MANIPAL UNIVERSITY JAIPUR

INSPIRED BY LIFE

Department of Artificial Intelligence & Machine Learning
School of Computer Science & Engineering

SYLLABUS

SEMESTER III

AIM2101: DATA STRUCTURES & ALGORITHMS [3 1 0 4]

Introduction: algorithm specification; Performance analysis: time and space complexity, asymptotic notation; C concepts: pointers, functions, arrays, passing arrays to functions through pointers, dynamic memory allocation, bubble sort , insertion sort , selection sort, structures, arrays of structures, passing structures to functions; **List:** ADT, array and its types, implementation, operations, linked list and its types, implementation and operations; **Stack:** ADT, implementations using array and linked list, operations and its applications; **Queue:** ADT, implementations using array and linked list, operations and its applications; **Tree:** terminologies, different types, representation of binary tree using array and linked structure, binary search tree, different operations (recursive and non-recursive), heap, heap sort, priority queue, AVL trees, B-tree; **Graph:** Introduction, representation, operations and applications; Searching techniques and hashing.

References:

1. Aaron M. Tenenbaum, Yedidyah Langsam, Moshe J. Augenstein, *Data Structures using* C, Pearson Education, 2013.
2. M. Tenenbaum et al., *Data Structures using C*, First edition, Pearson Education, 2019.
3. Ellis Horowitz, Sartaj Sahni and Susan Anderson-Freed, *Fundamentals of Data Structures in C*, University Press (India) Pvt. Ltd., 2014.
4. Alfred V. Aho, John E. Hopcroft and Jeffrey D. Ullman, *Data Structures and Algorithms*, Pearson Education, 2012.
5. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein, *Introduction to algorithms*, PHI, Third Edition, 2009.
6. Seymour Lipschutz,*Data Structures with C* (Schaum's Outline Series), McGraw Hill Education Private Limited, 2011.
7. Mark Allen Weiss, Data structures and Algorithm Analysis in C, Pearson, Second edition, 2014.

AIM2102: RELATIONAL DATABASE MANAGEMENT SYSTEM [3 1 0 4]

Introduction: DBMS Concepts, Database System Vs File System, Data Models, Schema & Instance, Schema architecture, Data independence, Data Base Languages and interfaces, Database system applications, Database users, Functions of DBA. **Data Modeling using the Entity Relationship Model:** ER model concepts, Entities, Attributes, Relationship & types, Relationship Constraints, Extended ER-Model Concept - Generalization, Specialization and Aggregation, Transforming ER diagram into the tables. **Relational Data models:** Domains, Tuples, Attributes, Relations, Characteristics of relations, Keys, Key attributes of relation, Relational database, Schemas, Integrity constraints. Referential integrity, Relational Algebra and Relational Calculus, Relational algebra operators – Unary, Binary, Set Operations. Tuple oriented and domain oriented relational calculus and its operations .**SQL:** Basic SQL Query, Creating Table and Views, SQL as DML, DDL and DCL, SQL Algebraic Operations, Joins, Sub-Queries, Aggregate Operations, Cursors, Dynamic SQL, Integrity Constraints in SQL, Triggers **Data Base Design:** Introduction to Normalization, Functional dependency, Normal forms, Decomposition, Armstrong's Axioms, Canonical Cover, Lossless Join & Dependency preservation Problems with null valued and dangling tuples, multivalued dependencies. **Transaction Processing Concepts:** Transaction Properties & States, Schedules, Serial & Concurrent Schedule, Serializability of schedules, conflict &

view serializable schedule, Recoverability, Recovery from transaction failures, log-based recovery, checkpoints, Deadlock handling. **Concurrency Control Techniques:** Concurrency control, Concept of Locks, Concurrency Control Protocols - Two Phase Locking Protocols, Time stamping protocols, validation-based protocol, multiple granularities, Multi version schemes, Recovery with concurrent transactions. **File Structures:** File Organization, Indexing, Primary, Clustered, Secondary Indexes, Hashing, Multilevel Indexing with B-Tree, B+ Tree.

References:

1. H. F. Korth, S. Sudarshan and A. Silverschatz, *Database System Concepts*, Six edition, TMH, New Delhi, 2017.
2. R. Elmasri and S. Navathe, *Fundamentals of Database systems*, Seven edition, Pearson Education, 2017.
3. C. J. Date, *Database Systems*, Eight edition, Prentice Hall of India, New Delhi, 2012.

AIM2103: PRINCIPLES OF ARTIFICIAL INTELLIGENCE [3 1 0 4]

Overview: Foundations, scope, problems, and approaches of AI; **Intelligent agents**: Reactive, deliberative, goal- driven, utility-driven, and learning agents; Artificial Intelligence programming techniques; **Problem-solving through Search**: Problem formulation, Sate-space, Production systems, Control Strategies, Informed and uninformed search, heuristic search methods, Forward and backward reasoning problem-reduction, A, A*, AO*, minimax, constraint satisfaction problem **Knowledge Representation and Reasoning**: Knowledge Based systems, Propositional Logic, syntax, semantics, inference, propositional theorem proving, Resolution- Horn clauses, Forward chaining and backward chaining, First Order Logic -Representation, Syntax and semantics, quantifiers, Inference in First Order Logic. Inductive, deductive learning. **Planning**: Planning as search, partial order planning, construction, and use of planning graphs; Blocks world problems **Representing and Reasoning with Uncertain Knowledge**: probability, connection to logic, independence, Bayes rule, Bayesian networks, probabilistic inference, sample applications; Hidden Markov Model, Maximum Entropy Markov Model, Conditional Random Field.

References:

1. S. Russell and P. Norvig, *Artificial Intelligence A Modern Approach*, Fourth edition, Pearson 2020
2. E. Rich et al., *Artificial Intelligence*, Third edition, Tata McGraw Hill, 2017
3. G. Antoniou et al., *A Semantic Web Primer*, MIT Press, 2012.

AIM2120: OBJECT ORIENTED PROGRAMMING USING PYTHON [3 1 0 4]

Introduction: Programming a computer, Programming languages; **Python basics:** Getting started with Python, Essentials of a Python program; Integers, Floating-point numbers, Strings; **Variables and scope:** Variables, Modifying values, Type conversion; **Selection control statements**: Selection: if statement, Boolean values, operators, and expressions; **Collections:** Lists, Tuples, Sets, Ranges, Dictionaries, Conversion, Sequences; **Loop control statements**: while, for statements, Nested loops, Iterables, iterators and generators, Comprehensions, The break and continue statements **Functions:** Input parameters, Return values, Default parameters,*args and **kwargs, Decoratorss, Lambdas. Generator functions and yield. **Object-Oriented programming:** OOP's Concepts, Classes, and Objects: Defining and using a class, Instance attributes, Class attributes, Class, decorators, inspecting an object, Constructor, Abstraction, Composition. Inheritance: Types of Inheritance. overriding magic methods; I/O and Errors Handling: Errors, exceptions, handling exceptions, Debugging programs, Logging, Testing. **Packaging:** Modules, Packages, Documentation, **File Handling:** Introduction, Access Methods, Read and write operation, Working with directories. **Python Libraries**: Pandas, Matplotlib, NUMPY, Introduction to GUI programming with Tkinter.

References:

1. D. Phillips, *Python 3 Object-Oriented Programming Build robust and maintainable software with object-oriented design patterns in Python 3.8*, Third edition, Packt Publishing, January 2018
2. W. J. Chun, *Core Python Applications Programming*, Third edition, Prentice Hall Publishers, 2012
3. J. Grus, *Data Science from Scratch: First Principles with Python*, First edition, O'Reilly Media, 2015.

AIM2121: OBJECT ORIENTED PROGRAMMING USING JAVA [3 1 0 4]

Basics of Java: Features of Java, Byte Code and Java Virtual Machine, JDK, Data types, Operator, Control Statements – If , else, nested if, if-else ladders, Switch, while, do-while, for, for-each, break, continue, Basics of Java: Features of Java, Byte Code and Java Virtual Machine, JDK, Data types, Operator, Control Statements – If , else, nested if, if-else ladders, Switch, while, do-while, for, for-each, break, continue. **Classes, Objects and Methods**: Class, Object, Object reference, Constructor, Constructor Overloading, Method Overloading, Recursion, Passing and Returning object form Method, new operator, this and static keyword, finalize () method, Access control, modifiers, Nested class, Inner class, Anonymous inner class, Abstract class. **Inheritance and Interfaces**: Use of Inheritance, Inheriting Data members and Methods, constructor in inheritance, Multilevel Inheritance – method overriding Handle multilevel constructors – super keyword, Stop Inheritance - Final keywords, Creation and Implementation of an interface, Interface reference, instance of operator, Interface inheritance, Dynamic method dispatch, Understanding of Java Object Class, Comparison between Abstract Class and interface. **Package:** Use of Package, CLASSPATH, Import statement, Static import, Access control. **Exception Handling:** Exception and Error, Use of try, catch, throw, throws and finally, Built in Exception, Custom exception, Throwable Class**. Multithreaded Programming:** Use of Multithread programming, Thread class and Runnable interface, Thread priority, Thread synchronization, Thread communication, Deadlock. IO Programming: Introduction to Stream, Byte Stream, Character stream, Readers and Writers, File Class, File Input Stream, File Output Stream, Input Stream Reader, Output Stream Writer, File Reader, File Writer, Buffered Reader. Collection Classes: List, Abstract List, Array List, LinkedList, Enumeration, Vector, Properties, Introduction to Java.util package.

References:

1. Brett McLaughlin, Gary Pollice, and David West, *Head First Object-Oriented Analysis and Design*, First edition, O'Reilly Media.

AIM2130: DATA STRUCTURES & ALGORITHMS LAB [0 0 2 1]

One Dimensional Arrays and Two Dimensional Arrays: Static and Dynamic Allocation, Passing of Arrays to Functions, Stack implementation using Array, Using Structures and Pointers, Programs on Evaluation of Expressions in Infix, Prefix and Postfix Notations, Programs on Conversion from One Notation to Other**, Queue :** Implementation Linear Queue, Circular Queue, Priority Queue using Array, Using Structures and Pointers **,**Tower of Hanoi, GCD, Fibonacci Definition, Binary Search, Prefix to Postfix etc. **, Link List:** Implementation of Singly, Doubly and Circular Linked Lists Using Pointers, Polynomial Addition, Sparse Matrices etc., **Tree:** Implementation of Binary Search Tree through Arrays and Pointers, Tree Traversals, Various Operations on Binary Search Tree, Huffman Algorithm, Josephus Problem etc, Implementation through Arrays and Pointers, Transitive Closure and Searching and Sorting algorithms**.**

References:

1. E. Horowitz et al., *Fundamentals of Data Structures in C*, University Press (India) Pvt. Ltd.,2014
2. M. Tenenbaum et al., *Data Structures using C*, First edition, Pearson Education, 2019
3. V. Aho et al., (1e), *Data Structures and Algorithms*, Pearson, 2012
4. T. H. Cormen et al., *Introduction to algorithms*, Third edition, PHI, 2010.
5. S. Lipschutz, *Data Structures with C (Schaum's Outline Series)*, First edition,Tata McGraw Hill Education Private Limited, 2011

AIM2131: RELATIONAL DATABASE MANAGEMENT SYSTEM LAB [0 0 2 1]

Introduction to SQL and its different command categories i.e., DDL, DML, DQL and DCL, Data Integrity Constraints and Built-in Functions, Design and implementing the data requirements of a simple DB application, Experiments on views, indexing, triggers, stored procedures, transaction.

References:
1. I. Bayross, *Teach yourself SQL & PL/SQL using Oracle 8i & 9i with SQLJ,* Third edition, BPB Publications, 2010.
2. A. Silberschatz et al., *Database System Concepts*, Six edition, McGraw Hill, 2013
3. R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems*, Seven edition, Addison-Wesley, 2017.

AIM2170: PROJECT BASED LEARNING -I [0 0 2 2]

This course aims at developing innovative skills in the students whereby they apply in totality the knowledge and skills gained through the course work in the solution of problem or by undertaking a project. Project work, therefore, should match the strengths of students. The project assignment can be individual assignment or a group assignment. The project work identified in collaboration with industry should be preferred. Each teacher is expected to guide the project work of 5-6 students. The project assignments may consist of: Programming customer-based applications, Web page designing (Only dynamic), Database applications, Software Development etc. Execution through mentor mentee policy.

SEMESTER IV

AIM2201: DESIGN & ANALYSIS OF ALGORITHMS [3 1 0 4]

Introduction: Algorithm Definition and Criteria of Algorithms, Iterative and Recursive algorithms, Performance Analysis: Priori and Posteriori Analysis, Asymptotic Notations, Space Complexity, Time Complexity, Performance measurement of iterative and recursive algorithms, **Solving Recurrence Relations:** Substitution Method, Iterative Method, Recursive Tree Method, Master Method, **Divide and Conquer:** Introduction, Binary Search, Finding Maximum and Minimum, Merge Sort, Quick Sort, Randomized Quick Sort, Closest Pair of Points, Integer Multiplication, Fast Fourier Transforms Graph Search Algorithm: Graph representation, Breadth First Search and Depth First Search, **Greedy Strategy:** Introduction, Knapsack Problem, Job Sequencing with Deadlines, Huffman Coding, Union and Find Operation (Set and Disjoint Set), Minimum Cost Spanning Tree Algorithms (Prim's and Kruskal's), Optimal Merge Patterns, Single Source Shortest Path (Dijkstra's Algorithm), Dynamic Programming: Introduction, Single Source Shortest Path (Bellman and Ford Algorithm), All Pair Shortest Path (Floyd Wrashal's Algorithm), Optimal Binary Search Trees, 0/1 Knapsack Problem, Travelling Salesperson Problem, Longest Common Subsequence, Matrix Chain Multiplication, Edit distance, Viterbi algorithm **Backtracking:** Introduction, N-Queens Problem, Graph Colouring and Hamiltonian Cycles, **Branch and Bound:** Introduction, FIFO and LC Branch and Bound, 0/1 Knapsack Problem, Travelling Salesman Problem, **String Matching:** Naïve String Matching, Rabin Karp Algorithm, Knuth-Morris-Pratt Algorithm, Boyer- Moore Algorithm, **Complexity Classes:** NP, NP-Complete and NP-Hard Problems, Polynomial time reductions, Satisfiability, Reduction from Satisfiability to Vertex Cover, Cook's Theorem.

References:
1. E. Horowitz et al., *Fundamental of Computer Algorithms*, Second edition, Universities Press, 2008.
2. T. H. Cormen et al., *Introduction to Algorithms*, Third edition, MIT press, 2010.
3. T. Roughgarden, *Algorithms Illuminated (Part 2): Graph Algorithms and Data Structures First edition,* Wiley, 2018

AIM2202: OPERATING SYSTEMS [3 1 0 4]

Introduction: Definition of operating systems, Single and multi-processor systems, Operating system services, System commands and system calls, Interrupt, System boot, Operating system structure, Types of OS, Multi-user, Multitasking, Embedded, Real-time, Network, Distributed. **Process and Thread:** Process concept, Operations on processes, Inter-process communication, UNIX pipes, Multithreading, Multithreaded models, Programs using PThread. **Process Scheduling:** Basic concepts, Scheduling criteria, Scheduling algorithms. **Synchronization:** Critical section problem, Dekker's algorithm, Peterson solution, Synchronization hardware, Semaphores, Classical problems of synchronization, Deadlock, Methods for handling deadlock- prevention, avoidance, detection, and recovery. **Memory Management**: Address binding, Logical vs Physical address space, Swapping, Contiguous memory allocation, Paging, Structure of Page Table, Segmentation, Demand Paging, Page Replacement Policies, Allocation of Frames, Thrashing. **File System Interface and Implementation:** File Concept, Access Methods, Directory and Disk Structure, File System Mounting, File System Structure, File System Implementation, Allocation Methods, Free Space Management**. Disk Management:** Disk Scheduling Algorithms, Disk Management, Swap Space Management. **Case Studies:** Linux, Windows, iOS, Android.

References:

1. A. Silberschatz, et al., *Operating System Concepts*, (9e), Wiley, 2018
2. A.S. Tanenbaum and H. Bos, *Modern Operating Systems*, (4e), Pearson, 2015
3. W. Stallings, *Operating Systems: Internals and Design Principles*, (9e), Pearson, 2018.

AIM2220: SOFTWARE ENGINEERING & PROJECT MANAGEMENT [3 1 0 4]

Software Engineering – importance – emergence - Phases of software development - Feasibility study Phases and Life cycle models of Software Development. Requirement Analysis, Design, Implementation, Testing, and Maintenance phases Different Software Life Cycle Models - Classical waterfall, Iterative, prototyping, Spiral, and Agile - Compare Lifecycle models. Requirements Analysis and **Design Requirement Analysis** – Analysis process, Requirements specification, desirable characteristics of an SRS, structure of an SRS document, Data Flow Diagrams - Role of Software Architecture and Architecture. **Software Design** - Software design concepts - Function Oriented Design and its Complexity Metrics -Object Oriented Design and its Complexity Metrics - Detailed Design. **Software Implementation and Testing-** Software Coding**-** Programming principles and coding guidelines - method of incrementally developing code - managing the evolving code Testing - Unit testing and Code Inspection - Testing concepts and testing process - Design of Test case and Test plan - Black-box testing - White box testing. **Software Project Management-**Software Project Management Framework - methods to estimate project time and cost, Resource. Planning for a Software Project. Management, Identification, Analysis, mitigation, and monitoring of Project Risks - Ensuring Project. Quality and quality management, Configuration Management, Change management, CMMI, Quality standards -ISO.

References:

1. B. Hughes et al., *Software Project Management*, Sixth Edition, McGraw Hill, 2017
2. P Jalote., *Software Project Management in Practice*, First Edition, Addison Wesley Professional, 2010.
3. R. S. Pressman*, Software Engineering: A practitioner's approach*, Eighth Edition, McGraw Hill, 2014
4. S. A. Kelkar, *Software Project Management: a concise study*, Third Edition, PHI Learning-New Delhi, 2013
5. S. H. Kan, *Metrics and Models in Software Quality Engineering*, Second Edition, Pearson, 2010.

AIM2221: AGILE SOFTWARE DEVLOPMENT [3 1 0 4]

Software Engineering — importance — emergence - Phases of software development - Feasibility study. Phases and Life cycle models of Software Development Requirement Analysis, Design, Implementation, Testing, and Maintenance phases Different Software Life Cycle Models - Classical waterfall, Iterative, prototyping, Spiral, and Agile Model. **Agile Model:** Need of Agile software development, agile context– Manifesto, Principles, Methods, Values, Roles, Artifacts, Stakeholders, and challenges. Business benefits of software agility. **Agile Project Planning:** Recognizing the structure of an agile team– Programmers, Managers, Customers. User stories– Definition, Characteristics, and content. Estimation– Planning poker, Prioritizing, and selecting user stories with the customer, projecting team velocity for releases and iterations. **Agile Project Design:** Fundamentals, Design principles–Single responsibility, Open-closed, Liskov substitution, Dependency-inversion, Interface-segregation. **Agile Project Design Methodologies:** Need of scrum, Scrum practices –Working of scrum, Project velocity, Burn down chart, Sprint backlog, Sprint planning and retrospective, Daily scrum, Scrum roles– Product Owner, Scrum Master, Scrum Team. Extreme Programming- Core principles, values, and practices. Kanban, Feature-driven development, Lean software development. **Agile Project Testing:** The Agile lifecycle and its impact on testing, test driven development– Acceptance tests and verifying stories, writing a user acceptance test, Developing effective test suites, Continuous integration, Code refactoring. Risk based testing, Regression tests, Test automation.

References:

1. Ken Schawber, Mike Beedle, *Agile Software Development with Scrum*, International Edition, Pearson.
2. Robert C. Martin, *Agile Software Development, Principles, Patterns and Practices*, First International Edition, Prentice Hall.
3. Pedro M. Santos, Marco Consolaro, and Alessandro Di Gioia, *Agile Technical Practices Distilled: A learning journey in technical practices and principles of software design*, First edition, Packt Publisher.
4.Lisa Crispin, Janet Gregory, *Agile Testing: A Practical Guide for Testers and Agile Teams*, International edition, Addison Wesley.
5. Alistair Cockburn, *Agile Software Development: The Cooperative Game*, Secon edition, Addison-Wesley.

AIM2230: DESIGN & ANALYSIS OF ALGORITHMS LAB [0 0 2 1]

Sorting & Searching Algorithm: Insertion sort, selection sort, binary search. Basic data structures: stacks and queues, graphs and trees, binary trees; Algorithmic paradigms: Recursion, divide-and-conquer, Merge sort, Quick sort. **Greedy:** Knapsack, Huffman encoding, dynamic programming, lower bounds and optimal algorithms; **Heaps:** Heaps, priority queues, min-max heaps, heap sort. Dynamic search structures, Binary search trees, height balancing, B-trees; **Algorithms on arrays:** Linear-time median finding, sorting in linear time (counting sort, radix sort, bucket sort), String matching (Rabin-Karp and Knuth-Morris-Pratt algorithms); **Graph algorithms Traversal:** (BFS, DFS, topological sort), Minimum spanning trees (Prim and Kruskal algorithms), shortest paths (Dijkstra's and Floyd-Warshal algorithms). Mini-Projects & Case Studies.

References:

1. E. Horowitz *et al*., *Fundamental of Computer Algorithms*, Second edition, Universities Press, 2008
2. T. H. Cormen *et al., Introduction to Algorithms*, Third edition, MIT press, 2010.
3. T. Roughgarden, *Algorithms Illuminated (Part 2): Graph Algorithms and Data Structures First edition, Wiley*, 2018

AIM2230: OPERATING SYSTEMS LAB [0 0 2 1]

Basic Linux commands: Illustration of shell functions, wild cards, redirection, pipes, sequencing, grouping, background processing, command substitution, sub shells, Shell programming. **System Calls:** File and process, I/O Redirection, IPC using Pipe and Signals. **PThread API:** Multithreaded programs, Synchronization programs using PThreads and Semaphores, CPU Scheduling, Deadlock, Memory Management. **Creating a Virtual Machine:** Virtual Machine Files and Snapshots, Virtual Machine Cloning and Exporting.

References:

1. W. R. Stevens and S. A. Rago, *Advanced Programming in the UNIX Environment*, Third edition, Addison-Wesley, 2017.
2. S. Das, *Unix Concepts and Applications*, Fourth edition, McGraw Hill, 2017
3. K. A. Robbins and S. Robbins, *Unix Systems Programming: Communication, Concurrency, and Threads*, Second edition, Prentice Hall, 2015.

AIM2270: PROJECT BASED LEARNING 2 [0 0 2 2]

This course aims at developing innovative skills in the students whereby they apply in totality the knowledge and skills gained through the course work in the solution of problem or by undertaking a project. Project work, therefore, should match the strengths of students. The project assignment can be individual assignment or a group assignment. The project work identified in collaboration with industry should be preferred. Each teacher is expected to guide the project work of 5-6 students. The project assignments may consist of: Programming customer-based applications, Web page designing (Only dynamic), Database applications, Machine Learning application, Software Development etc. Execution through mentor mentee policy.

PROGRAM ELECTIVE 1

AIM2140: COMPUTER ORGANIZATION & ARCHITECTURE [3 0 0 3]

Basic Structure of Computers: Computer Types, Functional Units, Basic Operational Concepts, Number Representation and Arithmetic Operations, Performance, **Instruction Set Architecture:** Memory Locations and Addresses, Instructions, and Instruction Sequencing Addressing Modes. **Arithmetic:** Addition and Subtraction of Signed Numbers, Design of Fast Adders, Multiplication of Positive Numbers, Signed Operand Multiplication, Fast Multiplication. **Memory Hierarchy:** Basics of Caches, Measuring and Improving Cache Performance. Processor Datapath and Control: Single and Multiple Bus Organization, Microprogram Control Unit, Hardware Control Unit Pipeline: Basic Concepts, Data Hazard, Instruction Hazard, Influence eon Instruction Hazard. Multicores, **Multiprocessors and Clusters:** Flynn’s classification, multi-core, superscalar, vector processor and GPU.

References:

1. C. Hamacher, Z. Vranesic and S. Zaky, *Computer Organization and Embedded Systems*, Six edition, McGraw Hill, 2017.
2. D. A. Patterson and J. L. Hennessy*, Computer Organization and Design: The Hardware and Software Interface*, Six edition, Morgan Kaufmann Publishers, 2020.
3. J. P. Hayes, *Computer Architecture and Organization*, Third edition, McGraw Hill, 2012.

AMI2141: CLOUD COMPUTING [3 0 0 3]

Introduction: Distributed Computing and Enabling Technologies, Cloud Fundamentals: Cloud Definition, Evolution, Architecture, Applications, deployment models, and service models. **Implementation:** Study of Cloud Computing Systems like Amazon EC2 and S3, Google App Engine, and Microsoft Azure, Build Private/Hybrid Cloud using open-source tools, Deployment of Web Services from Inside and Outside Cloud Architecture. MapReduce and its extensions to Cloud Computing, HDFS, and GFS**. Interoperability and Service Monitoring**: Issues with interoperability, Vendor lock-in, Interoperability approaches. SLA Management, Metering Issues, and Report generation. Resource Management and Load Balancing: Distributed Management of Virtual Infrastructures, Server consolidation, Dynamic provisioning and resource management, Resource Optimization, Resource dynamic reconfiguration, Scheduling Techniques for Advance Reservation, Capacity Management to meet SLA Requirements, and Load Balancing, various load balancing techniques. Migration and Fault Tolerance: Broad Aspects of Migration into Cloud, Migration of virtual Machines and techniques. Fault Tolerance Mechanisms. **Advances:** Grid of Clouds, Green Cloud, Mobile Cloud Computing.

References:

1. R. Buyya, J. Broberg, A. Goscinski, *Cloud Computing Principles and Paradigms*, Wiley Publishers, 2013.
2. B. Sosinsky, *Cloud Computing Bible*, Wiley, 2011.
3. M. Miller, *Cloud Computing: Web-based Applications that change the way you work and collaborate online*, Pearson, 2008.
4. D. S. Linthicum, *Cloud Computing and SOA Convergence in Your Enterprise: A Step-by-Step Guide*, Addision Wesley Information Technology Series, 2010.
5. T. Velte, A. T. Velte, R. Elsenpeter, Cloud *Computing: A Practical Approach*, McGraw Hill, 2017.