, NP, and NP-Complete, NP-Hard Problems. Coping with the Limitations of Algorithm Power: Backtracking, Branch-and-Bound. Dynamic Programming: Computing a Binomial Coefficient, The Knapsack Problem and Memory Functions, Warshall's and Floyd's Algorithms.

12 Hours

Tools / Languages: C Programming Language, GCC Compiler.

Text Book(s):

1: “Introduction to the Design and Analysis of Algorithms”, Anany Levitin, Pearson Education, Delhi (Indian Version), 3rd Edition, 2012.

Reference Book(s):

1: “Introduction to Algorithms”, Thomas H Cormen, Charles E Leiserson, Ronald L Rivest and Clifford Stein, Prentice-
Hall India, 3rd Edition, 2009.
2: “Fundamentals of Computer Algorithms”, Horowitz, Sahni, Rajasekaran, Universities Press, 2nd Edition, 2007.
3: “Algorithm Design”, Jon Kleinberg, Eva Tardos, Pearson Education, 1st Edition, 2006.

UE22CS242B: Operating Systems (4-0-0-4-4)

This course focuses on fundamental operating systems concepts including various algorithms and trade-offs for efficient management of resources such as CPU, Memory, Storage and I/ O. This course requires the student to have a desirable knowledge of Data Structures and its Applications.

Course Objectives:

- Focus on fundamental Operating System concepts.
- Provide an understanding of various components of the Operating System (OS).
- Delve deeper into various algorithms and associated trade-offs for efficient resource management such as process, disk, and memory management.
- Introduce design principles and trade-offs in the design of Operating Systems.

Course Outcomes:

At the end of the course, the student will be able to:

- Understand the principles and modules of Operating Systems.
- Understand the design of various algorithms for scheduling and their relative performance.
- Understand the concept of Deadlocks that typically occur in OS. Deadlocks - Avoidance and Detection.
- Implement Operating Systems Concepts related to process management, Concurrent processes, Threads and Memory Management.

Desirable Knowledge: UE22CS252A - Data Structures and its Applications.

Course Content:

Unit 1 : Introduction and Process Management

What Operating Systems Do, Operating-System Structure & Operations, Kernel Data Structures, Operating-System Services, Operating System Design and Implementation.

Shell programming: Overview of bash shell programming – variables, control flow

Processes: process concept, Process Scheduling, Operations on Processes, System calls for process management- fork (), vfork (), wait () and exec ().

CPU Scheduling: Basic Concepts, Scheduling Criteria, Scheduling Algorithms. Case Study: Linux Scheduling Policies.

Shell programming – cron

14 Hours

Unit 2 : IPC, Threads and Concurrency –

IPC – Introduction, Shared Memory systems, Message Passing, Communication in Client–Server Systems- Pipes, ordinary pipes and named pipes, system calls for shared memory, pipes and fifo's.

Threads: Overview, Multicore Programming, Multithreading Models, Thread Libraries, Thread Scheduling.

Process Synchronization: Background, The Critical-Section Problem, Peterson's Solution, Synchronization Hardware, Mutex Locks, Semaphores, Classic Problems of Synchronization- The Bounded-Buffer Problem, The Readers–Writers Problem, The Dining-Philosophers Problem, Synchronization Examples- Synchronization in Linux. System calls for threads creation and synchronization-POSIX Threads.

Deadlocks: System Model, Deadlock Characterization, Deadlock avoidance, Banker's Algorithm, Deadlock Detection.

14 Hours

Unit 3: Memory Management

Main Memory: Background- Basic Hardware, Address Binding, Logical Versus Physical Address Space, Dynamic Loading, Dynamic Linking and Shared Libraries, Swapping, Contiguous Memory Allocation, Segmentation, Paging, Structure of the Page Table.

Virtual Memory: Background, Demand Paging, Copy-on-Write, Page Replacement Algorithms-FIFO, LRU, Optimal, Allocation of Frames, Thrashing.

14 Hours

Unit 4 : File and Storage Management

File-System Interface: File Concept, system calls for file operations-open(), read(),write(), lseek(), close() and system call to retrieve file attributes and file types-stat(), lstat(), Access Methods, Directory and Disk Structure, system calls for reading directories, system calls to create hard links (link()) and symbolic links-symlink().
File-System Implementation: File-System Structure, File-System Implementation, Directory Implementation, Allocation Methods, File Sharing, Protection.
Storage management: Overview of Mass-Storage Structure, Disk Scheduling, Swap-Space Management, RAID Structure.
System Protection: Goals, Principles and Domain of Protection, Access Matrix, Implementation of the Access Matrix, Access Control
Shell programming - awk, sed

14 Hours

Tools/Languages/OS : C, Linux/Unix OS for system call implementation.

Text Book(s):
1: "Operating System Concepts", Abraham Silberschatz, Peter Baer Galvin, Greg Gagne 9th Edition, John Wiley & Sons, India Edition ,2016.
2: "Advanced Programming in the Unix Environment", Richard Stevens and Stephen A Rago, Pearson, 3rd edition, 2017.

Reference Book(s):
1: "Operating Systems, Internals and Design Principles", William Stallings, 9th Edition, Pearson, 2018.
2: "Modern Operating Systems", Andrew S Tanenbaum, 3rd edition, Pearson, 2007.
3: "Learning the bash shell", Cameron Newham, 3rd edition, O'Reilly, 2005.

UE21CS351A: Database Management System (5-0-2-7-5)

This course offers a solid theoretical foundation in Database Management System (DBMS) and explores the practical applications of DBMS in a real-world scenario. The course emphasizes on the creation and the design of relational database systems. It also introduces the databases that provides flexible schemas and scale easily with large amounts of data and high user loads.

Course Objectives:

- Understand fundamental concepts, terminology, and application of relational databases and Construct ER diagrams for a desired application and transform the same into a relational schema.
- Understand database design concepts and design relational databases and construct basic and advanced SQL queries.
- Understand, and apply Normal Forms in database design, Transactions, Concurrency control, Locking, and Recovery, implement them in a real-time setup, and transform the given Normal Form into the desired form.
- Understand the concepts of Database Security, and NoSQL databases such as MongoDB, Neo4j, and DynamoDB

Course Outcomes:

At the end of this course, the student will be able to:

- Demonstrate an ability to explain the basic concepts of database management, construct and transform ER diagrams into Relational Schema
- Design databases and construct simple and advanced SQL queries for given contexts.
- Explain Normal Forms, employ them in Database Design, and apply database security concepts in application contexts
- Demonstrate the ability to use semi structured and NoSQL databases

Course Content:

Unit 1: Introduction to Database Management

Database System Applications, Purpose, View of data, Database Languages, Database design- Introduction to databases, Database application architecture, Users and Administrators, E-R Model, reducing ER to a relational schema. Structure of relational databases, Database schema, and its constraints, Keys

14 Hours

Unit 2: Relational Model and Database Design

Relational operations (Algebra), Unary Operations - Unity, Binary, Aggregate Functions, Grouping, SQL overview, Data definition, Structure of SQL queries, Additional Basic Operations, Set Operations, Null Values, Aggregate Functions, Nested Subqueries, Database Modification, Join expressions, Views, Triggers, Functions, and Procedures

14 Hours

Unit 3: Advanced Design Concepts and Implementation

Functional Dependencies, Inference Rules, Closure, Equivalence, Minimal Cover Normal Forms Based on Primary Keys (1NF, 2NF, and 3NF), General Definitions of Second and Third Normal Forms Boyce-Codd Normal Form, Properties of Relational Decompositions, Overview of Higher Normal Forms. Database transactions, Concurrency control, Locking, Recovery, Database Security

14 Hours

Unit 4: Advanced Databases

Query Processing and Optimization, Accessing SQL from a Programming Language, Structured, Semi structured, Unstructured data, Introduction to NoSQL databases, CAP theorem, Document database (MongoDB), Key-Value database (DynamoDB), Graph databases (Neo4j) **14 Hours**

Lab/Hands-on sessions

1: Draw an ER diagram for a given problem statement
2: Conversion of an ER diagram into Relational schema
3: DDL – create, constraints, alter, rename, drop, truncate table, Views.
4: DML – Insert, Update, Delete, Transactions - commit, rollback, savepoint
5: SQL - Set operators: union, intersect, minus.
6: SQL – Aggregate functions.
7: SQL – Joins: inner, outer; Sub queries: correlated and uncorrelated

8: SQL – Creating Functions and Procedures
9: SQL – Creating Triggers and Cursors
10: XML- Database access
11: NoSQL database queries
12: High-level programming language accessing a database using an API.

Tools/ Languages: MySQL Workbench, Python, ERwin, Any other tool for ER modeling

Text Book(s):
1: "Database System Concepts", Silberschatz, H Korth and S Sudarshan, McGrawHill, 7th Edition, 2019.
2: "Fundamentals of Database Systems", Ramez Elamsri, Shamkant B Navathe, Pearson, 7th Edition, 2017.

Reference Book(s):
1: Database Management Systems, R Ramakrishnan, J Gehrke, 3rd Edition, McGraw Hill, 2002
2: Data on the Web: From Relations to Semistructured Data and XML, S Abiteboul, P Buneman, D Suciu, Morgan Kauffman, 1999