Revised Curriculum-2023

Master of Computer Applications

DEPARTMENT OF COMPUTER SCIENCE

JAMIA MILLIA ISLAMIA

Approved by the BoS held on 09.05.2023

© 2023, Jamia Millia Islamia, All Rights Reserved

INTRODUCTION

The Department of Computer Science is one of the youngest constituent units of the Faculty of Natural Sciences. It was established in the year 1999 with the objective of producing IT professionals of international standard and primary fulfill requirements of the booming IT industry and developing researchers. The Department offers three postgraduate programs including the regular programs MCA and PGDCA, and a self-financed evening program M.Sc. (Bioinformatics). In addition, it supports the execution of an undergraduate program B.Sc. in terms of the teaching of Computer Science as a subject. The objective of the course, MCA, is to prepare skilled and capable Computer professionals with a strong Conceptual background in Computer Science and Applications to solve challenging industrial problems. This program comprises courses in different areas of Computer Science, Computer Applications, Business Systems, and Mathematics. The rigorous exposure in these areas is aimed additionally at equipping the students with strong analytical and technical skills to work competently in diverse areas including software-related developments, teaching, and research.

PROGRAMME STRUCTURE

In order to achieve the aims and objectives set forth at the outset, a programme structure under the semester-based credit system is prescribed as follows. The programme structure prescriptions include the titles of the theory/lab courses along with L-T-P and assigned credits. Further, the respective detailed syllabi of theory courses are based largely on the latest available edition of the book prescribed first in the list, with minor portions or special-topics-of-interest are based on the respective reference books. Moreover, prescriptions of ten generic practical assignments are included as suggestive lab exercises, on which specific assignments for student(s) may be worked upon by the concerned teachers.

HIGHLIGHTS: MCA (2 YEARS) - REVISED CURRICULUM 2023

A.	**Motivation**	• Rapidly changing academic and technological scenarios around the world. • Conformance to guidelines from UGC, AICTE, NEP, and JMI ordinances. • Keeping abreast with the current and future industrial skill-set requirements, and placement-related tests. • Utilization of available faculty expertise/strength for the benefit of students. • Imminent revision on observations of FM, students and other stakeholders.
B.	**Bases and Constraints**	Guideline on Choice Based Credit System (CBCS) – with greater autonomy to students on the selection of courses – by reinforcing the following: 1. Keeping an enhanced set of Core, Elective, and CBCS courses. 2. All CBCS courses are to be open for any eligible student and vice-versa.
C.	**Course L-T-P**	• Theory Courses: 3-1-0 • Lab Courses: 0-0-4 • Elective/CBCS Courses: 3-1-0/3-0-2 • Minor Project: 0-1-6 • Major Project: 0-4-24
D.	**Special Considerations**	• Accommodation of Computer Science and supportive mathematical, management and interdisciplinary courses. • Applied computing and the professional nature of programs. • Balancing academic, technological, and industrial imperatives/prospects.
E.	**Better Skilling**	• PG Programs: Dedicated (Lab-courses-7, Minor-Project-1 & Major Project-1.
F.	**Bridge Courses**	Non-CS entrants must produce acceptable proof of passing at least 3 credits course in C/C++ programming of the UG/PG level. Or complete at least 3 credits course in C/C++ programming either from the Department or MOOC (NPTEL), as advised by the department.
G.	**MOOC (NPTEL) courses as CBCS**	• Students may choose CBCS courses (in consultation with co-ordinator) from approved list of NPTEL courses following the Department/University guidelines.
H.	**Remarks**	• Students are encouraged to register for courses of their interest/specialization (e.g. Communication/Management) from other Departments as CBCS courses. • CBCS courses of a minimum of 3 credits each that may be chosen from other departments subject to students' requirements and convenience. • Lab and project courses shall have independent practical and viva-voce examinations. • At least two typical elective courses may be offered, as long as at least 15 students sign up for each one – and these may be open to all PG students of the Department. • Department may float any other elective, beyond the listed ones, subject to demand, feasibility and endorsement of BOS. • Faculty members may be allotted two periods per week as a Tute-Slot in order to facilitate individual/small-group interaction for clarifying, discussing, submitting assignments, etc.

MCA PROGRAMME STRUCTURE (REVISED 2023)

SEM	CODE	COURSE TITLE	L-T-P	CR	Marks	SUMMARY
Bridge Course	CA01	Programming with C	3-0-2	4	100	Periods: 4 Credits: 4 Marks: 100
I	CA11	Algorithmics & Program Design	3-1-0	4	100	Periods: 32 Credits: 28 Total Marks: 800
	CA12	Digital Logic & Computer Architecture	3-1-0	4	100	
	CA13	Theoretical Computer Science	3-1-0	4	100	
	CA14	Database Management Systems	3-1-0	4	100	
	CA15	Programming with Java	3-1-0	4	100	
	CS16	Software Engineering	3-1-0	4	100	
	CA17	Lab-I (Oracle)	0-0-4	2	100	
	CA18	Lab-II (Java)	0-0-4	2	100	
II	CA21	Advanced Data Structures	3-1-0	4	100	Periods: 36$^+$ Credits: 30 Total Marks: 900
	CA22	Data Communication & Networks	3-1-0	4	100	
	CA23	Operating Systems & Shell Programming	3-1-0	4	100	
	CA24	Artificial Intelligence & Machine Learning	3-1-0	4	100	
	CS25	Elective-II	3-1-0	4	100	
	CS26	CBCS-I	3-1-0/3-0-2	4	100	
	CA27	Lab-III (Advanced Data Structures)	0-0-4	2	100	
	CA28	Lab-IV (Linux & System Programming)	0-0-4	2	100	
	CA29	Lab-V (AI & Machine Learning)	0-0-4	2	100	
III	CA31	Analysis & Design of Algorithms	3-1-0	4	100	Periods: 30$^+$ Credits: 28 Total Marks: 800
	CA32	Information & Cyber Security	3-1-0	4	100	
	CA33	Data Mining & Warehousing	3-1-0	4	100	
	CA34	Minor Project	0-2-4	4	100	
	CS35	Elective-III	3-1-0	4	100	
	CS36	CBCS-II	3-1-0/3-0-2	4	100	
	CA37	Lab-VI (Analysis & Design of Algorithms)	0-0-4	2	100	
	CA38	Lab-VII (Data Mining & Warehousing)	0-0-4	2	100	
IV	CA41	Major Project	0-4-20	14	500	Credits: 14
MCA Programme Summary*: Theory Courses (17) + Labs (7) + Minor Project (1) + Minor Project(1) = 26 *Excluding the Bridge course					**Total-Marks *= 3000**	**Total-Credits* = 100**

MCA (Elective Courses)

SEM	CODE	COURSE TITLE	L-T-P	CR
II	CS25.1	Software Testing & Quality Assurance	3-1-0	4
	CS25.2	Software Project Management	3-1-0	4
	CS25.3	Software Architecture	3-1-0	4
	CS25.4	Advanced DBMS	3-1-0	4
	CS25.5	Computer Graphics & Multimedia	3-1-0	4
	CS25.6	Statistical Computing	3-1-0	4
III	CS35.1	Computer Forensics	3-1-0	4
	CS35.2	Cryptographic Security	3-1-0	4
	CS35.3	Distributed Systems	3-1-0	4
	CS35.4	Digital Image Processing	3-1-0	4
	CS35.5	NLP with Deep Learning	3-1-0	4
	CS35.6	Advanced Software Engineering	3-1-0	4

PG-CBCS Courses

SEM	CODE	COURSE TITLE	L-T-P	CR
	CS26.1	e-Business Foundations	3-1-0	4
	CS26.2	Human-Computer Interaction	3-1-0	4
	CS26.3	IT Management	3-1-0	4
	CS26.4	Mobile Application Development	3-0-2	4
	CS26.5	Portal Development	3-0-2	4
	CS26.6	Modelling & Simulation	3-0-2	4
	CS26.7	Data Visualization	3-0-2	4
	CS26.8	SWAYAM (NPTEL)**	As prescribed	
III	CS36.1	J2EE Programming	3-0-2	4
	CS36.2	MATLAB Computations	3-0-2	4
	CS36.3	Data Analytics with R	3-0-2	4
	CS36.4	Full Stack Development	3-0-2	4
	CS36.5	Agile Methodology & DevOps	3-0-2	4
	CS36.6	Business Informatics	3-1-0	4
	CS36.7	IT Security Auditing	3-1-0	4
	CS36.8	IPR & Ethical Hacking	3-1-0	4
	CS36.9	SWAYAM (NPTEL)**	As prescribed	

** Approved NPTEL courses as per the Department/University guidelines.

Detailed Syllabi
(Core Courses)

Semester-I

CA11: Algorithmics & Program Design

LEARNING OUTCOMES

Design of an algorithm for a given problem.
Analysis of algorithms to get the best algorithms for a given problem.
Understanding of algorithm design process for well-known problems.

1. Algorithmic Problem Solving: Algorithms, Problem-Solving Aspect in Algorithm Devising, Top-down Design of Algorithm, Algorithm Implementation, Essential and Desirable Features of an Algorithm, Efficiency of Algorithms, Analysis of Algorithms, Pseudocodes, Importance of Developing Efficient Algorithms, Complexity Analysis of Algorithms: Every-Case Time Complexity, Worst-Case Time Complexity, Average-Case Time Complexity, Best-Case Time Complexity, Flowchart, Flowchart – Symbols and Conventions.
2. Basic Algorithms: Exchanging the Values of Two Variables, Counting, Summation of a Set of Numbers, Factorial Computation, Sine Function Computation, Generation of the Fibonacci Sequence, Reversing the Digits of an Integer, Base Conversion, etc., Recursive Algorithms.
3. Factoring: Finding the Square Root of Number, Smallest Divisor of an Integer, Greatest Common Divisor of Two Integers, Generating Prime Numbers, Computing Prime Factors of an Integer, Generation of Pseudo-random Numbers, Raising a Number to a Large Power, Computing the nth Fibonacci Number.
4. Arrays, Searching and Sorting: Single and Multidimensional Arrays, Array Order Reversal, Array Counting, Finding the Maximum/Minimum number in an array, Efficient Algorithm for Finding Max-Min in an Array, Partitioning an Array, Removal of Duplicates from an Ordered Array, Monotones Subsequence, Searching: Linear and Binary Array Search, Interpolation Search, Sorting: Sorting by Selection, Exchange, and Insertion, Sorting by Diminishing Increment, Sorting by Partitioning.
5. Programming: Introduction, Game of Life, Programming Style: Names, Documentation and Format, Refinement and Modularity, Coding, Testing, and Further Refinement: Stubs and Drivers, Program Tracing, Testing, Evaluation; Program Maintenance: Program Evaluation, Review, Revision and Redevelopment, and Problem Analysis, Requirements Specification, Coding, and Programming Principles.

REFERENCES

R.G. Dromy: How to Solve by Computer. Pearson (Unit 1-4)
R. Kruse, C.L. Tondo, B. Leung, and S. Mogalla: Data Structures and Program Design in C. Pearson (Unit-5)
L.A. Robertson: Simple Program Design, A Step-by-Step Approach. Thomson

CA12: Digital Logic & Computer Architecture

LEARNING OUTCOMES

The understanding of the concept used to develop the digital circuit.
The student will be able to develop the sequential, combinational circuit as well as various types of counters.
Understand to develop various types of counter design.

1. Number System and Data Representation: Binary, Octal, Decimal, and Hexa-Decimal Number Systems; Base Conversions; Binary Arithmetic; Complements: (r-1)‘s and r‘s Complement, Subtraction using Complements; Floating and Fixed-point Representation, Binary Codes for Decimal Digits: BCD Code, Excess-3 Code, 84-2-1 Code, 2421 Code, Reflected Code; Error Detection Code; ASCII, EBCDIC codes.
2. Boolean Algebra and Logic Gates: Boolean Algebra-Basic Definitions, Huntington‘s Postulate, Switching Algebra, Basic Theorems, and Properties; Boolean Functions: Basic Definition, Literals, Minimization of Boolean Functions by Algebraic Manipulation, Complement of a Boolean Function; Canonical and Standard Forms: Min-terms and Maxterms, Boolean Function as a Sum of the Minterms, Boolean Function as a Product of Maxterms, Conversion Between Canonical Forms, Standard Form of a Boolean Function; Other Logical Operations; Digital Logic Gates: Basic Gates – AND, OR, NOT; Universal Gates – NAND, NOR; Other Gates – XOR, XNOR, AND-OR-INVERT, and OR-AND-INVERT; Implementation of Boolean Functions.
3. Simplification of Boolean Functions: Karnaugh Maps Method: Two Variables K-Map; Three, Four and Five Variables KMaps; Product of Sum Simplification, Don‘t Care Condition, Simplification of a Boolean Function with Don‘t Care;
4. Combinational Logic: Design Procedure; Design of Half and Full Adder, Half and Full Subtractor, Code Conversion; Combinational Logic with MSI and LSI: Binary Parallel Adder, Decimal Adder, BCD Adder, Magnitude Comparator, Decoders, Encoder, Multiplexers, De-Multiplexer,
5. Sequential Logic and Computer Architecture: Flip-Flops: RS Flip Flop, Clocked RS, JK Flip Flop, D Type FF, T Type FF; Analysis of Clocked Sequential Circuits: State Table, State Diagram, State Equations, Flip Flop Input Functions; FF Characteristic Tables; FF Excitation Tables; Design of Sequential Circuits; Counter: Binary Counter, BCD Counter, Design of Counters. Registers: Register with Parallel Load, Shift Registers, Bidirectional Shift Register with Parallel Load, Serial Addition using Shift Registers; Counters: Ripple Counters, Binary Ripple Counters, BCD Ripple Counters, Synchronous Counters, Binary Synchronous Counter, Binary Synchronous Up-Down Counter, Binary Counter with Parallel Load, Timing Sequences, and Signals, Johnson Counter. Introduction to Computer Architecture, Addressing mode, pipeline, One address, two address, three address instruction format.

REFERENCES
Alam, M., & Alam, B. Digital Logic Design. PHI Learning Pvt. Ltd.
Mano, M. M. Digital logic and computer design. Pearson Education India.
Harris, S. L., & Harris, D. Digital design and computer architecture. Morgan Kaufmann.

CA13: Theoretical Computer Science

LEARNING OUTCOMES

The basics of Computation and Formal Languages.
Learn to design the DFA and NFA for the languages and vice-versa.
Deriving the corresponding languages from the PDAs.
Learn to design Turing machines.
Get acquainted with Computability and Decidability, tractability, and intractability.

1. Computation Fundamentals: Review of Set, Multi-set, Sequences, Relations, Functions, Special Integer Sequences, Poset, Lattice, and Boolean Algebra; Strong and Structural Induction; Topological Sorting; Pigeonhole Principle, Recurrence Relations, Solving Linear Recurrence Relations; Models of Computation: Languages, Grammars, and Automata.
2. Finite Automata: Concept of Automata, Computational Models and Formal systems; Automata Theory; Finite Automata: Deterministic Finite Automata (DFA), Languages of DFA; Non-Deterministic Finite Automata (NFA), Language of NFA, Equivalence of Deterministic and Non-deterministic Automata, Application of Automata: Finding String in Text, Recognizing a Set of Keywords, Finite Automata with Epsilon Transition.
3. Regular and Context-Free Languages: Regular Expressions, Finite Automata and Conversions, Regular Expressions and Regular Languages, Regular Grammar, Properties of Regular Languages, and Identifying Non-regular Languages. Context Free Grammars and Languages, Derivations, Derivation Trees, Relationship between Derivation and Derivation Trees, Ambiguity in Grammars and Languages, Ambiguous Grammar, Methods for Transforming Grammars. Normal Forms: Chomsky Normal Forms and Greibach Normal Form; Pumping Lemma for CFLs.
4. Pushdown Automata and CFL: Push Down Automata (PDA), Informal and Formal Definition of a Push Down Automata, Descriptions of a PDA, the Language Accepted by a Push Down Automata, Push Down Automata and Context-Free Languages, Context Free Grammar for Push Down Automata; Deterministic Push Down Automata.
5. Turing Machines and Variants: The Standard Turing Machine, Definition of a Turing Machine, Turing Machine Language Accepters, Other Models of the Turing Machine, Multi-tape Turing Machines, Multidimensional Turing Machines, Nondeterministic Turing Machines, The Universal Turing Machine; Recursive and Recursively Enumerable languages, Some Problems that Turing Machines, Computability, and Decidability.

REFERENCES
Linz: Introduction to Formal and Automata, JBL
Mishra: Theory of Computer Science, PHI
Rosen: Discrete Mathematical Structures for Computer Applications, TMH
J. Hopcrat, J.Ullman, R. Motwani: Introduction to Automata Theory, Languages, and Computation, 2nd edition, Addition Wesley

CA14: Database Management Systems

LEARNING OUTCOMES

Fundamentals of databases with their advantages and functionalities
Various database models. Details of RDBMS, its advantages, and usage
ER Diagrams and EER Diagrams
Normalization and its various usage in table optimization

1. Basic Concepts: Data, Database and DBMS; Database vs. Traditional File System Approach; Three Schema Architecture of DBMS, Data Independence; Categories of Database Management Systems: Hierarchical, Network and Relational Database Systems.
2. Database Models: Introduction, Categories of Database Models: High-level or Conceptual Data Models, Representational or Implementation Data Models, Low-level or Physical Data Models, Object Data Models. Entity relationship (ER) Model: Basic Concepts and their representations – Entity, Entity Type and Entity Set, Attributes and Keys, Relationships, Relationship Types, and Structural Constraints, Weak Entity, Naming Conventions & Design Issues in ER Model. ER and EER Diagrams.
3. Relational Database Model: Structure of Relational Model; Domains, Attributes, Tuples, and Relations; Characteristics of Relations; Relational Constraints – Domain Constraints, Key Constraints, Entity Integrity, and Referential Integrity Constraints; Relational Database Schema; Relational Algebra Operations – Select, Project, Rename, Union, Intersection, Set Difference, Join, and Division Operations; Aggregate Functions and Groupings.
4. Structured Query Language (SQL): Schema, Table and Domain Creation; Schema and Table Deletion; Table Modification; Insert, Delete, and Update Statements; SELECT- FROM-WHERE Structure; Renaming Attributes; Nested Queries and Set Comparisons; EXISTS and UNIQUE Functions; Aggregate Functions; Creating and Updating Views. Introduction to PL/SQL.

5. Functional Dependencies and Normalization: Informal Design Guidelines for Relation Schemas; Functional Dependencies; Inference Rules for Functional Dependencies; Normalization using Functional Dependencies – First Normal Form (INF), Second Normal Form (2NF), Third Normal Form (3NF), and Boyce-Codd Normal Form (BCNF); Multi-Valued Dependencies and Fourth Normal Form (4NF); Join Dependencies and Fifth Normal Form (5NF); Relation Decomposition and Insufficiency of Normal Forms; Dependency Preserving and Lossless Join Decompositions; Null Values and Dangling Tuples. Transaction Management and Concurrency Control: Transaction Concept; Transaction State; Concurrent Executions; Serializability and Recoverability; Testing for Serializability. Concurrency Control – Lock-Based Protocols and Timestamp-Based Protocols.

REFERENCES

R. Elmasri, S. B. Navathe: Fundamentals of Database Systems. Pearson
A. Silberschatz, H. F. Korth, and S. Sudarshan: Database System Concepts. McGraw Hill
J. Casteel: ORACLE 9i Developer: PL/SQL Programming. Thomson
Ivan Bayross: SQL, PL/SQL The Programming Language of Oracle. BPB

CA15: Programming with Java

LEARNING OUTCOMES

Understand the basic concepts and fundamentals of the platform-independent object-oriented language
Demonstrate skills in writing programs using classes, interfaces, inheritance, exception handling
Use the syntax and semantics of Java programming language and basic concepts of OOP.
Apply the concepts of Multithreading and Exception handling to develop efficient and error-free programs.
Able to connect Java applications with Databases using JDBC APIs, and to Design event-driven GUI applications

1. Introduction, Environment and Programming Structure: Java White Paper Buzzwords, History of Java, Common Misconceptions, Choosing a Development Environment: Command-Line Tools, Running a Graphical Application; A Simple Java Program, Comments, Data Types, Variables, Operators, Input and Output, Control Flow, Big Numbers, Arrays.
2. Class, Objects and Inheritance: Introduction to OOP, Predefined Classes, User Defined Classes, Static Fields and Methods, Method Parameters, Object Construction, Packages, CLASSPATH, Documentation Comments, Class Design; Inheritance: Super-classes and Subclasses, Types of Inheritance, Polymorphism, Abstract class, Object: The Cosmic Super class, Generic Array Lists, Object Wrappers and Autoboxing, Methods with a Variable Number of Parameters, Enumeration Classes, Reflection, Inheritance Guidelines, Interfaces.
3. String Handling, Exception Handling and Generic Programming: String Handling APIs: String, Immutable String, Methods of String Class, StringBuffer, StringBuilder, StringTokenizer. Exceptions: Dealing with Errors, Catching Exceptions, Guidelines for Using Exceptions, Assertions, Logging; Generic Programming: Definition, Generic Methods, Bounds for Type Variables, Generic Code and VM, Restrictions and Limitations, Inheritance Rules for Generic Types, Reflection and Generics.
4. Java Collections and Multithreading: Collection Interfaces, Concrete Collections, The Collections Framework, Algorithms, Legacy Collections, Multithreading: Threads, Interrupting Threads, Thread States, Thread Properties, Synchronization, Blocking Queues, Thread-Safe Collections, Callable and Futures, Executors, Synchronizers.
5. Java GUI Programming and JDBC: Introduction to Swing, Creating a Frame, Positioning a Frame, Displaying Information in a Component, Event Handling, Basics of Event Handling, Actions, Mouse Events, The AWT Event Hierarchy; JDBC: Basic JDBC Programming Concepts, JDBC Drivers, Statements, Executing Queries, Result Sets.

REFERENCES

Cay S. Horstmann: Core Java Volume I: Fundamentals. 11th edition, Pearson
Cay S. Horstmann: Core Java Volume II: Advanced Features. 11th edition, Pearson
H. Schildt: Java 2: The Complete Reference. 12th edition, TMH
Dietel & Dietel: Java How To Program. 11th Edition, Pearson
Balagurusamy: Programming with Java. 6th Edition, TMH

CS16: Software Engineering

LEARNING OUTCOMES

Analyse and specify software requirements, and model its software design.
Understand the software life cycle models and suitable models for the problem.
Illustrate design concepts and models and use suitable methods for software design

1. Software Process: Software Engineering and Development, Software and its Components; Software characteristics; the problem of Size and Complexity; Evolving Role of software; Changing Nature; Legacy Software and Software Myths; Software Engineering – A Layered approach, Process Framework, CMMI; Technology, Product and Process.
2. Software Process Models: Prescriptive Models: Waterfall Model; Incremental Process Models, RAD; Evolutionary Models – Prototyping, Spiral, and concurrent Models; The Unified Process – Phases and Work Products; Agile Process Models – Extreme Programming and Adaptive; and Dynamic Software Development – Scrum, Crystal, Feature Driven and Agile Modeling.
3. SE Principles and Practices: Software Engineering Practices – Essence and Principles; Communication Practices; Planning Process; Modelling Principles; Construction Practices – Coding principles and Concepts; Testing Principles and Deployment;

Computer-based Systems; System Engineering Hierarchy – System Modelling and Simulation; Business Process Reengineering; Product Engineering; system modeling.

4. Requirements Engineering and Modelling: Requirements Engineering Tasks; Requirements Engineering Process; Eliciting Requirements; Developing Use-Cases; Analysis Modelling; Negotiating Requirements; and Validations. Requirements Analysis; Analysis Modelling Approaches; Object Oriented Analysis; Scenario Based and Flow Oriented Modelling.

5. Design Concepts and Models: Design concepts and principles, Software Design and Software Engineering, Design Context, Process and Quality; Design Concept – Abstraction, architecture, Pattern, modularity, information hiding, functional independence, refinement, design classes; design models – data elements, interface elements, architecture elements; User Interface Design- Process and Models, User Interface Design-The Golden Rules, Component-Level Design.

REFERENCES

Roger S. Pressman: Software Engineering – A Practitioners' Approach. McGraw Hill
K.K. Aggarwal & Yogesh Singh: Software Engineering. New Age International Publishers
P. Jalote: Software Engineering. Narosa

CA17: Lab-I (Oracle)

Implementation of at least ONE specific assignment concerning each of the following:

1. SQL statements to create, update, and delete databases and tables
2. SQL statements to insert, update, and delete records from tables
3. SQL statements to create, update, and delete views
4. Nested SQL queries to handle complex information retrieval requirements.
5. Managing changes affecting the data using COMMIT, ROLLBACK and SAVEPOINT.
6. Providing security to databases using GRANT and REVOKE commands.
7. SQL queries using order by, group by and having clauses.
8. SQL sub queries, joins, views, nested queries, inner and outer joins.
9. SQL queries using aggregate functions like count, average, sum, etc.
10. PL/SQL blocks using basic data types, branching and looping constructs 12. Database triggers, functions/procedures and packages using PL/SQL

CA18: Lab-II (Java)

Implementation of at least ONE specific assignment concerning each of the following

1. Basic Data Types, Operators, Input and Output, Control Flow, Big Numbers, Vectors and Arrays.
2. Class, Objects, Inheritance, Packages, Generic Array Lists.
3. Dynamic memory allocation using new and delete operators, function and constructor overloading, and operator overloading.
4. Object Wrappers and Autoboxing, Varying Parameters, Enumeration Classes, Reflection.
5. String handling - String Comparison, String Concatenation, Substring finding, String tokenization.
6. Exception handling – Exception Catching and Handling, Assertions, Logging, Debugging.
7. Generic functions and classes, Virtual machine, Restrictions, Wildcards.
8. Collections and Multithreading, Interrupting Threads, Thread Synchronization, Blocking Queues, etc.
9. Graphical User Interfaces (GUI), Frame creation and positioning, Displaying images and event handling.
10. Java Database Connectivity (JDBC) using databases like SQL Server, Executing Queries and Viewing Results.

Semester-II

CA21: Advanced Data Structures

LEARNING OUTCOMES

Illustrate terminology and concepts of data structures.
Derive the mapping functions to map the indices of multi-dimensional arrays to the index of 1D array.
Design efficient algorithms for different matrix operations for various special matrices.
Design algorithms for various operations of different data structures.
Applications of various data structures to solve different problems.

1. List and Matrices: Data Structure, Linear Data Structure, Array Data Structure, Multi-Dimensional Array, Mapping of Indices of 2D and 3D Arrays to the Index of 1D Array, Matrix, Mapping of Indices of Matrix Elements to One Dimensional (1D) Array Index, Special Matrices, Triangular, Diagonal, Tri-Diagonal, Representation in Row Major and Column Major Order, Mapping of non-null Elements in 1D Array, Sparse Matrix, Single Linked List, Circular Linked List, Doubly Linked List, Circular Doubly Linked List, Applications of Linked Lists: Bin Sort, Radix Sort, Convex Hull.
2. Stacks and Queues: Stack Data Structure, Various Stack Operations, Representation and Implementation of Stack using Array and Linked List, Applications of Stack: Conversion of Infix to Postfix Expressions, Parenthesis Matching, Towers of Hanoi, Rat in a Maze, Implementation of Recursive Functions, Queue Data Structure, Various Queue Operations, Circular Queue, Representation and implementation of queues using Array and Linked List, Applications of Queue Railroad Car Rearrangement, Machine Shop Simulation, Image-Component Labeling, Priority Queues: Priority Queue Using Heap; Max and Min Heap; Insertion into Heap; Deletion from a Heap; Applications of Priority Queue: Heap Sort.
3. Trees: Binary Trees and their Properties; Representation of Binary Trees: Array-Based and Linked Representations; Binary Tree Traversals; Binary Search Trees (BST); Operations on BST: Search, Insertion and Deletion; BST with Duplicates; Applications of BST: Histogramming, Best- Fit Bin Packing, AVL Trees; AVL Tree Representation; Introduction to Red- Black and Splay Trees, B-Trees and their Representation; Operations on B-Tree: Search, Insertion and Deletion; B+-Trees.
4. Sorting, Searching, and Hashing: Insertion Sort, Bubble Sorting, Quick Sort, Merge Sort, Shell sort, Sequential search, binary search, Introduction to Hashing, Hash Table Representation, Hash Functions, Collision and Overflows, Linear Probing, Random Probing, Double Hashing, and Open Hashing.
5. Graphs and Disjoint Sets: Graph Terminology & Representations, Graphs & Multi- graphs, Directed Graphs, Representations of Graphs, Weighted Graph Representations; Graph Traversal Methods: Breadth-First Search and Depth-First Search; Spanning Tree and Shortest Path Finding Problems, Disjoint Sets, Various Operations of Disjoint Sets, Disjoint Sets Implementation.

REFERENCES

Sartaj Sahni: Data Structures, Algorithms and Applications in C++. Universities Press
D. Samanta: Classic Data Structures, PHI
Narasimha Karumanchi: Data Structures and Algorithms Made Easy. CareerMonk

CA22: Data Communication & Networks

LEARNING OUTCOMES

Identify the components required to build different types of networks.
Choose the required functionality at each layer for a given application.
Identify solutions for each functionality at each layer.
Trace the flow of information from one node to another node in the network.

1. Data Communication and Networking Overview: Communication Model; Data Communications; Data Transmission and Related Concepts – Guided Media, Unguided Media, Direct Link, Point-to-Point and Multipoint Guided Configuration, Simplex, Half-Duplex, and Full-Duplex Transmission, Frequency, Spectrum, Bandwidth; Time Domain and Frequency Domain Concepts; Analog and Digital Data Transmission; Transmission Impairments – Attenuation and Attenuation Distortion, Delay Distortion.
2. Computer Networks and Reference Models: Computer Networks and their Applications; Broadcast and Point-to-Point Networks; Personal Area Networks; LAN; MAN; WAN; Wireless Networks; Internetworks; Network Layers, Protocols, and Interfaces; Connection-Oriented and Connectionless Services. OSI Reference Model; TCP/IP Reference Model; Comparison of OSI and TCP/IP Models; Problems with OSI and TCP Models; Internet and its Usage; Internet Architecture; Connection-Oriented Networks – X.25, Frame Relay, and ATM.
3. Transmission Media: Twisted Pair, Coaxial Cable, Optical Fiber; Wireless Transmission – Antennas, Terrestrial Microwave, Satellite Microwave, Broadcast Radio, and Infrared; Wireless Propagation; Line of-Sight Transmission. Communication Satellites – Geostationary Satellites, Medium-Earth Orbit Satellites, Low-Earth Orbit Satellites; Satellites versus Fiber.
4. Data Link Layer: Design Issues; Error Detection and Correction; Elementary Data Link Protocols – Unrestricted Simplex Protocol, Simplex Stop-and-Wait Protocol; Sliding Window Protocols – One-Bit Sliding Window Protocol, Protocol Using Go Back N, Multiple Access Protocol – ALOHA (Pure and Slotted), Carrier Sense Multiple Access (CSMA) Protocols (Persistent and Nonpersistent), CSMA with Collision Detection, Collision-Free Protocols, Wireless LAN Protocols, etc.; Ethernet; Wireless LANS – 802.11 Protocol Stack, 802.11 Physical Layer, 802.11 MAC Sublayer Protocol, 802.11 Frame Structure, and Services.

5. The Network and Application Layer: Design Issues; Routing Algorithms – Optimality Principle, Shortest Path Routing, Flooding, IP Protocol; IP Addresses; Subnets; Subnet Mask; Internet Control Protocols, Domain Name System (DNS); DNS Name Space; Name Servers, Simple Mail Transfer Protocol (SMTP), POP3; World Wide Web and Hyper Text Transfer Protocol.

REFERENCES

A. S. Tanenbaum: Computer Networks. PHI
William Stallings: Data and Computer Communications. Pearson
Behrouz A. Forouzan: Data Communications and Networking (SIE). TMH

CA23: Operating Systems & Shell Programming

LEARNING OUTCOMES

To understand the design of an operating system and services provided by the OS.
To understand what a process is and how processes are synchronized and scheduled.
To acquire knowledge on different approaches to memory management.
To understand the structure and organization of the file system and disk.
Be familiar with various types of operating systems including UNIX, Linux, and Windows.

1. Introduction: Operating Systems functions, Computer-System Organization, Computer-System Architecture, Operating-System Structure, Operating-System Operations, Process Management, Memory Management, Storage Management, Protection and Security, Distributed Systems, Special-Purpose Systems, Computing Environments, Open Source Operating Systems, Operating-System Services, User Operating-System Interface, System Calls, Types of System Calls, System Programs, Operating-System Design and Implementation, Virtual Machines, Operating-System Generation, System Boot.
2. Process, Threads & Scheduling: Process Concept, Process Scheduling, Operations on Processes, Inter-process Communication, IPC Systems, Communication in Client-Server Systems, Threads: Overview, Multithreading Models, Thread Libraries, Threading Issues, Process Scheduling: Basic Concepts, Scheduling Criteria, Scheduling Algorithms, Thread Scheduling, Multiple-Processor Scheduling, Operating System Examples, Algorithm Evaluation.
3. Synchronization & Deadlocks: Background, The Critical-Section Problem, Peterson’s Solution, Synchronization Hardware, Semaphores, Classic Problems of Synchronization, Monitors, Synchronization Examples, Atomic Transactions, Deadlocks: System Model, Deadlock Characterization, Methods for Handling Deadlocks, Deadlock Prevention, Deadlock Avoidance, Deadlock Detection, Recovery from Deadlock.
4. Memory Management Strategies: Background, Swapping, Contiguous Memory Allocation, Paging, Structure of the Page Table, Segmentation, Virtual Memory: Background, Demand Paging, Copy-on-Write, Page Replacement, Allocation of Frames, Thrashing, Memory-Mapped Files, Allocating Kernel Memory, Operating-System Examples.
5. File-System & Shell Programming: File Concept, Access Methods, Directory and Disk Structure, File-System Mounting, File Sharing, Protection, File-System Structure, File-System Implementation, Directory Implementation, Allocation Methods, Free-Space Management, Efficiency and Performance, Recovery, NFS, The WAFL File System, Shell Programming: Types of shells, Shell functionality, Environment, writing & executing shell scripts, Debugging script, Making interactive scripts, Variables, default variables, Functions & file manipulations, Regular Expression & Filters.

REFERENCES

Silberschatz, P.B., Galvin, and G. Gagne: Operating System Concept. JW, 8th Ed.
W. Stallings: Operating Systems - Internals and Design Principles. Pearson

CA24: Artificial Intelligence & Machine Learning

LEARNING OUTCOMES

Understand the design of AI and Deep learning-based systems.
Designing AI-based expert systems using Prolog.
Solve different AI puzzles such as Sudoku, 8-puzzle, etc. using Prolog.
Build, train, and apply fully connected deep neural networks.
To implement efficient RNNs and SOM.

1. AI Techniques: History, AI techniques, Problem-solving using Search, Uninformed v/s Informed Search, Heuristic Search Techniques: Hill Climbing, Simulated Annealing, Best First Search: OR Graphs, Heuristic Functions, A* Algorithm, AND-OR Graphs, Adversarial Search: Zero-sum perfect information Games, Optimal Decisions and Strategies in Games, Mini-max Algorithm, Alpha-beta Pruning, Solving game puzzles: NIM, Chess, Chinese Checkers, etc.
2. Knowledge Representation & Reasoning: Propositional logic, Inference in First order logic, Forward v/s Backward chaining, Resolution, Resolution-refutation; PROLOG: Logic Programming and Horn Clauses; CUT and FAIL operators, Built-in Goals, Negation, recursive rules, recursive Lists processing, CLP library, PROLOG programs for solving puzzles such as Sudoku, Cryptarithmetic, etc.
3. Feed-forward Neural network: Machine learning (m/l) types & applications, Artificial Neural Networks: Activation Functions; Loss functions; Training/learning functions, Different Models: McCulloch and Pitts Model; Perceptron Learning, Multi-layer Feed-forward model: Delta rule and error Backpropagation learning; Deep learning, CNN architecture, layers, learning, & applications.

4. Recurrent Neural Network: RNN architecture and learning; different models and their applications in sequence classification, long short-term memory (LSTM), and Gated Recurrent Unit (GRU).
5. Unsupervised m/l methods: Self-organizing maps (SOM): Competitive Learning, Feature maps, Implementation, and training; DBSCAN, Autoencoder, PCA, etc.

REFERENCES

Tom M. Mitchell: Machine Learning. McGraw Hill
E. Rich & K. Knight: Artificial Intelligence. TMH
Michael Nielsen: Neural Networks and Deep Learning (free online book)
Stuart Russel & Peter Norvig: Artificial Intelligence– A Modern Approach. Pearson
Padhy: Artificial intelligence and intelligent systems. Oxford University Press
Prolog Programming for Artificial Intelligence, August 2011, by Ivan Bratko, Addison-Wesley; 4th edition
Deep Learning (Adaptive Computation and Machine Learning series) Illustrated Edition, by Ian Goodfellow, Yoshua Bengio, Aaron Courville, MIT Press, London

CA27: Lab-III (Advanced Data Structures)

Implementation of at least ONE specific assignment concerning each of the following:

1. Creation of Matrix and Special Matrices classes OR Implementation of the different functions of the Matrix and Special Matrices.
2. Creation of Array class OR implementation of the different functions of the Array D.S.
3. Creation of the Different types of Linked list classes OR Implementation of the various functions of the Different types of Linked list D.S.
4. Implementation of the bucket and radix Sort algorithms using single linked list D.S. and convex hull using doubly circular linked list D.S.
5. Creation of the Generic Stack class and implementation of the infixTopostfix, EvaluatePostfix, tower of Hanoi problem, Rat on maz problems, and recursive functions such as factorial(n), Fib(n), gcd(m, n), DecimalToBinary(n) using Stack D.S.
6. Creation of the Generic CircularQueue class and implementation of the Railroad Car Rearrangement, Image-Component Labeling using Queue.
7. Creation of Array and linked representation of the tree data structure class; linked representation of the Expression tree data structure.
8. Creation of the Heap tree data structure and implementation of the heapsort and priority Queue D.S.
9. Creation of the binary search tree data structure.
10. Implementation of various functions of the graph; depth-first search and breadth-first search of a graph.

CA28: Lab-IV (Linux & System Programming)

1. Advanced commands used in the Unix operating system such as setting access permissions for the existing files and directories, setting default permissions for the newly created files and directories, creating groups, changing ownerships of the files, and sharing files among groups.
2. Sorting content and performing input/output (I/O) redirections.
3. Cutting or slicing the file vertically, pasting content, splitting files, counting characters, words, and lines in files or other content.
4. Displaying the top and bottom contents of a file, presenting content page-wise, and displaying the manual of any command.
5. Comparing files, eliminating and displaying duplicate lines in two files, and displaying and suppressing the unique and common content in two files.
6. Creating and running simple Bourne shell scripts.
7. Using command line parameters in the shell scripts.
8. Using conditional statements and loops.
9. Reading input, displaying output, testing data, translating content, and searching for patterns in files using different commands.
10. Displaying the exit status of the commands, and applying command substitution.

CA29: Lab V AI & Machine Learning

Implementation of at least ONE specific assignment concerning each of the following:

1. Implementing different PROLOG language features e.g.; recursive rules, cut and fail operators, knowledge base, LIST processing, etc to build an expert system.
2. Solving puzzles e.g.; Sudoku, water-jug problem using PROLOG.
3. Solving Cryptarithmetic puzzle using PROLOG.
4. Build, train, and apply a fully connected feed-forward network for MNIST data classification.
5. Build, train, and apply a fully connected feed-forward network for Regression problems.
6. Build, train, and apply a convolutional neural network for image classification in Python.
7. To implement recurrent neural networks for sequence data classification in Python.
8. To train SOM for clustering applications in Python.
9. To do anomaly detection using Autoencoders in Python.
10. To train SOM/Autoencoder for data reduction in Python.

Semester-III

CA31: Analysis & Design of Algorithms

LEARNING OUTCOMES

Asymptotic notation for representing the time complexity of an algorithm in order notations.
Learn various algorithm development approaches such as divide and Conquer, Dynamic Programming, Greedy, etc.
To Device different algorithms using various approaches for well-known problems.
Learn the concept of Intractable problems such as NP, NP-Complete, NP-Hard, etc.

1. Algorithms Analysis and Divide and Conquer Approach: Time Complexity, Complexity Representation using Order Notations: Big-o (O), Theta (Θ), Big-Omega (Ω), Small-o (o) and Small-Omega (ω) Notations; Properties of Complexity Notations; Limit Approach to Determine Order, Master Theorem. Algorithm Design Techniques (ADT): Divide and Conquer Approach – Divide, Conquer, and Combine Steps; Design and Analysis of Binary Search (Recursive and Non-recursive), Merger Sort, Quicksort, and Strassen‘s Matrix Multiplication Algorithms.
2. Dynamic Programming Approach: Introduction to Dynamic Programming; Difference Between Divide-and- Conquer and Dynamic Programming Approaches; Binomial Coefficient Finding using Dynamic Programming; Dynamic Programming and Optimization Problems: Chained Matrix Multiplication and Longest Common Subsequence Problems; Travelling Salesman Problem.
3. Greedy Approach: Introduction to Greedy Approach; Components of Greedy Approach: Selection Procedure, Feasibility Check, and Solution Check; Minimum Spanning Tree Generation: Prim‘s and Kruskal‘s Algorithms; Dijkstra‘s Algorithm for Single-Source Shortest Paths; Scheduling: Single Server and Multi-Server Scheduling, Scheduling with Deadlines; Huffman Code; The Knapsack Problem (Greedy Approach vs Dynamic Programming): 0-1 Knapsack and Fractional Knapsack Problems.
4. Backtracking Approach: Introduction to Backtracking; Backtracking Technique: State Space Tree, Promising and Non-Promising Nodes, Pruned State Space Tree; Backtracking Algorithms for n-Queens, Sum-of-Subsets, Graph Coloring, and 0-1 Knapsack Problems.
5. Branch-and-Bound Method and Intractable Problems: Introduction to Branch-and-Bound Method; Solving 0-1 Knapsack Problem using Branch-and-Bound Method: Breadth-First Search with Branch-and- Bound Pruning, Best-First Search with Branch-and-Bound Pruning; Solving Traveling Salesman Problem using Branch-and-Bound Method. Intractable Problems: NP-hard and NP-complete problems, Some examples of NP hard and NP complete with examples.

REFERENCES

R. Neapolitan & K. Naimipour: Foundations of Algorithms. Jones & Bartlett
T. H. Cormen etc.: Introduction to Algorithms. PHI
E. Horowitz, S. Sahani, and S. Rajasekaran: Fundamentals of Computer Algorithms. Galgotia

CA32: Information & Cyber Security

LEARNING OUTCOMES

Articulation and application of principled security analysis, design, and implementation.
Application and development security authentication and authorization mechanism.
Analyses of security protocols in use in fast-expanding cyberspace.
Assessment and assurance of software-specific cyber security.
Devise security solutions and customized programs for operations security.

1. Security Context & Principles: Contextualizing Information Security; IS Expertise & Business Systems. Security Management Practices: Security Architecture and Models; Physical Security; Operations Security; ACM Systems and Methodology; Cryptography; Telecommunications, Network and Internet Security; and Application Development Security; Security Goals-CIA triad; Security Principles – Defense in depth, Breadth, etc.
2. Authentication & Authorization: Authentication Methods, Passwords: Keys vs Passwords, Choosing Passwords, Attacking Systems via Passwords, Password Verification, Math of Password Cracking, Other Password Issues; Biometrics: Types of Errors, Examples, Error Rates & Biometrie; Something You Have, Two-Factor Authentication, Single Sign-On and Web Cookies; Orange Book & Common Criteria; Access Control Matrix- ACLs and Capabilities, Confused Deputy, Multilevel Security Models, Inference Control, CAPTCHA, Firewalls Defense in Dept & IDSs.
3. Typical Security Protocols: Simple Security Protocols; Authentication Protocols - Symmetric Keys, Public Keys, Session Keys, Perfect Forward Secrecy, Mutual Authentication, Session Key, and PFS, Timestamps; Authentication and TCP, Zero Knowledge Proofs, Best Authentication Protocol; Real-World Security, IPSec - IKE Phase-1&2, IPSec and IP Datagrams, Transport and Tunnel Modes, ESP and AH; Kerberos - Kerberized Login, Ticket, Security; WEP- Authentication, Encryption, Non-Integrity, Other WEP Issues; GSM -Architecture, Security Architecture, Authentication Protocol, Security.
4. Software Flaws and Malware: Software Flaws - Buffer Overflow, Incomplete Mediation, Race Conditions; Malware – Brain, Morris Worm, Code Red, SQL Slammer, Trojan Horse, Malware Detection, Cyber Diseases Versus Biological Diseases; Botnets, Miscellaneous Software-Based Attacks - Salami, Linearization Attacks, Time Bombs, Trusting Software.
5. Insecurity in Software: SRE, Anti-Disassembly Techniques, Anti-Debugging Techniques; Software Tamper Resistance, Metamorphism; Digital Rights Management, Enterprise DRM, DRM Failures; Software Development - Open vs Closed Source Software, Finding Flaws and Other Software Development Issues; OS Security Functions, Separation and Memory Protection; Trusted Operating System - MAC, DAC, Trusted Path and TCB.

REFERENCES
Stamp (2021): Principles of Information security, Wiley.
Merkow and Breithaupt: Information Security - Principles and Practices, PE.
Whitman and Mattord: Principles of Information Security, Course Technology.

CA33: Data Mining & Warehousing

LEARNING OUTCOMES
Understand the fundamental concepts of data mining and warehousing.
Learn data preprocessing techniques, multidimensional data models, and association rules.
Develop applications using data mining concepts and understand the design of data warehouse.

1. Data Mining: Introduction, Data warehouses, Transactional databases, Advanced-Data Information Systems and Applications, Data Mining Functionalities, Classification of data mining systems, data mining task primitives, Integration of data mining systems with a data warehouse system, Data Preprocessing: Descriptive data summarization, Data cleaning, Data Integration and Transformation, Data Reduction, Data discretization, and Concept hierarchy generation.
2. Data Warehouse and OLAP Technology: Multidimensional data model, Data Warehouse architecture and Implementation: OLAP, ROLAP, MOLAP, HOLAP, etc., Data Cubes, Indexing OLAP data, OLAP queries, Full Cube Computation, BUC, Star-cubing, Discovery-driven exploration of data cubes.
3. Frequent Patterns, Associations and Classification: Association Rules, Frequent Itemsets, Closed Itemsets, Apriori algorithm, Generating association rules from frequent itemsets, Mining Closed Frequent Itemsets, Correlation Analysis, Metarule guided mining of Association Rules, Constraint Pushing, Classification v/s Prediction methods, Classification by Decision Tree Induction, Bagging and Boosting.
4. Data Mining techniques: Rule-based Classification, Rule extraction from a Decision Tree, Support Vector Machines for linear and non-linear separable data, Classification by Association Rule Analysis, k- Nearest-Neighbor Classifier, Case-based Reasoning, Prediction: Linear v/s Non-linear Regression, Accuracy and Error measures: Hold-out method, Cross-validation, Bootstrap, estimating confidence intervals, ROC curves
5. Clustering: Types of data in Cluster Analysis, Categorization of Clustering methods, Partitioning Methods: k-means, kMedoids, CLARANS, Hierarchical Methods: BIRCH, ROCK, Desity-based Methods: DBSCAN.

REFERENCES
Han & Kamber: Data Mining - Concepts and Techniques, Elsevier
Witten, Frank & Hall: Data Mining: Practical Machine Learning Tools and Techniques, Elsevier.
Mohammed Zaki, Wagner Meira: Data Mining and Analysis: Fundamental Concepts and Algorithms, Cambridge Press

CA34: Minor Project

Students are required to take minor project as per the Department/University guidelines.

CA37: Lab-VI (Analysis & Design of Algorithms)

Implementation of at least ONE specific assignment concerning each of the following:
1. Implementations of various divide and conquer algorithms.
2. Chained Matrix Multiplication and Longest Common Subsequence Problems; Minimum Spanning Tree Generation.
3. Implementations of dynamic programming-based algorithms.
3. Implementations of the algorithms based on the greedy approach.
4. Implementations of the algorithms based on the backtracking approach.
5. Solving 0-1 Knapsack Problem; Breadth-First Search with Branch-and-Bound Pruning.
6. Solving the Traveling Salesman Problem using various approaches.
7. Implementations of the algorithms based on branch and bound approach.
8. Travelling Salesman Problem.
9. Scheduling: Single Server and Multi-Server Scheduling, Scheduling with Deadlines; Huffman Code.
10. Implementation of 0-1 Knapsack; Fractional Knapsack Problems; Backtracking Algorithms for n-Queens, Sum-of-Subsets, etc.

CA38: Lab-VII (Data Mining & Warehousing)

Implementation of at least ONE specific assignment concerning each of the following:
1. Discuss whether or not each of the following activities is a data mining task. If the answer is yes, then also specify which one of the following categories it will belong to-
   i) Classification, ii) Association Analysis, iii) Clustering, iv) Regression, or v) Anomaly Detection
2. Sorting a student database based on student identification numbers.
3. Problems related to classification.
4. Problems related to association analysis.
5. Problems related to clustering. analysis
6. Problems related to regression analysis.
7. Problems related to anomaly detection.
8. WEKA workbench to invoke several different machine learning schemes.
9. Use of both the graphical interface (Explorer) and command line interface (CLI) of Weka.
10. Use the following learning schemes, with the default settings to analyze the weather data. For test options, first choose ‘Use training set’ then choose ‘Percentage Split’ using the default 66% percentage split. Report model percent error rate.
    ZeroR (majority class), OneR, Naive Bayes Simple, J4.8
    Which of these classifiers are you more likely to trust when determining whether to play? Why?

Semester-IV

CA41: Major Project

Students are required to take major project as per the Department/University guidelines.

Detailed Syllabi
(Elective Courses)

ELECTIVE COURSES

CS25.1: Software Testing & Quality Assurance

LEARNING OUTCOMES

Explain the context and concepts, and apply techniques related to software testing.
Develop a viable and optimistic strategy for testing.
Perform object-oriented testing in practice.
Test web applications for their functionalities and environment.
Illustrate case tools and use them during testing and other phases.

1. Testing Fundamentals: Testing Terminology, Types of Testing; Basis Path Testing: Flow Graph Notation, Independent Program Paths, Deriving Test Cases and Graph Metrics; Control Structure Testing: Condition Testing, Data Flow Testing, and Loop Testing; Black Box Testing: Graph-Based Testing methods, Equivalence Partitioning, Boundary Value Analysis; Objected Oriented Testing Methods: Testing Specialized Environments, Architecture, and Applications: Testing GUI, Client Server Architecture, Documentation and Help Facilities, and Real-Time Systems.
2. Testing Strategies: Strategic Approach to Software Testing: Verification and Validation, Organizing for Software Testing, Strategies for Conventional Architecture, Object Oriented Structure and Criterion for Completion of Testing; Strategic Issue; Significance and Potential; Testability; Unit and Integration Testing; OO Strategies: Unit Testing in OO Context, Integration Testing for OO Context; Validation Testing: Criteria, Configuration, Review, Alpha and Beta Testing; System Testing: Recovery Testing, Security Testing, Stress Testing, and Performance Testing; Art of Debugging: Debugging Process, Debugging Strategies and Fixing Errors.
3. Object-Oriented Testing: Broadening the View of Testing, Testing OOA and OOD Models, Object-Oriented Testing Strategies, Test Case Design for OO Software, Testing Methods Applicable at the Class Level, Interclass Test Case Design: Multiple Class and Tests Derived from Behavior Model.
Testing Web Applications: Testing Concepts for Web Applications; Dimensions of Quality, Errors in Web App Environment, Testing Strategy and Test Planning; Testing Process Overview; Content Testing: Objectives and Database Testing; User Interface
4. Testing: Testing Strategy, Interface Mechanism, Interface Semantics, Usability Testing, and Compatibility Tests; Component Level Testing: Navigation Testing: Syntax and Semantics; Configuration Testing: Server and Client Side Issues; Security Testing; Performance Testing: Objectives, Load and Stress Testing.
5. CASE: Concept of CASE, Building Blocks for CASE, A Taxonomy of CASE Tools, Integrated CASE Environments, The Integration Architecture, The CASE Repository, The Role of the Repository in I-CASE, Features and Content.

REFERENCES

Pressman: Software Engineering, TMH
Aggarwal and Singh: Software Engineering, NAI
Jalote: Software Engineering, Narosa

CS25.2: Software Project Management

LEARNING OUTCOMES

Generic software project management, Management Principles, and step-wise project planning.
Software process models, and software estimation techniques.
Activity planning, project schedules, and PERT Techniques.
Cost monitoring, Earned Value Analysis, and Software Configuration Management.
Organizational behavior, monitoring and Project close-out.

1. Software Project Management and Evaluation: Project, Software Projects vs Other Projects, Technical Project Management, Activities, Plans, Methods and Methodologies, Categorizing Software Projects, Project Charter, Stakeholders, Setting Objectives, The Business Case, Project Success and Failure, Management, Management Control, Project Management Life Cycle, Traditional vs Modern Project Management Practices; Introduction, Business Case, Cost-benefit and Risk Evaluation, Managing Allocation, Strategic Programme Management, Aids, Reservations and Benefits Management.
2. Project Planning and Approaches: Step Wise Project Planning- Selection to Execution; Planning; Selection Approach: Introduction, Build or Buy; Choosing Methodologies and Technologies, Software Processes and Process Models, Choice of Process Models, Structure vs Speed of Delivery, The Waterfall, Spiral, Software Prototyping, Other models; Categorizing Prototypes, Incremental Delivery, Rapid Application Development, Agile Methods, Extreme Programming (XP), Scrum, Lean Software Development, Managing Iterative Processes, Selecting the Most Appropriate Process Model.
3. Effort Estimation and Activity Planning: Estimates, Problems with Over-and-Under-Estimates, Software Estimation Techniques: Bottom-up, Top-down Approach and Parametric, Expert Judgement, Analogy, Albrecht Function Point Analysis, Function Points Mark II, COSMIC Full Function Points; Staffing Pattern, Schedule Compression, Capers Jones Estimating Rules of Thumb; Activity Planning: Objectives, When-to, Schedules, Projects, and Activities; Sequencing and Scheduling Activities, Network Planning Models, Formulating a Network Model, Time Dimension, Forward and Backward Pass, Critical Path, Activity Float, and Shortening Duration.

4. Risk Management and Resource Allocation: Risk, Categories, Management Approaches; A Framework for Dealing with Risk: Identification, Assessment, Planning, Management, Evaluation; Boehm's Top 10 Risks and Counter Measures, PERT Technique, Monte Carlo Simulation, Critical Chain Concepts; Resource Allocation: Introduction, Nature of Resources, Identifying Resource Requirements, Scheduling Resources, Creating Critical Paths, Counting the Cost, Being Specific, Publishing the Resource Schedule, Cost Schedules, Scheduling Sequence.

5. Monitoring, Control and Closure: Creating the Framework, Collecting the Data, Review, Visualizing Progress, Cost Monitoring, Earned Value Analysis, Prioritizing Monitoring, Getting the Project Back to Target, Change Control, Software Configuration Management (SCM); Project Closeout.

REFERENCES

Hughes et al.(2018). Software Project Management, 6ed, Tata McGraw Hill
Mehta, S. (2017). Project Management and Tools & Technologies – An overview, 1st Ed, SPD
Royce, W. (2005). Software Project Management. Pearson

CS25.3: Software Architecture

LEARNING OUTCOMES

Explain the function of software architecture and modern development methodologies.
Use domain-driven design to model and write appropriate software quality attributes.
Use agile approaches to complete software development projects.
Design and develop code for implementing software architecture patterns.
Articulate and implement the core principles and practices into software architectures.

1. **Software Concepts:** Software-Distinguishing Features, Types, and Recent Developments; SDLC – Waterfall, Prototype, and Agile; Software Architecture, Architectural Structures and Views, Good Architecture; Importance-Inhibiting or Enabling a System's Quality Attributes, Managing Change, Predicting System Qualities, Early Design Decisions, Constraints on Implementation, Influences on Organizational Structure, Incremental Development, and Transferable/Reusable Model.

2. Software Quality Attributes: Functionality, Quality Attribute Considerations, Specifying Quality Attribute Requirements: Quality Attribute Scenarios, Achieving Quality Attributes through Architectural Patterns and Tactics, Designing with Tactics, and Analysing Quality Attribute Design Decisions: Tactics-Based Questionnaires; Quality Attribute Tactics–Deployability, Modifiability, Energy efficiency, Testability, Integrability, Safety, Performance, and Usability.

3. Architectural Solutions: Software Interfaces-Interface Concepts, Designing an Interface, and Documenting the Interface; Virtualization-Shared Resources, Virtual Machines, VM Images, Containers, Containers and VMs, Container Portability, Pods and Serverless Architecture; Cloud and Distributed Computing-Cloud Basics, Mobile Systems–Energy, Network Connectivity, Sensors and Actuators, Resources, and Life Cycle. Spring and Spring Boot, GitHub, Maven, basic UI, database use and microservices structures, UML diagram

4. **Architectural Requirements and Design:** Gathering Architecturally Significant Requirements (ASRs) from Requirements Documents, Gathering ASRs by Interviewing Stakeholders, Gathering ASRs by Understanding the Business Goals, Capturing ASRs in a Utility Tree, Attribute-Driven Design (ADD), The Steps of ADD, More on ADD Step, More Design Concepts, Producing Structures, Creating Preliminary Documentation during the Design, Performing Analysis of the Current Design, and Reviewing the Iteration Goal and Achievement of the Design Purpose

5. Evaluating an Architecture: Evaluation as a Risk Reduction Activity, Key Evaluation Activities, Performing the Evaluation, Contextual Factors, The Architecture Trade-off Analysis Method, and Lightweight Architecture Evaluation.

REFERENCES

Bass et al.: Software Architecture in Practice, Addison-Wesley Professional
Ingeno, J. : Software Architect's Handbook , Packt Publishing
Taylor, R.,et al.: Software Architecture: Foundations, Theory, and Practice, Wiley
Clements, P.: Documenting Software Architectures: Views and Beyond, 2002, Addison-Wesley

CS35.1: Computer Forensics

LEARNING OUTCOMES

Knowledge of digital investigations, collection, storage, and cataloging of evidence.
Securing/preservation of digital assets using the best acceptable practices, appropriate laws, and relevant regulations.
Identification and response to security attacks, preservation of the crime scene, various tools, and file systems.
Skills concerning major forensics tools and performing digital investigations related to forensics cases.
Performing Windows, network, virtual, email, social media, and cloud forensics.

1. Forensic Fundamentals: Overview, Roles of Forensics Investigator, Forensics Readiness, Steps for Forensics, Digital Evidence, Issues Facing Computer Forensics; Crime Basics, Crime Categories, and Cybercrimes; Information Warfare, Electronic Attack, and

Terrorism; e-Evidence Technology, Issues, Discovery and e-Discovery; Digital Evidence and Investigations; Computer Forensics Investigation – Preparation, Procedures, Process, and Professional Conduct; Investigator's Lab Requirements and Tool Box. **2. Data Acquisition and Processing:** Overview, goals, objectives and Issues; Storage Formats – Raw, Proprietary, and advanced; Determining Acquisition Methods; Contingency Planning and Image Acquisition; Common acquisition Tools; Validating data Acquisition; RAID Data Acquisition; Remote Network Acquisition Tools; digital evidence – Identification, Search, Collection, Seizure, Storage and review. **3. Windows Forensics and Tools:** Windows Overview, File System, File Structures; Volatile and Non-Volatile Information; NTFS Disks and examination; Disk encryption – BitLocker and 3rd Party Tools; Windows Registry – Uses and Examination; Digital Forensics Tools – Types, Utility, Testing, Validation and Comparison; Digital Forensics Road Map; Digital Forensics Software (FTK/EnCase/AUTOPSY) and Hardware Tools; Anti Forensics and Probable Counters. **4. Networks, Virtual and Live Forensics:** Overview, Network Components and their Forensic importance, Securing, Investigating and Examining Networks; Virtual Machine – Concept, Practices, Issues and Networks; Virtual Forensics; Packet Sniffing, Capture and Analyses; Website Penetration: WHOIS, nslookup, etc; Live Acquisitions. E-mail, Social Media, and Cloud Forensics: Overview, Roles and Issues; Email Investigation – Messages, Headers; Tracing and Examination – Emails and Network Logs; Email Forensics Tools (AXIOM/Hex/Outlook); Site Report Generation (Netcraft); Social Media Forensics; Mobile Phones; Cloud Forensics – Services, Deployment, Issues, Vendors, Investigation, and Tools.
REFERENCES Nelson et al. Guide to Computer Forensics and Investigations, 6ed, Cengage Learning Volonino et al. Computer Forensics: Principles and Practices, Prentice Hall Milind. Digital Forensics: An Investigative Approach,| Practical Edition Paperback

Detailed Syllabi
(CBCS Courses)

PG-CBCS COURSES

CS26.1: e-Business Foundations

LEARNING OUTCOMES

Explaining Systems; and IT-related terminology, trends, challenges, obstacles, and prospects.
Illustration and performing work-centered analysis on business systems.
Analyzing and developing business process models.
Identifying typical information systems, and roles in current e-business perspective.
Evaluation of the performance of IT and e-Business and vice-versa.

1. Toward E-business Systems: Systems, Business Systems, Building and Maintaining Systems, IT-Based Innovations in Every Business Function, Product Design Systems, CAD Software, Procurement Systems, Supply Chain Management, Electronic Data Interchange, Manufacturing, Sales and Marketing Systems, Delivery Systems, Customer Service Systems, Finance Systems, Dramatic Progress in Processing Data, Recent Trends in IT; Applying IT to the Real World.
2. Business Systems: Frameworks and Models, Viewing Businesses as Systems, Businesses as Systems Consisting of Business Processes, The Value Chain, Business Processes and Functional Areas of Business, Information Systems and Work Systems, Increasing Overlap Between Information Systems and Work Systems, Framework for Thinking About Any System in Business, WCA Framework, Five Perspectives for Viewing a Work System, Architecture: System Components and How They Operate Together, Performance: How Well the System Operates, Analyzing an IT-Enabled System From a Business Professional's Viewpoint, Work-Centered Analysis Method, Limitations and Pitfalls
3. Business Processes and Models: Business Processes, Process Characteristics: Degree of Structure, Range of Involvement, Level of Integration, Rhythm, Complexity, Degree of Reliance on Machines, etc, Communication and Decision Making; Evaluating Business Process Performance: Activity Rate and Output, Consistency, Productivity, Cycle Time, Downtime and Security, Basic Communication and Decision-Making Concepts.
4. Typical Information Systems: Information System Categories related to Specific Functional Areas of Business, IS Categories applicable Functional Areas; Office Automation Systems; Communication Systems: Teleconferencing, E-Mail, Fax, SMS, Groupware, Internet, Intranets, Extranets, Knowledge Management, and Group Support Systems, Transaction Processing Systems, MIS and Executive Information Systems, Decision Support Systems, Case-Based Reasoning, and Intelligent Systems, Enterprise Systems, Limitation and Uses of Typical Information Systems.
5. Customer, Product and IT: Customer's View of Products and services, The Customers' Experience, Evaluating Products and Services: Cost, quality, responsiveness, Reliability and Conformance to standards, Performance Variables of IT: Functional Capabilities and Limitations, Ease of use, Compatibility and Maintainability, Approaches of Organizational Computing-Centralized, Personal, Distributed, Networked and Client-Server, Current Limits of Software; Types of Software, Programming viewed as Business Process, Major Developments in Programming – Special purpose, Spreadsheets and CASE, Artificial Intelligence and Intelligent Systems.

REFERENCES

Alter: Information Systems – The Foundations of E-Business, Pearson
Combe: Introduction to E-business Management and Strategy, Elsevier
Colin Combe Haag and Cummings: Information Systems Essentials, TMHl

CS26.2: Human-Computer Interaction

LEARNING OUTCOMES

Acquire first-hand experience with useful HCI techniques in practice.
Comprehend the latest topics in multimedia, global information systems, and web-based models for rich interaction.
Articulate social and contextual models and theories related to HCI design processes,
Appreciate and illustrate the role of interaction design, universal access, and rich interaction.
Apply design principles and established rules during user-friendly interface designs.

1. HCI Overview: Need, Issues in Human-Computer Interaction and Significance; Overview of Human Sensory Capabilities and Limitations: Input-Output Channels and Design Focus; Analysis of Design Experiments; Human Sensory Capabilities and Limitations: Memory, Types of Memory, Comparative association with Computer memory, Thinking and Emotions; Human Sensory Capabilities and Limitations: Individual Differences and Psychology of Design of Interactive Systems.
2. Computers: Text Entry, Design Focus- Numeric Pads, Positioning, Pointing and Display Devices; Display Devices, Devices for Virtual Reality and 3D interactions, Physical Controls, Sensors, and Special Devices; Paper: Printing and Scanning, Readability of Text, Font Issues, etc.
3. Interaction: Models of Interaction, Framework for HCI and Ergonomics; Industrial Interfaces; Interaction Styles: Introduction and Types with Examples, Navigation, Elements of WIMP Interface, Interactivity and its context, Experience and edutainment, etc; Interaction Paradigms: Introduction, Different Paradigms of Interaction, Timesharing, Batch Processing, Personal computing, Distributed etc; Language vs Action; Hypertext, Multimodality, WWW, Sensor-based Context-aware Interactions etc.

4. Interaction Design Basics: Introduction, Design Basics and Process of Design, Scenarios, Navigation Design: Local structure, Global Structure, Dialog, etc; Interaction Design Basics: Introduction, User Focus, Cultural Probes and Scenarios; Screen Design, Alignment and Layout, Screen Colors, etc.
5. Interaction Design Rules: Introduction, Principles to Support Usability: Learn ability, Flexibility, and Robustness; Design Rules: Standards, need, significance; Underlying Theory, Usability factors, etc.; Design Guidelines: Fundamental Guidelines, Golden Rules and Heuristics; Sheiderman Eight Rules; Norman's seven Principles for Transformation; HCI Patterns

REFERENCES
Dix et al.: Human-Computer Interaction, Pearson
Carlo et al.: Human-Computer Interaction, PHI
Mosley & Posey: Just Enough Software Test Automation, Pearson

CS36.1: J2EE Programming

LEARNING OUTCOMES
Understand core concepts of J2EE programming.
Ability to develop enterprise applications.
Learn the concepts of Servlets, and Java Server Pages.
Develop database-driven web applications using Enterprise Java Beans, and Web Services.

1. J2EE Overview & Multi-tier Architecture: Overview of J2SE, J2EE, Advantages of Java, Birth of J2EE, Why J2EE; Distributed Systems, The Tier, J2EE Multi-tier architecture, Implementation of Client-tier, Web-tier, EJB-tier, and EIS-tier, Challenges; J2EE best practices: Enterprise Application Strategy, The Enterprise Application - Client, Session Management, Web-tier and JSPs, EJB-tier, MVC, Maintainable Classes, Performance Enhancement, Power of Interfaces, Threads, and Notification.
2. Java Servlets & JDBC: Overview of HTML, XML, and XHTML, Java and XML, Parsing XML, Java Servlets and CGI Programming, A Simple Java Servlet, Anatomy of a Java Servlet, Life Cycle of the Servlet, Deployment Descriptor, Reading data from client, reading HTTP request headers, working with cookies, tracking sessions. Overview of JDBC, JDBC Drivers, JDBC Packages, JDBC Process, Database Connection, Statement, ResultSet, Transaction Processing, and Servlet program with JDBC.
3. Java Server Pages: Overview of JSP, JSP versus Servlet, JSP Tags: Variables and Objects, Directives, Scripting Elements, Standard Actions, Implicit Objects, Scope, Java Server Pages with Beans, Tomcat, User Sessions, Cookies, Session Objects, JSP with JDBC, Creating Custom JSP Tag Libraries.
4. Enterprise Java Beans: The EJB Container, EJB Classes, EJB Interfaces, Deployment Descriptions: Anatomy, Environment elements, referencing EJB, Sharing resources, Security elements, Query elements, Relationship elements, Assembly elements. Session Java Beans - stateless vs stateful, Entity Java Beans - Container-managed persistence, Bean-managed persistence. Message-driven Beans, JAR, WAR, EAR Files.
5. JavaMail, CORBA and RMI: JavaMail API and Java Activation Framework, Protocols, Exceptions, Send Email Message, Retrieving Email Messages, Deleting Email Message. CORBA: The Concept of Object Request Brokerage, Java IDL and CORBA, The IDL Interface. Java RMI: Remote Method Invocation Concept, Server Side, Client Side.

REFERENCES
Jim Keogh: J2EE: The Complete Reference. Mc Graw Hill
H. Schildt: Java 2: The Complete Reference. Mc Graw Hill
Kogent Solutions Inc.: Java Server Programming Java EE 7 (J2EE 1.7), Black Book, Dreamtech Press
Subrahmanyam Allaramaju et al.: Professional JSP J2EE 1.3 Edition. Wrox Press

CS36.3: MATLAB Computation

LEARNING OUTCOMES
Understanding the fundamentals of Matlab, including data structures, matrices, vectors, functions, and scripts.
Ability to use the built-in functions, control structures, and write and debug Matlab code.
Ability to visualize and analyze data using tools like graphs, histograms, and scatter plots.
Knowledge of advanced Matlab concepts such as OOP, optimization, and simulation.
Ability to solve practical engineering and scientific problems.

1. Vectors and Matrices: MATLAB Desktop Environment, Data types, Variables, and Assignment Statements, Numerical Expressions, Operator precedence, Random number generation, Characters, and Encoding, Relational Expressions, Creating matrix variables, dimensions, Scalar and Array Operations on Vectors and Matrices, Matrix Multiplication, Logical vectors, Meshgrid functions, Saving workspace, Importing and Exporting data
2. MATLAB Scripts: Scripts with Input and Output, Scripts to Produce and Customize simple plots, Introduction to File Input/Output (Load and Save), User-Defined Functions that Return a Single Value, Commands and Functions, Vectors and Matrices as function arguments, Selection statements: if, else, else if, switch, Loop statements: while, for, Vectorizing Code, MATLAB Program Organization, Application: Menu-Driven Modular Program, "is" Functions in MATLAB Variable, Scope, M-files etc.

3. String Manipulation and Cell Arrays: Creating String Variables, Operations on Strings, “is” Functions for Strings, Converting between String and Number Types, creating Cell Arrays, Referring to and Displaying Cell Array Elements and Attributes, Storing Strings in Cell Arrays, Structures, and operations,
4. Advanced File Input and Output: Opening and Closing a File, file identifier and file modes, Reading from file: fscanf, fgets, fgetl, textscan, Lower-Level File I/O Functions, Writing, and Reading Spreadsheet Files, Using MAT-files for Variables
5. Advanced problem solving with MATLAB: 2-D plot types, logarithmic scale plots, pie charts, and histograms, customizing plots using cell arrays and string functions, 3-D plot functions, Built-in statistical and set operations, Sorting and Indexing, Sights and **Sounds**, programming GUIs, etc.

REFERENCES

Stormy Attaway: Matlab, A Practical Introduction to Programming and Problem-Solving, Elsevier Inc.
Amos Gilat, MATLAB: An Introduction with Applications, Wiley
Cleve Moler, Numerical Computing with MATLAB, SIAM

(NAAC Accredited A++ Grade)

Department of Computer Science

Jamia Millia Islamia (Central University)

Jamia Nagar, New Delhi-110025 (INDIA)

https://www.jmi.ac.in/computerscience

Email: computerscience@jmi.ac.in