ಬೆಂಗಳೂರು
ನಗರ ವಿಶ್ವವಿದ್ಯಾನಿಲಯ

BENGALURU
CITY UNIVERSITY

Office of the Registrar, Central College Campus, Dr. B.R. Ambedkar Veedhi, Bengaluru – 560 001.
PhNo.080-22131385, E-mail: registrarbcu@gmail.com

No.BCU/BoS/Syllabus-PG/Science/ 332 /2025-26 Date: 23.09.2025

NOTIFICATION

Sub: Syllabus for the Post Graduate Courses in the Faculty of Science– reg

Ref: 1. Recommendations of the Boards of Studies in the Faculty of Science
2. Academic Council resolution No.04 dated.22.09.2025
3. Orders of Vice-Chancellor dated. 23.09.2025

****

The Academic Council in its meeting held on 22.09.2025 has approved the syllabus prepared by different Board of Studies for the Post Graduate Courses in the Faculty of Science. Accordingly, the following CBCS Syllabus for the Semester PG Courses of Science Faculty are hereby notified for implementation effective from the academic year 2025-26.

Sl. No.	Programmes
1.	M.Sc. Chemistry – I & II Semester
2.	M.Sc. Biochemistry – I to IV Semester
3.	M.Sc. Physics – I & II Semester
4.	M.Sc. Mathematics – I to IV Semester
5.	M.Sc. Psychology– I to IV Semester
6.	M.Sc. Counselling Psychology – I to IV Semester
7.	M.Sc. Fashion & Apparel Design – I to IV Semester
8.	M.Sc. Zoology – I & II Semester
9.	M.Sc. Botany – I to IV Semester
10.	M.Sc. Computer Science – I & II Semester
11.	M.Sc. Speech Language Pathology – I to IV Semester
12.	Master of Computer Applications – I & II Semester

The detailed Syllabi for above subjects are notified in the University Website: www.bcu.ac.in for information of the concerned.

REGISTRAR

Copy to;

1. The Registrar(Evaluation), Bengaluru City University
2. The Dean, Faculty of Science, BCU.
3. The Principals of the concerned affiliated Colleges of BCU- through email.
4. The P.S. to Vice-Chancellor/Registrar/Registrar (Evaluation), BCU.
5. Office copy / Guard file / University Website: www.bcu.ac.in

ಬೆಂಗಳೂರು ನಗರ ವಿಶ್ವವಿದ್ಯಾನಿಲಯ

BENGALURU CITY UNIVERSITY

BENGALURU CITY UNIVERSITY

Syllabus of

Master of Science in Computer Science (MSc-CS)

(CBCS Scheme)

Effective from the Academic Year 2025 – 2026

Board of Studies in Computer Science for PG

1	Prof. Ramesh B Kudenatti Department of Mathematics Bengaluru City University, Bengaluru-560056	Chairman
2	Prof. Guru D S Department of Studies in Computer Science University of Mysore, Mysore-570006	Member
3	Prof. Aziz Makandar Department of Computer Science Karnataka State Akkamahadevi Women University, Jnanashakti Campus, Vijayapura-586109	Member
4	Prof. Suneetha Department of Computer Science, Karnataka State Open University, Muktha Gangothri, Mysuru-570006	Member
5	Prof. Veena R Department of MCA, Seshadripuram College, Seshadripuram, Bengaluru-560020	Member
6	Prof. Kiran Kumar M N Department of Computer Applications, BMS College of Commerce and Management, Bengaluru-560004	Member
7	Prof. Latha B Department of Computer Science Vijaya College, R V Road, Basavanagudi, Bengaluru-560004	Member
8	Prof. R Shanthi Krishna Department of Computer Applications, SSMRV College, Jayanagar, Bengaluru-560041	Member
9	Prof. Roopa HR Department of Computer Applications, Seshadripuram Institute of Commerce and Management, Seshadripuram, Bengaluru-560020	Member
10	Shri Seby Kallarakkal CEO-Nabler Web Solutions, Bengaluru-560052	Member
11	Prof Hanumanthappa M Senior Professor, Department of Computer Science Bangalore University, Bangalore	Co-opted Member
12	Shri Manjunatha Aradhya Founder-Director & CEO, ABC–Technology Training & Upskilling, Bengaluru	Co-opted Member

Details of MSc in Computer Science

1	**Name of the Course**	MSc in Computer Science
2	**Discipline Course**	Computer Science
3	**Duration of the Course**	Two (02) years
4	**Year of Implementation (Revised)**	2025-2026 (I & II Semesters) 2026-2027 (III & IV Semesters)
5	**Eligibility**	BCA/BVoc (IT)/BSc with Computer Science as one of the cognate subjects, with minimum of 50% aggregate marks, and the minimum requirement for SC/ST/Cat-I students is in accordance with BCU Rules and Regulations
6	**Intake**	As per the BCU Rules and Regulations
7	**Admission & Fees Structure**	As per the BCU Rules and Regulations
8	**Medium of Instruction**	English
9	**Attendance**	As per the BCU Rules and Regulations
10	**Internal Assessment for Theory and Practical**	There shall be two internal tests of 20% of marks each and the average of both tests, and 10% for assignment/seminar shall be considered for final computation of marks.
11	**Allocation of Marks for Practical Assessment**	Writing & execution of two programs: 30 marks Viva-voce: 05 marks
12	**Scheme of Examination and Announcement of Results**	As per the BCU Rules and Regulations

Programme Outcome (PO)

PO 1	**Apply** knowledge of computer science and mathematics to analyze and solve real-world problems.
PO 2	**Design and develop** efficient software systems and applications that meet specified user requirements.
PO 3	**Modern** programming languages, tools and emerging technologies to implement solutions.
PO 4	**Analyze and interpret** experimental data to **evaluate** system performance and draw valid conclusions.
PO 5	**Integrate** knowledge across disciplines to **adapt** to new environments and complex projects.
PO 6	**Communicate** effectively through oral and written forms with both technical and non-technical audiences.
PO 7	**Demonstrate** professional, ethical, legal and social responsibilities in computing practices.
PO 8	**Collaborate and contribute** effectively as an individual and as a member of diverse teams.
PO 9	**Engage** in lifelong learning to **upgrade** skills and knowledge on far with technological advancements.
PO 10	**Apply and evaluate** computing knowledge to **contribute** towards societal, environmental and sustainable development.

ASSESSMENT

Weightage for the Assessments (in percentage)

Type of Course	Formative Assessment	Summative Assessment
Theory	30%	70 %
Practical	30%	70 %

Detailed Structure for MSc-CS Course

Semester	Course Code	Paper Title	Teaching Hours / Week	Marks		Duration of Exam in Hours	Credits
				Exam	IA		
I	MSC101T	Computer Architecture	04	70	30	03	04
	MSC102T	Data Structures using Java	04	70	30	03	04
	MSC103T	Python Programming	04	70	30	03	04
	MSC104T	Advanced Software Engineering	04	70	30	03	04
	MSC105T	Analysis & Design of Algorithms	04	70	30	03	04
	MSC101P	Data Structures using Java Lab	04	35	15	03	02
	MSC102P	Python Programming Lab	04	35	15	03	02
	MSC101SC	Soft Core - Quantitative Techniques and Teaching Aptitude	02	35	15	1.5	02
Total Credits							**26**
II	MSC201T	Database Applications	04	70	30	03	04
	MSC202T	Advanced Java Programming	04	70	30	03	04
	MSC203T	Artificial Intelligence	04	70	30	03	04
	MSC204T	Advanced Operating Systems	04	70	30	03	04
	MSC205T	Data Communication & Networks	04	70	30	03	04
	MSC201E	a. Finite Automata and Formal Languages b. Multimedia and its Applications	04	70	30	03	04
	MSC201P	Database Applications Lab	04	35	15	03	02
	MSC202P	Advanced Java Programming Lab	04	35	15	03	02
Total Credits							**28**

SEMESTER – I

Theory	MSC101T: Computer Architecture
Teaching Hours : 04 Hours/Week	Credits: 04
Duration of Exam: 03 Hours	Maximum Marks: 100 (Exam 70 + IA 30)

Course Outcomes

COs	Description
CO1	Understand the concepts of number systems, logic circuits, data representation, and computer arithmetic operations.
CO2	Analyze the structure and organization of computers, instruction execution, CPU design, and differentiate between CISC and RISC architectures.
CO3	Apply register transfer language, micro-operations, control unit design, and input/output organization to analyze instruction-level parallelism.
CO4	Study advanced processor architectures, memory hierarchy, multiprocessor systems, and parallel computing models for performance evaluation.

UNIT I: Number Systems, Computer Arithmetic and Logic Circuits 14 Hours

Number Systems: Binary, Octal, Hexadecimal, Decimal numbers, base conversion, Binary arithmetic, Data Representation: Fixed-point, Floating-point, One's and Two's Complement, Signed numbers - Character codes: ASCII, EBCDIC, Error Detection and Correction codes.

Boolean Algebra, K-Map Simplification, Logic Gates, Universal Gates, Combinational Circuits: Half Adder, Full Adder, Multiplexer, De-multiplexer, Encoder, Decoder, Sequential Circuits: Flip-Flops, Shift Registers, Counters.

UNIT II: Computer Structure and Basic Organization 14 Hours

Structure of Computers: Computer types, Fundamental units, Von Neumann Architecture, Bus Structures, Performance- Basic Computer Organization and Design: Instruction codes, Registers, Computer Instructions - Instruction cycle, Timing and Control - Memory-reference instructions - Input-Output and Interrupt systems - Central Processing Unit: Stack organization, Instruction formats, Addressing modes, Data transfer and manipulation, CISC vs. RISC architectures.

UNIT III: Register Transfer, Control, and I/O Organization 14 Hours

Register Transfer Language (RTL), Bus operations - Memory transfers - Micro-operations: Arithmetic, Logic, and Shift operations- Arithmetic Logic Shift Unit (ALSU), Microprogrammed control: Control memory, Address sequencing, Design of control unit, Input/Output Organization: Programmed I/O, Memory-mapped I/O, Interrupt-driven I/O - Instruction Level Parallelism (ILP): Hazards and limitations.

UNIT IV: Advanced Architectures and Parallel Computing 14 Hours

Processors and Memory Hierarchy: Advanced processor architecture, Scalar & Vector processors - Memory hierarchy technology, Virtual memory systems, Cache memory organization, Shared memory organization, Sequential & weakly consistent models - Multiprocessor and Multi-core systems , Shared-memory and Distributed-memory, multiprocessors , SIMD, MIMD, PRAM, VLSI models , Multithreaded and Dataflow models - Performance of Multi-Processor systems.

Text Books:

1. Mano M Morris, Computer System Architecture, 3rd Edition, Pearson India (2019).
2. William Stallings, Computer Organization and Architecture Designing for Performance, 11 Edition, Pearson.
3. K. Kwang, Advanced Computer Architecture: Parallelism, Scalability, Programmability, Tata McGraw-Hill, 3rd Edition, 2015.

Reference Books:

1. Dezso Sima, Terence Fountain, and Peter Kacsuk , "Advanced Computer Architectures Design Space Approach" Pearson Education, 2009
2. Carl Hamacher, ZvonksVranesic, and Safwat Zaky ,"Computer Organization And Embedded Systems "6th edition , McGraw Hill, 2011.

Theory	**MSC102T: Data Structures using Java**
Teaching Hours : 04 Hours/Week	**Credits: 04**
Duration of Exam: 03 Hours	**Maximum Marks: 100 (Exam 70 + IA 30)**

Course Outcomes

COs	**Description**
CO1	Demonstrate proficiency in Java programming fundamentals, including arrays, strings, control structures, and object-oriented programming constructs to solve computational problems.
CO2	Apply advanced Java features such as inheritance, interfaces, exception handling, file handling, and Java Collections Framework to design modular and reusable programs.
CO3	Implement and analyze linear data structures such as stacks, queues, and linked lists for solving real-world applications effectively.
CO4	Apply advanced data structures such as trees, graphs, and efficient searching/sorting techniques to develop optimized algorithms for complex problem-solving.

UNIT – I: Introduction to Java **14 Hours**

Features of Java, JVM architecture, JDK and JRE, Java program structure, compilation and execution. Data types, variables, operators, expressions, type casting and control structures (decision making and looping). Arrays – one-dimensional and two-dimensional arrays, array manipulation and applications. Strings – String and StringBuffer classes, String operations. Introduction to object-oriented programming in Java – classes, objects, constructors, method overloading, static members.

UNIT – II: Java Concepts for Data Structures **14 Hours**

Inheritance – types, super and sub-classes, method overriding, dynamic method dispatch. Abstract classes and interfaces. Packages and access specifiers. Exception handling – types of exceptions, try–catch–finally, throw and throws. Input/Output in Java – byte streams, character streams, file handling. Introduction to Java Collections Framework – ArrayList, LinkedList, Stack, Queue, HashMap.

UNIT – III: Fundamentals of Data Structures and their applications 14 Hours

Introduction to Data Structures, need and applications. Stacks – implementation using arrays and linked lists, operations, applications (expression conversion, evaluation). Queues – linear queue, circular queue, double-ended queue, priority queue, implementation and applications. Linked Lists – singly linked list, doubly linked list, circular linked list, insertion, deletion, traversal and applications.

UNIT – IV: Data Exploring Trees, Graphs, Searching and Sorting 14 Hours

Trees–binary trees, binary search trees, tree traversals (inorder, preorder, postorder), applications. Balanced trees– AVL trees (concepts and rotations).Graphs– representation (adjacency matrix, adjacency list), traversal (BFS, DFS), applications. Sorting techniques – bubble sort, selection sort, insertion sort, quick sort, merge sort, heap sort. Searching techniques – linear search, binary search, hash tables and collision handling techniques.

Text Books:

1. E. Balagurusamy, Programming with Java – A Primer, McGraw Hill, 6th Edition, 2017.
2. YashavantKanetkar, Data Structures Through Java, BPB Publications, 2nd Edition, 2003.
3. Mark Allen Weiss, Data Structures and Problem Solving Using Java, Addison-Wesley, 4th Edition, 2012.

Reference Books:

1. Herbert Schildt, Java: The Complete Reference, McGraw Hill, 11th Edition, 2018.
2. Goodrich, Tamassia, Goldwasser, Data Structures and Algorithms in Java, Wiley, 6th Edition, 2014.
3. Horowitz, Sahni, Mehta, Fundamentals of Data Structures in C++/Java, UniversitiesPress, 2nd Edition, 2008.

Theory	**MSC103T: Python Programming**
Teaching Hours : 04 Hours/Week	**Credits: 04**
Duration of Exam: 03 Hours	**Maximum Marks: 100 (Exam 70 + IA 30)**

Course Outcomes

COs	**Description**
CO1	Illustrate the fundamentals of Python programming by applying variables, operators, control structures, strings, and lists to solve basic computational problems.
CO2	Apply and implement Python data structures (tuples, sets, dictionaries) and functions (including recursion and higher-order functions) to develop efficient solutions.
CO3	Demonstrate proficiency in object-oriented programming concepts in Python, including inheritance, polymorphism, and exception handling, and utilize modules and packages for modular programming.
CO4	Analyze and apply advanced Python features such as file handling, regular expressions, and libraries (NumPy, Pandas, Matplotlib) for data analysis, visualization, and database connectivity.

Unit I: Introduction to Python Programming **14 Hours**

Python Basics: History, Features, Python Interpreter, IDEs - Variables, Data Types, Operators, Expressions - Control Structures: Conditional Statements (if, if-else, nested if) - Loops: for, while, nested loops, loop control statements - Strings: Operations, String methods, Slicing, String formatting - Lists: Creation, Indexing, Slicing, List methods.

Unit II: Python Data Structures & Functions **14 Hours**

Tuples: Operations, Tuple methods, Packing and Unpacking - Sets: Operations, Methods, Applications Dictionaries: Keys, Values, Methods, Operations - Functions: Defining functions, Arguments (positional, keyword, default, variable-length) - Lambda Functions, Map, Filter, Reduce - Recursion in Python.

Unit III: Object-Oriented Programming & Modules in Python **14 Hours**

Classes and Objects in Python - Constructors, Instance &Class Variables, Methods- Inheritance: Single, Multiple, Multilevel, Hierarchical - Method Overriding, Polymorphism, Abstract Classes - Exception Handling: try, except, finally, raise, custom exceptions - Modules and Packages: Importing, Creating, Built-in modules (math, random, datetime).

Unit IV: Advanced Python – File Handling & Libraries **14 Hours**

File Handling: Text and Binary files, File operations (read, write, append) - Regular Expressions: re module, Pattern matching, Searching, Substitution - Python Libraries: NumPy – Arrays, Vectorized operations - Pandas – DataFrames, Series, Data manipulation - Matplotlib – Plotting graphs, Visualization - Introduction to Database Connectivity with Python (SQLite/MySQL).

Text Books:

1. John M Zelle – Python Programming: An Introduction to Computer Science. 3rd Edition published in 2016.
2. Mark Lutz – Learning Python : Powerful Object-Oriented Programming, 6th Edition, O'Reilly Media, Feb'2025.
3. Wes McKinney – Python for Data Analysis: Data Wrangling with Pandas, Numpy and Jupyter, 3rd Edition, O'Reilly Media, 2022.

Reference Books:

1. Charles Dierbach – Introduction to Computer Science using Python: A Computational Problem-Solving Focus, Wiley, 2nd Edition, 2015.
2. ReemaThareja – Python Programming: Using Problem Solving Approach, Oxford University Press, 2nd Edition, 2017.
3. Allen B. Downey – Think Python: How to Think Like a Computer Scientist, 3rdEdition, Green Tea Press, 2024.

Theory	MSC104T: Advanced Software Engineering
Teaching Hours : 04 Hours/Week	Credits: 04
Duration of Exam: 03 Hours	Maximum Marks: 100 (Exam 70 + IA 30)

Course Outcomes

COs	Description
CO1	Analyze software process models and apply requirements engineering techniques to model and validate software requirements using UML.
CO2	Apply software design principles, methodologies, and design patterns to develop effective software architectures, and evaluate their applicability in real-world systems.
CO3	Demonstrate the ability to plan and conduct software testing using manual and automated tools, implement SQA practices, and utilize metrics for assessing software quality and reliability.
CO4	Evaluate software project management practices including estimation, planning, risk management, and version control, and explore emerging trends such as DevOps, cloud-based development, AI in SE, and software security.

Unit I: Software Process Models and Requirements Engineering 14 Hours

Evolution of Software Engineering– Classical vs. Modern perspectives -Software Development Life Cycle (SDLC) models: Spiral, V-Model, Incremental, Agile, and DevOps model - Requirement Engineering: Process, Requirement elicitation techniques, Requirement analysis and specification - Requirement modeling using UML – Use cases, Activity Diagrams, Sequence Diagrams - Software Prototyping and Requirements Validation.

Unit II: Software Design and Architecture 14 Hours

Principles of Software Design: Modularity, Abstraction, Cohesion, Coupling, Information Hiding - Design Methodologies: Structured Design, Object-Oriented Design - Design Patterns: Creational, Structural, Behavioral patterns (examples with UML) - Software Architecture: Styles (Layered, Client-Server, Service-Oriented, Microservices) - Component-Based Design and Reuse - Case Study: Applying design patterns in real-world applications.

Unit III: Advanced Software Testing, Quality and Metrics 14 Hours

Testing Strategies: Unit Testing, Integration Testing, System Testing, Regression Testing - Test Automation and Tools (JUnit, Selenium, PyTest) - White-box and Black-box testing techniques - Software Quality Assurance (SQA) frameworks and standards (CMMI, ISO, Six Sigma) - Software Metrics: Process Metrics, Project Metrics, Product Metrics - Defect Management and Reliability Engineering.

Unit IV: Software Project Management and Emerging Trends 14 Hours

Project Estimation: Function Point Analysis, COCOMO-II - Project Planning, Scheduling and Tracking (PERT, CPM, Gantt Charts, Agile Scrum framework) - Risk Management in Software Projects - Configuration Management and Version Control Systems (Git, GitHub, CI/CD concepts) - Emerging Trends: Cloud-based Development, DevOps, AI in Software Engineering, Software Security Engineering - Case Studies in Successful and Failed Projects.

Text Books:

1. Ian Sommerville – Software Engineering, 10th Edition, Pearson Education, 2019.
2. Roger S. Pressman & Bruce R. Maxim – Software Engineering: A Practitioner's Approach, 9th Edition, McGraw Hill, 2020.
3. Len Bass, Paul Clements, Rick Kazman – Software Architecture in Practice, 4th Edition, Addison-Wesley, 2021.

Reference Books:

1. Mary Shaw & David Garlan – Software Architecture: Perspectives on an Emerging Discipline, PHI, 1996.
2. PankajJalote – An Integrated Approach to Software Engineering, 3rd Edition, Springer, 2018.
3. Rajib Mall – Fundamentals of Software Engineering, 5th Edition, PHI Learning, 2018.

Theory	**MSC105T: Analysis & Design of Algorithms**
Teaching Hours : 04 Hours/Week	**Credits: 04**
Duration of Exam: 03 Hours	**Maximum Marks: 100 (Exam 70 + IA 30)**

Course Outcomes

COs	**Description**
CO1	Understand the fundamental concepts of algorithms, problem-solving patterns, asymptotic analysis, and mathematical evaluation of recursive and non-recursive algorithms.
CO2	Apply brute force, decrease-and-conquer, divide-and-conquer techniques, and graph traversal algorithms to solve computational problems.
CO3	Develop efficient solutions using advanced algorithm design strategies such as space–time tradeoffs, dynamic programming, and greedy techniques.
CO4	Analyze computational complexity, apply backtracking and branch-and-bound methods, and evaluate parallel and probabilistic algorithms for real-world applications.

UNIT I: Foundations of Algorithms **14 Hours**

Introduction: Algorithms, Fundamentals of Algorithmic Problem Solving, Important Patterns - Fundamental Data Structures (review) - Fundamentals of Algorithm Analysis: The Asymptotic Framework, Asymptotic Notations, Basic Efficiency Classes - Mathematical Analysis of Non-recursive and Recursive Algorithms - Empirical Analysis of Algorithms - Algorithm Visualization.

UNIT II: Divide, Conquer, and Traversals **14 Hours**

Brute Force Methods: Selection Sort, Bubble Sort, Sequential Search, Exhaustive Search - Decrease and Conquer: Insertion Sort, Topological Sorting, Combinatorial Object Generation - Divide and Conquer: Merge Sort, Quick Sort, Binary Tree Traversals, Strassen's Matrix Multiplication - Graph Traversals: Depth-First Search (DFS), Breadth-First Search (BFS) .

UNIT III: Advanced Design Techniques **14 Hours**

Space and Time Tradeoffs: Counting Sort, Input Enhancement, Hashing - Dynamic Programming: Principle of Optimality, Bellman's Equation, Optimal Binary Search Trees, Knapsack Problem, Warshall's Algorithm, Floyd's Algorithm -Greedy Technique: Prim's Algorithm, Kruskal's Algorithm, Dijkstra's Algorithm, Huffman Coding.

UNIT IV: Complexity, Backtracking, and Parallel Algorithms **14 Hours**

Limitations of Algorithm Power: Lower-Bound Arguments, Decision Trees, P, NP, and NP-Complete Problems. Coping with Limitations: Backtracking (n-Queens, Hamiltonian Circuit, Subset-Sum) -Branch and Bound: Assignment Problem, Knapsack Problem, Traveling Salesman Problem -Parallel & Randomized Algorithms: Models, Speedup, Efficiency, Sorting Networks, Applications in Graph Problems - Probabilistic Algorithms: Las Vegas, Monte Carlo, Game-Theoretic Techniques.

Textbooks :

1. Anany Levitin – Introduction to the Design and Analysis of Algorithms, Pearson,3rd Edition, 2012.
2. Thomas H. Cormen, Charles E. Leiserson, Ronald Rivest, Clifford Stein – Introduction to Algorithms, MIT Press, 4th Edition, 2022.
3. Ellis Horowitz, Sartaj Sahni, Sanguthevar Rajasekaran – Fundamentals of Computer Algorithms, Universities Press, 2nd Edition, 2008.

Reference Books:

1. Jon Kleinberg, ÉvaTardos – Algorithm Design, Pearson, 1st Edition, 2005.
2. Robert Sedgewick, Kevin Wayne – Algorithms, Addison Wesley, 4th Edition, 2011.
3. Michael T. Goodrich, Roberto Tamassia – Algorithm Design: Foundations, Analysis, and Internet Examples, Wiley, 1st Edition, 2001.

Lab	**MSC101P: Data Structures using Java Lab**
Teaching Hours : 04 Hours/Week	**Credits: 02**
Duration of Exam: 03 Hours	**Maximum Marks: 50 (Exam 35 + IA 15)**

1. Write a Java program to perform addition, subtraction, and multiplication of two matrices.
2. Implement a Java program to check whether a given string is a palindrome using character arrays.
3. Write a Java program to solve Towers of Hanoi problem using recursion.
4. Implement a Java program to sort an array of integers using Quick Sort algorithm.
5. Implement a Java program to perform push, pop, and display operations on a stack using arrays.
6. Write a Java program to convert an infix expression to postfix expression using a stack.
7. Implement a Java program to evaluate a postfix expression using stack.
8. Write a Java program to perform insert, delete, and display operations on a queue using arrays.
9. Write a Java program to create a singly linked list and perform insertion and deletion operations.

10. Implement a Java program to create a doubly linked list and traverse it in both directions.
11. Write a Java program to add two polynomials using singly linked lists.
12. Write a Java program to create a BST and perform inorder, preorder, and postorder traversals.
13. Implement a Java program to sort a list of integers using Heap Sort.
14. Write a Java program to represent a graph using adjacency matrix and adjacency list.

Lab	**MSC102P: Python Programming Lab**
Teaching Hours: 04 Hours/Week	**Credits: 02**
Duration of Exam: 03 Hours	**Maximum Marks: 50 (Exam 35 + IA 15)**

Part - A

1. Write a Python program to accept name, age, and CGPA from the user and display them in a formatted way using f-strings. Convert age into months.
2. Write a Python program to input two integers and perform arithmetic operations: sum, integer division, remainder, and power. Check whether the first is divisible by the second. Handle division by zero.
3. Write a Python program to read marks of 5 subjects, compute total, percentage, and display grade based on given conditions. Also check pass/fail (pass if all subjects ≥40).
4. Write a Python program using loops: (a) Display first n Fibonacci numbers. (b) Accept a list of integers and display minimum, maximum, sum, and average.
5. Write a Python program to accept a paragraph of text and display: number of words, number of sentences, and number of characters. Also display the 3 most frequent words.
6. Write a Python program to accept a list of student details as tuples (id, name, score). Convert them into a dictionary with id as key, display unique scores as a set, and sort students by scorein descending order.
7. Write a Python program using functions: (a) Define a function to check if a number is prime. (b)Use filter() to extract primes between 1 and 100. (c) Use map() to square those primes. (d) Usereduce() to multiply the first 10 primes.
8. Write a recursive Python function for factorial, GCD, and binary search. Also provide an iterative version of one function and compare their results.
9. Write a Python program using OOP to define a BankAccount class with attributes acc_no, owner, and balance. Include methods deposit(), withdraw(), and display(). Demonstrate the class with multiple objects.
10. Write a Python program to read student marks from a CSV file, calculate total and grade, and write the results into a new CSV file. Handle missing values using exception handling.

PART - B

1. Write a Python program that reads a list of integers from the user and:
    a) Stores them in a list.
    b) Finds and prints the largest, smallest, sum, and average of the numbers.
    c) Removes duplicates and displays the unique values as a set.
2. Write a Python program to create a dictionary that stores names of students as keys and their marks as values.
    a) Accept at least 5 entries from the user.
    b) Display all student names and marks.
    c) Find and print the student with the highest marks.
    d) Search for a student name entered by the user and display their marks (if found).
3. Write a Python program to process an inventory dataset stored in a **CSV file**. The program s should:
    a) Read product details (product_id, product_name,category, price, quantity) from a CSV file.
    b) Search for products by name (case-insensitive).
    c) Display the top 3 most expensive products in each category.
    d) Identify products with quantity less than a given threshold and save them into another CSV file
4. Write a Python program using OOP concepts to design Person, Student, and Teacher classes. Implement abstract class Member with abstract method get_role_info(). Demonstrate method overriding, polymorphism, and multiple inheritance with a TA (Teaching Assistant) class.
5. Write a Python program using NumPy to generate a 100×5 dataset of random numbers. Perform column-wise mean, standard deviation, normalization, covariance, and eigenvalue computation. Plot any two columns using Matplotlib.
6. Write a Python program using Pandas to read a CSV file of sales data (date, product, category, quantity, unit_price). Clean missing values, convert date into datetime format, compute monthly sales by category, and generate a pivot table of product vs month showing revenue. Save results to Excel.
7. Write a Python program using Matplotlib to visualize sales data: (a) line plot of monthly revenue, (b) stacked bar chart of revenue by category, and (c) histogram of order quantities. Save all graphs as PNG files.
8. Write a Python program using SQLite to create a database with tables Students and Marks. Insert records from CSV, compute average marks, display top 5 students, and list students who failed. Use parameterized queries to prevent SQL injection.
9. Write a Python program to develop a Student Management System (CLI based). The program should allow adding, updating, deleting, and searching student records. Data should be stored in SQLite. Include validation and option to export data into CSV file.
10. Write a Python program to perform Exploratory Data Analysis (EDA) on a dataset such as Iris/Titanic. Perform data cleaning, missing value handling, encoding categorical data, visualizations, and train a simple classifier (e.g., logistic regression). Display accuracy and confusion matrix. Generate a report with results and plots.

Theory	MSC101SC: Quantitative Techniques & Teaching Aptitude
Teaching Hours : 02 Hours/Week	Credits: 02
Duration of Exam: 1.5 Hours	Maximum Marks: 50 (Exam 35 + IA 15)

Course Outcomes

COs	Description
CO1	Apply the concepts of number systems, arithmetic, algebra, probability, and applied quantitative techniques to solve real-world problems.
CO2	Analyze and interpret quantitative data using tabulation, charts, graphs, and reasoning techniques for logical decision-making.
CO3	Demonstrate an understanding of teaching aptitude by identifying learner characteristics, teaching methods, evaluation systems, and the use of teaching aids in classroom practice.
CO4	Evaluate classroom communication strategies, comprehension skills, and logical reasoning methods for effective teaching–learning practices.

UNIT – I: Quantitative Techniques **15 Hours**

Number Systems & Arithmetic: Numbers property, simplification, divisibility, HCF & LCM, decimal fractions, square & cube roots, logarithms, antilogarithms, surds and indices. Algebra & Probability: Permutation and combination, probability, ratio & proportion, partnership, averages, percentages, profit & loss, simple & compound interest. Applied Quantitative Problems: Time & work, problems on ages, calendar, clock, pipes & cistern, time & distance, problems on trains, boats & streams. Data Interpretation & Reasoning: Data interpretation (tabulation, bar graphs, pie charts, line graphs), odd man out, number series, letter series, coding-decoding, relationships, classification.

UNIT – II: Teaching Aptitude & Communication **15 Hours**

Teaching Aptitude: Nature, objectives, characteristics and basic requirements of teaching; learner's characteristics; factors affecting teaching; methods of teaching; teaching aids; evaluation systems. Classroom Communication: Nature, characteristics, types, and barriers of communication; effective classroom communication strategies. Comprehension Skills: Reading comprehension passage with questions for evaluation of understanding and interpretation. Analytical & Logical Reasoning: Structure of arguments; deductive and inductive reasoning; verbal analogies, verbal classification; logical diagrams (simple and multi-diagrammatic relationships, Venn diagrams, analytical reasoning).

Text Books :

1. R.S. Aggarwal, Quantitative Aptitude, S. Chand & Company, New Delhi, 2024-25.
2. Govind Prasad Singh &Rakesh Kumar, Text Book of Quickest Mathematics, KiranPrakashan,1st Edition,Kiran Prakashan,2021.
3. Lal, Jain & K.C. Vashistha, UGC-NET/JRF/SET Teaching Aptitude, UpkarPrakashan, 2025.

Reference Books:

1. UGC NET Paper I: Teaching & Research Aptitude, Two Brothers Publications Pvt. Ltd., 10th Edition, 2024.
2. NTA UGC NET Paper I: Teaching & Research Aptitude by KVS Madaan, Pearson, 2025.
3. Arihant Experts, Teaching Aptitude& Attitude, Arihant Publications, 2025.

SEMESTER - II

Theory	MSC201T: Database Applications
Teaching Hours : 04 Hours/Week	**Credits: 04**
Duration of Exam: 03 Hours	**Maximum Marks: 100 (Exam 70 + IA 30)**

Course Outcomes

COs	Description
CO1	Understand the fundamentals of database systems, data models, ER modeling, relational algebra, and SQL operations.
CO2	Apply normalization techniques for effective database design and implement PL/SQL constructs such as cursors, triggers, procedures, and indexing methods.
CO3	Analyze transaction management, concurrency control, recovery techniques, and distributed database concepts for reliable and secure data processing.
CO4	Evaluate emerging database applications including data warehousing, data mining, NoSQL, and big data technologies with real-time case studies.

UNIT – I: Fundamentals of Database Systems **14 Hours**

Introduction to Database Systems – Characteristics, Advantages of DBMS over File Systems - Data Models – Hierarchical, Network, Relational, Object-oriented - Entity–Relationship (ER) Model – Entities, Attributes, Relationships, ER Diagrams, Enhanced ER concepts - Relational Model – Concepts, Keys, Relational Algebra (Selection, Projection, Union, Difference, Join, Division) - Introduction to SQL – DDL, DML, DCL, TCL statements.

UNIT – II: Database Design & Programming **14 Hours**

Database Design Process– Mapping ER Model to Relational Schema - Functional Dependencies and Normalization –1NF, 2NF, 3NF, BCNF, Multi-valued & Join Dependencies, 4NF, 5NF - PL/SQL –Cursors, Triggers, Stored Procedures, Functions, Packages, Error Handling - Indexing and Hashing Techniques.

UNIT – III: Database Applications and Features **14 Hours**

Transaction Management –ACID properties, States of a Transaction, Concurrency Control (Lock-based, Timestamp-based, Deadlock handling). Recovery Techniques–Log-based, Shadow Paging, Checkpointing-Distributed Databases–Concepts, Architecture, Fragmentation, Replication, Transparency - Object-Relational Databases –Concepts, Features, Implementation.

UNIT – IV: Emerging Database Applications **14 Hours**

Data Warehousing –Architecture, ETL Process, Introduction to OLAP operations- Data Mining –Classification, Clustering, Association Rules, Applications - NoSQL Databases –Key-Value, Document, Column-oriented, Graph Databases - Big Data & Cloud Databases– Concepts, Use cases (MongoDB, Cassandra, HBase) - Case Studies of Real-time Applications –E-commerce, Banking, Healthcare.

Text Books:

1. Abraham Silberschatz, Henry Korth, and S. Sudarshan – Database System Concepts, 2019, 7th Edition, McGraw Hill.
2. Ivan Bayross – SQL, PL/SQL: The Programming Language of Oracle, BPB Publications,2010.
3. C.J. Date – An Introduction to Database Systems, 2003, 8th Edition, Pearson Education.

Reference Books:

1. Ramez Elmasri, Shamkant B. Navathe – Fundamentals of Database Systems, 7th Edition, Pearson.
2. Raghu Ramakrishnan& Johannes Gehrke – Database Management Systems, 3rd Edition, McGraw Hill.
3. Pramod J. Sadalage, Martin Fowler – NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence, Addison-Wesley,2012.

Theory	**MSC202T: Advanced Java Programming**
Teaching Hours : 04 Hours/Week	**Credits: 04**
Duration of Exam: 03 Hours	**Maximum Marks: 100 (Exam 70 + IA 30)**

Course Outcomes

COs	**Description**
CO1	Apply GUI programming concepts using AWT, Swing, and JavaFX with effective event handling to design interactive desktop applications.
CO2	Implement multithreading, concurrency utilities, generics, and advanced collections to develop efficient and concurrent applications.
CO3	Develop programs using Java I/O, NIO, serialization, and networking concepts to build robust client-server communication models.
CO4	Construct database-driven and web-based applications by integrating JDBC, Servlets, and JSP with appropriate transaction and session management.

UNIT I: Event Handling and GUI Development in Java 14 Hours

Event Handling: Event delegation model, Action Event, Mouse Event, Key Event, Window Event, Swing: Swing Components, Dialogs and Menus, Introduction to JavaFX, Scene Graph, Controls, Layouts.

UNIT II: Multithreading, Generics and Collections Framework 14 Hours

Multithreading: Thread lifecycle, creating and running threads, synchronization, inter-thread communication, Concurrency utilities, Generics in Java–Generic classes, methods, bounded types, wildcards, Collection Framework: TreeSet, TreeMap, Priority Queue.

UNIT III: Java I/O, Streams and Networking 14 Hours

Java I/O: Byte and Character streams, Buffered streams, Serialization, Deserialization, File handling with NIO package, File Channel, Buffers, Paths, Basics of networking in Java: Socket Programming – TCP & UDP, Server Socket - URL, URL Connection classes, Building client-server applications.

UNIT IV: Database Connectivity and Web Programming Basics 14 Hours

JDBC Architecture – Drivers, Connection, Statement, Prepared Statement, Callable Statement, Executing SQL queries using JDBC, ResultSet, Metadata, Transaction Management, Servlet Programming– Servlet lifecycle, Request & Response handling, Session management, JSP – Elements, Directives, JSTL, Expression Language, Introduction to Java Web Application Architecture (3-tier concept).

Text Books:

1. Herbert Schildt, Java: The Complete Reference, 13th Edition, McGraw Hill, 2024.
2. Rajiv Chopra, Advanced Java Programming, 2nd Edition, Dreamtech Press, 2018.

Reference Books:

1. Cay S. Horstmann, Core Java Volume I & II, 11th Edition, Pearson, 2019.
2. Joe Wigglesworth & Paula McMillan, Java Programming: Advanced Topics, Cengage Learning, 4th Edition, 2017.

Theory	**MSC203T: Artificial Intelligence**
Teaching Hours : 04 Hours/Week	**Credits: 04**
Duration of Exam: 03 Hours	**Maximum Marks: 100 (Exam 70 + IA 30)**

Course Outcomes

COs	**Description**
CO1	Understand Intelligent agents and their environments, and apply problem-solving techniques to classical AI problems.
CO2	Apply various search strategies, game-playing techniques, and knowledge representation methods to model and solve AI problems effectively.
CO3	Apply logical and probabilistic reasoning, advanced knowledge representation, and machine learning techniques to develop intelligent systems.
CO4	Understand the structure and applications of expert systems, apply knowledge engineering and reasoning techniques, and evaluate their integration with machine learning and societal impacts.

UNIT - I Foundations of Artificial Intelligence and Problem Solving 14 Hours

AI problems, foundation of AI and history of AI ,Intelligent agents: Agents and Environments, the concept of rationality, the nature of environments, structure of agents, problem solving agents, problem formulation. Examples of AI problems- Water-Jug problem, Chess Problem, TSP problem, Cryptaritimatic puzzle, Tower of Hanoi Problem. Magic Square.

UNIT –II Search Strategies, Game Playing, and Knowledge Representation 14 Hours

Searching- Searching for solutions, uninformed search strategies – Breadth first search, depth first Search. Search with partial information (Heuristic search) Hill climbing, A*, AO* Algorithms, Problem reduction, Means End Analysis, Game Playing-Adversarial search, Games, mini-max algorithm, optimal decisions in multiplayer games, Problem in Game playing,

Alpha-Beta pruning, Evaluation functions. Introduction to Knowledge, types of knowledge, Knowledge representation issues, predicate logic- logic programming, semantic nets- frames and inheritance, constraint propagation, representing knowledge using rules, rules based deduction systems.

UNIT -III Reasoning, Logic, and Learning 14 Hours

Reasoning under uncertainty, review of probability, Baye's probabilistic interferences and Dempster–Shafer theory. First order logic. Inference in first order logic, propositional vs. first order inference, unification & lifting, forward chaining, Backward chaining, Resolution, Advanced knowledge representation techniques, Semantic net, Frames, Learning from observation Inductive learning, Decision trees, Explanation based learning, Statistical Learning methods, Reinforcement Learning.

UNIT- IV Expert Systems, Knowledge Engineering, and Advanced Reasoning 14 Hours

Introduction to Expert Systems: Definition, basic concepts, architecture/structure of expert systems- Role of human experts and knowledge engineers - How expert systems work- Applications and Types of Expert Systems- Knowledge Engineering and Acquisition-Reasoning Techniques in Expert Systems.

Text Books:

1. S. Russel and P. Norvig, "Artificial Intelligence – A Modern Approach", 4th Edition, 2021, Pearson Education.
2. David Poole, Alan Mackworth, "Artificial Intelligence : Foundations of Computational Agents, Oxford University Press. 3rd Edition, 2023.
3. G. Luger, "Artificial Intelligence: Structures and Strategies for complex problem-solving", Sixth Edition, 2020, Pearson Education.

Reference Books:

1. Nils J. Nilsson, "Artificial Intelligence: A new Synthesis", Elsevier Publishers. 2020.
2. Wolfgang Ertel," Introduction to AI", 3rd Ed., 2022, Springer.
3. Kevin Knight, Elaine Rich & Shivashankar B. Nair – Artificial Intelligence, (3rd Ed., 2017, McGraw Hill.

Theory	MSC204T: Advanced Operating System
Teaching Hours : 04 Hours/Week	Credits: 04
Duration of Exam: 03 Hours	Maximum Marks: 100 (Exam 70 + IA 30)

Course Outcomes

COs	Description
CO1	Understand and analyze the structure of operating systems, process management, scheduling, memory management, and synchronization in multiprocessor systems.
CO2	Understand distributed system architectures, communication mechanisms, synchronization models, and apply logical/causal clock concepts to distributed computing problems.
CO3	Analyze distributed deadlock detection, resolution strategies, and agreement protocols, and evaluate their applications in reliable systems.
CO4	Apply concepts of database operating systems and mobile operating systems (ARM/Intel architectures, concurrency, power management) to modern computing environments.

UNIT I: Fundamentals of Operating System 14 Hours

Structure of OS, processes and scheduling, System Call, Interface, Life Cycle of a Process, Process States, Process Control Block, Process Dynamics, Process synchronization, Process scheduling and Allocation, memory management, Process synchronization, Critical-Section Problem, Peterson's Solution, Synchronization Hardware, Semaphores, Classic Problems of Synchronization, Multiprocessor Operating System.

UNIT II: Distributed System Concepts 14 Hours

Architecture of Distributed System, Characteristics of Distributed System, Advantages and Disadvantages of Distributed System, Use cases of Distributed System, issues in distributed operating systems, communication networks, communication primitives. Inherent limitations of a distributed system, lamport's logical clocks, vector clocks, causal ordering of messages, termination detection. Distributed Mutual Exclusion – Introduction, classification of mutual exclusion and associated algorithms, Introduction to Distributed resource management.

Unit III: Deadlock Detection and Agreement Protocols 14 Hours

Distributed Deadlock Detection: Introduction, approaches to detect deadlock in the distributed system, deadlock handling strategies in distributed systems, issues in deadlock detection and resolution, control organizations for distributed deadlock detection, Agreement protocols – Introduction, a classification of agreement problems, solutions to the Byzantine agreement problem, Applications of agreement algorithms.

Unit IV- Database Operating Systems and Mobile Operating Systems 14 Hours

Database Operating Systems: Requirements of Database OS, Transaction process model, Synchronization primitives, Concurrency control algorithms, Mobile Operating Systems: ARM and Intel architectures, Power Management , Mobile OS Architectures, Underlying OS, Kernel structure and native level programming, Runtime issues, Approaches to power management.

Text Books :

1. Andrew S. Tanenbaum & Herbert Bos – Modern Operating Systems, 5th Edition, Pearson, 2022.
2. Maarten van Steen & Andrew S. Tanenbaum – Distributed Systems, 4th Edition, Self-Published (distributed-systems.net), 2023.
3. Remzi H. Arpaci-Dusseau & Andrea C. Arpaci-Dusseau – Operating Systems: Three Easy Pieces (OSTEP), Version 1.10, Arpaci-Dusseau Books, 2023 (freely available online).

Reference Books:

1. Silberschatz, Galvin & Gagne – Operating System Concepts, 10th Edition, Wiley, 2018/2021.
2. William Stallings – Operating Systems: Internals and Design Principles, 9th Edition, Pearson, 2018
3. Mukesh Singhal & Niranjan G. Shivaratri – Advanced Concepts in Operating Systems: Distributed, Database, and Multiprocessor Operating Systems, McGraw-Hill, Reprint Edition, 2001.

Theory	**MSC205T: Data Communication and Networks**
Teaching Hours : 04 Hours/Week	**Credits: 04**
Duration of Exam: 03 Hours	**Maximum Marks: 100 (Exam 70 + IA 30)**

Course Outcomes

COs	**Description**
CO1	Understand the process of data communication and classification of computer networks based on architecture, transmission, and applications.
CO2	Compare and evaluate guided and unguided transmission media, multiplexing techniques, and switching methods for efficient data transfer.
CO3	Apply error detection and correction techniques and analyze wireless standards.
CO4	Illustrate network communication models, addressing, topologies, and connecting devices in computer networks.

UNIT I: Fundamentals of Data Communication & Computer Networks 14 Hours

Process of data communication and its components: Transmitter, Receiver, Medium, Message, Protocols, Standards, Standard organizations, Bandwidth, Data Transmission Rate, Baud Rate and Bits per second, Modes of Communication (Simplex, Half duplex, Full Duplex),Analog Signal and Digital Signal, Analog and Digital Transmission: Analog To Digital, Digital To Analog Conversion, fundamentals of Computer Network: Definition And Need Of Computer Network, Applications, Network Benefits , Classification of Network: LAN, WAN,MAN.

UNIT II: Transmission Media and Switching Techniques 14 Hours

Communication Media: Guided Transmission Media Twisted-Pair Cable, Coaxial Cable, Fiber-Optic Cable, Unguided Transmission Media: Radio Waves, Microwaves, Infrared, Satellite ,Line-of-Sight Transmission, Point-to-Point, Broadcast ,Multiplexing: Frequency-Division Multiplexing, Time Division Multiplexing, Switching: Circuit-switched network, Packet switched network.

UNIT III: Error Detection, Correction and Wireless Standards 14 Hours

Types of Errors, Forward Error Correction Versus Retransmission, Framing: Fixed Sized and Variable Sized Framing, Error Detection: Repetition codes, Parity bits, Checksums, CRC, Error Correction: Automatic Repeat Request (ARQ), Hamming Code, Wireless LAN IEEE 802.11 standard Architecture, Features of IEEE 802.11,Bluetooth Architecture: Piconet, Scatternet, Mobile Generations: 3G, 4G and 5G.

UNIT IV: Network Communication Models and Addressing 14 Hours

OSI MODEL: Layered Architecture, Encapsulation, Layers in OSI Model, TCP/IP Layers and their functions: Addressing: Physical Address, Logical Address, Port Address, IP Address-Concept, Notation, Address Space, IPv4 Addressing: Classful and Classless Addressing, subnet mask, supernetting, subnetting, IPv6 Addressing scheme and basic structure, Network Computing Model: Peer To Peer, Client Server, Network Topologies: Introduction, Definition, Selection criteria, Types of Topology- Star, Mesh, Tree, Hybrid, Network Connecting Devices: Switch, Router, Repeater, Bridge, Gateways and Modem.

Text Books :

1. Behrouz A. Forouzan, Data Communications and Networking, 6th Edition ,McGraw Hill Education,2022.
2. Andrew S Tanenbaum, Computer Networks, 6th Edition. Pearson Education, 2022.

Reference Books:

1. P.C .Gupta, Data communications and Computer Networks,2 nd Edition , PHI,2013.
2. S. Keshav, An Engineering Approach to Computer Networks, 2nd Edition, Pearson Education.

Theory	**MSC201E: Finite Automata and Formal Languages**
Teaching Hours : 04 Hours/Week	**Credits: 04**
Duration of Exam: 03 Hours	**Maximum Marks: 100 (Exam 70 + IA 30)**

Course Outcomes

COs	Description
CO1	Understand the concepts of automata theory including alphabets, strings, languages, DFAs, NFAs, and Moore/Mealy machines, and apply conversion techniques between various finite automata.
CO2	Construct and analyze regular expressions and finite automata, apply algebraic laws and pumping lemma, and evaluate closure and decision properties to establish language regularity.
CO3	Develop context-free grammars for different languages, analyze derivations and parse trees, and examine ambiguity in grammars to determine language properties.
CO4	Design pushdown automata and Turing machines for given problems, demonstrate equivalence between CFGs and PDAs, and apply computational models to solve decision problems.

UNIT I: Introduction to Finite Automata **14 Hours**

Introduction, Structural Representations, Automata and Complexity, the Central Concepts of Automata Theory – Alphabets, Strings, Languages, Problems. Nondeterministic Finite Automata: Formal Definition, an application, Text Search, Finite Automata with Epsilon-Transitions. Deterministic Finite Automata: Definition of DFA, How a DFA Process Strings, The language of DFA, Conversion of NFA with €-transitions to NFA without €-transitions. Conversion of NFA to DFA, Moore and Melay machines.

UNIT II: Regular Expressions **14 Hours**

Finite Automata and Regular Expressions, Applications of Regular Expressions, Algebraic Laws for Regular Expressions, Conversion of Finite Automata to Regular Expressions. Pumping Lemma for Regular Languages, Statement of the pumping lemma, Applications of the Pumping Lemma. Closure Properties of Regular Languages: Closure properties of Regular languages, Decision Properties of Regular Languages, Equivalence and Minimization of Automata.

UNIT III: Context-Free Grammars **14 Hours**

Definition of Context-Free Grammars, Derivations Using a Grammar, Leftmost and Rightmost Derivations, the Language of a Grammar, Sentential Forms, Parse Tress, Applications of Context-Free Grammars, Ambiguity in Grammars and Languages.

UNIT IV: Introduction to PDA and Turing Machine **14 Hours**

Definition of the Pushdown Automaton, Design of PDA ,the Languages of a PDA, Equivalence of PDA's and CFG's, Acceptance by final state, Acceptance by empty stack, Deterministic Pushdown Automata, Applications of PDA,CFG to PDA, PDA to CFG, Turing machine, problems on Turing machine.

Text Books:

1. John E Hopcroft, Rajeev Motwani, Jeffrey D. Ullman, "Introduction to Automata Theory, Languages and Computation", 3rd Edition, Pearson Education.
2. A M Padmareddy "Finite Automata and Formal Languages", 4th Edition, Pearson Education India.

Reference Books:

1. Peter Linz, " An introduction to Formal Languages and Automata", 6th Edition, Jones & Bartlett Learning.
2. Mishra and Chandrashekaran, "Theory of Computer Science– Automata Languages and Computation", 3rd Edition, PHI

Theory	MSC201E: Multimedia and its Applications
Teaching Hours : 04 Hours/Week	Credits: 04
Duration of Exam: 03 Hours	Maximum Marks: 100 (Exam 70 + IA 30)

Course Outcomes

COs	Description
CO1	Understand the basic concepts of components, applications, tools, and hardware/software needs of multimedia.
CO2	Identify and apply text standards, fonts, graphics, color models, image formats, and devices used in multimedia.
CO3	Compare audio and video technologies, formats, and tools for recording and editing.
CO4	Use animation, compression, streaming, and web technologies to design simple multimedia applications for the web.

UNITI: Introduction to Multimedia **14 Hours**

Multimedia – Introduction, applications, Components, Hardware and Software requirements. Characteristics of effective multimedia presentation. Overview of Multimedia Authoring Tools. Interactive media, Hypermedia. Distributed Multimedia Applications. Multimedia Database Systems. Virtual Reality.

Unit II: Basics of Text and Graphics **14 Hours**

Text – Introduction, Standards – ASCII, Unicode. Fonts. Graphics – Introduction, Types, Color and color models. Digital Images, Image processing. Graphic file formats. Graphics input and output devices.

Unit III: Introduction to Audio and Video **14 Hours**

High Definition Television and Desktop computing – Knowledge based Multimedia systems. Audio – Introduction, attributes, Mono Vs Stereo audio, audio channels. Analog Vs Digital audio. Musical Instrument Digital Interface (MIDI). File formats, audio recording and editing – hardware and software. Video – Introduction, Video frames and frame rate. Video recording and editing – hardware and software File formats.

Unit IV: Multimedia on the Web **14 Hours**

Animation – Introduction, Principles, Types and Uses. Animation Software. Compression techniques – Introduction, need and types of compression – lossy and lossless. CODECS.- webinars. Introduction, Bandwidth**.** Text on the web – Dynamic and embedded font technology**.** Audio and video on the web, buffering and streaming**.** Webcasting, video conferencing.

Text Books :

1. Computer Graphics & Multimedia – by A.P. Godse, Technical Publications; 2nd edition.
2. Multimedia: Making It Work, Eighth Edition – by Tay Vaughan, McGraw-Hill Osborne; 8th edition.

Reference Books:

1. Computer Graphics, Multimedia and Animation – by Pakhira (Author), Malay K (Author), Publisher: Prentice Hall India Learning Private Limited, 2nd edition (2010).
2. Multimedia – by M. Mahalakshmi (Author), Publisher: Margham Publications (2013).

Lab	MSC201P: Database Applications Lab
Teaching Hours : 04 Hours/Week	**Credits: 02**
Duration of Exam: 03 Hours	**Maximum Marks: 50 (Exam 35 + IA 15)**

1. **Basic SQL Queries** – Create a database for a college system. Write queries using SELECT, WHERE, ORDER BY, DISTINCT.
2. **Joins & Nested Queries** – Perform operations using INNER JOIN, LEFT JOIN, RIGHT JOIN, and subqueries.
3. **Relational Algebra Implementation** – Write SQL queries for UNION, INTERSECT, EXCEPT, DIVISION equivalents.
4. **Database Design & Normalization** – Given an unnormalized relation, normalize it up to **BCNF** and implement the schema in SQL.
5. **Views & Indexing** – Create views for restricted access and implement indexing on frequently used attributes.
6. **Cursors** – Write PL/SQL programs using **implicit and explicit cursors** for employee database traversal.
7. **Stored Procedures & Functions** – Develop procedures/functions to compute salary increments, generate student grade sheets.
8. **Triggers** – Implement **row-level and statement-level triggers** (e.g., before insert/update on salary table).
9. **Exception Handling in PL/SQL** – Write PL/SQL blocks to handle user-defined and predefined exceptions.
10. **Transaction Management** – Demonstrate **commit, rollback, and savepoint** with banking transactions.
11. **Concurrency Control** – Simulate **lost update problem** and show how locking prevents it.
12. **Distributed Database Simulation** – Create **fragments of tables** in two schemas and query them to demonstrate data distribution.
13. **Data Warehousing & OLAP** – Create a simple sales warehouse; perform **roll-up, drill-down, slice, dice** operations.
14. **Data Mining** – Use a dataset (e.g., student performance or sales data) and implement **classification (decision tree)** or **clustering (k-means)** using SQL/PLSQL or a tool like Weka/Oracle Data Miner.
15. **NoSQL Database (MongoDB/Cassandra)** – Perform **CRUD operations** and queries on a document/column-based NoSQL DB.

Lab	MSC202P: Advanced Java Programming Lab
Teaching Hours : 04 Hours/Week	**Credits: 02**
Duration of Exam: 03 Hours	**Maximum Marks: 50 (Exam 35 + IA 15)**

1. Design a **simple calculator application** using **AWT/Swing components** (JFrame, JButton, JTextField) and handle arithmetic operations with event listeners.
2. Create a **student registration form** using **Swing components** (JTextField, JComboBox, JRadioButton, JCheckBox, JTable) and display the entered details in a table.
3. Write a program to demonstrate **mouse and keyboard event handling** (MouseEvent, KeyEvent) with interactive output on the GUI.
4. Implement a **tabbed interface** (JTabbedPane) for managing multiple views such as Profile, Courses, and Results using Swing.
5. Develop a program that demonstrates **multithreading** by creating multiple threads for printing odd/even numbers with proper synchronization.
6. Implement a **producer–consumer problem** using inter-thread communication (wait/notify).
7. Write a program using **Generics** to create a **type-safe stack/queue** and demonstrate push/pop operations.
8. Create a **priority-based task scheduler** using **ExecutorService, Callable, and Future** classes.
9. Write a program to perform **file copy operation** using **byte and character streams** with buffered I/O.
10. Demonstrate **serialization and deserialization** of Employee objects (with name, id, salary) to a file.
11. Build a **client-server chat application** using **TCP sockets**, where multiple clients can send/receive messages from the server.
12. Develop a **JDBC application** to insert, update, delete, and display records from a student database using PreparedStatement.
13. Write a program to demonstrate **transaction management in JDBC** with commit and rollback operations.
14. Create a **Servlet application** that accepts student login credentials and validates them against a database.
15. Develop a **JSP application** for an online shopping cart using JSTL and Session Management.