Department of Computer Science and Engineering

Scheme of Examination and Syllabus for Post Graduate Programme

Under Learning Outcome Based Curriculum Framework-Choice Based Credit System (LOCF-CBCS) as per NEP-2020 w.e.f. session 2025-26 (in phased manner)

M.Sc. Computer Science

Guru Jambheshwar University of Science &Technology
Hisar-125001, Haryana

(A+ NAAC Accredited State Govt. University)

Guru Jambheshwar University of Science and Technology Hisar-125001, Haryana

('A+' NAAC Accredited State Govt. University)

Name of the Programme: Computer Science PG Programs (M.Sc.)
Scheme of Examination for Affiliated Degree College w.e.f. session 2025-26
According to National Education Policy-2020

SEMESTER-I

M.Sc. (Computer Science)

Type of Course	Course Code	Nomenclature of Paper/Course	Credits	Contact Hours	Internal Marks	External Marks	Total Marks	Duration of Exam (Hrs.)
Discipline Specific Course	P25MCS101T	Advanced Database Systems	4	4	30	70	100	3
	P25MCS102T	Linux and Shell programming	4	4	30	70	100	3
	P25MCS103T	Web Engineering	4	4	30	70	100	3
	P25MCS104T	Computer Networks	4	4	30	70	100	3
Practicum Course	P25MCS105P	Linux and Shell programming Lab	2	4	15	35	50	3
	P25MCS106P	Web Engineering Lab	2	4	15	35	50	3
Discipline Elective Course	P25MCS111T	Design and Analysis of Algorithms	4	4	30	70	100	3
Value-Added Courses (VAC)		To be opted from VAC Pool	2	2	15	35	50	2
Total Credits			**26**				**650**	

*Value Added Course Offered by the Department of Computer Science & Engineering

Type of Course	Course Code	Nomenclature of Paper/Course	Credits	Contact Hours	Internal Marks	External Marks	Total Marks	Duration of Exam (Hrs.)
Value-Added Courses (VAC)	U25VAC110T	IoT in Smart Cities and Automation	2	2	15	35	50	2

Guru Jambheshwar University of Science and Technology Hisar-125001, Haryana
('A+' NAAC Accredited State Govt. University)

Name of the Programme: Computer Science PG Programs (M.Sc.)
Scheme of Examination for Affiliated Degree College w.e.f session 2025-26
According to National Education Policy-2020

SEMESTER-II

M.Sc. (Computer Science)

Type of Course	Course Code	Nomenclature of Paper/Course	Credits	Contact Hours	Internal Marks	External Marks	Total Marks	Duration of Exam (Hrs.)
Discipline Specific Course	P25MCS201T	Advance Data Structures	4	4	30	70	100	3
	P25MCS202T	Python Programming	4	4	30	70	100	3
	P25MCS203T	Artificial Intelligence	4	4	30	70	100	3
	P25MCS204T	Software Engineering	4	4	30	70	100	3
Practicum Course	P25MCS205P	Advance Data Structures Lab	2	4	15	35	50	3
	P25MCS206P	Python Programming Lab	2	4	15	35	50	3
Discipline Elective Course	P25MCS211T	Advanced Operating Systems	4	4	30	70	100	3
Seminar	P25MCS201S		2				50	2
Internship	P25MCS201I		4				100	
Total Credits			**30**				**750**	

M.Sc. Computer Science

Advanced Database Systems

(Semester-I) Discipline Specific Course (DSC)

Paper Code: P25MCS101T

60 Hrs (4 Hrs /Week) — **External Marks: 70**

Credits: 4 — **Internal Marks: 30**

Exam. Time: 3 Hrs — **Total Marks:100**

Note: The examiner is required to set nine questions in all. The first question will be compulsory consisting of seven short questions covering the entire syllabus consisting of 2 marks each. In addition to this, eight more questions (each question may be of 2-3 parts) will be set consisting of two questions from each unit. The student/candidate is required to attempt five questions in all selecting one question from each unit including compulsory Question No. 1. All questions will carry equal marks.

Course Objective: The objective of this course is to provide students with knowledge of database concepts, architecture, query languages, normalization techniques, and advanced datab ase models essential for designing and managing modern database systems.

Unit I

Database System Concepts and Architecture: Three-Schema Architecture and Data Independence. Entity Relationship Model: Entity Types, Entity Sets, Attributes and Keys, Relationship Types and Instances, ER Diagrams, Naming Conventions and Design Issues. Relational Model Constraints. Enhanced Entity Relat ionship Model: Subclasses, Superclasses, Inheritance, Specialization and Generalization.

Unit II

SQL: Data Definition and Data Types, DDL, DML, and DCL, Join Operations, Views and Queries in SQL, Specifying Constraints and Indexes in SQL. PL/SQL: Architecture of PL/SQL, Basic Elements of PL/SQL, PL/SQL Transactions, Cursors and Triggers. Relational Algebra: Unary and Binary Relational Operations.

Unit III

Functional Dependencies, Normal Forms Based on Primary Keys (1NF, 2NF, 3NF, BCNF), Multivalued Dependencies, 4NF, Join Dependencies, 5NF, Domain Key Normal Form. Query Processing and Optimization. Introduction to Transaction Processing and Desirable Properties of Transactions. Concurrency Control Techniques. Database Backup and Recovery.

Unit IV

Overview of Object-Oriented Database Model. Databases for Advanced Applications: Architecture for Parallel and Distributed Databases. Active Database Concept and Triggers. Temporal Database Concepts. Spatial and Multimedia Databases. Geographical Information Systems. Mobile Databases. Web Databases. XML Schema and XML Query.

Text and Reference Books:

1. Abraham Silberschatz, Henry F. Korth, S. Sudarshan, Database System Concepts, McGraw Hill.
2. Ramez Elmasri, Shamkant B. Navathe, Fundamentals of Database Systems, Pearson Education.
3. C.J. Date, An Introduction to Database Systems, Addison Wesley. Ian Sommerville,
4. Raghu Ramakrishnan, Johannes Gehrke, Database Management Systems, McGraw Hill.
5. Peter Rob, Carlos Coronel, Database Systems: Design, Implementation, and Management, Cengage Learning.

Course Outcomes: After completing this Course, the learner will be able to:

CO1: Recall fundamental concepts of database systems, architectures, and models.
CO2: Explain SQL, PL/SQL, relational algebra operations, and query formulation.
CO3: Apply normalization techniques and transaction concepts to design reliable databases.
CO4: Analyze query optimization, concurrency, and recovery techniques for database management.
CO5. Evaluate advanced database models and technologies for modern applications.

M.Sc. Computer Science

Linux and Shell Programming

(Semester-I) Discipline Specific Course (DSC)

Paper Code: P25MCS102T
60 Hrs (4 Hrs /Week) **External Marks: 70**
Credits: 4 **Internal Marks: 30**
Exam. Time: 3 Hrs **Total Marks:100**

Note: The examiner is required to set nine questions in all. The first question will be compulsory consisting of seven short questions covering the entire syllabus consisting of 2 marks each. In addition to this, eight more questions (each question may be of 2-3 parts) will be set consisting of two questions from each unit. The student/candidate is required to attempt five questions in all selecting one question from each unit including compulsory Question No. 1. All questions will carry equal marks.

Course Objective: The objective of this course is to provide fundamental knowledge and hands -on skills in Linux operating system concepts, file systems, commands, process management, and shell programming to prepare students for system level programming and administration.

Unit I

Introduction to Linux: History of Linux, Linux distributions, Features of Linux, Linux architecture, Accessing Linux system, Starting and shutting down system. Linux File System: Introduction to files, Linux file types, File system structure: boot block, super block, inode block, data block.

Unit II

Linux Commands: General-purpose commands, Directory-oriented commands, file-oriented commands file access permissions: chmod, umask, chgrp. Process Management: Basics, Process states and transitions, Foreground and background jobs, Process scheduling: cron, crontab, at.

Unit III

System Administration: Booting and shutting down process, Creating, mounting and unmounting file systems, Managing user accounts: creating, modifying, deleting users. Networking Tools: ping, nslookup, netstat, ftp. Development Tools: make command and makefiles, Debugging with gdb.

Unit IV

Pipes and Filters: Connecting processes with pipes, Redirecting input and output, Filters: sort, grep, uniq, cut, paste. Shell Programming: Shell types, Shell variables and exporting, Conditional statements, Looping, Command line arguments, writing shell scripts to automate tasks.

Text and Reference Books:

1. Yashwant Kanetkar, Unix & Shell Programming, BPB Publications.
2. Richard Petersen, The Complete Reference – Linux, McGraw-Hill.
3. M.G. Venkateshmurthy, Introduction to Unix & Shell Programming, Pearson Education.
4. Stephen Prata, Advanced UNIX – A Programmer’s Guide, SAMS Publication.
5. Sumitabha Das, Your Unix - The Ultimate Guide, Tata McGraw-Hill.

Course Outcomes: After completing this Course, the learner will be able to:

CO1: Recall and define the fundamental concepts, features, architecture, and basic components of Linux operating system.
CO2: Explain file system structure, essential Linux commands, and process management concepts.
CO3: Apply Linux commands and shell scripting to manage files, processes, and user accounts in real-world scenarios.
CO4: Analyze system operations, networking tools, and performance aspects for administrative tasks.
CO5. Evaluate the effectiveness of shell scripts, scheduling utilities, and security practices in Linux system management.

M.Sc. Computer Science

Web Engineering

(Semester-I) Discipline Specific Course (DSC)

Paper Code: P25MCS103T
60 Hrs (4 Hrs /Week) **External Marks: 70**
Credits: 4 **Internal Marks: 30**
Exam. Time: 3 Hrs **Total Marks:100**

Note: The examiner is required to set nine questions in all. The first question will be compulsory consisting of seven short questions covering the entire syllabus consisting of 2 marks each. In addition to this, eight more questions (each question may be of 2-3 parts) will be set consisting of two questions from each unit. The student/candidate is required to attempt five questions in all selecting one question from each unit including compulsory Question No. 1. All questions will carry equal marks.

Course Objective: The objective of this course is to provide students with a comprehensive understanding of web development technologies, including client-side and server-side scripting, database integration, asynchronous using AJAX, and web optimization techniques. The course aims to equip students with practical skills to design, develop, and deploy responsive, interactive, and dynamic web applications while emphasizing security, usability, and performance.

Unit I

Introduction: Web browsers and its functions, web optimizations; Static page design; designing static web pages with HTML5.0-HTML basic, multimedia, Graphics, Form tags, CSS 2.0 concept and its properties & CSS 3.0 properties i.e. borders, backgrounds, fonts, text effects, Buffering, Weblog, Web Cache Poisoning.

Unit II

JavaScript: Document Object Model (DOM), Obtaining user inputs, memory concepts, Operators, Control Structures, Looping constructs, break, continue statements, Programmer defined functions, Scoping rules, Recursion and iteration, Array declaration and allocation, passing arrays to function, Objects: String, Date, Boolean, Window, document; using cookies, form validation in JavaScript, Handling Events Using JavaScript.

Unit III

PHP: Installing and Configuring MySQL and PHP, Basic Security Guidelines, Variables, Data Types, Operators and Expressions, Constants, Flow Control Functions; Switching Flow, Loops, Code Blocks and Browser Output, Objects, Strings Processing, Form processing, Connecting to database, cookies, Session, dynamic contents.

Unit IV

Introduction to AJAX: Exploring different web technologies, Creating a simple AJAX application, Interacting with the Web Server Using the XMLHttpRequest Object, Create an XMLHttpRequest Object, Interact with the Web Server. Differentiating AJAX and Non-AJAX application.

Text and Reference Books:

1. Kogent Learning, Web Technologies: HTML, JavaScript, PHP, Java, JSP, XML, AJAX – Black Book, Wiley India Pvt. Ltd Ramez Elmasri, Shamkant B. Navathe, Fundamentals of Database Systems, Pearson Education.
2. Deitel H.M., Deitel P.J., Internet & World Wide Web: How to program, Pearson Education.
3. Boronczyk, Naramore, Beginning PHP, Apache, MySQL Web Development, Wiley India Pvt. Ltd.
4. Thomas Powell, Ajax: The Complete Reference.
5. Maro Fischer, Website Boosting: Search Engine, Optimization, Usability, Website Marketing, Firewall Media, New Delhi.

Course Outcomes: After completing this Course, the learner will be able to:

CO1: Recall the basic structure and functioning of web browsers, web optimizations, static web design principles, and CSS properties.
CO2: Explain JavaScript programming concepts including DOM manipulation, control structures, user input handling, form validation, and event-driven programming.
CO3: Apply PHP scripting for server-side programming, form handling, database connectivity, and dynamic content generation.
CO4: Analyze the architecture of AJAX-based applications and differentiate between synchronous and asynchronous web communications.
CO5. Evaluate the performance and interactivity of web applications using modern client-server technologies and optimization techniques.

M.Sc. Computer Science

Computer Networks

(Semester-I) Discipline Specific Course (DSC)

Paper Code: P25MCS104T
60 Hrs (4 Hrs /Week) **External Marks: 70**
Credits: 4 **Internal Marks: 30**
Exam. Time: 3 Hrs **Total Marks: 100**

Note: The examiner is required to set nine questions in all. The first question will be compulsory consisting of seven short questions covering the entire syllabus consisting of 2 marks each. In addition to this, eight more questions (each question may be of 2-3 parts) will be set consisting of two questions from each unit. The student/candidate is required to attempt five questions in all selecting one question from each unit including compulsory Question No. 1. All questions will carry equal marks.

Course Objective: The objective of this course is to gain a fundamental understanding of computer networks, including their design, protocols, and operation. This will enable students to analyze network communication processes, troubleshoot network issues, and configure basic network devices.

Unit I

Introduction to Computer Network: Types of Networks, Network Topologies, OSI and TCP/IP Reference Models;
Data Communications Concepts: Digital Vs. Analog communication; Parallel and Serial Communication; Synchronous, Asynchronous and Isochronous Communication; Communication modes: simplex, half duplex, full duplex; Multiplexing;
Transmission media: Wired-Twisted pair, Coaxial cable, Optical Fiber, Wireless transmission: Terrestrial, Microwave, Satellite, and Infrared.

Unit II

Communication Switching Techniques: Circuit Switching, Message Switching, Packet Switching.
Data Link Layer Fundamentals: Framing, Error Detection and Error Correction, Flow Control protocols, Media Access Protocols: ALOHA, Carrier Sense Multiple Access (CSMA), CSMA with Collision Detection (CSMA/CD), Token Ring, Token Bus.

Unit III

High-Speed LAN: Standard Ethernet, Fast Ethernet, Gigabit Ethernet, 10G; Wireless LANs: IEEE 802.11, Bluetooth.
Network Layer: IP Addressing and Routing, Network Layer Protocols: IPv4 (Header Format and Services) ARP, ICMP (Error Reporting and Query message); IPv6 (Header Format and Addressing).

Unit IV

Transport Layer: Process-to-Process Delivery: UDP, TCP; Application Layer: Domain Name System (DNS); SMTP; HTTP; WWW.
Network Security: Security Goals; Attacks; Cryptography; Confidentiality: Symmetric-Key and Asymmetric – Key Ciphers; Message Integrity & Authentication; Digital Signature; Certificates; IPSec; Firewalls; SSL.

Text and Reference Books:

1. Andrew S. Tanenbaum, Computer Networks, PHI.
2. Behrouz A Forouzan, Data Communications and Networking, Mc-Graw Hill Education.
3. Michael A. Gallo, William M. Hancock, Computer Communications and Networking Technologies – CENGAGE learning.
4. William Stallings, Data and Computer Communications, PHI.

Course Outcomes: After completing this Course, the learner will be able to:

CO1: Identify: basic components and functions of computer networks.
CO2: Understand: principles of data communication, network models and how different layers interact
CO3: Apply: appropriate networking concepts and protocols to solve configuration and troubleshooting problems.
CO4: Analyze and compare various routing algorithms congestion control techniques and network architectures.
CO5: Design simple network architecture based on user requirements

M.Sc. Computer Science

Linux and Shell Programming Lab
(Semester-I) Practicum Course (DSC)

Paper Code: P25MCS105P
60 Hrs (4 Hrs /Week) — **External Marks: 35**
Credits: 2 — **Internal Marks: 15**
Exam.Time: 3 Hrs — **Total Marks: 50**

Note: An internal practical examination is conducted by the course coordinator. The end-of-semester practical examination is conducted jointly by external and internal examiners. The external examiner is appointed by the COE of the university from the panel of examiners approved by BOSR of the Department of Computer Science and Engineering, GJUS&T Hisar, and the internal examiner is appointed by the Chairperson of the Department.

Course Objective: The objective of this course is to provide practical knowledge and skills in Linux commands and shell scripting to solve computational problems, automate tasks, and manage system-level operations efficiently.

List of Laboratory Assignments:

1. Basic Linux command
2. Basic Shell Programming (Fibonacci Series generation, Factorial of a given number, Checking for Armstrong number)
3. Designing an Arithmetic calculator
4. Generation of Multiplication table
5. Base Conversion (Decimal to Binary, Binary to Decimal)
6. Finding the information about the Login name and File name.
7. Write a shell script to exchange the contents of two variables.
8. Write a shell script, which accepts three subject marks scored by a student and declare the result.
9. Write a shell script program to find area of a square, rectangle, circle and triangle.
10. Write a shell script to print integer numbers from 1 to 20.

Students are advised to do laboratory/practical practice not limited to above set of experiments: The lab assignments are evenly spread over the semester. Every student is required to prepare a file of laboratory experiments done.

Course Outcomes:
After completing this course, the learner will be able to:
CO1: Recall and execute basic Linux commands for file, directory, and process operations.
CO2: Explain and demonstrate simple shell scripts to perform arithmetic operations and basic logic.
CO3: Apply shell scripting techniques to develop programs for number generation, base conversion, and user interaction.
CO4: Analyze and design shell scripts to automate systemtasks such as login information, variable manipulation, and decision making.
CO5: Evaluate the efficiency and correctness of shell scripts for system management and automation.

M.Sc. Computer Science
Web Engineering Lab
(Semester-I) Practicum Course (DSC)

Paper Code: P25MCS106P
60 Hrs (4 Hrs /Week) **External Marks: 35**
Credits: 2 **Internal Marks: 15**
Exam.Time: 3 Hrs **Total Marks: 50**

Note: An internal practical examination is conducted by the course coordinator. The end-of-semester practical examination is conducted jointly by external and internal examiners. The external examiner is appointed by the COE of the university from the panel of examiners approved by BOSR of the Department of Computer Science and Engineering, GJUS&T Hisar, and the internal examiner is appointed by the Chairperson of the Department.

Course Objective: The objective of this course is to provide practical knowledge and hands-on experience in designing and developing dynamic, interactive, and responsive web applications using HTML, CSS, JavaScript, PHP, and AJAX. The course emphasizes client-server interaction, form processing, and real-time web technologies.

List of Laboratory Assignments:

1. Design a basic static web page using HTML5 with multimedia and form elements.
2. Create a personal web page applying CSS3 styling including borders, backgrounds, and text effects.
3. Develop a responsive web page using CSS media queries for mobile and desktop screens.
4. Implement JavaScript functions for form validation and basic arithmetic calculations.
5. Design a JavaScript-based interactive web page using event handling and DOM manipulation.
6. Develop a dynamic registration form using PHP with proper form processing and data validation.
7. Implement a user login system using PHP and MySQL database connectivity.
8. Create a simple web application using AJAX to retrieve server data without page reload.
9. Design a multi-page website with SEO optimization (appropriate use of meta tags, headers, and alt attributes).
10. Develop a client-server application using PHP and AJAX to display dynamic content based on user input.

Students are advised to do laboratory/practical practice not limited to above set of experiments: The lab assignments are evenly spread over the semester. Every student is required to prepare a file of laboratory experiments done.

Course Outcomes:
After completing this course, the learner will be able to:

CO1: Recall fundamental concepts of web page development using HTML, CSS, and JavaScript.
CO2: Explain the structure of web applications and the client-server interaction using JavaScript and PHP.
CO3: Apply front-end and back-end programming techniques to design interactive and dynamic web pages.
CO4: Analyze and develop real-time web applications using AJAX and client-server communication principles.
CO5: Evaluate the performance, usability, and responsiveness of web applications.

M.Sc. Computer Science

Design and Analysis of Algorithms

(Semester-I) Discipline Elective Course (DEC)

Paper Code: P25MCS111T
60 Hrs (4 Hrs /Week) — **External Marks: 70**
Credits: 4 — **Internal Marks: 30**
Exam. Time: 3 Hrs — **Total Marks:100**

Note: The examiner is required to set nine questions in all. The first question will be compulsory consisting of seven short questions covering the entire syllabus consisting of 2 marks each. In addition to this, eight more questions (each question may be of 2-3 parts) will be set consisting of two questions from each unit. The student/candidate is required to attempt five questions in all selecting one question from each unit including compulsory Question No. 1. All questions will carry equal marks.

Course Objective: The objective of this course is to provide students with comprehensive knowledge and skills in designing and analyzing algorithms. The course emphasizes problem-solving techniques, algorithmic strategies such as divide and conquer, dynamic programming, greedy methods, graph algorithms, and an introduction to computational complexity and NP-completeness.

Unit I

Introduction to Algorithms: Role of algorithms in computing, analyzing algorithms, designing algorithms, asymptotic notations. Divide and Conquer: Solving recurrence equations: Back substitution method, Recursion tree method, Master's theorem. Probabilistic Analysis and Randomized Algorithms: The hiring problem, Indicator random variables, Randomized algorithms, Probabilistic analysis and further uses of indicator random variables.

Unit II

Trees: Red-black trees and Splay trees. Dynamic Programming: Elements of dynamic programming, Matrix chain multiplication, Longest common subsequence, Optimal binary search trees. Greedy Techniques: Elements of greedy techniques, Activity selection problem, Huffman codes, Knapsack problem.

Unit III

Graph Algorithms: Topological sort, Strongly connected components, Single source shortest path: Analysis of Dijkstra's Algorithm, Limitations of Dijkstra's Algorithm, Negative weight cycle, Bellman-Ford algorithm. All Pairs Shortest Path: Relation of shortest path and matrix multiplication, Analysis of Floyd Warshall algorithm. Maximum Flow: Flow network, Ford-Fulkerson method. Strings: Storage of strings, Naive string-matching algorithm, Rabin-Karp algorithm, String matching with finite automata, Knuth-Morris-Pratt algorithm.

Unit IV

Computational Geometry: Line-segment properties, Convex hull, Closest pair of points. Computational Complexity: Notion of polynomial time algorithms, Complexity classes: P, NP, NP-Hard and NP-Complete, Polynomial time verification, Reducibility, NP-Completeness, Examples of NP-Complete and NP-Hard problems: Traveling Salesman Problem, Knapsack, Bin Packing, Satisfiability, Vertex Cover, Clique, Independent Set.

Text and Reference Books:

1. Cormen, Leiserson, Rivest, Introduction to Algorithms, PHI India.
2. Neapolitan R., Foundations of Algorithms, Jones and Bartlett Learning.
3. Cooper A., Computability Theory, Chapman and Hall/ CRC Press.
4. A.V. Aho, J.E. Hopcroft, and J.D. Ullman, The Design and Analysis of Computer Algorithms, Pearson Education India.
5. Anany Levitin, Introduction to the Design and Analysis of Algorithms, Pearson Education.
6. R.C.T Lee, S.S. Tseng, R.C. Chang, Y.T. Tsai, Introduction to Design and Analysis of Algorithms: A Strategic Approach, Tata McGraw Hill.
7. Steven Skiena, The Algorithm Design Manual, Springer India.

Course Outcomes: After completing this Course, the learner will be able to:

CO1: Recall the fundamental concepts of algorithm design, asymptotic notations, and basic algorithm analysis techniques.
CO2: Explain various algorithmic strategies such as divide and conquer, dynamic programming, and greedy approaches for solving computational problems.
CO3: Apply appropriate data structures and algorithms to solve graph, string, and optimization problems.
CO4: Analyze the efficiency of graph algorithms, string matching algorithms, and maximum flow problems to determine their computational complexity.
CO5: Evaluate the complexity classes, NP-completeness, and hardness of problems by applying reduction techniques and identifying computational limitations.

***Value Added Course Offered by the Department of Computer Science & Engineering**

M.Sc. Computer Science

IoT in Smart Cities and Automation

(Semester-I) Value-Added Course (VAC)

Paper Code: U25VAC110T

30 Hrs (2 Hrs /Week)	**External Marks: 35**
Credits:2	**Internal Marks: 15**
Exam Time: 2 Hrs	**Total Marks: 50**

Note: The examiner is required to set five questions in all. The first question will be compulsory consisting of five short questions covering the entire syllabus consisting of 3 marks each. In addition to this, four more questions (each question may be of 2 parts) will be set consisting of two questions from each unit. The student/candidate is required to attempt three questions in all selecting one question from each unit consisting of 10 marks each including compulsory Question No. 1.

Course Objective: The objective of this course is to introduce students to the application of Internet of Things (IoT) in the development of smart cities and automation systems. The course aims to familiarize students with the integration of IoT technologies in areas like smart transportation, energy management, waste management, public safety, home automation, and industrial control. It covers IoT components, network protocols, sensor data processing, and cloud connectivity for building intelligent, connected urban and automated environments.

Unit I

Introduction to IoT and Smart Cities: Definition and scope of IoT, evolution and architecture of IoT systems, key components – sensors, actuators, controllers, gateways, cloud platforms. Introduction to Smart Cities – characteristics, technologies involved, and objectives.

Smart Infrastructure Applications: Use of IoT in smart lighting, smart waste management, environmental monitoring, energy-efficient buildings, and intelligent transport systems. Overview of sensor modules used in these domains. Introduction to microcontrollers (Arduino, NodeMCU/ESP32) and basic interfacing with sensors.

Unit II

Automation and Communication in IoT Systems: Automation in smart homes (smart appliances, surveillance), smart industries (remote monitoring, predictive maintenance), and public safety systems. IoT communication protocols – MQTT, CoAP, HTTP; wireless technologies – Zigbee, LoRa, Wi-Fi, Bluetooth.

Text and Reference Books:

1. Arshdeep Bahga and Vijay Madisetti, "Internet of Things: A Hands-On Approach", Universities Press.
2. Maciej Kranz, "Building the Internet of Things", Wiley.
3. H. Chaouchi, "Smart Cities: Foundations, Principles, and Applications", Wiley.
4. Adrian McEwen and Hakim Cassimally, "Designing the Internet of Things", Wiley.
5. N. Kumar, P. Tiwari, "IoT and Smart Cities: Concepts, Technologies and Applications", CRC Press.

Course Outcomes: After completing this Course, the learner will be able to:

CO1: Recall the fundamental architecture, components, and objectives of IoT systems and smart cities.
CO2: Explain the role of sensors, actuators, and microcontrollers in implementing smart infrastructure and automation.
CO3: Apply IoT communication protocols and wireless technologies in real-world smart applications.
CO4: Analyze the integration of cloud platforms for data monitoring, control, and visualization in IoT systems.
CO5: Evaluate the challenges of security, privacy, and scalability in deploying IoT-based smart and automated environments.

M.Sc. Computer Science
Advance Data Structures
(Semester-II) Discipline Specific Course (DSC)

Paper Code: P25MCS201T
60 Hrs (4 Hrs /Week) **External Marks: 70**
Credits: 4 **Internal Marks: 30**
Exam. Time: 3 Hrs **Total Marks: 100**

Note: The examiner is required to set nine questions in all. The first question will be compulsory consisting of seven short questions covering the entire syllabus consisting of 2 marks each. In addition to this, eight more questions (each question may be of 2-3 parts) will be set consisting of two questions from each unit. The student/candidate is required to attempt five questions in all selecting one question from each unit including compulsory Question No. 1. All questions will carry equal marks.

Course Objective: The course aims to empower students with a comprehensive skill set in data structures, fostering both theoretical understanding and practical application, preparing them for analyzing and applying algorithms, design and apply efficient algorithms using data structures, understand the significance of efficient data and file organization, develop coding proficiency in data structure applications.

Unit I

Introduction to Data Structures: Classification of Data Structures, Arrays
Stacks & Queues: Representation of Stacks, Stack Operations, Applications, Queues, Operations on Queues, Circular Queues, Dequeue, Priority Queues, Applications.
Introduction to Algorithms: Role of algorithms in computing, Complexity of algorithms, analyzing algorithms, designing algorithms, asymptotic notations.

Unit II

Linked List: List concepts, Definition and List as ADT, Static and Dynamic List Structure and implementation, Types of linked list, Operations on Linked List, Singly linked list, Circular Linked List, Doubly Linked List, Doubly Circular Linked List, Inserting, traversing and deleting nodes at beginning, end and specified positions in these linked lists, Linked implementation of a stack and queue in singly linked list.
Tree: Definition and basic terminologies of tree, Binary Tree: Introduction, Types of Binary Tree, Level and depth, height-balance tree(AVL), Operations in Binary Search Tree (BST): Insertion, Deletion, Searching, Tree Traversal: Pre-order traversal , In-order traversal (sorted list of Nodes), Post-order traversal, Applications of Binary Tree (Huffman tree, expression tree).

Unit III

Divide and Conquer: Solving recurrence equations: back substitution method, recursion tree method, master's theorem. Analysis of heap sort and quick sort; Counting sort, Radix sort, Bucket sort, Lower bounds for sorting.
Dynamic Programming (DP): Elements of DP, Matrix chain multiplication, Longest common subsequence, optimal binary search trees.

Unit IV

Greedy Techniques (GT): Elements of GT, Activity selection problem, Huffman codes, Knapsack Problem.
Graph Algorithms: Negative weight cycle, Bellman-Ford algorithm. All Pairs Shortest Path. Maximum Flow: Flow network, Ford-Fulkerson method.
Strings: Storage of strings, Naive string-matching algorithm, Rabin-Karp algorithm, String matching with finite automata, Knuth-Morris-Pratt algorithm

Text and Reference Books:

1. R.S. Salaria, Data Structures & Algorithms, Khanna Book Publishing Co. (P) Ltd.
2. P.S. Deshpande and O.G. Kakde, C & Data Structures, Wiley Dreamtech India Private Limited.
3. T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein, Introduction to Algorithms, Third Edition, MIT

Press
4. G.A.V Pai, Data Structures and Algorithms, McGraw-Hill.
5. Neapolitan R., Foundations of Algorithms, Jones and Bartlett Learning.
6. Seymour Lipschutz, Data Structures, McGraw-Hill, Schaum's Outlines, New Delhi.
7. Alfred V. Aho, John E. Hopcroft, Jeffrey D. Ullman, Data Structures and Algorithms, Pearson Education.

Course Outcomes: After completing this Course, the learner will be able to:

CO1: Identify fundamental concepts and types of data structures.
CO2: Understand operations, advantages, limitations and use cases of different data structures.
CO3: Apply data structures for solving basic computational problems and implementing algorithms.
CO4: Analyze and compare time and space complexities of different data structures and associated algorithms.
CO5. Design efficient solutions using appropriate data structures for real-world applications and problem solving.

M.Sc. Computer Science
Python Programming
(Semester-II) Discipline Specific Course (DSC)

Paper Code: P25MCS202T
60 Hrs(4Hrs /Week) **ExternalMarks:70**
Credits:4 **InternalMarks: 30**
Exam.Time:3Hrs **TotalMarks:100**

Note: The examiner is required to set nine questions in all. The first question will be compulsory consisting of seven short questions covering the entire syllabus consisting of 2 marks each. In addition to this, eight more questions (each question may be of 2-3 parts) will be set consisting of two questions from each unit. The student/candidate is required to attempt five questions in all selecting one question from each unit including compulsory Question No. 1. All questions will carry equal marks.

Course Objective: The primary objective of this course is to provide students with a strong foundation in Python programming. It aims to develop the ability to write efficient, structured, and error-free Python programs by covering fundamental and advanced programming concepts. This includes understanding Python syntax, data structures, control flow, functions, object-oriented programming, exception handling, file operations, and the use of essential libraries such as NumPy, Pandas, and Matplotlib. By the end of the course, students will be equipped to solve real-world problems and develop basic applications using Python.

Unit I

History & Features of Python, Installation & Execution, Python IDEs (IDLE, PyCharm, Jupyter), Python syntax, Variables, Data Types, Type conversion, Operators, Expressions, Input/Output functions, Decision Making: if, if-else, if-elif, Loops: for, while, break, continue, pass

Unit II

Lists, Tuples, Sets, Dictionaries, Operations & Methods on data structures, Functions, Defining and calling functions, Arguments (default, keyword, variable-length), Return values, Scope and Lifetime, Lambda functions, Recursion, Importing standard & user-defined modules math, random, datetime, etc.

Unit III

Classes and Objects, Constructors (_init_), Instance and Class Variables, Types of Methods: Instance, Class, Static, Inheritance (Single, Multiple, Multilevel), Method Overriding, super() function

Unit IV

Exception Handling: try, except, else, finally, Built-in Exceptions, User-defined Exceptions, File Handling: Opening, Reading, Writing files, Working with text and binary files, with statement, Python Libraries: Introduction to NumPy, Pandas, Matplotlib (basic usage)

Text and Reference Books:

1. **"**Learning Python" **by** Mark Lutz
2. "Python Programming: Using Problem Solving Approach" **by** Reema Thareja
3. "Python: The Complete Reference" **by** Martin C. Brown **– McGraw-Hill**

Course Outcomes: After completing this Course, the learner will be able to:

CO1: Understand and apply the fundamentals of Python programming, including syntax, variables, data types, operators, and control structures.
CO2: Design and implement Python programs using loops, conditional statements, functions, and modular coding techniques.
CO3: Utilize standard Python libraries such as math, random, datetime, and external libraries like **NumPy**, **Pandas**, and **Matplotlib** for basic data processing and visualization.
CO4: Develop problem-solving skills by writing and debugging Python programs for real-world applications.
CO5. Demonstrate the ability to work with modular code and reusable components, including user-defined functions and modules.

M.Sc. Computer Science
Artificial Intelligence
(Semester-II) Discipline Specific Course (DSC)

Paper Code: P25MCS203T
60 Hrs (4 Hrs /Week) — **External Marks: 70**
Credits: 4 — **Internal Marks: 30**
Exam. Time: 3 Hrs — **Total Marks: 100**

Note: The examiner is required to set nine questions in all. The first question will be compulsory consisting of seven short questions covering the entire syllabus consisting of 2 marks each. In addition to this, eight more questions (each question may be of 2-3 parts) will be set consisting of two questions from each unit. The student/candidate is required to attempt five questions in all selecting one question from each unit including compulsory Question No. 1. All questions will carry equal marks.

Course Objective: The objective of this course is to provide the in-depth coverage of basic principles of Artificial Intelligence and its sub fields; apply various AI based algorithms to solve real life problems; It provides the usage of different techniques for their exquisite utilization. It provides the concept of expert systems. It provides the knowledge about the Machine Learning techniques.

Unit I

Introduction: Artificial Intelligence (AI)- Introduction, The history of AI, Representation of Knowledge, Knowledge-Based Systems, State Space Search. Production Systems: Problem Characteristics, Types of Production Systems. Intelligent Agents and Environments, the concept of rationality, the nature of environments, structure of agents, problem-solving agents, problem formulation, Knowledge-based agents

Unit II

Solving Problems by Searching- Problem Solving Agents, Searching for Solutions- Infrastructure for search algorithms, Measuring problem-solving performance.
Uninformed Search Strategies-Breadth-first search, Uniform-cost search, Depth-first search, Depth-limited search, Iterative deepening depth-first search, Bidirectional search, Comparing uninformed search strategies.

Unit III

Informed (Heuristic) Search Strategies- Generate & test, Hill Climbing, Best First Search, A* and AO* Algorithm, Constraint Satisfaction, Means-Ends Analysis. Game playing: Minimax Search, Alpha-Beta Cutoffs, Waiting for Quiescence, Comparing informed search strategies.
Knowledge Representation Propositional Logic: Representation, Inference, Reasoning Patterns, Resolution, Forward and Backward Chaining. First-order Logic: Representation, Inference, Reasoning Patterns, Resolution, Forward and Backward Chaining.

Unit IV

Architecture of expert systems, Steps to build Expert Systems - Role of expert systems – Knowledge Acquisition – Meta knowledge, Heuristics.
Introduction to Machine Learning, Supervised Learning: Classification: Decision Trees, k-NN, Regression: Linear and Logistic Regression; Unsupervised Learning, Clustering: k-Means, Hierarchical Clustering, Neural Networks: Perceptron, Multi-Layer Perceptron (MLP)Backpropagation and Activation Functions

Text and Reference Books:

1. Stuart Russell and Peter Norvig, Artificial Intelligence: A Modern Approach, 3rd Edition, Pearson Education.
2. Elaine Rich, Kevin Knight, and Shivashankar B. Nair, Artificial Intelligence, 3rd Edition, Tata McGraw-Hill.
3. Dan W. Patterson, Introduction to Artificial Intelligence and Expert Systems, Pearson Education.
4. Tom M. Mitchell, Machine Learning, McGraw-Hill.
5. Christopher M. Bishop, Pattern Recognition and Machine Learning, Springer.

Course Outcomes: After completing this Course, the learner will be able to:
CO1: Identify basic terminology related to artificial intelligence.
CO2: Understand the concepts of reasoning under uncertainty.
CO3: Apply various AI based algorithms to solve real life problems.
CO4: Analyze and compare the usage of different search techniques for their exquisite utilization.
CO5. Implement AI techniques in the aspects of computing.

M.Sc. Computer Science
Software Engineering
(Semester-II) Discipline Specific Course (DSC)

Paper Code: P25MCS204T
60 Hrs (4 Hrs /Week) **External Marks: 70**
Credits: 4 **Internal Marks: 30**
Exam. Time: 3 Hrs **Total Marks:100**

Note: The examiner is required to set nine questions in all. The first question will be compulsory consisting of seven short questions covering the entire syllabus consisting of 2 marks each. In addition to this, eight more questions (each question may be of 2-3 parts) will be set consisting of two questions from each unit. The student/candidate is required to attempt five questions in all selecting one question from each unit including compulsory Question No. 1. All questions will carry equal marks.

Course Objective: The objective of this course is to provide students with a comprehensive understanding of software engineering principles, methodologies, project management techniques, and testing strategies essential for designing, developing, and maintaining high-quality software systems.

Unit I

Introduction to Software Engineering, System Engineering vs Software Engineering, Software Evolution, Software Characteristics, Components, Software Crisis – Problems and Causes, Software Feasibility, Software Process Models: V-Model, Waterfall, Iterative Enhancement, Incremental, RAD, Prototyping, Spiral, Concurrent Development, Rational Unified Process, AGILE. Challenges in Software Engineering.

Unit II

Software Project Management: Planning, Scope Management, Cost Estimation – LOC, Function Point Analysis, COCOMO, Putnam Resource Allocation Model. Project Scheduling and Resource Management: Gantt-Chart, PERT, CPM, Histogram, Team Building, Organization Charts. Project Monitoring and Risk Management, Software Configuration Management, Software Quality Assurance, Project Monitoring Techniques. Software Requirement Analysis: Structured Analysis, Object Oriented Analysis, Data Modelling, Software Requirement Specification, DFDs, Data Dictionaries, Decision Trees, Decision Tables, Structured English, ER Diagrams.

Unit III

Design and Implementation of Software: Basic Fundamentals, Design Methodology (Structured and Object Oriented), Design Approaches, User Interface Designing Tools and Techniques, Design Complexity, Monitoring and Control, Coding, Halstead's Software Science, McCabe's Cyclomatic Complexity. Software Reliability: Metrics and Specification, Fault Avoidance and Tolerance, Exception Handling, Defensive Programming, Component Based Development.

Unit IV

Software Testing: Fundamentals, Validation and Verification, White-Box and Black-Box Testing Techniques (Control Flow, Data Flow, Loop, Mutation, Load, Stress, Performance, Boundary Valu e, Equivalence Class, Decision Table, Cause Effect Graph Testing). Testing Strategies: Unit, Integration, Validation and System Testing, Alpha and Beta Testing, Debugging, Static Testing Strategies. Software Maintenance: Maintenance Characteristics, Maintainability, Maintenance Tasks and Side Effects.

Text and Reference Books:

1. K. K. Aggarwal and Yogesh Singh, Software Engineering, New Age International Private Limited; Fourth edition. Richard Petersen, The Complete Reference – Linux, McGraw-Hill.
2. R. S. Pressman, Software Engineering – A Practitioner's Approach, 6th ed., McGraw Hill Int. Ed.
3. Richard Fairley, Software Engineering Concepts, McGraw Hill Education.

4. Pankaj Jalote, An Integrated Approach to Software Engineering, Narosa.
5. Ian Sommerville, Software Engineering, Pearson Education; Tenth edition.

Course Outcomes: After completing this Course, the learner will be able to:
CO1: Recall fundamental concepts, models, and characteristics of software engineering and software process models.
CO2: Explain project management activities, cost estimation, and requirement analysis techniques.
CO3: Apply design principles, coding standards, and software reliability practices to implement software systems.
CO4: Analyze testing techniques, strategies, and maintenance practices to ensure software quality.
CO5. Evaluate software engineering methodologies and tools for real-world software development and management.

M.Sc. Computer Science
Advance Data Structures Lab
(Semester-II) Practicum Course (DSC)

Paper Code: P25MCS205P
60 Hrs (4 Hrs /Week) — **External Marks: 35**
Credits: 2 — **Internal Marks: 15**
Exam.Time: 3 Hrs — **Total Marks: 50**

Note: An internal practical examination is conducted by the course coordinator. The end-of-semester practical examination is conducted jointly by external and internal examiners. The external examiner is appointed by the COE of the university from the panel of examiners approved by BOSR of the Department of Computer Science and Engineering, GJUS&T Hisar, and the internal examiner is appointed by the Chairperson of the Department.

Course Objective: The objective of this course is to deepen understanding of fundamental and advanced data structures. and to develop the ability to analyse, implement and apply these structures for efficient algorithm design and real- world problem solving.

List of Laboratory Assignments:

1. Write a program to declare and initialize an array, then find and print the largest and smallest elements in the array.
2. Write a program to demonstrate basic data structure operations such as creating, inserting, deleting, and displaying elements in an array.
3. Write a program to insert and delete an element at a specified position in a linear array.
4. Write a program to implement and demonstrate a sequential search on an array.
5. Write a program to implement and demonstrate a binary search on a sorted array.
6. Write a program to implement and demonstrate the Bubble sort algorithm.
7. Write a program to perform operations: push, pop, and display.
8. Write a program to implement a queue using an array and perform basic queue operations: enqueue, dequeue, and display.
9. Write a program to implement and demonstrate the Selection sort algorithm.
10. Write a program to implement and demonstrate the Insertion sort algorithm.
11. Write a program to implement and demonstrate the Quick sort algorithm.
12. Write a program to implement and demonstrate the Merge sort algorithm.
13. Write a program to implement a singly linked list and perform insertion, deletion, and traversal operations.
14. Write a program to implement a binary tree and perform in-order, pre-order, and post-order traversal using recursion.
15. Write a program to implement a binary search tree (BST) and perform insertion, deletion, and search operations.

Students are advised to do laboratory/practical practice not limited to above set of experiments: The lab assignments are evenly spread over the semester. Every student is required to prepare a file of laboratory experiments done.

Course Outcomes: After completing this course, the learner will be able to:
CO1: Recall and identify the properties of advanced data structures.
CO2: Explain the working principles, advantages, limitations of advanced data structures in various contexts.
CO3: Apply advanced data structures to solve problems like searching, sorting, indexing and memory management.
CO4: Analyze the time and space complexities of algorithms using advanced data structures under different input conditions.
CO5: Evaluate and compare different data structures to select the most appropriate one for a given problem.

M.Sc. Computer Science
Python Programming Lab
(Semester-II) Practicum Course (DSC)

Paper Code: P25MCS206P
60 Hrs (4 Hrs / Week) **External Marks: 35**
Credits: 2 **Internal Marks: 15**
Exam Time: 3 Hrs **Total Marks: 50**

Note: An internal practical examination is conducted by the course coordinator. The end-of-semester practical examination is conducted jointly by external and internal examiners. The external examiner is appointed by the COE of the university from the panel of examiners approved by BOSR of the Department of Computer Science and Engineering, GJUS&T Hisar, and the internal examiner is appointed by the Chairperson of the Department.

Course Objective: The objective of this lab course is to provide hands-on experience in Python programming by applying theoretical concepts through practical implementation. It aims to help students develop problem-solving skills, strengthen their understanding of Python syntax, data structures, functions, and object-oriented programming, and become proficient in using file handling, exception management, and essential Python libraries such as NumPy, Pandas, and Matplotlib. By the end of the course, learners will be capable of developing basic applications and writing efficient, modular Python code to solve real-world problems.

List of Laboratory Assignments:

1. Write a Python program to print "Hello, World!" and user-defined input.
2. Demonstrate type conversion between different data types.
3. Write a program to use arithmetic, relational, and logical operators.
4. Implement decision-making using `if`, `if-else`, and `if-elif` constructs.
5. Develop programs using `for` and `while` loops, including `break`, `continue`, and `pass` statements.
6. Create and manipulate lists, tuples, sets, and dictionaries.
7. Perform operations such as append, insert, delete, sort on a list.
8. Write functions using default, keyword, and variable-length arguments.
9. Implement a recursive function (e.g., factorial, Fibonacci series).
10. Use built-in modules like `math`, `random`, and `datetime` with examples.
11. Create a class and demonstrate instance and class variables.
12. Write a Python program with a constructor (`__init__`) and method definitions.
13. Implement single, multilevel, and multiple inheritance.
14. Demonstrate method overriding and use of `super()` function.
15. Create and use instance, class, and static methods.
16. Write a program to handle exceptions using `try`, `except`, `else`, and `finally`.
17. Create a user-defined exception class and use it in a program.
18. Read from and write to text files using `with open()` context manager.
19. Perform basic operations on binary files (read/write).
20. Plot basic graphs using Matplotlib (line, bar, pie charts).
21. Use NumPy to create and manipulate arrays.
22. Load data into Pandas DataFrame and perform basic operations like filtering and sorting.

Students are given ten or more laboratory assignments with soft and hard deadlines. The lab assignments are evenly spread over the semester. Every student is required to prepare a file of laboratory experiments done.

Course Outcomes: After completing this course, the learner will be able to:
CO1: Develop and execute Python programs using basic syntax, operators, and control structures.
CO2: Implement programs using Python's built-in data structures such as lists, tuples, sets, and dictionaries.
CO3: Design and use functions with various types of arguments, recursion, and built-in modules.
CO4: Apply object-oriented programming concepts including classes, inheritance, and polymorphism in Python.
CO5: Handle exceptions and perform file input/output operations using text and binary files.

M.Sc. Computer Science
Advanced Operating Systems
(Semester-II) Discipline Elective Course (DEC)

Paper Code: P25MCS211T
60 Hrs (4 Hrs /Week) **External Marks: 70**
Credits: 4 **Internal Marks: 30**
Exam. Time: 3 Hrs **Total Marks:100**

Note: The examiner is required to set nine questions in all. The first question will be compulsory consisting of seven short questions covering the entire syllabus consisting of 2 marks each. In addition to this, eight more questions (each question may be of 2-3 parts) will be set consisting of two questions from each unit. The student/candidate is required to attempt five questions in all selecting one question from each unit including compulsory Question No. 1. All questions will carry equal marks.

Course Objective: The objective of this course is to provide in-depth knowledge of advanced operating system concepts including process synchronization, memory management, distributed systems, protection mechanisms, and fault-tolerant systems. The course draws on foundational and advanced material to prepare students for research and development in system-level software.

Unit I

Introductory Concepts: Operating system functions, structure, and design approaches. Types of operating systems: Batch, Multi-programming, Time-sharing, Desktop, Multi-processor, Distributed, Clustered, Real-time, Handheld, and Open-source OS. Operating System Structures: System components, OS services, system calls, and system programs.

CPU Scheduling: Process concepts, states and transitions, scheduling criteria and algorithms, multiple processor scheduling, inter-process communication.

Unit II

Concurrent Processes and Synchronization: Critical section problem, semaphores, monitors. Classical synchronization problems: Producer-consumer, Reader-writer, Dining philosophers. Deadlocks: Characterization, prevention, avoidance, detection, and recovery techniques.

Unit III

Memory Management: Swapping, Paging, Segmentation. Virtual Memory: Demand paging, Page replacement algorithms, Thrashing. Storage Management: File concepts, Access methods, Directory structures, Mounting, File sharing, Protection mechanisms. File system implementation, Directory implementation, Allocation methods, and Recovery. Disk scheduling: Criteria and algorithms.

Unit IV

Protection and Security: Protection goals, domains, access matrix. Security concepts, threats, tools, and classification. Distributed Systems: Network-based OS types, structures, and protocols. Distributed file systems: Remote file access, replication. Distributed synchronization: Mutual exclusion, concurrency control, and deadlock handling.

Text and Reference Books:

1. Mukesh Singhal and N. G. Shivaratri, “Advanced Concepts in Operating Systems”, McGrawHill, 2000
2. Abraham Silberschatz, Peter B. Galvin, G. Gagne, “Operating System Concepts”, Sixth Addison Wesley Publishing Co., 2003.
3. Andrew S. Tanenbaum, “Modern Operating Systems”, Second Edition, Addison Wesley, 2001

Course Outcomes: After completing this Course, the learner will be able to:
CO1: Recall the fundamental functions, structures, and types of operating systems.
CO2: Explain process scheduling, synchronization techniques, and deadlock handling mechanisms.
CO3: Apply memory and storage management techniques including paging, segmentation, and disk scheduling.
CO4: Analyze protection models, security strategies, and access control mechanisms.
CO5. Evaluate distributed system models, communication protocols, and file system coordination strategies.