Muffakham Jah College of Engineering and Technology

(Sultan-ul-Uloom Education Society)

An Autonomous Institution

Approved by AICTE and Affiliated to Osmania University

Accredited by NAAC with A+ Grade and NBA Accreditation for CE, CSE, ECE and ME

Computer Science Engineering (Artificial Intelligence)

(R-25 Regulation)

SCHEME OF INSTRUCTION & EXAMINATION

B.E. CSE (Artificial Intelligence)

SEMESTER-III

S. No.	Course Code	Course Title	Scheme of Instruction				Scheme of Examination			Credits
			L	T	D/P	Contact Hrs./ Week	CIE	SEE	Duration in Hrs.	
Theory Courses										
1	25BS211MT	Probability & Statistics	3	-	-	3	40	60	3	3
2	25PC212CS	Data Structures	3	1	-	4	40	60	3	4
3	25PC213CS	Discrete Mathematics	3	-	-	3	40	60	3	3
4	25PC214CS	OOP Using Java	3	-	-	3	40	60	3	3
5	25PC215DS	Foundation of Data Science	3	-	-	3	40	60	3	3
Practical / Laboratory Courses										
6	25PC251DS	Data Science Lab	-	-	2	2	40	60	3	1
7	25PC252CS	Data Structures Lab	-	-	2	2	40	60	3	1
8	25PC253CS	OOP using Java Lab	-	-	2	2	40	60	3	1
9	25ES254EC	Skill Development Course-I (IoT)	-	-	2	2	50	-	0	1
Total			**15**	**1**	**8**	**24**	**370**	**480**	**24**	**20**

PROBABILITY AND STATISTICS		
L:T:P (Hrs./week):3:0:0	**SEE Marks : 60**	**Course Code :25BS211MT**
Credits : 3	**CIE Marks : 40**	**Duration of SEE : 3 Hours**

COURSE OBJECTIVES	**COURSE OUTCOMES** On completion of the course, the students will be able to
1. Understand fundamental concepts of probability theory and random variables. 2. Apply discrete and continuous probability distributions in engineering problems. 3. Analyze statistical data using sampling distributions and estimation techniques. 4. Perform hypothesis testing for means, proportions and variances. 5. Apply correlation, regression and basic statistical tools in real-life engineering applications.	1. Solve problems involving probability axioms, conditional probability and Bayes' theorem 2. Evaluate statistical parameters of discrete probability distribution. 3. Evaluate statistical parameters of continuous probability distribution. 4. Perform regression analysis to compute the coefficient of correlation to interpret data. 5. Testing of hypothesis of few unknown statistical parameters using types of sampling, Sampling distribution of means, Sampling distribution of variance, Estimations of statistical parameters.

CO/PO &PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	3	3	-	2	-	-	-	-	-	-	-	1	–
CO2	3	3	-	2	1	-	-	-	-	-	-	1	–
CO3	3	3	-	2	1	-	-	-	-	-	-	2	1
CO4	3	3	2	3	2	-	-	-	-	1	-	2	1
CO5	3	3	2	3	2	-	-	-	-	1	-	2	2

Unit-I

Probability: Introduction, Conditional Probability, Baye's Theorem (Without proof), Random variable, Discrete Probability Distribution, Continuous Probability Distribution, Probability Mass Function, Probability Density Function, Expectation, Variance.

Unit-II

Discrete Probability Distributions: Binomial Distribution and Poisson Distributions- Evaluation of Statistical Parameters for these Distributions- Mean, Variance and Moment Generating Function.

Unit-III

Continuous Probability Distributions: Uniform Distribution, Exponential Distribution and Normal Distribution- Evaluation of Statistical Parameters for these Distributions-Mean, Variance and Moment Generating Function.

Unit-IV

Curve Fitting by the Method of Least Squares-Fitting of Straight Line, Fitting of Parabola, Fitting of other Curves ($y = ae^{bx}, y = ax^b, y = ab^x$), Coefficient of Correlation, Regression Coefficient, Lines of Regression, Rank Correlation Coefficient.

Unit-V

Test of Significance -Large samples- Test for single mean, difference of means, single proportion, difference of proportions -Test of significance for small samples- chi-square test for goodness of fit and independence of attributes, student's t-test, F -test for ratio of variances,

Suggested Reading:

1. R. K Jain S.R.K Iyengar, Advanced Engineering Mathematics, Narosa Publication. 4^{Th} Edition, 2014.
2. B. S. Grewal, Higher Engineering Mathematics, Khanna Publication 43^{rd} Edition, 2014.
3. S.C Gupta and V.K Kapoor, Fundamental of Mathematical Statistics, Sultan Chand & sons, New Delhi, 2014.

DATA STRUCTURES		
L:T:P (Hrs./week): 3:1:0	**SEE Marks : 60**	**Course Code :25PC212CS**
Credits : 4	**CIE Marks : 40**	**Duration of SEE : 3 Hours**

COURSE OBJECTIVES	**COURSE OUTCOMES** On completion of the course, the students will be able to
1. Develop the ability to analyze algorithm efficiency and implement Abstract Data Types using appropriate Data Structures. 2. Understand and implement linear and non-linear data structures for solving computational problems 3. Apply tree-based and hashing techniques for efficient storage, retrieval and manipulation of data. 4. Analyze and implement searching, sorting and graph based algorithms using appropriate data structures.	1. Analyze algorithm efficiency using asymptotic notations, implement array-based data structures, and string processing techniques. 2. Implement linked list and hashing techniques for efficient storage, retrieval, and manipulation of data. 3. Apply stack and queue data structures to solve computational problems involving expression evaluation and scheduling applications. 4. Construct and analyze tree-based data structures including BSTs, AVL trees, and heaps for efficient searching and sorting. 5. Apply graph algorithms, sorting, and searching techniques to solve real-world computational problems.

CO/PO &PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	3	2	2	2	2	-	-	-	-	1	2	2	2
CO2	3	3	3	2	2	-	-	-	1	1	2	2	2
CO3	3	3	3	2	2	-	-	-	1	1	2	2	2
CO4	3	3	3	2	2	-	-	-	1	1	2	3	2
CO5	3	3	3	3	2	-	-	-	1	1	2	3	3

Unit-I

Algorithms: Introduction, Algorithm Specifications, Recursive Algorithms, Performance Analysis of an algorithm- Time and Space Complexity, Asymptotic Notations**.**

Arrays: Arrays ADT, Polynomials, Sparse matrices, Strings-ADT, Pattern Matching.

Unit-II

Linked Lists: Singly Linked Lists and Chains, Polynomials, Operations for Circularly linked lists, Equivalence Classes, Sparse matrices, Doubly Linked Lists.

Unit-III

Stacks and Queues: Stacks, Stacks using Arrays, and Stacks using dynamic arrays, Evaluation of Expressions, Evaluating Postfix Expression, Infix to Postfix. Linked Stacks.

Queues: Queues ADT, operations, Circular Queues, Linked Queues.

Hashing: Static Hashing, Hash Tables, Hash Functions, Overflow Handling

Unit-IV

Trees: Introduction, Binary Trees, Binary Tree Traversals, Heaps**.**

Binary Search trees (BST): Definition, Searching an element, Insertion into a BST, Deletion from a BST**.**

Efficient Binary Search Trees: AVL Trees: Definition, Searching an element, Insertion into a AVL.

Unit-V

Graphs: Graph Abstract Data Type, Elementary Graph operations (DFS and BFS), Minimum Cost Spanning Trees (Prim's and Kruskal's Algorithms).

Sorting and Searching: Insertion sort, Quick sort, Merge sort, Heap sort, Shell sort, Sorting on Several Keys, List and Table Sorts, Summary of Internal Sorting, Linear and Binary Search algorithms.

Suggested Reading:

1. Horowitz E, Sahni S and Susan Anderson-Freed, Fundamentals of Data structures in C, 2nd Edition (Reprint 2024), Universities Press.
2. Tanenbaum A. M, Langsam Y. Augenstein M. J, Data Structures using C, Second Edition (2008), Pearson.
3. Gilberg R. F and Forouzan B. A, Data structures: A Pseudocode Approach with C, Second Edition (2007), Cengage Leaming.
4. Thomas H. Cormen, Charles E. Leiserson, Ronald L Rivest, Clifford Stein, Introduction to Algorithms, Fourth Edition (2022), MIT Press.
5. Kushwaha D. S and Misra A.K, Data structures A Programming Approach with C, Second Edition (2014), PHI.

DISCRETE MATHEMATICS		
L:T:P (Hrs./week):3:0:0	**SEE Marks : 60**	**Course Code :25PC213CS**
Credits : 3	**CIE Marks : 40**	**Duration of SEE : 3 Hours**

COURSE OBJECTIVES	**COURSE OUTCOMES** On completion of the course, the students will be able to
1. **Understand** the fundamental concepts of mathematical logic, predicate logic, and proof techniques used in discrete mathematics. 2. **Apply** the principles of set theory, relations, functions, lattices, and partial ordering to solve mathematical and computational problems. 3. **Analyze** graph-theoretic concepts, trees, and their applications in computer science, including graph traversal, graph coloring, and spanning trees. 4. **Develop** problem-solving skills using combinatorial techniques, counting principles, permutations, combinations, binomial theorem, inclusion-exclusion principle, and pigeonhole principle. 5. **Solve** recurrence relations and utilize generating functions to analyze algorithms and discrete structures.	1. Apply propositional and predicate logic, normal forms, quantifiers, and proof techniques to formulate and validate logical arguments. 2. Analyze and solve problems involving sets, relations, functions, lattices, equivalence relations, partial orders, and Hasse diagrams. 3. Model and solve real-world problems using graph theory and trees, including graph traversal, connectivity, graph coloring, Euler and Hamilton paths, and minimum spanning trees. 4. Employ combinatorial methods, including counting principles, permutations, combinations, binomial theorem, inclusion-exclusion principle, and pigeonhole principle, to solve discrete mathematical problems. 5. Formulate and solve linear recurrence relations and apply generating functions and divide-and-conquer techniques in algorithm analysis.

CO/PO &PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	3	3	1	1	-	-	-	-	-	-	1	1	–
CO2	3	3	1	1	-	-	-	-	-	-	1	1	–
CO3	3	3	2	2	1	-	-	1	-	-	1	2	1
CO4	3	3	1	2	-	-	-	-	-	-	1	2	1
CO5	3	3	2	2	1	-	1	-	-	-	2	2	2

Unit-I

Mathematical Logic: Statements and Notations, Connectives, Well-Formed Formulas, Truth Tables, Tautology, Equivalence, Implication, Normal Forms, Quantifiers, Universal Quantifiers.

Predicates: Predicate Logic, Free & Bound Variables, Rules of Inference, Consistency, Proof of

Contradiction.

Unit-II

Set Theory and Relations: Basic Concepts of Set Theory, Relations and Ordering, Properties of Binary Relations, Equivalence, Transitive Closure, Compatibility and Partial Ordering Relations, Hasse Diagram.
Functions: Composition of Functions, Inverse Functions, Recursive Functions, Lattice and Its Properties.

Unit-III

Graphs: Graphs and Graph Models, Graph Terminology and Special Types of Graphs, Subgraphs, Representing Graphs and Graph Isomorphism, Connectivity, Euler and Hamilton Paths, Euler Circuits, Planar Graphs, Graph Coloring, Chromatic Numbers.
Trees: Introduction to Trees, Properties, Applications of Trees, Tree Traversal, Minimal Spanning Trees.

Unit-IV

Elementary Combinatorics: Basics of counting, Combinations & Permutations with and without Repetitions and Constrained Repetitions.
Binomial Theorem: Binomial Coefficients, Binomial and Multinomial theorems, The Principle of Inclusion — Exclusion and Its Applications, Pigeon Hole Principle and Its Applications.

Unit-V

Recurrence Relations: Recurrence Relations, Solving Linear Homogenous Recurrence Relations, Solving Linear Non-Homogenous Recurrence Relations, Divide and Conquer Algorithms and Recurrence Relations, Generating Functions.

Suggested Reading:

1. Discrete Mathematics and its Applications, Kenneth H. Rosen, VIII Edition, McGraw Hill Education, 2019.
2. Elements of Discrete Mathematics- A Computer Oriented Approach- C. L. Liu, D. P. Mohapatra, lIl Edition, Tata Mc Graw Hill, 2017.
3. Discrete Mathematics for Computer Scientists & Mathematicians, Joe L. Mott, A. Kandel,T. P. Baker, II Edition, PHI, 2015.
4. Discrete Mathematical Structures Theory and Application- Malik & Sen, I Edition, Cengage Learning, 2012.
5. Discrete Mathematics with Applications, Thomas Koshy, I Edition, Elsevier, 2005.

OOP USING JAVA		
L:T:P (Hrs./week):3:0:0	**SEE Marks : 60**	**Course Code :25PC213CS**
Credits : 3	**CIE Marks : 40**	**Duration of SEE : 3 Hours**

COURSE OBJECTIVES	**COURSE OUTCOMES** On completion of the course, the students will be able to
1. To understand object-oriented programming principles. 2. To develop Java programs using core and advanced OOP concepts. 3. To implement data handling using Collections and file processing. 4. To apply multithreading concepts in Java applications. 5. To develop simple database-driven applications using JDBC.	1. Explain OOP principles and Java program structure 2. Develop Java programs using classes, inheritance, interfaces and exception handling 3. Implement data structures using Java Collections and Generics 4. Demonstrate multithreading and file handling concepts 5. Build simple database applications using JDBC

CO/PO &PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	3	2	2	2	2	-	-	-	-	1	2	1	2
CO2	3	3	3	2	2	-	-	-	1	1	2	2	3
CO3	3	3	3	2	2	-	-	-	1	1	2	2	3
CO4	3	3	3	2	2	-	-	-	1	1	2	2	3
CO5	3	3	3	3	2	-	-	-	1	1	2	2	3

Unit-I

Java Fundamentals & OOP

OOP principles (Abstraction, Encapsulation, Inheritance, Polymorphism), Java Architecture (JVM, JRE, JDK) , Structure of Java Program, Data Types, Variables, Type Casting, Operators & Control Statements, Arrays, Classes and Objects, Constructors, Methods & Overloading, this, static, final, String and StringBuilder classes.

Unit-II

Advanced OOP & Exception Handling

Inheritance & Method overriding, super keyword, Abstract Classes, Interfaces (including Functional Interfaces), Packages & Access Modifiers, Exception Handling: try-catch-finally-throw-throws, Custom Exceptions, **Introduction to Lambda Expressions** (Basic syntax only)

Unit-III

Collections & Generics

Generics, Overview of Java Collections Framework, List, Set, Map Interfaces, ArrayList, LinkedList, HashSet, TreeSet, HashMap classes, Comparable & Comparator, Iterators, **Basic Stream API** (**filter, map – intro level**)

Unit-IV

Multithreading & File Handling

Java Thread Model, Creating Threads (Thread class & Runnable), Thread Lifecycle, Synchronization, Inter-thread Communication, AutoBoxing, Java Input/Output: exploring java.io, Java I/O classes and interfaces, File Handling: File class, Byte Streams, Character Streams, Serialization

Unit-V

JDBC & Mini Application

JDBC Architecture, Types of JDBC Drivers, steps to Connect to Database, Statement vs PreparedStatement, CRUD Operations, Basic Transaction Handling, Mini **Case Study: Student Management / Employee Record System**

Suggested Reading:

1. The Complete Reference Java, 12th Edition – Herbert Scheldt
2. JDBC: Practical Guide for Java Programmers, 2nd Edition Gregory D. Speegle
3. Head First Java, 3rd Edition – Kathy Sierra & Bert Bates
4. Core Java, Volume I: Fundamentals, 12th Edition - Cay S. Horstmann
5. Database Programming with JDBC and Java, 2nd Edition - George

FOUNDATIONS OF DATA SCIENCE		
L:T:P (Hrs./week):3:0:0	**SEE Marks : 60**	**Course Code :25PC215DS**
Credits : 3	**CIE Marks : 40**	**Duration of SEE : 3 Hours**

COURSE OBJECTIVES	**COURSE OUTCOMES** On completion of the course, the students will be able to
1. Understand the fundamental concepts, principles, lifecycle, and interdisciplinary applications of Data Science. 2. Acquire knowledge of data collection, preprocessing, transformation, and statistical techniques for preparing data for analysis. 3. Develop proficiency in using Python-based data science libraries such as NumPy and Pandas for data manipulation, storage, and analysis. 4. Apply exploratory data analysis (EDA) and visualization techniques to summarize, interpret, and derive meaningful insights from datasets. 5. Understand the fundamentals of hypothesis testing, regression, and the machine learning workflow for building data-driven solutions.	1. Explain the concepts, workflow, tools, and real-world applications of Data Science. 2. Perform data pre-processing, transformation, and descriptive statistical analysis on structured datasets. 3. Utilize Python libraries to effectively manipulate, analyze, and manage datasets. 4. Conduct exploratory data analysis and visualization to identify patterns and relationships in data. 5. Apply hypothesis testing, basic regression analysis, and the machine learning workflow to formulate and evaluate data-driven solutions.

CO/PO &PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	3	2	2	2	2	-	-	-	-	1	2	2	2
CO2	3	3	3	2	2	-	-	-	1	1	2	3	2
CO3	3	3	3	2	2	-	-	-	1	1	2	3	3
CO4	3	3	3	2	2	-	-	-	1	1	2	3	3
CO5	3	3	3	3	2	-	-	-	1	1	2	3	3

Unit-I

Introduction: Data Science, Computational Thinking, Skills for Data Science, Tools for Data Science.

Data science in practice: Finance, Public Policy, Politics, Healthcare, Urban Planning, Education, Libraries. Relevance of Data Science to Other Fields, The Relationship between Data Science and Information Science: Information vs. Data Users in Information Science.

Data: Introduction, Data Types and Data Collections: Open Data, Social Media Data, Multimodal Data, Data Storage and Presentation. Data Pre-processing: Data Cleaning, Data Integration, Data Transformation, Data Reduction, Data Discretization.

Unit-II

Data Science Tools and Techniques: Introduction: Data Analysis and Data Analytics, Descriptive Analysis, Diagnostic Analytics, Predictive Analytics, Prescriptive Analytics, Exploratory Analysis, Mechanistic Analysis.

Statistics: Understanding attributes and their types, Types of attributes, Discrete and continuous attributes, measuring central tendency: Mean, Mode, Median, measuring dispersion, Skewness and kurtosis, understanding relationships using covariance and correlation coefficients.

Descriptive Statistics: Understanding statistics, distribution functions uniform distribution, normal distribution, exponential distribution, binomial distribution. Cumulative distribution function, descriptive statistics.

Unit-III

Python Libraries for Data Science: Numpy and Pandas: Arithmetic with Numpy Arrays, Pseudorandom Number Generators, Universal Functions, Array-oriented programming with arrays, File input/output with arrays, Linear Algebra, Introduction to Pandas data structures, DataFrame, Indexing, Sorting and Ranking, Summarization and computing descriptive Statistics.

Data Loading, Storage, and File Formats: reading and writing data in text formats, Binary Data Format, Interacting with Web APIs and Databases.

Unit-IV

The Fundamentals of Exploratory Data Analysis: Significance of EDA, Comparison of EDA with Classical and Bayesian Analysis, Getting Started with EDA, Visual Aids for EDA, Case Study: EDA with Personal Email.

Data Transformation: Transformation Techniques - data deduplication, Replacing Values, Handling Missing Data, Outlier Detection and Filtering, Renaming Axis Indexes, Outlier detection and filtering, Benefits of Data Transformation**.**

Correlation: Introduction, types of analysis: Univariate, Bivariate, and Multivariate analysis, Multivariate Analysis using Titanic Dataset, Simpson's Paradox.

Unit-V

Hypothesis Testing and Regression: Hypothesis Testing, understanding Regression, Model Development and Evaluation, Types of Machine Learning: Understanding Supervised Learning, Unsupervised Learning, Reinforcement Learning, Unified Machine Learning Workflow.

Suggested Reading:

1. Shah, Chirag. A hands-on introduction to data science. Cambridge University Press, 2020.
2. Mukhiya, Suresh Kumar, and Usman Ahmed. Hands-On Exploratory Data Analysis with Python: Perform EDA techniques to understand, summarize, and investigate your data. Packt Publishing Ltd, 2020.
3. Wes, McKinney. Python for data analysis. O'Reilly Media, Inc., 2012.

DATA SCIENCE LAB		
L:T:P (Hrs./week):0:0:2	**SEE Marks : 60**	**Course Code :25PC251DS**
Credits : 1	**CIE Marks : 40**	**Duration of SEE : 3 Hours**

COURSE OBJECTIVES	**COURSE OUTCOMES** On completion of the course, the students will be able to
1. Develop proficiency in Python programming and utilize fundamental programming constructs, data structures, file handling, and exception handling. 2. Apply Python-based data science libraries such as **NumPy and Pandas** for data manipulation, pre-processing, transformation, and statistical analysis. 3. Perform exploratory data analysis, data visualization, and statistical interpretation using real-world datasets to identify patterns, trends, and relationships. 4. Implement data pre-processing techniques, including handling missing values, outlier detection, feature transformation, and normalization for preparing datasets for analysis. 5. Develop and evaluate supervised and unsupervised machine learning models using appropriate algorithms and performance measures for solving real-world data-driven problems.	1. Develop Python programs using essential programming constructs, file handling, and exception handling techniques for data science applications. 2. Apply NumPy and Pandas libraries for data manipulation, pre-processing, transformation, and statistical analysis of structured datasets. 3. Perform exploratory data analysis and visualization using appropriate Python libraries to extract meaningful insight from datasets. 4. Apply statistical techniques and data pre-processing methods and feature transformation for effective data analysis. 5. Develop and evaluate supervised and unsupervised machine learning models using appropriate algorithms and performance metrics to derive data-driven solutions.

CO/PO &PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	3	2	2	3	2	-	-	-	-	1	2	2	3
CO2	3	3	3	3	2	-	-	-	1	1	2	3	3
CO3	3	2	3	3	3	-	-	-	1	1	3	3	3
CO4	3	3	3	3	2	-	-	-	1	1	2	3	3
CO5	3	3	3	3	3	-	-	-	1	1	3	3	3

List of Experiments

1. Write Python programs to demonstrate the use of variables, data types, operators, control structures, functions, strings, lists, tuples, dictionaries, sets, file handling, and exception handling.

2. Develop Python programs using **NumPy** to perform array creation, indexing, slicing, reshaping, broadcasting, vectorized arithmetic operations, random number generation, matrix operations, and basic linear algebra computations.
3. Develop Python programs using **Pandas** to create Series and DataFrames, perform indexing, filtering, sorting, grouping, aggregation, merging, concatenation, and descriptive statistical analysis on the **Students Performance Dataset**.
4. Load the **Titanic Dataset (Kaggle)** from CSV, Excel, JSON, and SQLite database formats using Pandas, perform data import/export operations.
5. Perform data cleaning and preprocessing on the **Titanic Dataset (Kaggle)** by handling missing values, removing duplicate records, encoding categorical attributes, normalizing numerical features, and transforming the dataset for analysis.
6. Compute descriptive statistical measures including mean, median, mode, variance, standard deviation, skewness, kurtosis, covariance, and correlation for the **Wine Quality Dataset (UCI/Kaggle)** using Python.
7. Generate line plots, bar charts, histograms, scatter plots, box plots, violin plots, heat maps, and pair plots for the **Iris Dataset** using Matplotlib and Seaborn, and interpret the visualizations.
8. Perform Exploratory Data Analysis (EDA) on the **Titanic Dataset (Kaggle)** by generating summary statistics, identifying missing values and outliers, analyzing feature distributions, and visualizing relationships among variables.
9. Perform univariate, bivariate, and multivariate analysis on the **Heart Disease Dataset (Kaggle)** by computing covariance, correlation coefficients, correlation matrices, and heat maps, and interpret the relationships among features.
10. Perform time series analysis on the **Individual Household Electric Power Consumption Dataset (UCI Machine Learning Repository)** by preprocessing timestamps, analyzing trends and seasonality, computing moving averages, rolling statistics, and visualizing temporal patterns.
11. Identify and handle missing values and detect outliers in the **House Prices Dataset (Kaggle)** using mean, median, mode and interpolation, Interquartile Range (IQR), and Z-score techniques.
12. Develop supervised learning models using **Linear Regression** and **Decision Tree Classification** on the **Diabetes Dataset (Scikit-learn)** and evaluate the models using appropriate performance metrics.
13. Develop unsupervised learning models using **K-Means Clustering**, **Hierarchical Clustering**, and **Principal Component Analysis (PCA)** on the **Mall Customer Segmentation Dataset (Kaggle)** and visualize the resulting clusters.

DATA STRUCTURES LAB		
L:T:P (Hrs./week):0:0:2	**SEE Marks : 60**	**Course Code : 25PC252CS**
Credits : 1	**CIE Marks : 40**	**Duration of SEE : 3 Hours**

COURSE OBJECTIVES	**COURSE OUTCOMES** On completion of the course, the students will be able to
1. To develop skills to design and analyse simple linear and nonlinear data structures. 2. To gain programming skills to implement sorting and searching algorithms 3. To Strengthen the ability to identify and apply the suitable data structures for the given real world problem. 4. To Gain knowledge in practical applications of data structures	1. Implement linear data structures such as arrays, stacks, queues, and their variations to perform basic operations like insertion, deletion, and traversal. 2. Apply linked lists (singly, doubly, and circular) for dynamic memory management and use them to implement linear data structures such as stacks and queues. 3. Use standard searching (linear and binary) and sorting techniques (selection, insertion, merge, quick, heap) to process and organize data efficiently. 4. Construct tree structures such as binary trees, binary search trees, and AVL trees, and apply recursive traversal techniques. 5. Apply graph traversal (DFS and BFS), hashing techniques, and minimum spanning tree algorithms (Prim's and Kruskal's) for solving computing problems.

CO/PO &PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	3	2	-	-	3	-	-	-	-	-	-	3	2
CO2	3	2	-	-	3	-	-	-	-	-	-	3	3
CO3	3	3	-	-	3	-	-	-	-	-	-	2	3
CO4	3	3	-	2	3	-	-	-	-	-	-	3	2
CO5	3	3	-	3	3	-	-	-	-	-	-	2	3

List of Experiments

Write a program using arrays:

a. To check whether the given matrix is sparse or not and display triplet representation.
b. To add two given polynomials.

2. Write a program to implement the following linear data structures using arrays:
   a. Stacks
   b. Queues
   c. Circular Queue
3. Write a program to implement the following stack applications:
   a. String Reversal.
   b. Infix to Postfix Conversion.
   c. Postfix Expression Evaluation.
   d. Balanced Parenthesis
4. Write a program to implement insertion, deletion and search operation on Singly Linked List
5. Write a program to implement insertion, deletion and search operation on Doubly Linked List.
6. Write a program to implement insertion, deletion and search operation on Circular Linked List.
7. Write a program to implement the following using singly linked list:
   a. Stacks
   b. Queues
8. Write a program to implement the following searching techniques using arrays:
   a. Linear search
   b. Binary search
9. Write a program to build a hash table using linear probing and search for a given element.
10. Write a program to implement inorder, preorder, and postorder traversals in a Binary Tree.
11. Write a program to implement insertion, deletion and search operation in a Binary Search Tree.
12. Write a program to implement insertion, deletion and search operation in an AVL Tree.
13. Write a program to implement DFS and BFS graph traversal techniques.
14. Write a program to find the minimum cost spanning tree for a given connected graph using the following algorithms:
    a. Prims Algorithm
    b. Kruskal's Algorithm.
15. Write a program to implement the following sorting techniques:
    a. Selection sort
    b. Insertion sort
    c. Merge sort
    d. Quick sort
    e. Heap sort

OOP USING JAVA LAB		
L:T:P (Hrs./week):0:0:2	**SEE Marks : 60**	**Course Code** : 25PC253CS
Credits : 1	**CIE Marks : 40**	**Duration of SEE : 3 Hours**

COURSE OBJECTIVES	**COURSE OUTCOMES** On completion of the course, the students will be able to
1. Develop Java programs using core object-oriented principles. 2. Handle exceptions and implement basic multithreaded applications. 3. Perform file operations including reading, writing, and serialization. 4. Use collections effectively for data storage and manipulation. 5. Apply advanced Java features for efficient problem-solving.	1. Apply object-oriented programming concepts such as classes, inheritance, polymorphism, abstract classes, and interfaces in Java programs 2. Develop programs using exception handling and multithreading with proper synchronization mechanisms. 3. Perform file handling operations using Java I/O streams, including serialization and deserialization. 4. Utilize Java Collection Framework components (List, Set, Map) and implement object comparison techniques. 5. Apply modern Java features such as lambda expressions, Stream API, and regular expressions for efficient data processing.

CO/PO &PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	3	2	3	-	3	-	-	-	-	-	2	3	2
CO2	3	3	3	1	3	-	-	-	-	-	2	3	3
CO3	2	2	3	-	3	-	-	-	-	-	1	2	2
CO4	2	3	3	-	3	-	-	-	-	-	2	1	2
CO5	2	2	3	-	3	-	-	-	1	-	3	2	3

List of Experiments:

1. Write a Java program to illustrate Command Line Arguments.
2. Write a Java program to illustrate java.util.Scanner class for input.
3. Write a Java program to illustrate the concept of classes with methods (Box class).
4. Write a Java program to illustrate the concept of constructor Overloading (Box class).
5. Write a Java program to illustrate the concept of method Overloading (Test class).

6. Write a Java Program that reads a line of integers, and then displays each integer, and the sum of all the integers Use java.util.StringTokenizer class
7. Write a Java program to illustrate the concept of Single level and Multi level Inheritance.
8. Write a Java program to illustrate the concept of Method Overriding (Figure, Triangle, Rectangle class). (demonstrate Runtime polymorphism – Dynamic method dispatch).
9. Write a Java program to illustrate the concept of Abstract Classes (Method Overriding).
10. Write a Java program to demonstrate the Interfaces.
11. Write a Java Program to implement Lambda Functions.
12. Write a Java Program to implement Regular Expressions.
13. Write a Java program to demonstrate checked and unchecked Exceptions.
14. Write a Java Program to demonstrate user defined Exceptions.
15. Write a Java program to illustrate collection classes like Array List, Linked List, using Iterator interface and List Iterator interface.
16. Write a Java program to illustrate the concept of Hashmap (key-value).
17. Write a Java Program to implement Stream API.
18. Write a Java program to illustrate the concept of threading using Thread Class and runnable Interface.
19. Write a Java program to illustrate the concept of Thread synchronization.
20. Write a Java program that reads a file name from the user, and then displays information about whether the file exists, whether the file is readable, whether the file is writable, the type of file and the length of the file in bytes.
21. Write a Java program to illustrate the concept of I/O Streams (FileInputStream, FileReader, FileOutputStream, FileWriter).
22. A program to demonstrate the usage of Filter Stream (DataInputStream and DataOutuputStream) and Buffered I/O streams.
23. Write a Java program to implement Serialization / De-serialization concept.
24. Write a Java program for to illustrate CRUD Operations on a database using JDBC.
25. Write a Java program for to illustrate PreparedStatement with ResultSet.

SKILL DEVELOPMENT COURSE: IOT		
L:T:P(Hrs. / Week)0:0:2	**CIE Marks : 50**	**Course Code : 25ES254EC**
Credit : 1		

COURSE OBJECTIVES	**COURSE OUTCOMES** On completion of the course, the students will be able to
1. Introduce the fundamental concepts and architecture of the Internet of Things (IoT). 2. Develop practical skills in interfacing sensors, actuators, and embedded IoT platforms. 3. Enable students to acquire, process, and communicate sensor data using wired and wireless technologies. 4. Familiarize students with cloud-based IoT platforms for monitoring and control. 5. Design and implement domain-specific IoT solutions for engineering applications.	1. Explain the basic concepts, architecture, and components of IoT systems. 2. Interface sensors and actuators with embedded IoT platforms. 3. Develop IoT applications using communication protocols and cloud platforms. 4. Analyze real-time sensor data for monitoring and automation. 5. Design and demonstrate an IoT solution addressing an engineering problem.

CO/PO &PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	3	2	2	-	3	-	-	-	1	-	-	1	2
CO2	3	3	2	2	3	-	-	-	-	-	-	2	3
CO3	2	2	3	2	3	-	-	-	1	-	-	2	3
CO4	2	2	3	3	3	-	-	-	1	-	-	2	3
CO5	2	3	3	3	3	1	1	-	3	2	2	3	3

List of Experiments

1. Interface an LED with ESP32/Arduino/Raspberry Pi and write a program to blink the LED using digital output.
2. Interface a Push Button with ESP32/Arduino/Raspberry Pi and write a program to control an LED based on the button state.
3. IR/LDR Sensor Interfacing: Interface an IR or LDR sensor with ESP32/Arduino/Raspberry Pi and write a program to detect an object/light intensity and control an LED accordingly.
4. Interface a 16×2 LCD (I2C) with ESP32/Arduino/Raspberry Pi and write a program to display text and sensor readings.
5. Interface a DHT11 Temperature and Humidity Sensor with ESP32/Arduino/Raspberry Pi and display the measured values on the Serial Monitor/LCD.

6. Interface a Relay Module with ESP32/Arduino/Raspberry Pi and write a program to control a DC motor or AC load using a push button or sensor input.
7. Develop a Bluetooth Low Energy (BLE) or Bluetooth Serial application using ESP32 to control an LED or relay from a smartphone.

8. Interface an HC-SR04 Ultrasonic Sensor with ESP32/Arduino/Raspberry Pi and measure the distance to an object. Display the measured distance on the Serial Monitor/LCD.
9. Acquire temperature and humidity data from the DHT11 sensor using ESP32 and upload the sensor readings to the ThingSpeak Cloud. Visualize and analyze the data using ThingSpeak charts.
10. Develop a mobile-based IoT application using ESP32 to remotely monitor temperature and humidity and control an LED or relay through the Blynk IoT mobile dashboard.
11. Develop an embedded web server on ESP32 to monitor sensor data and control an LED/relay through a web browser over a Wi-Fi network.

Mini Project:
Students shall implement a complete domain-specific IoT application (Smart Home, Smart Agriculture, Smart Energy Monitoring, Smart Environment Monitoring, Smart Healthcare, etc.) using ESP32/Arduino/Raspberry Pi, integrating sensors, actuators, wireless communication, and cloud/mobile monitoring.

SCHEME OF INSTRUCTION & EXAMINATION

B.E. CSE (Artificial Intelligence)

SEMESTER-IV

S. No.	Course Code	Course Title	Scheme of Instruction				Scheme of Examination			Credits
			L	T	D/P	Contact Hrs./ Week	CIE	SEE	Duration in Hrs.	
Theory Courses										
1	25PC215CS	Computer Organization and Architecture	3	-	-	3	40	60	3	3
2	25PC221AI	Artificial Intelligence	3	-	-	3	40	60	3	3
3	25PC222AI	Full Stack Development	3	1	-	4	40	60	3	4
4	25PC224DS	Database Systems	3	-	-	3	40	60	3	3
5	25PC314CS	Design and Analysis of Algorithms	3	-	-	3	40	60	3	3
Practical / Laboratory Courses										
6	25PC255AI	Artificial Intelligence Lab	-	-	2	2	40	60	3	1
7	25PC256DS	Database Systems Lab	-	-	2	2	40	60	3	1
8	25PC257AI	Full Stack Development Lab	-	-	2	2	40	60	3	1
9	25PW258AI	Mini Project	-	-	2	6	40	60	3	3
Total			**15**	**1**	**08**	**28**	**360**	**540**	**27**	**22**

ARTIFICIAL INTELLIGENCE		
L:T:P (Hrs./week):3:0:0	**SEE Marks : 60**	**Course Code :25PC221AI**
Credits : 3	**CIE Marks : 40**	**Duration of SEE : 3 Hours**

COURSE OBJECTIVES	**COURSE OUTCOMES** On completion of the course, the students will be able to
1. Understand the basic concepts, foundations, and applications of Artificial Intelligence. 2. Learn problem-solving methods and search techniques in AI. 3. Study logic, reasoning and knowledge representation techniques. 4. Understand uncertainty handling and machine learning approaches. 5. Learn expert systems, their architecture, and AI paradigms.	1. Understand the fundamental concepts of Artificial Intelligence, formulate classical state-space problems and analyze the behavior of intelligent agents in different environments 2. Apply search algorithms uninformed, informed and adversarial search techniques for problem solving task efficiently. 3. Represent knowledge and perform logical reasoning using propositional logic, predicate logic, logic programming, semantic networks, and frames. 4. Analyze methods for reasoning under uncertainty using probabilistic models and explain machine learning techniques 5. Describe expert systems, applications and compare connectionist and symbolic AI approaches.

CO/PO &PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	3	3	2	1	2	–	–	1	–	–	2	3	2
CO2	3	3	3	2	2	–	–	–	–	–	2	3	2
CO3	3	3	3	2	2	–	–	1	–	–	2	3	2
CO4	3	3	3	3	2	–	–	2	–	–	2	3	3
CO5	3	3	2	2	2	–	–	1	–	–	3	3	3

Unit-I

Introduction to Artificial Intelligence

Introduction, Brief History, Intelligent Systems, foundations of AI, Sub-Areas of AI. AI Applications in Healthcare, Finance, Autonomous Systems, Generative AI.

State Space Representation, Problem Formulation, Classical State Space Problems.

Agents

Agents and Environments, the concept of Rationality, Performance measures, the nature of

Environments, The Structure of Agents, Simple Reflex Agents, Model-Based Agents, Goal-Based Agents, Utility-Based Agents, Learning Agents.

Unit-II

Problem Solving

Problem Solving by Searching, Uninformed Search, Informed Search, Heuristic Functions, Greedy Best-First Search, A* Search, Local Search Algorithms, Hill Climbing, Simulated Annealing.

Adversarial Search

Game Trees, Minimax Algorithm, Alpha-Beta Pruning, Iterative Deepening.

Unit-III

Logic Concepts and Logic Programming

Introduction, Propositional Calculus, Propositional Logic, Natural Deduction System, Axiomatic System, Predicate Logic, Logic Programming.

Knowledge Representation

Introduction, Approaches to Knowledge Representation, Knowledge Representation using Semantic Network, Knowledge Representation using Frames.

Unit-IV

Uncertainty Measures

Bayesian Belief Networks, Certainty Factor Theory, Dempster-Shafer Theory

Introduction to Machine Learning: Machine Learning Systems, Supervised and Unsupervised Learning

Unit-V

Expert System

Introduction, Phases in Building Expert Systems, Expert System Architecture, Expert System versus Traditional Systems, Rule-Based Expert Systems, Truth Maintenance Systems, Applications of Expert System, List of Shells and Tools, Symbolic AI Vs Connectionist AI.

Suggested Reading:

1. Artificial Intelligence-A Modern Approach, Second Edition by Stuart Russell, Peter Norvig.
2. Artificial Intelligence, Saroj Kaushik, Cengage Learning.
3. Artificial Intelligence, Third Edition by Elaine Rich, Kevin Knight, Shivashankar B Nair, Tata-McGraw Hill.

COMPUTER ORGANIZATION AND ARCHITECTURE		
L:T:P (Hrs./week):3:0:0	**SEE Marks : 60**	**Course Code :25PC215CS**
Credits : 3	**CIE Marks : 40**	**Duration of SEE : 3 Hours**

COURSE OBJECTIVES	**COURSE OUTCOMES** On completion of the course, the students will be able to
1. Understand the fundamental concepts of computer organization, data representation, register transfer operations, and instruction execution. 2. Explain the organization and operation of the CPU, control unit, memory hierarchy, and input/output subsystems of a computer. 3. Analyze processor organization, pipeline processing, and memory management techniques to understand the performance of modern computer systems.	1. Explain the fundamental concepts of computer organization, data representation, register transfer operations, and instruction execution. 2. Describe the organization and operation of computer registers, bus systems, micro operations, instruction sets, and control units. 3. Apply CPU organization concepts, instruction formats, addressing modes, and program control techniques to solve basic computer organization problems. 4. Explain the principles of pipeline processing, parallel processing, and RISC/CISC architectures for improving processor performance. 5. Explain memory hierarchy, cache organization, virtual memory, and input/output organization used in modern computer systems.

CO/PO &PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	3	2	-	-	-	-	-	-	-	-	1	1	–
CO2	3	2	-	-	2	-	-	-	-	-	-	1	1
CO3	3	2	2	-	2	-	-	-	-	-	-	2	1
CO4	3	2	1	1	2	-	-	-	-	-	1	2	2
CO5	3	3	1	1	2	-	-	-	-	-	1	2	2

Unit-I

Fundamental Concepts and Data Representation

Introduction to computer systems, Functional units of a computer, Basic operational concepts, Computer organization, Computer design, Computer architecture.

Data types, Number systems, Complements, Fixed-point representation, Floating-point representation.

Register Transfer Operations

Register transfer language, Register transfer, Bus and memory transfers.

Unit-II

Micro operations

Arithmetic micro operations, Logic micro operations, Shift micro operations, Arithmetic logic shift unit.

Basic Computer Organization

Computer registers, Common Bus system, Computer instructions, Timing and control, Instruction cycle.

Unit-III

Instruction Set

Register-reference instructions, Memory-reference instructions, Input-output and Interrupt.

Control Unit Design

Control memory, Address sequencing, Microprogram example, Design of control unit

Unit-IV

CPU Organization

General register organization, Stack organization, Instruction formats, Addressing modes, Data transfer and manipulation, program control.

Parallel Processing and Pipeline

Parallel Processing, Pipeline, Instruction pipeline, RISC, CISC.

Unit-V

Memory Organization

Memory hierarchy, Main memory, Auxiliary memory, Associative memory, Cache memory, Virtual memory.

Input–Output Organization

I/O interface, Asynchronous serial transfer, asynchronous communication interface, Modes of Transfer, Priority interrupt, DMA.

Suggested Reading:

1. Computer System Architecture, M. Morris Mano, 3rd Edition, Pearson Education.
2. Computer Organization and Architecture: Designing for Performance, William Stallings, 11th Edition, Pearson.
3. Computer Organization and Embedded Systems, Carl Hamacher et al., 6th Edition, McGraw-Hill Education.
4. Structured Computer Organization, Andrew S. Tanenbaum and Todd Austin, 6th Edition, Pearson Education.

FULL STACK DEVELOPMENT		
L:T:P (Hrs./week):3:1:0	**SEE Marks : 60**	**Course Code : 25PC222AI**
Credits : 4	**CIE Marks : 40**	**Duration of SEE : 3 Hours**

COURSE OBJECTIVES	**COURSE OUTCOMES** On completion of the course, the students will be able to
1. Understand the architecture of full-stack systems for AI applications. 2. Develop backend services and APIs integrating machine-learning models. 3. Design responsive and interactive front-end interfaces. 4. Manage databases and data pipelines for AI systems. 5. Deploy AI-enabled web applications using modern tools and platforms	1. Explain the architecture, components, and workflow of AI-based full-stack systems. 2. Develop responsive front-end applications, backend REST APIs using modern web technologies, and Python frameworks. 3. Integrate databases and machine learning models into full-stack applications using appropriate data processing techniques. 4. Design and implement AI-enabled full-stack applications with authentication, data visualization, and frontend-backend integration. 5. Deploy AI-enabled web applications using cloud and containerization technologies while applying version control and collaborative development practices.

CO/PO &PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	3	2	1	–	2	–	–	1	–	–	1	2	2
CO2	3	3	3	2	3	–	–	–	–	–	2	2	3
CO3	3	3	3	2	3	–	–	–	–	–	2	3	3
CO4	3	3	3	3	3	–	–	2	–	–	2	3	3
CO5	3	3	3	3	3	–	–	2	1	1	3	3	3

Unit-I

Introduction to Full Stack Development and Front-End Technologies

Overview of Full Stack Development, Architecture of AI-Based Full Stack Systems, Frontend, Backend, Database and AI Layer, Case Studies of AI Web Applications, HTML5 Fundamentals, CSS3 Styling, JavaScript Basics, Responsive Web Design Principles, Introduction to ReactJS, Components, Props, State, Hooks, Building Interactive User Interfaces.

Unit-II

Backend Development and Database Systems

Introduction to Flask Framework, Introduction to FastAPI Framework, REST API Design Principles, API Development using JSON, SQL using SQLite, NoSQL using MongoDB, Backend-Database Integration, CRUD Operations, Data Storage and Retrieval Techniques.

Unit-III

Machine Learning Integration and Data Processing

Machine Learning Workflow, Data Preprocessing Techniques, Data Validation, Data Processing Pipelines, Pipeline Design and Implementation, Model Serialization using Pickle and Joblib, Integration of Machine Learning Models with Backend Services, Prediction API Development.

Unit-IV

Full Stack Integration, Visualization and Deployment

Frontend-Backend Communication using Fetch API and Axios, Dynamic Data Rendering, Data Visualization using Plotly and Chart.js, Dashboard Development, Authentication Mechanisms using JWT and OAuth, Cloud Deployment Basics, Deployment using Streamlit, Render and Hugging Face Spaces, Docker Fundamentals, CI/CD Concepts, API Testing and Optimization.

Unit-V

AI-Based Full Stack Application Development

Development of AI-Based Full Stack Web Applications, End-to-End System Integration, Git and GitHub for Version Control, Project Documentation using README, Peer Code Review, Testing and Deployment of AI Applications, Mini Project Development, Project Demonstration and Presentation.

Suggested Readings:

1. M. Grinberg, Flask Web Development: Developing Web Applications with Python, 2nd ed. Sebastopol, CA, USA: O'Reilly Media, 2018. ISBN: 978-1491991732.
2. A. Géron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 3rd ed. Sebastopol, CA, USA: O'Reilly Media, 2022. ISBN: 978-1098125967.
3. A. Géron, Hands-On Machine Learning with Scikit-Learn and PyTorch. Sebastopol, CA, USA: O'Reilly Media, 2025. ISBN: 979-8341607989.
4. S. Chen et al., Deep Learning and Machine Learning: From Theory to Practice. 2024. [Online]. Available: https://arxiv.org/abs/2410.19849
5. K. Chen et al., Deep Learning and Machine Learning: Design Patterns for Big Data Analytics. 2024. [Online]. Available: https://arxiv.org/abs/2410.03795
6. M. Makai, Full Stack Python. Self-published, 2016. [Online]. Available:

DATABASE SYSTEMS		
L:T:P (Hrs./week):3:0:0	**SEE Marks : 60**	**Course Code : 25PC224DS**
Credits : 3	**CIE Marks : 40**	**Duration of SEE : 3 Hours**

COURSE OBJECTIVES	**COURSE OUTCOMES** On completion of the course, the students will be able to
1. Understand the fundamentals, architecture, and components of database management systems. 2. Apply SQL to define, manipulate, and retrieve data from relational databases. 3. Design relational databases using ER modelling and relational algebra concepts. 4. Apply functional dependencies and normalization techniques to design efficient database schemas. 5. Understand indexing, transaction processing, and concurrency control mechanisms for efficient database management.	1. Explain the concepts, architecture, components, applications, and advantages of database systems over file processing systems. 2. Design relational databases and apply SQL (DDL, DML, constraints, joins, views, triggers, and transactions) for database management. 3. Apply ER modelling, relational algebra, and normalization techniques to design efficient relational databases. 4. Analyse functional dependencies, normalization, indexing, and database design techniques to ensure data integrity and performance. 5. Evaluate transaction processing, concurrency control, indexing, and recovery mechanisms for reliable and secure database systems.

CO/PO &PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	3	2	2	3	1	-	-	-	-	-	1	1	1
CO2	3	2	3	1	1	-	-	-	-	-	1	2	3
CO3	3	3	3	2	1	-	-	-	-	-	2	2	2
CO4	3	3	2	2	-	1	-	-	-	-	2	2	2
CO5	2	3	2	2	1	-	-	-	-	-	2	3	2

Unit-I

Introduction to Database Systems

Database-System Applications, Purpose of Database Systems, Characteristics and Advantages of DBMS over File Processing Systems, View of Data, Data Abstraction (Physical, Logical, and View Levels), Data Independence, Database Languages (DDL, DML, DCL, and TCL), Database Design, Database Engine, Database and Application Architecture (Two-Tier and Three-Tier), Database Users and Administrator (DBA), History and Evolution of Database Systems.

Unit-II

SQL and Advanced SQL

SQL: Introduction to SQL, Overview of SQL Query Language, SQL Data Definition, Basic Structure of SQL Queries, Additional Basic Operations, Set Operations, Null Values, Aggregate Functions, Nested Subqueries, Modification of Database, Join Expressions, Views, Transactions, Integrity Constraints, and Triggers.

Unit-III

Data Modeling and Relational Databases

Entity–Relationship Model: Entity Sets, Attributes, Keys, Relationship Sets, Mapping Cardinalities, Participation Constraints, Weak Entity Sets, Extended ER Features, and Reduction of ER Diagrams to Relational Schemas. **Relational Algebra:** Selection, Projection, Union, Set Difference, Cartesian Product, Rename, Join, Division, Extended Relational Algebra Operations, and Relational Algebra Queries.

Unit-IV

Database Design and Normalization

Functional Dependencies, Armstrong's Axioms, Closure of Functional Dependencies, Decomposition, Lossless Join and Dependency Preservation, Database Design Process, Normalization, Anomalies, First Normal Form (1NF), Second Normal Form (2NF), Third Normal Form (3NF), Boyce–Codd Normal Form (BCNF), Fourth Normal Form (4NF), Fifth Normal Form (5NF), and Denormalization.

Unit-V

Indexing and Transaction Processing

Indexing: Basic Concepts, Ordered Indices, B+-Tree Index Files, B-Tree Index Files.

Transaction Processing and Concurrency Control: Transaction Concepts, ACID Properties, Conflict Serializability, Transaction States. Concurrency Control: Lock-Based Protocols, Multiple Granularity, Timestamp-Based Protocols.

Suggested Readings:

1. Abraham Silberschatz, Henry F Korth, S Sudarshan, Database System Concepts, McGraw-Hill International Edition, 6th Edition, 2010

2. Raghu Ramakrishnan and Johannes Gehrke, ―Database Management Systems‖, Tata McGraw-Hill, New Delhi,2008.

3. RamezElmasri and Shamkant B Navathe, ―Fundamentals of Database Systems‖, Addison Wesley, USA,2010.

DESIGN AND ANALYSIS OF ALGORITHMS		
L:T:P (Hrs./week):3:0:0	**SEE Marks : 60**	**Course Code : 25PC314CS**
Credits : 3	**CIE Marks : 40**	**Duration of SEE : 3 Hours**

COURSE OBJECTIVES	**COURSE OUTCOMES** On completion of the course, the students will be able to
1. To review elementary data structures , order notation and algorithm analysis 2. To introduce Divide-and-Conquer algorithm design strategy and problems solved using it. 3. To introduce Greedy and Dynamic Programming algorithm design strategies and problems solved using it. 4. To introduce Backtracking and Branch-and-Boound algorithm design strategies and problems solved using it. 5. To introduce the concepts of NP-hard and NP-complete problems and parallel algorithms	1. Analyze the time and space complexity of algorithms; explain the different elementary data structures. 2. Design algorithms for various computing problems using the Divide and Conquer Strategy. 3. Design algorithms for various computing problems using the Greedy and Dynamic Programming Strategies. 4. Design algorithms for various computing problems using the Backtracking, Branch and Bound Strategies 5. Explain NP-Hard, NP-Complete problems and parallel algorithms.

CO/PO &PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	3	3	1	1	1	–	–	–	–	–	1	2	1
CO2	3	3	2	2	1	–	–	–	–	–	2	2	1
CO3	3	3	3	2	1	–	–	–	–	–	2	3	2
CO4	3	3	3	2	1	–	–	–	–	–	2	3	2
CO5	3	3	2	2	2	–	–	–	–	–	2	3	2

Unit-I

Introduction & Elementary

Data Structures: Introduction, Fundamentals of algorithm(Line Count, Operation Count), Analysis of algorithms(Best, Average, Worst case), Asymptotic Notations(O,Ω,Θ) Recursive Algorithms, Analysis using Recurrence Relations, Master's Theorem.

Review of elementary data structures–Graphs:

BFS, DFS, Bi-Connected Components. Sets: representation, UNION, FIND operations.

Unit-II

Divide-and-Conquer Method

The general method, Binary search, Finding maximum and minimum, Merge sort, Quick sort.

Brute Force:

Knapsack, Traveling salesman problem, Convex-Hull.

Unit-III

Greedy Method

Knapsack problem, Minimum spanning trees, Single source shortest path, Job sequencing with deadlines, Optimal storage on tapes, Optimal merge pattern

Dynamic programming method

All pairs shortest paths, Optimal binary search tress, 0/1 Knapsack problem, Reliability design, Traveling salesman problem

Unit-IV

Back tracking

N-queens problem, Graph coloring, Hamiltonian cycles

Branch-and-bound

FIFO & LC branch and Bound methods, 0/1 Knapsack problem, Traveling salesperson

Unit-V

NP-hard and NP-complete problems

Non-Deterministic algorithms, the classes: P, NP, NP Complete, NP Hard, Satisfiability problem

Proofs for NP Complete Problems

Clique, Vertex Cover.

Parallel Algorithms: Introduction, models for parallel computing

Suggested Readings:

1. Horowitz E, Sahni S, Fundamentals of Computer Algorithms, 2ndEdition, Universities Press, 2007
2. Thomas H.Cormen, Charles E.Leiserson, Ronald L. Rivest and Clifford Stein, "Introduction to Algorithms", Third Edition, PHI Learning Private Limited, 2012
3. Michael T. Goodrich, Roberto Tamassia, Algorithm Design: Foundations, Analysis and Internet Examples, John Wiley & Sons,2002

ARTIFICIAL INTELLIGENCE LAB		
L:T:P (Hrs./week):0:0:2	**SEE Marks : 60**	**Course Code : 25PC255AI**
Credits : 1	**CIE Marks : 40**	**Duration of SEE : 3 Hours**

COURSE OBJECTIVES	**COURSE OUTCOMES** On completion of the course, the students will be able to
1. To develop practical skills in implementing classical Artificial Intelligence algorithms for problem solving and intelligent search. 2. To provide hands-on experience in logic programming, knowledge representation, and rule-based reasoning using appropriate AI tools. 3. To enable students to implement and evaluate fundamental machine learning and neural network techniques for intelligent applications.	1. Implement intelligent search algorithms, including Breadth First Search (BFS), Depth First Search (DFS), Greedy Best-First Search, and A* Search, to solve classical AI problems. 2. Develop logic programs using Prolog to represent knowledge and perform logical inference through facts, rules, and queries. 3. Design and implement simple rule-based expert systems for real-world problem domains. 4. Apply probabilistic learning techniques by implementing the Naïve Bayes classifier for classification problems.

CO/PO &PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	3	3	3	2	3	–	–	–	–	–	2	3	3
CO2	3	3	3	2	3	–	–	–	–	–	2	3	3
CO3	3	3	3	3	3	–	–	1	–	–	2	3	3
CO4	3	3	3	3	3	–	–	1	–	–	2	3	3
CO5	3	3	3	2	3	–	–	–	–	–	2	3	3

List of Experiments

1. Write a Program to implement Breadth First Search.
2. Solve the classical Water-Jug Problem using Breadth First Search.
3. Write a Program to implement Depth First Search.
4. Solve the classical Missionaries and Cannibals Problem using Depth First Search.
5. Write a Program to implement Greedy Best-First Search Algorithm.
6. Write a Program to implement A* Algorithm.
7. Implement logic programming using Prolog by defining facts, rules, and queries, and develop a Family Tree application.

8. Develop a simple Rule-based Expert System for applications such as
    i. Medical Diagnosis
    ii. Career Guidance
    iii. Course Recommendation
9. Develop a Knowledge Representation System using Semantic Networks or Frames for one of the following applications
    i. Animal Classification
    ii. Hospital Management
    iii. Library Management
10. Implement the Naïve Bayes Classifier using scikit-learn on the Iris dataset.

DATABASE SYSTEMS LAB		
L:T:P (Hrs./week):0:0:2	**SEE Marks : 60**	**Course Code : 25PC256DS**
Credits : 1	**CIE Marks : 40**	**Duration of SEE : 3 Hours**

COURSE OBJECTIVES	**COURSE OUTCOMES** On completion of the course, the students will be able to
1. To develop skills in SQL for database definition, manipulation, and query processing. 2. To learn advanced database concepts such as joins subqueries, integrity constraints, and views. 3. To gain practical knowledge of PL/SQL programming including procedures, functions, triggers, and exception handling 4. To understand transaction management, locking mechanisms, and real-time database applications through case studies 5. Get acquainted with GUI development tools and design.	1. Apply DDL, DML, TCL, and DCL commands for database creation and management. 2. Construct few complex SQL queries. 3. Design relational database tables with appropriate integrity constraints. 4. Develop PL/SQL programs to demonstrate the basic construct of programming. 5. Design case studies for practical database use cases such as student information, library systems, and payroll management.

CO/PO &PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	3	2	2	1	1	-	-	-	-	-	1	1	3
CO2	3	2	2	2	-	-	-	-	-	-	2	2	3
CO3	3	3	3	2	-	-	-	-	-	-	1	2	3
CO4	3	2	3	2	-	-	-	-	1	-	2	2	3
CO5	3	3	3	3	-	1		2	2	2	2	3	3

List of Programs

1. Practice DDL Commands.
2. Practice DML Commands.
3. Practice Different Operators.
4. Practice Aggregate Functions and Sub Queries.
5. Practice Group by, Having and Order by Clauses.
6. Practice Various Types of Joins and Built in Functions.

7. Create tables with Complex Integrity Constrains.
8. Practice TCL and DCL Commands and Create Different Views
9. PL/SQL Programs on Control Structures, and Triggers.
10. PL/SQL Programs on Cursors and Exception Handling.
11. PL/SQL Programs on Functions, Procedures, Packages.
12. Create forms and generate reports for student information, library information etc.

FULL STACK DEVELOPMENT LAB		
L:T:P (Hrs./week):0:0:2	**SEE Marks : 60**	**Course Code : 25PC257AI**
Credits : 1	**CIE Marks : 40**	**Duration of SEE : 3 Hours**

COURSE OBJECTIVES	**COURSE OUTCOMES** On completion of the course, the students will be able to
1. Gain hands-on experience in developing responsive front-end applications using modern web technologies. 2. Practice designing backend services, RESTful APIs, and database connectivity for AI-enabled applications. 3. Explore the integration of machine learning models, data processing pipelines, and backend services. 4. Build end-to-end AI-enabled full-stack web applications by integrating frontend, backend, databases, and visualization tools. 5. Acquire practical experience in testing, version control, collaboration, and deployment of AI-based web applications using various development platforms.	1. Design and develop responsive front-end applications using HTML5, CSS3, JavaScript, and ReactJS. 2. Implement RESTful backend services using Flask/FastAPI integrated with SQLite and MongoDB databases. 3. Develop AI-enabled applications by integrating data preprocessing pipelines and machine learning models with backend services. 4. Build and test complete full-stack AI applications incorporating authentication, frontend-backend communication, and interactive data visualization. 5. Deploy AI-enabled web applications using Git/GitHub, Docker, Streamlit, Render, or Hugging Face Spaces while following collaborative software development practices.

CO/PO &PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	3	2	3	1	3	–	–	–	–	–	1	2	3
CO2	3	3	3	2	3	–	–	–	–	–	2	2	3
CO3	3	3	3	3	3	–	–	1	–	–	2	3	3
CO4	3	3	3	3	3	–	–	2	1	1	2	3	3
CO5	3	3	3	3	3	–	–	2	2	2	3	3	3

List of Experiments

1. Design a Responsive Personal Portfolio Website using HTML5, CSS3, Flexbox, Grid, Forms, and Media Queries.
2. Develop Interactive Web Pages using JavaScript (DOM Manipulation, Events, Form Validation, Local Storage, JSON).

3. Build a ReactJS Student Management UI using Components, Props, State, Hooks, Routing and Responsive Layout.
4. Develop REST APIs using Flask (GET, POST, PUT, DELETE) for Student Management System. Test APIs using Postman.
5. Develop REST APIs using FastAPI with automatic Swagger documentation and JSON request/response handling.
6. Database Integration: Connect Flask/FastAPI with SQLite and MongoDB. Implement complete CRUD operations.
7. Data Preprocessing Pipeline: Load dataset using Pandas, perform missing value treatment, encoding, normalization, feature selection and validation.
8. Train and Integrate an ML Model: Train a classification/regression model using Scikit-learn, serialize using Pickle/Joblib, and expose prediction through a REST API.
9. Frontend–Backend Integration: Connect ReactJS frontend with FastAPI/Flask backend using Fetch API/Axios and display live predictions.
10. Dashboard Development: Create interactive dashboards using Plotly and Chart.js for visualizing ML predictions, analytics, and database statistics.
11. Authentication & Deployment: Implement JWT Authentication, Git/GitHub version control, Dockerize the application and deploy using Streamlit/Render/Hugging Face Spaces.
12. Mini Project Implementation: Develop an end-to-end AI-enabled Full Stack application with frontend, backend, database, ML model, authentication, and visualization.
13. Mini Project Demonstration: Project presentation, GitHub repository evaluation, README preparation, peer code review, resume & portfolio update, viva voce

MINI PROJECT		
L:T:P (Hrs./week):0:0:6	**SEE Marks : 60**	**Course Code : 25PW258AI**
Credits : 3	**CIE Marks : 40**	**Duration of SEE : 3 Hours**

COURSE OBJECTIVES	**COURSE OUTCOMES** On completion of the course, the students will be able to
1. To enhance practical and professional skills. 2. To familiarize tools and techniques of systematic literature survey and documentation 3. To expose the students to industry practices and team work. 4. To encourage students to work with innovative and entrepreneurial ideas.	1. Demonstrate the ability to synthesize and apply the knowledge and skills acquired in the academic program to the real-world problems. 2. Evaluate different solutions based on economic and technical feasibility 3. Effectively plan a project and confidently perform all aspects of project management 4. Demonstrate effective coding, written, presentation and oral communication skills.

CO/PO &PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	3	3	2	2	3	–	–	–	1	–	1	3	3
CO2	2	3	3	2	2	1	1	–	1	–	2	3	2
CO3	1	2	3	2	2	1	–	1	3	2	3	2	3
CO4	1	–	2	–	3	–	–	1	2	3	2	2	3

The students are required to carry out mini projects in any of the areas such as Data Structures, Microprocessors and Interfacing, Database Systems, Operating Systems, Design and Analysis of Algorithms, Software Engineering, Data Communications, Web Programming & Services, Computer Networks, Compiler Construction, and Object Oriented System Development.

Students are encouraged to undertake problem statements available on the Smart India Hackathon (SIH) Portal, contributed by Ministries, Public Sector Undertakings (PSUs), Multinational Companies (MNCs), and Non-Governmental Organizations (NGOs).

The project could be classified as hardware, software, modeling, simulation etc. The project should involve one or many elements of techniques such as analysis, design, and synthesis.

The department will appoint a project coordinator who will coordinate the following:

1. Grouping of students (maximum of 3 students in a group)
2. Allotment ofprojects and project guides.

3. All projects allotment is to be completed by the 4th week of the semester so that the students get sufficient time for completion of the project.
4. Disseminate guidelines given by monitoring committee comprising of senior faculty members to the students and their guides.
5. Three periods of contact load will also be assigned to each project guide for project guidance and monitoring at regular intervals.
6. Sessional marks are to be awarded by the monitoring committee.
7. Common norms will be established for the final presentation and documentation of the project report by the respective departments.
8. Students are required to submit a presentation and report on the mini project at the end ofthe semester.