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 Computer Applications (MCA)

(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. KiranKumar 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 Master of Computer Applications

1	**Name of the Course**	Master of Computer Applications- MCA
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**	Bachelor's degree with Computer Science /Mathematics/Statistics/Business Mathematics/ Business Statistics as one of the core subjects with a minimum of 50% aggregate marks at 10+2 or at graduation level with specific considerations for reserved categories.
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 problems: 30 marks Viva-voce: 05 marks
12	**Scheme of examination and Announcement of Results**	As per the BCU Rules and Regulations

Programme Outcome (PO):

PO1	Apply knowledge of mathematics, science and computing appropriately to model the software applications
PO2	Assimilate and use state of the art computing technologies, tools and techniques necessary for computing practices.
PO3	Design a system, component, or process to meet desired needs within realistic constraints such as economic, environmental, social and ethical contexts
PO4	Have an ability to design, implement and evaluate sustainable computational solutions for various complex problems as per needs and specifications.
PO5	Communicate effectively with the computing community, and with society, about complex computing activities by being able to comprehend and write effective reports, design documentation, and make effective presentations.
PO6	Manage projects and function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO7	Recognize the need for and prepare themselves to engage in independent and life-long learning, engage in self-learning for continual development as a computing professional for the betterment of individuals, organizations, research community and society.
PO8	Apply ethical principles and commit to professional responsibilities and human values.
PO9	Utilize the education necessary to understand the impact of computing solutions in a Global and societal context
PO10	Innovate and contribute value and wealth for the benefit of the society.

ASSESSMENT

Weightage for the Assessments (in percentage)

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

Detailed Structure for MCA Course

Semester	Course Code	Paper Title	Teaching Hours / Week	Marks		Duration of Exam in Hours	Credits
				Exam	IA		
I	MCA101T	Computational Mathematics and Statistics	04	70	30	03	04
	MCA102T	Data Structures using Java	04	70	30	03	04
	MCA103T	Operating System Concepts and Design	04	70	30	03	04
	MCA104T	Advanced Database Management Systems	04	70	30	03	04
	MCA105T	Software Project Management	04	70	30	03	04
	MCA101E	a. Mobile Computing b. Software Testing with Selenium	04	70	30	03	04
	MCA101P	Data Structures using Java Lab	04	35	15	03	02
	MCA102P	Database Management Systems lab	04	35	15	03	02
Total Credits							**28**
II	MCA201T	Artificial Intelligence	04	70	30	03	04
	MCA202T	Advanced Algorithms	04	70	30	03	04
	MCA203T	Python Programming	04	70	30	03	04
	MCA204T	Computer Networks	04	70	30	03	04
	MCA205T	Cryptography & Information Security	04	70	30	03	04
	MCA201E	a. Automata Theory b. Digital Image Processing	04	70	30	03	04
	MCA201P	Artificial Intelligence using Python Lab	04	35	15	03	02
	MCA202P	Advanced Algorithms using Java Lab	04	35	15	03	02
Total Credits							**28**

SEMESTER–I

Theory	MCA101T: Computational Mathematics and Statistics
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 propositions, logical connectives, truth tables, logical equivalence for simplification. Apply concepts of sets and their operations, explore role of functions as mathematical models in computer science
CO2	Apply basic counting principles to solve problems, use of permutation & combination, learn binomial theorem concept and properties and apply them to computer science.
CO3	Apply probability theorems, conditional probability and Bayes' rule, and analyze independent events.
CO4	Analyze populations and samples using sampling distributions, frequency tables, and statistical measures such as mean, variance, and moments, and apply point/interval estimation, and maximum likelihood methods for statistical inference.

UNIT I: Fundamentals of Logic, Sets and Functions 14 Hours

Propositional Logic–Propositions, Conditional Statements, Truth Tables of Compound Propositions, Precedence of Logical Operators, Propositional Equivalences–Logical Equivalences, Using De Morgan's Laws, Predicates and Quantifiers – Predicates, Quantifiers, Rules of Inference - Valid Arguments in Propositional Logic, Rules of Inference for Propositional Logic, Sets – Introduction, The Power Set, Cartesian Products, Set Operations - Set Identities, Generalized Unions and Intersections Functions – Introduction, One-to-One and Onto Functions.

UNIT II: Counting, Permutation and Combinations 14 Hours

The Basics of Counting – Basic Counting Principles, The Inclusion-Exclusion Principle, The Pigeonhole Principle – Introduction, The Generalized Pigeonhole Principle, Permutations and Combinations – Permutations, Combinations, Binomial Coefficients – The Binomial Theorem, Generalized Permutations and Combinations – Permutations with Repetition, Combinations with Repetitions, Permutations with Indistinguishable Objects.

UNIT-III: Basic Probability and Random Variables 14 Hours

Random Experiments, Sample Spaces Events, the Concept of Probability the Axioms of Probability, Some Important Theorems on Probability Assignment of Probabilities, Conditional Probability Theorems on Conditional Probability, Independent Events, Bayes Theorem or Rule. Random Variables, Discrete Probability Distributions, Distribution Functions for Random Variables, Distribution Functions for Discrete Random Variables, Continuous Random Variables.

UNIT -IV: Sampling and Estimation Theory **14 Hours**

Population and Sample, Statistical Inference Replacement Random Samples, Random Numbers Population Parameters Sample Statistics Sampling Distributions, Frequency Distributions, Relative Frequency Distributions, Computation of Mean, Variance, and Moments for Grouped Data. Unbiased Estimates and Efficient Estimates Point Estimates and Interval Estimates. Reliability Confidence Interval Estimates of Population Parameters, Maximum Likelihood Estimates.

TEXT BOOKS:

1. Kenneth H Rosen: Discrete Mathematics and its Applications, 7th Edition, McGraw Hill publications. 2017.
2. Ralph P Grimaldi, B V Ramana: Discrete and Combinatorial Mathematics, An Applied Introduction, 5th Edition, Pearson Education, 2007.

REFERENCE BOOKS

1. Murray R. Spiegel, John J. Schiller and R. AluSrinivasan, Probability & Statistics,3rd Edition, Schaum's Outline Series, Tata McGraw-HillPublishers,2018
2. John Vince, Foundation Mathematics for Computer Science, First Edition, Springer,2015

Theory	**MCA102T: 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.
2. YashavantKanetkar, Data Structures Through Java, BPB Publications.
3. Mark Allen Weiss, Data Structures and Problem Solving Using Java, Addison-Wesley.

Reference Books:

1. Herbert Schildt, Java: The Complete Reference, McGraw Hill.
2. Goodrich, Tamassia, Goldwasser, Data Structures and Algorithms in Java, Wiley.
3. Horowitz, Sahni, Mehta, Fundamentals of Data Structures in C++/Java, Universities Press.

Theory	MCA103T: Operating System Concepts and Design
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 design principles, structures, and goals of operating systems, and analyze their evolution from traditional to modern OS architectures.
CO2	Apply process management and CPU scheduling techniques, and evaluate concurrency, synchronization, and deadlock-handling strategies in multiprocessor environments.
CO3	Design and analyze memory management schemes, including paging, segmentation, virtual memory, and protection mechanisms for efficient and secure execution.
CO4	Evaluate and implement file system structures, I/O subsystem designs, storage management techniques, and OS-level security principles through case studies of modern operating systems.

UNIT – I : Introduction to Operating Systems 14 Hours

Objectives, role of operating systems in computer systems, fundamental OS design principles (abstraction, mechanism vs. policy, modularity, portability), evolution of OS from batch systems to modern OS. Operating System Design Goals: Efficiency, reliability, security, scalability, and user convenience. Modern OS Design Challenges: Multicore processors, mobile OS vs. desktop OS, real-time operating systems, and energy-aware OS design. OS Structures: Monolithic systems, layered systems, microkernel, modular kernel, hybrid systems, container-based OS structures. System Calls and Interfaces: System call design, user–kernel interface, virtual machines and containers as design concepts-Docker, cgroups, namespaces. Case Study in OS Design: Overview of UNIX/Linux kernel architecture and design decisions.

UNIT – II: Process, Scheduling and Deadlocks 14 Hours

Process concept, states, operations, process control block (PCB), design issues in process management. Threads and Multithreading Models: User-level vs. kernel-level threads, multithreading benefits and challenges in OS design. CPU Scheduling Design: Design objectives, scheduling criteria, evaluation techniques, deterministic modeling, scheduling in multicore environments. Concurrency & Synchronization: Critical section design problems, hardware vs. software solutions, semaphores, monitors, synchronization in multiprocessors. Deadlock Handling: System design issues, prevention, avoidance (Banker's algorithm), detection and recovery strategies.

UNIT – III: Memory Management Strategies 14 Hours

Objectives of memory management, design constraints, OS role in memory hierarchy management. Memory Allocation: Fixed and dynamic partitioning, fragmentation handling. Paging and Segmentation: Design trade-offs, page table structures (hierarchical, inverted), segmentation with paging. Virtual Memory Design: Page replacement policies (FIFO, Optimal, LRU, Clock), thrashing control, working set model, frame allocation policies. Design Issues in Memory Protection: Protection, sharing, kernel memory allocation, memory isolation in virtualized systems.

UNIT – IV: File System, I/O and Security **14 Hours**

File concepts, access methods, directory structures, file system design goals. File System Implementation Issues: Allocation methods (contiguous, linked, indexed), free space management, journaling, log-structured file systems. I/O System Design: I/O hardware, interrupt-driven I/O, device driver design, kernel I/O subsystem. Secondary Storage Management: Disk scheduling design (FCFS, SSTF, SCAN, C-SCAN, LOOK), RAID levels. Security and Protection Design: Goals, access matrix, role-based access control, authentication, encryption, secure OS design principles (least privilege, secure defaults). Case Studies in OS Design: Design principles in Windows NT/Windows 10 and Linux.

TEXT BOOKS:

1. Abraham Silberschatz, Peter B. Galvin, Greg Gagne – Operating System Concepts, 9th Edition, Wiley, 2018.
2. Andrew S. Tanenbaum, Herbert Bos – Modern Operating Systems, 4th Edition, Pearson, 2015.

REFERENCE BOOKS:

1. William Stallings – Operating Systems: Internals and Design Principles, 9th Edition, Pearson, 2018.
2. D. M. Dhamdhere – Operating Systems: A Concept-based Approach, 3rd Edition, McGraw Hill, 2017.

Theory	**MCA104T: Advanced Database Management Systems**
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 architecture, data models, relational concepts, and SQL operations of database management systems, and design simple schemas for real-world applications
CO2	Apply normalization, ER/EER mapping, and concurrency control for consistent and efficient database design.
CO3	Analyze and optimize query processing strategies, and demonstrate the use of distributed, parallel, and data warehouse systems in handling large-scale data
CO4	Evaluate advanced and emerging database models, and apply database security and protection mechanisms in modern applications.

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

Objectives, characteristics, advantages over traditional file systems. Database System Architecture: Levels of abstraction, data independence, DBMS components, system environment. Data Models: Relational, hierarchical, network, Entity–Relationship (ER), and Enhanced ER (EER) models. Relational Model Concepts: Attributes, tuples, relations, keys, constraints. Relational Algebra and Calculus: Basic operations, query formulation, tuple and domain relational calculus. Structured Query Language(SQL): DDL, DML, integrity constraints, simple and nested queries, joins, aggregate functions, views. Case Study: Schema design for a university database.

UNIT – II: Database Design and Recovery **14 Hours**

Functional dependencies, multivalued dependencies, normalization (1NF, 2NF, 3NF, BCNF, 4NF, 5NF). ER/EER-to-Relational Mapping: Design methodology with examples. Transaction Management: ACID properties, transaction states, concurrent executions, schedules. Concurrency Control: Lock-based protocols, two-phase locking, deadlock handling, timestamp ordering, optimistic concurrency control. Recovery System Design: Log-based recovery, checkpoints, shadow paging, ARIES algorithm.

UNIT – III: Query Processing and Optimization **14 Hours**

Steps in query processing, query compilation, query cost estimation, join algorithms, heuristic and cost-based optimization. Distributed Databases: Fragmentation and replication strategies, data transparency, distributed query processing, distributed transaction management, two-phase commit protocol. Parallel Databases: Intra-query and inter-query parallelism, partitioning techniques, parallel join strategies. Data Warehousing and OLAP: Data warehouse architecture, star and snowflake schemas, ETL process, OLAP operations (roll-up, drill-down, slice, dice, pivot). Case Study: Data warehouse design for a retail or banking system.

UNIT – IV: Advanced Data Models **14 Hours**

Object-oriented databases, object-relational databases, XML databases, temporal and spatial databases. NoSQL Databases: Introduction, CAP theorem, data models – key-value stores, column-oriented stores, document-oriented stores, and graph databases. MongoDB: Data model, CRUD operations, indexing, aggregation framework, replication and sharding. Big Data Systems: Introduction to Hadoop ecosystem, HDFS, MapReduce, Spark SQL overview. Database Security and Protection: Discretionary access control, mandatory access control, role-based security, database auditing. Emerging Trends: In-memory databases, cloud databases, blockchain databases, NewSQL systems.

TEXT BOOKS:

1. Abraham Silberschatz, Henry F. Korth, S. Sudarshan – Database System Concepts, 7th Edition, McGraw Hill, 2020.
2. RamezElmasri, Shamkant B. Navathe – Fundamentals of Database Systems, 7th Edition, Pearson, 2017.
3. Raghu Ramakrishnan, Johannes Gehrke – Database Management Systems, 3rd Edition, McGraw Hill, 2003.
4. Jiawei Han, MichelineKamber, Jian Pei – Data Mining: Concepts and Techniques, 3rd Edition, Elsevier, 2011 (for Data Warehousing & OLAP).
5. Kristina Chodorow – MongoDB: The Definitive Guide, 3rd Edition, O'Reilly Media, 2019.

REFERENCE BOOKS:

1. Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer Widom – Database Systems: The Complete Book, 2nd Edition, Pearson, 2009.
2. Markus Winand – SQL Performance Explained, Markus Winand Publishing, 2012.
3. C. J. Date – An Introduction to Database Systems, 8th Edition, Pearson, 2003.
4. Alex Petrov – Database Internals: A Deep Dive into How Distributed Data Systems Work, O'Reilly, 2019.

Theory	MCA105T: Software Project Management
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 software project management, life cycle models, estimation techniques, and planning approaches
CO2	Apply scheduling methods, resource allocation strategies, risk management, and quality frameworks in project design.
CO3	Analyze project monitoring and control techniques, configuration management tools, and contract management practices.
CO4	Evaluate advanced project management practices including Agile, DevOps, communication, leadership, and project closure strategies.

UNIT–I: Introduction to Software Project Management 14 Hours

Objectives, importance of project management in software development, characteristics of software projects compared with traditional engineering projects, and role of project manager. Project Life Cycle and Processes: Project initiation, planning, execution, monitoring, and closure. Agile vs. traditional approaches to project management. Software Project Planning: Effort and cost estimation, work breakdown structure (WBS), project scope management, and deliverables. Estimation Techniques: Lines of Code (LOC), Function Point Analysis (FPA), and COCOMO models (Basic, Intermediate, Detailed).

UNIT–II: Project Scheduling and Quality 14 Hours

Scheduling principles, task allocation, activity planning. Network analysis using CPM and PERT techniques. Resource Allocation and Project Staffing: Human resource planning, staffing patterns, team structures, and software team roles.Risk Management: Identification, risk assessment, quantitative and qualitative analysis, risk mitigation, and contingency planning. Project Quality Management: Concepts of software quality, ISO 9001, CMMI framework, Six Sigma. Quality planning, assurance, and control strategies.

UNIT–III: Project Monitoring and Control 14 Hours

Tracking project progress, milestone analysis, Earned Value Management (EVM). Monitoring cost, schedule, and quality. Software Configuration Management (SCM): Version control, change control, and release management. Tools for SCM (Git, SVN).Contract Management and Procurement: Types of contracts, outsourcing strategies, vendor evaluation, SLA (Service Level Agreement).Case Studies in Project Control: Success and failure case studies of software projects with emphasis on monitoring and design decisions.

UNIT–IV: Advanced Topics in Project Management 14 Hours

Agile project management (Scrum, Kanban), hybrid models, DevOps integration. Project Communication and Leadership: Communication planning, stakeholder management, leadership styles, and conflict resolution in software teams. Project Closure and Evaluation: Steps in closing projects, post-project analysis, knowledge management, project maturity models. Project Management Tools and Environments: Use of MS Project, JIRA, Trello, and other PM tools. Emerging trends in project execution and design.

TEXT BOOKS:

1. Bob Hughes, Mike Cotterell, Rajib Mall – Software Project Management, 6th Edition, McGraw Hill, 2017.
2. Walker Royce – Software Project Management: A Unified Framework, Addison-Wesley, 1998.
3. Pankaj Jalote – Software Project Management in Practice, Pearson, 2002.

REFERENCE BOOKS:

1. Roger S. Pressman, Bruce R. Maxim – Software Engineering: A Practitioner's Approach, 9th Edition, McGraw Hill, 2019.
2. Harold Kerzner – Project Management: A Systems Approach to Planning, Scheduling, and Controlling, 12th Edition, Wiley, 2022.
3. Ian Sommerville – Software Engineering, 10th Edition, Pearson, 2016.
4. Mike Cohn – Agile Estimating and Planning, Prentice Hall, 2005.
5. Ken Schwaber – Agile Project Management with Scrum, Microsoft Press, 2004.

Theory	**MCA101E: Mobile Computing**
Teaching Hours: 04 Hours/Week	**Credits: 04**
Duration of Exam: 03 Hours	**Maximum Marks: 100 (Exam 70 + IA 30)**

Course Outcomes

COs	**Description**
CO1	To understand the evolution, principles, and challenges of mobile computing and wireless communication systems.
CO2	Apply networking and transport layer protocols, wireless LAN standards, and routing mechanisms to support mobility in computing environments.
CO3	Analyze the design of mobile systems, operating platforms, and application development processes with considerations for usability and resource efficiency.
CO4	Evaluate emerging trends, security mechanisms, and real-world applications of mobile computing in diverse domains.

UNIT–I: Introduction to Mobile Computing **14 Hours**

Evolution of mobile computing, characteristics and applications, limitations and challenges. Mobile and Wireless Communication Fundamentals: Spectrum, frequency allocation, multiplexing (FDMA, TDMA, CDMA, OFDMA). Cellular Systems: GSM architecture, GPRS, 3G/4G/5G networks. Wireless Transmission: Medium characteristics, modulation, signal propagation, multiplexing and spread spectrum techniques. Mobile IP and Wireless Access Protocols: Mobile IP features, tunneling, handoff strategies, WAP architecture, protocols, and applications.

UNIT–II: Mobile Network and Transport Layer **14 Hours**

Mobile TCP, indirect and snooping TCP, fast retransmission and recovery, transaction-oriented TCP. Ad Hoc Networks: Properties, applications, and challenges; routing protocols (AODV, DSR, DSDV, ZRP). Wireless LANs: IEEE 802.11 architecture, Bluetooth, Wi-Fi standards, MAC management, and QoS support. Mobile Application Support Protocols: DNS, DHCP, HTTP adaptations in mobile environments.

UNIT–III: Mobile Architecture and Applications **14 Hours**

Mobile device architecture, hardware considerations, energy management, and power-aware design. Mobile Operating Systems: Features of Android, iOS, and other platforms; middleware support. Mobile Application Development Concepts: Architecture of mobile apps, development lifecycle, UI/UX considerations. Location Management: Principles of location tracking, location-based services (LBS), GPS, assisted GPS, and mobile positioning techniques.

UNIT–IV: Emerging Trends in Mobile Computing **14 Hours**

Internet of Things (IoT) and mobile devices, mobile cloud computing, and fog computing. Mobile Security: Security requirements, authentication, encryption, secure routing, and intrusion detection in wireless and mobile networks. Mobile Databases: Characteristics, transaction models, synchronization issues, and data dissemination. Case Studies and Applications: Mobile payment systems, healthcare apps, smart cities, AR/VR-based mobile applications.

TEXT BOOKS:

1. Jochen Schiller – Mobile Communications, 2nd Edition, Pearson Education, 2012.
2. Ashok K. Talukder, Roopa R. Yavagal – Mobile Computing: Technology, Applications, and Service Creation, 2nd Edition, McGraw Hill, 2010.
3. Raj Kamal – Mobile Computing, Oxford University Press, 2018.

REFERENCE BOOKS:

1. William Stallings – Wireless Communications and Networks, 2nd Edition, Pearson, 2015.
2. Martyn Mallick – Mobile and Wireless Design Essentials, Wiley, 2003.
3. Frank Adelstein et al. – Fundamentals of Mobile and Pervasive Computing, McGraw Hill, 2005.
4. Wei-Meng Lee – Beginning Android Programming with Android Studio, Wiley, 2015.
5. Sandeep Kumar – Mobile Computing, Pearson, 2021.

Theory	MCA102E: Software Testing with Selenium
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 principles of software testing, the testing life cycle, and the role of testing across different stages of software development.
CO2	Apply test design techniques, test management practices, and automation concepts to improve the effectiveness and efficiency of testing processes.
CO3	Analyze and implement automated testing frameworks using industry-standard tools, incorporating user interaction handling, reporting, and design patterns.
CO4	Evaluate advanced automation strategies, integration with continuous delivery pipelines, and emerging trends in test automation for real-world applications.

UNIT–I: Introduction to Software Testing **14 Hours**

Fundamentals of testing, verification vs. validation, quality assurance vs. quality control, defect life cycle, cost of defects. Software Development Life Cycle (SDLC) and Testing: Testing in different phases – unit, integration, system, and acceptance testing. Software Testing Life Cycle (STLC): Requirement analysis, test planning, test design, test execution, defect reporting, test closure. Test Levels and Types: Black box vs. white box testing, static vs. dynamic testing, regression testing, re-testing, smoke and sanity testing, exploratory testing.

UNIT–II: Test, Management and Automation **14 Hours**

Equivalence class partitioning, boundary value analysis, decision tables, state transition, use case testing, error guessing. Test Management: Test planning, test strategy, metrics, and reporting. Test Automation Concepts: Advantages, limitations, criteria for automation. Selenium Overview: Introduction to Selenium, features, Selenium components (Selenium IDE, Selenium WebDriver, Selenium Grid), and installation setup.

UNIT–III: Selenium WebDriver **14 Hours**

Architecture, browser drivers, locators (ID, name, class, link text, XPath, CSS selectors), handling dynamic elements. Automating User Actions: Keyboard and mouse events, handling alerts, popups, frames, windows, waits (implicit and explicit). TestNG Framework: Features, annotations, grouping, parameterization, parallel test execution, reporting. Page Object Model (POM): Concepts, implementation, benefits in test design.

UNIT–IV: Advanced Selenium Concepts **14 Hours**

Data-driven testing using Apache POI, cross-browser testing, distributed testing with Selenium Grid. Continuous Integration (CI): Integrating Selenium with Jenkins, Maven, and Git. Selenium with BDD: Introduction to Cucumber, Gherkin syntax, writing feature files, step definitions, and executing scenarios. Emerging Trends in Test Automation: Mobile application testing with Appium, AI in test automation, cloud-based testing platforms (e.g., BrowserStack, Sauce Labs). Case Studies: Automating a sample web application using Selenium end-to-end.

TEXT BOOKS:

1. Paul C. Jorgensen – Software Testing: A Craftsman's Approach, 4th Edition, CRC Press, 2013.
2. Unmesh Gundecha – Selenium WebDriver Practical Guide, Packt Publishing, 2015.
3. Navneesh Garg – Test Automation using Selenium WebDriver with Java, Kindle Edition, 2016.

REFERENCEBOOKS:

1. Glenford J. Myers, Corey Sandler, Tom Badgett – The Art of Software Testing, 3rd Edition, Wiley, 2011.
2. Rex Black – Practical Guide to Software Testing, Wiley, 2002.
3. Alan Richardson – Selenium Simplified: A Tutorial Guide, Compendium Developments, 2015.
4. Mark Collin – Mastering Selenium WebDriver, Packt Publishing, 2015.
5. Ajay Pandey – Hands-on Selenium WebDriver with Java, BPB Publications, 2020.

Practical	**MCA101P: Data Structures using JAVA**
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.
15. Implement a Java program to perform Breadth First Search (BFS) and Depth First Search (DFS) on a graph.
16. Write a Java program to implement a hash table using linear probing for collision handling.

Practical	MCA102P: Database Management Systems lab
Teaching Hours: 04 Hours/Week	Credits: 02
Duration of Exam: 03 Hours	Maximum Marks:50 (Exam 35 + IA 15)

PART A

1. **Create and Manage University Database- Perform the following operations:**

- Create tables: Student, Course, and Enrolment with appropriate attributes and constraints (Primary key, Foreign key).
- Insert minimum 5 records into each table.
- Display all student details enrolled in a specific course.
- Update a student's name using UPDATE command.
- Delete a course using DELETE command.

2. **Perform the following operations:**

- Create tables: `Employee`, `Department`, and `Salary`.
- Insert sample records into each.
- Display list of employees working in a specific department using `JOIN`.
- Display total salary paid per department using `GROUP BY` and aggregate function.
- Create a view that displays employee name, department, and salary.

3. **Perform the following operations:**

- Create tables: Books, Authors, Borrowers.
- Insert sample records for at least 5 books and 3 borrowers.
- Retrieve book details borrowed by a specific borrower using nested queries.
- Count how many books each borrower has taken using aggregate functions.
- Create a view showing borrower name and books borrowed.

4. **Perform the following operations:**

- Create tables: `Patient`, `Doctor`, and `Appointment`.
- Insert sample data for at least 5 patients and 3 doctors.
- Retrieve patient names who have appointments with a specific doctor.
- Display total number of appointments per doctor.
- Create an index on `Doctor_ID`

5. **Execute the following queries:**

- Create tables: `Customer`, `Product`, and `Orders`.
- Insert sample data for 5 customers and 5 products.
- Display all products ordered by a specific customer using JOIN.
- Find customers who haven't placed any orders using `NOT IN`.
- Create a view that shows product name, quantity, and customer name.

6. **Execute the following queries:**

Given a single unnormalized `Sales` table (book_id, book_title, author_name, customer_name, customer_address, purchase_date):

- Convert it to 1NF.
- Further convert to 2NF.
- Normalize to 3NF and show the resulting tables.
- Explain primary and foreign keys in the resulting schema.

7. **Perform the following operations:**

- Create tables: `Teacher`, `Class`, `Subject`.
- Insert 5 records each.
- Display subject-wise list of teachers using JOIN.
- Count how many teachers are assigned per class.
- Create a view to show class name and total number of subjects taught.

8. **Create a table Account(Account_No, Customer_Name, Balance) with appropriate data types and primary key constraint.**

- Insert at least 3 sample records into the table.
- Write a PL/SQL block that:
    - Accepts an account number and withdrawal amount.
    - Checks if the account has sufficient balance.
    - If sufficient, deducts the amount and updates the balance.
    - If not, displays an appropriate message.
    - Includes exception handling for invalid account numbers and other errors.

PART B

9. Create a table Bank_Account(Account_No, Customer_Name, Balance) with suitable data types. Execute the following:

- Insert at least 3 records into the table.
- Write a PL/SQL block to simulate two transactions:
- Use SAVEPOINT to mark intermediate points.
- Use ROLLBACK to revert to a savepoint.
- Use COMMIT to finalize changes.
- Display the changes before and after rollback and commit to observe the effects.

10. Create a MongoDB collection named Students with fields such as student_id, name, department, and marks.

- Perform the following operations using appropriate MongoDB commands:
- Insert at least 5 student records.
- Update the marks of a student based on student_id.
- Delete a student record.
- Search for all students in a particular department.

11. **Perform the following operations:**

- Use the Students collection created earlier with fields including department and marks.
- Perform aggregation operations using MongoDB's aggregation pipeline:
- Count the number of students in each department.
- Calculate the average marks per department.

12. **Perform the following operations:**

- Create a sample .txt file containing plain English text.
- Upload the file to the Hadoop Distributed File System (HDFS).
- Run the default Hadoop**WordCount** program to process the uploaded file.
- Display and interpret the output that shows word frequencies.

13. **Perform the following operations:**

- Load a CSV file containing student information with fields such as student_id, name, GPA, and department using Spark SQL.
- Register the data as a temporary view or DataFrame.
- Write and execute queries to:
- Display all student records.
- Filter and show students with GPA > 8.
- Group students by department and display count.

14. **Perform the following operations:**

- Create two new users in the database with appropriate usernames and passwords.
- Grant SELECT and INSERT privileges on a table (e.g., Students) to one of the users.
- Log in as that user and perform operations to verify access.
- Revoke the privileges and attempt the same operations to observe the effect.
- Discuss the difference between **GRANT** and **REVOKE** operations.

SEMESTER–II

Theory	MCA201T: 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 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 14Hours

Searching- Searching for solutions, uniformed 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-Adversial 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 14Hours

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 & lifts 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 14Hours
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.

TEXTBOOKS:

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	**MCA202T: Advanced 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	Demonstrate algorithmic foundations, problem-solving strategies, and asymptotic complexity analysis.
CO2	Apply greedy techniques and graph algorithms to solve optimization and traversal problems effectively.
CO3	Develop solutions using dynamic programming and backtracking approaches for complex computational problems.
CO4	Analyze advanced algorithmic strategies such as branch and bound, approximation, randomized methods, and evaluate computational complexity classes.

UNIT- I: Fundamentals of Algorithms 14 Hours
Algorithms and problem-solving techniques, algorithm design fundamentals, asymptotic notation and complexity classes, recurrence relations and solving techniques, analysis of recursive and non-recursive algorithms, divide-and-conquer strategy – binary search, merge sort, quick sort, matrix multiplication, Strassen's algorithm.

UNIT –II: Graph Algorithms **14 Hours**

Graph traversal algorithms – breadth-first search (BFS), depth-first search (DFS), minimum spanning trees – Prim's and Kruskal's algorithms, shortest path algorithms – Dijkstra's and Bellman-Ford, greedy method – activity selection, Huffman coding, job sequencing with deadlines, greedy knapsack problem.

UNIT –III: Dynamic Programming and Backtracking **14 Hours**

Dynamic programming – matrix chain multiplication, longest common subsequence (LCS), optimal binary search trees, 0/1 knapsack problem, Floyd-Warshall algorithm; Backtracking – n-queens problem, graph coloring, Hamiltonian cycle, subset sum problem.

UNIT –IV: Branch and Bound **14 Hours**

Branch and bound – traveling salesman problem (TSP), knapsack problem; Approximation algorithms – vertex cover, set cover; Complexity theory – P, NP, NP-complete, and NP-hard problems; Introduction to randomized algorithms; String matching algorithms – Knuth-Morris-Pratt (KMP), Rabin-Karp.

TEXT BOOKS:

1. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, Introduction to Algorithms, 3rd Edition, MIT Press, 2009.

REFERENCEBOOKS:

1. Jon Kleinberg, ÉvaTardos, Algorithm Design, Pearson, 2006.
2. Ellis Horowitz, SartajSahni, SanguthevarRajasekaran, Fundamentals of Computer Algorithms, 2nd Edition, Universities Press, 2008.
3. Steven S. Skiena, The Algorithm Design Manual, 2nd Edition, Springer, 2008.

Theory	**MCA203T: 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	Understand Python programming fundamentals including syntax, control structures, functions and string manipulations.
CO2	Apply built-in and advanced data structures, file handling techniques, and exception management to solve computational problems.
CO3	Develop object-oriented solutions in Python by applying principles of abstraction, encapsulation, inheritance, and polymorphism.
CO4	Analyze and visualize data, implement system-level programming concepts, and integrate Python with external resources to build real-world applications.

UNIT- I: Basics of Python **14 Hours**

Python Interpreter/Shell, Identifiers, Keywords, Statements, Expressions, Variables, Operators, Precedence, Data Types, Indentation, Comments, Input/Output, Type Conversions, type() and is, Mutable vs Immutable, Dynamic Typing. Control Flow: if, else, elif, nested blocks. Loops: for, while, range, break, continue, pass, else. Functions: Definitions, Built-in Functions, Parameters (*args, **kwargs), Scope, Lambda, Recursion, Command Line Arguments. Strings: Operations, Slicing, Methods, String Formatting, Regular Expressions.

UNIT –II: Data Structures in Python **14 Hours**

Lists, Tuples, Dictionaries, Sets: Creation, Indexing, Slicing, Comprehensions, Built-in Methods, Nested Structures. Advanced Collections: defaultdict, Counter, deque, heapq. Iterables and Iterators. File Handling: Text and Binary Files, with statement, File Methods, Exception Handling.CSV and JSON Files, Serialization using pickle.

UNIT –III: Object-Oriented Concepts **14 Hours**

Classes and Objects, Constructors, Destructors, Instance vs Class Variables, Methods, Static and Class Methods, Inheritance, Polymorphism, Method Overriding, Operator Overloading, Encapsulation, Abstract Classes. Advanced OOP Concepts: Iterators, Generators, Decorators, Regular Expressions, Python Memory Management and Garbage Collection. Exception Handling: try, except, finally, Raising and Creating Custom Exceptions. CoreLibraries: NumPy: Arrays, Indexing, Broadcasting, Mathematical Operations. Pandas: Series, DataFrames, Indexing, Filtering, Handling Missing Data, Grouping, Merging.

UNIT –IV: Data Handling, Visualization, and System Programming **14 Hours**

Data handling using CSV, JSON, and APIs, Data cleaning and preprocessing using Pandas, Web scraping basics, NumPy arrays and operations, Pandas Series and DataFrames, Indexing and aggregation, Visualization using Matplotlib (line, bar, scatter, histograms), Seaborn (heatmaps, boxplots, pairplots), Plotly (interactive charts, geo-visualization), Basics of socket programming, Multithreading and multiprocessing in Python, Database connectivity using SQLite and MySQL, Mini projects on data analysis, random walks, and dice simulation.

TEXT BOOKS:

1. Mark Lutz, Learning Python, 5th Edition, O'Reilly, 2013.

REFERENCE BOOKS:

1. Allen B. Downey, Think Python: How to Think Like a Computer Scientist, 2nd Edition, O'Reilly, 2015.
2. Charles Dierbach, Introduction to Computer Science Using Python, Wiley, 2015. |
3. Wesley J. Chun, Core Python Applications Programming, 3rd Edition, Prentice Hall, 2012

Theory	MCA204T: Computer 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 fundamental concepts of computer networks, network models, physical layer transmission media, and performance metrics.
CO2	Apply error detection/correction methods, data link protocols, and medium access control techniques for reliable and efficient data transmission.
CO3	Analyze network layer functions, addressing schemes, routing algorithms, IPv6 protocols, and emerging networking paradigms such as SDN and NFV.
CO4	Evaluate transport and application layer protocols, congestion control mechanisms, QoS strategies, and basic network security and cloud networking concepts.

UNIT- I: Introduction to Networks 14 Hours

Introduction to Data Communications and Networks, Network Types (LAN, MAN, WAN, PAN, Wireless Networks), Internet Evolution and Architectures, Network Models: Protocol Layering, OSI Model in-depth, TCP/IP Protocol Suite, Physical Layer: Transmission Media Types (guided and unguided), Transmission Impairments and Noise, Channel Capacity and Shannon's Theorem, Performance Metrics (throughput, latency, jitter, bandwidth).

UNIT –II: Data Link Layer and Advanced MAC Techniques 14 Hours

Data Link Layer: Link-Layer Addressing and ARP, Error Detection and Correction: Block Coding, Cyclic Redundancy Check (CRC), Checksum Techniques, Data Link Control Protocols (HDLC, PPP), Logical Link Control (LLC).Medium Access Control (MAC) Sublayer: Random Access Protocols - Pure and Slotted ALOHA, CSMA, CSMA/CD, CSMA/CA, Carrier Sensing Mechanisms. Advanced Scheduling Techniques: Reservation Systems, Polling, Token Passing (Ring and Bus Topologies).Channelization Techniques: FDMA, TDMA, CDMA, OFDMA, Introduction to MAC Protocols in Wireless Networks (Wi-Fi, Bluetooth).

UNIT –III: Network Layer and Routing Protocols with IPv6 and Mobile IP 14 Hours

Network Layer Services and Functions, Packet Switching Concepts, Network Layer Performance Metrics.IPv4 Addressing and Subnetting, Classful and Classless Addressing, CIDR, NAT, DHCP. Network Layer Protocols: IP, ICMPv4, Mobile IP Fundamentals. Routing Algorithms: Distance Vector, Link State, Path Vector. Unicast Routing Protocols: RIP, OSPF, BGP. IPv6 Addressing and Header Structure, IPv6 Extensions and Transition Mechanisms, IPv6 Routing. Introduction to Software Defined Networking (SDN) and Network Function Virtualization (NFV).

UNIT –IV: Transport Layer, Application Layer Protocols, and QoS 14 Hours

Transport Layer Services and Protocols: UDP and TCP – Features, Segment Structure, Connection Management (3-way handshake, termination), Flow Control, Error Control, Congestion Control Algorithms (Tahoe, Reno, Vegas).Advanced TCP Concepts: TCP Variants,

SCTP Overview. Application Layer Protocols: HTTP/HTTPS, FTP, SMTP, POP3, IMAP, DNS Architecture and Resolution Process. Quality of Service (QoS): QoS Requirements, Flow Control Mechanisms, Integrated Services (IntServ), Differentiated Services (DiffServ), Traffic Shaping and Policing. Introduction to Network Security Concepts (basic cryptography, VPNs), Cloud Networking Overview.

TEXT BOOKS:

1. Andrew S. Tanenbaum, David J. Wetherall, Computer Networks, 5th Edition, Pearson, 2010.

REFERENCE BOOKS:

1. Behrouz A. Forouzan, Data Communications and Networking, 5th Edition, McGraw Hill, 2013.
2. William Stallings, Data and Computer Communications, 10th Edition, Pearson, 2013.
3. James F. Kurose, Keith W. Ross, Computer Networking: A Top-Down Approach, 6th Edition, Pearson, 2013.

Theory	**MCA205T: Cryptography and Information Security**
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 fundamental concepts of security and classical cryptography to demonstrate basic encryption and decryption techniques.
CO2	Apply symmetric and asymmetric cryptographic algorithms, key management, and digital signature techniques for secure communication.
CO3	Analyze the role of cryptographic hashing, authentication methods, and network security protocols in protecting data and communication.
CO4	Evaluate system-level threats, secure coding practices, cloud and application security mechanisms, and emerging security trends.

Unit- I: Security Fundamentals and Classical Cryptography 14 Hours

Goals of Security- Confidentiality, Integrity, Availability, Authentication, Non-repudiation, Security attacks and threats, Security services and mechanisms. Classical Cryptography: Caesar cipher, Monoalphabetic and Polyalphabetic substitution ciphers, Playfair cipher, Transposition ciphers. Introduction to Modern Cryptography: Overview of symmetric and asymmetric schemes, Cryptanalysis basics.

Unit –II: Symmetric and Asymmetric Cryptography 14 Hours

Symmetric Key Cryptography: Detailed study of DES, 3DES, AES (with focus on AES-128, AES-256), modes of operation (ECB, CBC, CFB, OFB, CTR), Key management techniques. Asymmetric Cryptography: RSA algorithm (key generation, encryption/decryption), Diffie-Hellman key exchange, Elliptic Curve Cryptography (ECC), Digital Signatures (RSA and ECC based), Introduction to Post-Quantum Cryptography concepts.

Unit –III: Hashing and Network Security **14 Hours**

Cryptographic Hash Functions: MD5, SHA-1, SHA-2 family (SHA-256, SHA-512), SHA-3, HMAC, Applications of hash functions. Authentication Protocols: Kerberos, Challenge-Response Mechanisms, Password-based authentication, OAuth and OpenID Connect (overview).Network Security Protocols: SSL/TLS versions and handshake protocols, IPsec (AH, ESP), VPNs, Firewalls – Types and Architecture, Intrusion Detection and Prevention Systems (IDS/IPS).

Unit –IV: System Security and Applications **14 Hours**

Malware: Viruses, Worms, Trojans, Ransomware, Advanced Persistent Threats (APT).Antivirus and Endpoint Security Solutions. Secure Coding Principles and Common Vulnerabilities (OWASP Top 10), Buffer Overflow, SQL Injection, Cross-site Scripting (XSS).Social Engineering Attacks and Countermeasures. Security in E-commerce: SSL/TLS for secure transactions, Payment Security standards (PCI-DSS).Cloud Security Fundamentals: Cloud service models, Shared responsibility model, Data protection in cloud, Identity and Access Management (IAM).Emerging Topics: Blockchain basics, Zero Trust Security Model, Security considerations in IoT and Mobile Devices.

TEXT BOOKS:

1. William Stallings, Cryptography and Network Security: Principles and Practice, 6th Edition, Pearson, 2014.

REFERENCEBOOKS:

1. Behrouz A. Forouzan, Cryptography and Network Security, McGraw Hill, 2015.
2. Bruce Schneier, Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd Edition, Wiley, 2015.
3. Charlie Kaufman, Radia Perlman, Mike Speciner, Network Security: Private Communication in a Public World, 2nd Edition, Pearson, 2016.

Theory	**MCA201E (a) – Automata Theory**
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 the principles of formal proof, mathematical induction, and finite automata to represent and simplify regular languages.
CO2	Analyze and construct regular expressions, grammar and context-free languages using closure properties, pumping lemmas, and normal forms.
CO3	Design and compare computational models such as PDA, DPDA, and transducers for acceptance of CFLs and evaluate their equivalence with CFGs.
CO4	Evaluate the power and limitations of Turing machines by solving problems on decidability, undecidability, and reductions.

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

The Principle of Mathematical Induction, Introduction to formal proof, Additional forms of Proof, Inductive Proofs. Finite Automata: Introduction, Deterministic Finite Automata (DFA): Formal definition, simpler notations (state transition diagram, transition table), language of a DFA. Nondeterministic Finite Automata (NFA): Definition of NFA, language of an NFA, Equivalence of Deterministic and Nondeterministic Finite Automata, Applications of Finite Automata, Finite Automata with Epsilon Transitions, Eliminating Epsilon transitions, Minimization of Deterministic Finite Automata.

UNIT –II: Regular Expressions **14 Hours**

Introduction, Identities of Regular Expressions, Finite Automata and Regular Expressions- Converting from DFA's to Regular Expressions, Converting Regular Expressions to Automata, applications of Regular Expressions. Regular Grammars: Definition, regular grammars and FA, FA for regular grammar, Regular grammar for FA. Proving languages to be non-regular. Properties of Regular Languages: The Pumping Lemma for regular languages, Applications of the pumping lemma closure properties of regular languages, Decision properties of regular languages, Equivalence and minimization of automata. Context Free Grammer (CFG): Derivation Trees, Sentential Forms, Rightmost and Leftmost derivations of Strings. Ambiguity in CFG's, Minimization of CFG's, CNF, GNF, Pumping Lemma for CFL's, Enumeration of Properties of CFL.

UNIT –III: Pushdown Automata Introduction **14 Hours**

Definition, Formal definition of pushdown automata, A graphical notation for PDA's, Instantaneous descriptions of a PDA Pushdown Automata: Definition, Model, Acceptance of CFL, Acceptance by Final State and Acceptance by Empty stack and its Equivalence, Equivalence of CFG and PDA. Deterministic Pushdown Automata: Definition of a deterministic PDA, Regular languages and deterministic PDA's, DPDA's and context-free languages, DPDA's and ambiguous grammars. Transducers: Moore machine, Mealy machine, Difference between Moore & Mealy machines, Properties, Equivalence of Moore & Mealy machines. Context Sensitive Languages: Linear bounded automata, Chomsky's hierarchy of languages.

UNIT –IV: Introduction to Turing Machines **14 Hours**

The Turing Machine: The instantaneous descriptions for Turing machines, Transition diagrams for Turing machines, The language of a Turing machine, Turing machines and halting programming techniques for Turing machines, Extensions to the basic Turing machine, Restricted Turing machines, Turing machines and computers. Undecidability: A language that is not recursively enumerable, Enumerating the binary strings, Codes for Turing machines, the diagonalization language, An undecidable problem that is RE: Recursive languages, Complements of recursive and RE languages, The universal languages, Undecidability of the universal language. Undecidable Problems About Turing Machines: Reductions, Turing machines that accept the empty language. Post's correspondence problem: Definition of post's correspondence problem, The "Modified" PCP, Other undecidable problems: Undecidability of ambiguity for CFG's. Unsolvable Problems and Computable Functions: A no recursive Language and Unsolvable Problem, Reducing one problem to another: The Halting Problem, Other unsolvable Problems involving TMs, Rice's Theorem and More Unsolvable problems.

TEXT BOOKS:

1. John E. Hopcroft , Rajeev Motwani, Jeffrey D. Ullman (2007), Introduction to Automata Theory, Languages and Computation, 3rd Edition, Pearson Education, India.
2. K L P Mishra, N. Chandrashekaran (2003), Theory of Computer Science-Automata Languages and Computation, 2nd Edition, Prentice Hall of India, India.

REFERENCE BOOKS:

1. Harry. R. Lewis and C. H. Papadimitriou - Elements of the Theory of Computation, Second Edition, PHI, 2003,
2. John C. Martin - Introduction to Languages and the Theory of Computation, Fourth Edition, TMH, 2011
3. MichealSipser - Introduction of the Theory and Computation, Thomson Brokecole, Second Edition, 1997 4. C K. Nagpal - Formal Languages and Automata Theory, Oxford Higher Education, April 2011.

Theory	**MCA201E (b) – Digital Image Processing**
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 image formation, sensing, sampling, quantization, and apply enhancement techniques in both spatial and frequency domains.
CO2	Demonstrate proficiency in various image transforms and employ filtering techniques for image restoration under different degradation models.
CO3	Utilize compression methods and perform morphological operations (dilation, erosion, opening, closing) to solve real-world image processing tasks.
CO4	Evaluate segmentation strategies, extract meaningful features, and perform object recognition/classification for applications in fields like medical imaging, remote sensing, and computer vision.

UNIT- I: Introduction **14 Hours**

Fundamentals of digital image processing, human visual system, image sensing and acquisition, sampling and quantization. Image enhancement – spatial domain and frequency domain techniques, histogram equalization, smoothing and sharpening filters.

UNIT –II: Image Transforms **14 Hours**

Fourier transform, DFT, DCT, Walsh, Hadamard, and Wavelet transforms. Image restoration – degradation models, noise models, inverse filtering, Wiener filtering.

UNIT –III: Image Compression **14 Hours**

Lossless and lossy compression, Huffman coding, run-length coding, JPEG, MPEG. Morphological image processing – dilation, erosion, opening, closing, hit-or-miss transform, applications.

UNIT –IV: Image Analysis **14 Hours**

Segmentation techniques – edge detection, thresholding, region-based segmentation. Feature extraction – shape, texture, color. Object recognition and classification. Applications in medical imaging, remote sensing, and computer vision.

TEXT BOOKS:

1. Rafael C. Gonzalez, Richard E. Woods – Digital Image Processing, 4th Edition, Pearson, 2017
2. Anil K. Jain – Fundamentals of Digital Image Processing, 1st Edition, Prentice Hall (PTR).

REFERENCE BOOKS:

1. Bhabatosh Chanda, Dwijesh Dutta Majumder – Digital Image Processing and Analysis, 2nd Edition, PHI Learning Pvt. Ltd., 2011

Practical	**MCA201P: Artificial Intelligence Using Python 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 program to implement uninformed search (BFS, DFS).
2. Write a program to implement informed search (A*).
3. Write a program to solve 8-puzzle problem.
4. Write a program to implement constraint satisfaction problem.
5. Write a program for hill climbing algorithm.
6. Write a program for genetic algorithm.
7. Write a program for decision tree learning.
8. Write a program for naïve Bayes classifier.
9. Write a program for perceptron learning.
10. Write a program for feedforward neural network.

PART B

11. Write a program to perform text classification using scikit-learn.
12. Write a program to implement sentiment analysis using Python.
13. Write a program to implement image classification using Keras.
14. Write a program to build a chatbot using NLP.
15. Write a program to implement reinforcement learning (Q-learning).
16. Write a program to apply clustering (K-means).
17. Write a program to apply regression analysis.
18. Write a program to perform named entity recognition (NER).
19. Write a program to visualize decision boundaries.
20. Write a program for recommendation system using collaborative filtering.

Practical	MCA202P: Advanced Algorithms using Java 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 program to perform file handling operations in Java.
2. Write a program to implement sorting algorithms in Java.
3. Write a program to apply searching techniques in Java.
4. Write a program to perform matrix operations.
5. Write a program to demonstrate exception handling.
6. Write a program to implement multithreading.
7. Write a program to use Java collections (List, Set, Map).
8. Write a program to demonstrate JDBC connectivity.
9. Write a program to read and process CSV files.
10. Write a program for object serialization and deserialization.

PART B

11. Write a program for regression analysis in Java.
12. Write a program to perform clustering using Java libraries.
13. Write a program for association rule mining.
14. Write a program to implement text mining with Java.
15. Write a program for visualization using JavaFX.
16. Write a program to integrate Java with MongoDB.
17. Write a program for sentiment analysis with Java.
18. Write a program for recommendation system.
19. Write a program for classification using Weka in Java.
20. Write a program for time-series forecasting in Java.