UNIVERSITY OF CALCUTTA

Notification No. CSR/67/2025

It is notified for information of all concerned that in terms of the provisions of Section 54 of the Calcutta University Act, 1979, (as amended), and, in the exercise of her powers under 9(6) of the said Act, the Vice-Chancellor has, by an order dated 22.09.2025, approved the Course Structure & Syllabus for Major Courses of semester-5 & 6 of 4-year Honours and Honours with Research Courses of studies and revised course structure for Minor Courses of 4-year Honours and Honours with Research Courses of studies in Computer Science under CCF,2022.

Detail Syllabus of Minor Courses was Published under CSR/82/2024, dt. 26.09.2024.

The above shall take effect from the Odd semester examinations, 2025 and onwards.

13/10/2025

SENATE HOUSE

Kolkata-700073

13.10.2025

Prof.(Dr.) Debasis Das

Registrar

University of Calcutta

B.Sc. (Honours and Honours with Research), 4 - Years degree program in Computer Science under credit framework (CCF).

(2024)

Semester – V & VI

Semester - 5

Theory Paper	Credit	Contact hours	Practical/Tutorial Paper	Credit	Contact hours
Design & Analysis of Algorithms	03	45	Graph algorithms Lab using C++	1	30
Data Communication and Networking	03	45	Networking Lab	1	30
Theory of Computation	03	45	Tutorial	1	30
Database Management System (DBMS)	03	45	RDBMS Lab	1	30

Theory: Design & Analysis of Algorithms
Credits - 03, Contact hours - 45.

Introduction to Algorithms: Definition, Characteristics, Recursive and Non-recursive algorithms	3 hours
Asymptotic Complexity Analysis of Algorithms: Space and Time Complexity, Efficiency of an algorithm, Growth of Functions, Polynomial and Exponential Complexity, Asymptotic Notations: Big O Notation and Small o notation, Big Ω and Small ω, Big Θ and Small φ Notations, Properties: Best case/worst case/average case analysis of well-known algorithms.	6 hours
Algorithm Design Techniques: Concepts and simple case studies of Greedy algorithms. Divide and conquer: Basic concepts, Case study of selected searching and sorting problems using divide and conquer techniques: Dynamic programming: General issues in Dynamic Programming.	12 hours
Graph Representation and Algorithm: Graph traversal algorithms: BFS, DFS, Minimal spanning trees: Prim's Algorithm, Kruskal's Algorithm, Shortest path algorithms: Floyd's Algorithm, Floyd-Warshall Algorithm, Dijkstra's Algorithm, Graph Colouring Algorithms.	22 hours
Classification of Problems: Concept of P, NP.	2 hours

Text/References Books

1. Introduction to Algorithms, Cormen, Leiserson, Rivest and Stein, TMH.
2. The Design and Analysis of Algorithms, Aho, Hopcroft and Ullman, Pearson Education.
3. The Art of Computer Programming, D.E. Knuth, Pearson Education.
4. Algorithm Design, Jon Kleiberg and Eva Tardos,Pearson Education.
5. Data Structures and Algorithms - K.Mehlhorn.
6. Computer Algorithms, S.Baase, Pearson Education.
7. Fundamentals of Computer Algorithms, E. Horowitz and Sahani, Galgotia
8. Combinational Algorithms- Theory and Practice, E.M. Reingold, J. Nievergelt and N. Deo, PHI.

Practical: Graph algorithms Lab using C++
Credits - 01, Contact hours - 30.

Pre-requisite;

1. Programming Foundations in C++

- Syntax, data types, loops, conditionals, and functions
- STL basics — especially vector, queue, stack, and map
- Dynamic memory and pointers (essential for building graph structures)

2. Graph Theory Fundamentals

- Terminology: vertices, edges, adjacency, degree, cycles
- Graph types: directed vs undirected, weighted vs unweighted
- Representations: adjacency matrix vs adjacency list

3. Math & Logic Readiness

- Basic combinatorics and logic reasoning
- Set theory (for understanding disjoint sets and connectivity)
- Matrices (helpful for adjacency matrix representations)

4. Compilers

- C++ compiler GCC
  or
- MSVC (Microsoft Visual C++), Visual Studio Community Edition.

Laboratory exercises to be based on Graph Theory using C++ and based on the following;

Implementation of Graph algorithms: Single Spanning Tree Generation using - BFS, DFS, Minimal Spanning Tree Generation using - Prim's Algorithm, Kruskal's Algorithm, Shortest Path finding using Floyd's Algorithm, Floyd-Warshall Algorithm, Dijkstra's Algorithm, Graph Partitioning Algorithm.

Sample questions

1. Write a C++ program to generate a single spanning tree using Breadth-First Search (BFS). Trace and display visited nodes.
2. Develop a Depth-First Search (DFS)-based C++ algorithm to construct a spanning tree from a connected undirected graph. Compare traversal order with BFS.
3. Design and implement Prim's Algorithm using adjacency list and priority queue in C++. Display the edges included in the MST and their weights.
4. Construct Kruskal's Algorithm using disjoint set union in C++. Identify cycle prevention strategy and display final edge set of the MST.
5. Write a C++ program to compute the shortest paths from a single source using Dijkstra's Algorithm. Provide output trace with distances and paths.
6. Implement Floyd's Algorithm to find all-pairs shortest paths in a weighted graph using adjacency matrix representation. Analyze time complexity.
7. Demonstrate Floyd-Warshall Algorithm with intermediate node storage. Display final distance matrix and reconstructed paths.
8. Apply a basic graph partitioning strategy using BFS clustering in C++. Divide the graph into subgraphs and count nodes in each.
9. Implement a heuristic-based Graph Partitioning Algorithm in C++ and evaluate inter-partition edge cuts. Visualize the result if possible.

Note: ***The questions provided are illustrative samples. Students are encouraged to practice additional exercises aligned with the prescribed syllabus topics for comprehensive understanding.***

Theory: Data Communication and Networking
Credits - 03, Contact hours - 45.

Overview of Data Communication and Networking **Introduction:** Data communications Components, data representation, direction of data flow (simplex, half duplex, full duplex). **Network Hardware:** Physical structure (type of connection, topology), categories of network (LAN, MAN, WAN). **Internet:** Brief history, Protocols and standards, **Reference models:** OSI reference model, properties of all the layers, TCP/IP reference model, their comparative study.	03 hours
Physical Layer **Data & Signals:** Analog & Digital Data and Signals, periodic and non-periodic signals, composite signals, bandwidth, bit rate, transmission of digital signals. **Transmission Impairments:** Attenuation, Distortion and Noise. **Data Rate Limits:** Noiseless Channel: Nyquist Data rate, Noisy Channel: Shannon's Capacity, calculation of data rate using both limits. **Digital Transmission** **Digital to Digital Conversion:** Line coding, schemes (RZ, NRZ, Manchester, Differential Manchester), block coding. **Analog to Digital Conversion:** Sampling, Nyquist rate of sampling, Pulse code modulation (PCM), Delta Modulation (DM), Adaptive Delta Modulation (ADM), parallel and serial transmission. **Analog Transmission** **Digital to Analog conversion:** Amplitude shift keying (ASK), Frequency Shift Keying (FSK), Phase Shift Keying (PSK), Quadrature Amplitude Modulation (QAM). **Analog to Analog Conversion (qualitative):** Amplitude Modulation (AM), Frequency Modulation (FM), Phase Modulation.	11 hours
Bandwidth Utilization Techniques Multiplexing: FDM, Synchronous & Statistical TDM, WDM.	03 hours
Transmission Media **Guided media:** Twisted pair, Coaxial, Fiber optics. **Unguided:** Radio waves, microwaves, Infrared, Antenna, Communication satellites (qualitative study only).	04 hours
Switching and Telephone network Circuit switched networks, Packet Switched networks, Virtual Circuit switch. Major components of telephone network, Dial up modem, DSL and ADSL modems, Cable TV for data transfer (qualitative study only)	03 hours
Data link Layer Types of errors, framing (character and bit stuffing), error detection & correction methods, Linear and cyclic codes, checksum. **Protocols:** Stop & wait ARQ, Go-Back- N ARQ, Selective repeat ARQ, HDLC (qualitative study only). **Physical addressing:** MAC address and its format	03 hours
Medium Access sub layer Point to Point Protocol, **Token Ring:** Reservation, Polling. **Multiple access protocols:** Pure & Slotted ALOHA, CSMA, CSMA/CD, CSMA/CA. **Channelization:** FDMA, TDMA, CDMA (Qualitative study only). **Wired and Wireless LAN:** Standards, fast Ethernet, Protocol 802.11, Bluetooth.	05 hours

Network layer **Internetworking & devices:** Repeaters, Hubs, Bridges, Switches, Router, Gateway, **Addressing:** IP addressing, Subnetting, Routing techniques: static vs. dynamic routing, **Protocols:** RARP, ARP, IP, ICMP.	07 hours
Transport layer **Process to Process delivery:** UDP, TCP	02 hours
Application Layer Introduction to DNS, Remote logging, FTP, Electronic mail, WWW & HTTP.	04 hours

Text/ Reference Books

1. Data Communication and Networking, B.A. Forouzan, Tata McGraw Hill.
2. Computer Networks, A.S. Tanenbaum, Pearson Education.
3. Data and Computer Communication, W. Stallings, Pearson Education.
4. Data & Computer Communication, Black, PHI.
5. Internet & World Wide Web: How to program, Harvey M. Deitel& Paul J. Deitel.

Practical: Networking Lab
Credit: 01, Contact hour: 30.

1. Simulate Simplex, Half Duplex, and Full Duplex Communication Use serial and Ethernet ports to show directional data flow between two nodes.
2. Create LAN, MAN, and WAN Topologies Use appropriate devices (switches, routers, cloud) to build various category-wise networks.
3. Design Physical Topologies Implement star, bus, ring, and mesh topologies using appropriate devices.
4. Model Protocol Stack using TCP/IP Reference Model Layers Simulate communication between clients and servers, annotating each layer's activity.
5. Compare OSI vs TCP/IP Reference Models Create two scenarios showing layered encapsulation in both stacks and interpret packet flows.
6. Visualize Bandwidth and Bit Rate Concepts Use different cable types and configure link speeds to demonstrate throughput variations.
7. Simulate Signal Impairments Using Distorted Packets Introduce corruption manually and trace signal behavior across a noisy channel.
8. Compare Guided Media: Coaxial vs Fiber vs Twisted Pair Deploy multiple link types and measure latency and throughput across them.
9. Implement Multiplexing Models (FDM & TDM) Use time-sequenced communication across nodes to simulate multiplexed channel behavior.
10. Simulate Packet Switching Using Routers Design a scenario where data packets are rerouted based on availability.
11. Create a Virtual Circuit Setup Between Devices Use Label Switching or logical pathways to model a fixed route data exchange.
12. Demonstrate MAC Addressing and Frame Inspection Use simulation mode to trace MAC-level transmission and address resolution.
13. Simulate Error Detection Using Checksums and CRC Introduce errors and verify how they're flagged and corrected.
14. Frame Construction Using Bit and Character Stuffing Manual framing via scripts or event annotation showcasing stuffing techniques.
15. Compare CSMA/CD vs CSMA/CA Protocols Use wired Ethernet vs wireless 802.11 environments to simulate contention handling.

16. Create WLAN Using 802.11 Standard Devices Configure a wireless network with access points, laptops, and mobile nodes.
17. Simulate Bluetooth and PAN Communication Establish small-scale device-to-device networks to reflect low-energy protocol behavior.
18. Configure Static and Dynamic Routing Using RIP/EIGRP/OSPF Route between multiple routers with table inspection and simulation.
19. Implement IP Addressing and Subnetting Assign subnets to devices and verify connectivity across routed domains.
20. Simulate TCP and UDP Communication Use PCs and servers to demonstrate reliable vs unreliable transport with packet inspection.

Simulator: CISCO Packet Tracer

Note: ***The questions provided are illustrative samples. Students are encouraged to practice additional exercises aligned with the prescribed syllabus topics for comprehensive understanding.***

Theory: Theory of Computation
Credits - 03, Contact hours - 45.

Finite Automata Definition of a Finite Automaton, Model, Representation, Classification – with respect to output function Mealy and Moore Machines, with respect to State Transition – Deterministic and Non-Deterministic Machine, Examples, conversion algorithms Mealy to Moore and Moore to Mealy, Finite and Infinite state machines, Finite Automaton, Deterministic and Non-Deterministic Finite automaton, non-deterministic to equivalent Deterministic Automaton-Optimized and Non-optimized technique ideas and algorithms, Acceptability of String by a Finite Automaton.	12 hours
Formal Languages and Grammar Introduction to Formal Grammar and Language, Chomsky's Classification of Grammar – Type-0, Type-1 or Context Sensitive, Type-2 or Context Free and Type-3 or Regular Grammar, Illustration of each of these classes with example, Sentential form, Sentences – Languages or strings, Derivations, Ambiguous Grammar and Language, Designing of Grammar for a language, Find the Language for given Grammar, Definition and basic idea about Push Down Automaton.	11 hours
Regular Expression Basic Idea and Definition, Regular Expression basic Identities, Arden's Theorem – Statement (without Proof) and application for reduction of equivalent regular expressions, Regular expression to Finite Automata conversion, State Transition System to Regular Expression conversion algorithm by Arden's Algebraic Method, FA to Regular Grammar and Regular Grammar to FA conversion algorithms and applications.	12 hours
Turing Machine Concepts of Turing Machine, Formal Definitions, Classifications – Deterministic and Non-Deterministic Turing Machines, Simple Design of Turing Machines: Odd / even count and concepts of Universal Turing Machines, Difference and Similarities between Turing Machine and a General-Purpose Computer, Definition and significant of Halting Problem in Turing Machine.	10 hours

Text/ Reference Books

1. Introduction to Automata Theory, Languages, and Computation by John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman, 3rd Edition, Pearson.
2. Theory of Computer Science (Automata, Languages & Computation) by K L P Misra & N Chandrasekharan, 3rd Edition, PHI.
3. Introduction to Theory of Computation by MichealSipser, 3rd Edition, Cengage Learning.
4. Switching and Finite Automata Theory by ZviKohavi, Niraj.K.Jha, 3rd Edition, TMH.
5. Formal Language and Automata, P. Linz, Narosa.

Practical: Tutorial
Credits - 01, Contact hours - 30.

Theory: Database Management System (DBMS)
Credits - 03, Contact hours - 45.

Introduction Drawbacks of Legacy System; Advantages of DBMS; Layered Architecture of Database, Data Independence; Data Models; Schemas and Instances; Database Languages; Database Users, DBA; Data Dictionary.	03 hours
Entity Relationship (ER) Modelling Entity, Attributes and Relationship, Structural Constraints, Keys, ER Diagram of Some Example Database, Weak and strong Entity Set, Specialization and Generalization, Constraints of Specialization and Generalization, Aggregation.	04 hours
Relational Model Basic Concepts of Relational Model; Relational Algebra; Tuple Relational Calculus; Domain Relational Calculus.	08 hours
Integrity Constraints Domain Constraints, Referential Integrity, View.	04 hours
Relational Database Design Problems of Un-Normalized Database; Functional Dependencies (FD), Derivation Rules, Closure of FD Set, Canonical Cover; Normalization: Decomposition to 1NF, 2NF, 3NF or BCNF Using FD; Lossless Join Decomposition Algorithm; Dependency preservation.	10 hours
SQL Basic Structure, Data Definition, Constraints and Schema Changes; Basic SQL Queries (Selection, Insertion, Deletion, Update); Order by Clause; Complex Queries, Aggregate Function and Group by Clause; Nested Sub Queries; Views, Joined Relations; Set Comparisons (All, Some); Derived Relations.	11 hours
Record Storage and File Organization (Concepts only) Fixed Length and Variable Length Records; Spanned and Un-Spanned Organization of Records; Primary File Organizations and Access Structures Concepts; Unordered, Sequential, Hashed; Concepts of Primary and Secondary Index; Dense and Sparse Index; Index Sequential Files; Multilevel Indices.	05 hours

Text/ Reference Books

1. Fundamentals of Database Systems 6th Edition, R. Elmasri, S.B. Navathe, Pearson Education.
2. Database Management Systems, R. Ramakrishanan, J. Gehrke,3rd Edition, McGraw-Hill.
3. Database System Concepts 6th Edition, A. Silberschatz, H.F. Korth, S. Sudarshan, McGraw Hill.
4. Database Systems Models, Languages, Design and application Programming, R. Elmasri, S.B. Navathe, Pearson Education.
5. SQL and Relational Theory: How to Write Accurate SQL Code, Christopher J. Date, O'Reilly Media.
6. Database Systems: A Practical Approach to Design, Implementation and Management, Thomas M. Connolly and Carolyn E. Begg, Pearson.

Practical: RDBMS Lab
Credits - 01, Contact hours - 30.

1. Design a student-course enrollment schema using Excel (legacy format), then recreate the same in MySQL using normalized tables to highlight redundancy reduction and relational integrity.
2. Create a login system using MySQL that demonstrates layered architecture. Separate user credentials (user layer), schema definitions (schema layer), and storage engine settings (storage layer).
3. Alter a table's structure (e.g., add/remove columns) and demonstrate how MySQL preserves application-level logic, illustrating data independence.
4. Draw an ER diagram for a college admission system. Then, create MySQL tables for Student, Course, and Department reflecting primary/foreign key relationships.
5. Model weak and strong entities by implementing Dependent and Employee tables in MySQL. Use partial key constraints and foreign key links.
6. Demonstrate specialization/generalization by creating an entity Vehicle, and specialized entities Car and Truck in MySQL using table inheritance logic.
7. Convert your ER diagram from Exercise 4 into relational schemas and implement the tables in MySQL.
8. Use MySQL to perform relational algebra operations:
   Selection: Retrieve students from a particular department
   Projection: List unique course names
   Union: Combine student lists from two departments
   Join: Get student names with course titles they enrolled in*
9. Write queries using tuple and domain relational calculus concepts translated into SQL. Example: "Find all students who have enrolled in Math but not Physics."
10. Create a student-course database in MySQL. Apply domain constraints (e.g., NOT NULL, CHECK), and referential constraints using FOREIGN KEYs.
11. Define a MySQL view that shows the average marks of students per department. Write SQL to query from that view.
12. Design a student performance table with repeated subject columns. Show data redundancy and anomalies during updates.
13. Normalize the above table into 1NF, 2NF, 3NF, and BCNF. Create each normal form version using MySQL CREATE TABLE statements.
14. Demonstrate lossless join and dependency preservation using SQL joins. Verify consistency of decomposed relations.
15. *Create a Sales database in MySQL. Write SQL queries to:
    SELECT all products
    INSERT a new order
    UPDATE product price
    DELETE an obsolete order*

16. Write complex queries using GROUP BY and aggregate functions to calculate total sales per region and highest priced product per category.
17. Use nested subqueries and SET comparison operators (ALL, ANY, SOME) to find customers with purchases exceeding others.
18. Join customer and order tables. Create a view for total spending per customer. Query the view to find top spenders.
19. Write derived table queries (subqueries inside FROM) to filter high-value orders exceeding ₹20,000.
20. Apply schema constraints using PRIMARY KEY, FOREIGN KEY, and CHECK. Perform ALTER TABLE operations to modify schema.

RDBMS software tool: MYSQL

Note: ***The questions provided are illustrative samples. Students are encouraged to practice additional exercises aligned with the prescribed syllabus topics for comprehensive understanding.***

Semester - 6

Theory Paper	Credit	Contact hours	Practical/Tutorial Paper	Credit	Contact hours
Software Engineering	03	45	Tutorial	1	30
Programming in Python	03	45	Programming in Python	1	30
Linear Algebra & Statistical Methods	03	45	Linear Algebra & Statistical Methods using Python	1	30

Theory: Software Engineering
Credits - 03, Contact hours - 45.

Introduction Defining system, open and closed system, modelling of system through computer hardware, communication systems, external agents and software systems; Importance of Engineering Methodology towards computerization of a system.	3 hours
Software Life Cycle Classical and Iterative Waterfall Model; Spiral Model; Prototype Model; Evolutionary model and its importance towards application for different system representations, Comparative Studies.	6 hours
Software Requirement and Specification Analysis Requirements Principles and its analysis principles; Specification Principles and its Representations Software Design Analysis – Different level of DFD Design, Physical and Logical DFD, Use and Conversions between them, Decision Tables and Trees, Structured analysis, Coupling and Cohesion of different modules Software Cost Estimation Modelling –COCOMO.	15 hours
Software Testing Software Verification and Validation; Testing objectives, Testing Principles, Testability; Error and Faults; Unit Testing, White Box and Blank Box Testing, Test Case Design: Test Vector, Test Stub.	15 hours

Software Quality Assurances Concepts of Quality, Quality Control, Quality Assurance, IEEE Standard for Statistical Software Quality Assurances (SSQA) criterions.	06 hours

Text/References Books

1. Software Engineering: A Practitioner's Approach by R.S. Pressman, McGraw-Hill.
2. An Integrated Approach to Software Engineering by P. Jalote, Narosa Publishing House.
3. Software Engineering by K.K. Aggarwal and Y. Singh, New Age International Publishers.
4. Software Engineering by I. Sommerville, Addison Wesle.
5. Software Engineering for Students by D. Bell, Addison-Wesley.
6. Fundamentals of Software Engineering by R. Mall, PHI.

Practical: Tutorial
Credits - 01, Contact hours - 30.

Theory: Programming In Python
Credits - 03, Contact hours - 45.

Introduction Introducing Python, features of Python, the paradigms, chronology and uses, setting up Python on Windows and other Operating systems, introducing IDLE, Installation of Anaconda and miniconda.	02 hours
Parts of Python Programming Language Identifiers, Keywords, Statements and Expressions, Variables, Operators, Precedence and Associativity, Data Types, Indentation, Comments, Reading Inputs, print output, Type Conversions, The type() function and Is operator, Dynamic and strongly typed language.	04 hours
Control Flow Statements If, if.....else, if.....elif...else decision control statement, nested if statement, While loop, Continue and break, catching exceptions using try and except.	04 hours
Functions Built in functions, commonly used modules, Function Definition and Calling the Function, return and void function, Scope and life time of variables, default parameters, keyword arguments, *args and **kwargs, command line arguments.	04 hours
String Creating and storing strings, basic string operation, Accessing Characters in string by index, String slicing and joining, string methods, formatting strings.	05 hours
Lists Creating lists, basic list operations, indexing and slicing in lists, built-in functions used on lists, list methods, del statements,	05 hours
Dictionaries Creating Dictionary, accessing and modifying key value, built-in functions used on dictionaries, dictionary methods.	03 hours

Tuples and Sets Creating tuples, basic tuple operations, Indexing and slicing in Tuples, built-in functions on tuples, relation between tuples and lists, relation between tuples and dictionaries, Tuple methods, Zip() function, Sets, Sets methods, frozen set.	04 hours
Files Types of files, creating and reading text data, file method to read and write data, reading and writing binary, the pickle module, reading and writing CSV files, python os and os.path modules,	03 hours
Regular expression Using special characters, regular expression methods, named groups in python regular, regular expression with glob module.	03 hours
Object Oriented Programming Classes and objects, creating classes in python, creating objects in python, constructor method, classes with multiple objects, class versus data attributes, encapsulation, inheritance, polymorphism.	04 hours
Introduction to Data Science Functional programming, JSON and XML in Python, Numpy with Python, Pandas, altair	04 hours

Text/ Reference Books

1. Introduction to Computation and Programming Using Python: With Application to Understanding Data, Guttag, John V. MIT Press.
2. Learn Python 3 the Hard Way: A Very Simple Introduction to the Terrifyingly Beautiful World of Computers and Code, Shaw, Zed A, Addison-Wesley Professional.
3. Think Python 2e. Green Tea Books, Downey, Allen B.
4. Practical Programming: An Introduction to Computer Science Using Python 3.6. Pragmatic Bookshelf, Gries, Paul, Jennifer Campbell, and Jason Montojo.
5. Introduction to Python Programming, Gowrishankar S, Veena A, Taylor and Francis, CRC Press.
6. Python Cookbook, David Beazley and Brian K. Jones, O'Reilly.

Practical: Programming in Python
Credits - 01, Contact hours - 30.

1. Develop a Python program that reads marks from three different subjects and calculates their average.
2. Create a Python script to convert weight from kilograms to pounds.
3. Write a program that takes the lengths of the three sides of a prism as input (integers) and computes its surface area using the formula: 2ab + 2bc + 2ca.
4. Design a Python program to calculate the speed of a plane, given a distance of 395,000 meters and a travel time of 9,000 seconds.
5. Build a program that determines how long it will take to empty a swimming pool of dimensions 12m × 7m × 2m using a pump with a flow rate of 17 cubic meters per hour.
6. Write a Python script to convert temperature from Celsius to Fahrenheit (input as a float).
7. Create a program to compute the total number of seconds in one full day.
8. Develop a Python program to calculate the acceleration of a car that starts from rest and reaches a velocity of 10 m/s in 20 seconds, using the formula: $(v_{final} - v_{initial})$/time.

9. Write a program to display Pascal's triangle.
10. Write a program to display the following pattern using nested loops.
    1 1
    22 21
    333 321
    4444 4321
    55555 54321
11. Write a program that uses a while loop to add up all the even numbers between 100 and 200.
12. Write a program to print the sum of the following series
    a. **1 + ½ + 1/3 +. …. + 1/n**
    b**. 1/1 + 2^2/2 + 3^3/3 + ……. + n^n/n**
13. Write a program to find the depreciation value of an asset (property) by reading the purchase value of the asset (amt), year of the service (year) and the value of depreciation.
14. Write a program that accepts a string from the user and display the same string after removing vowels from it.
15. Write a function to insert a string in the middle of the string.
16. Write a program to sort a string lexicographically.
17. Write a program to replace a string with another string without using built-in methods.
18. Write a program to concatenate two strings into another string without using the + operator.
19. Write a program to strip a set of characters from a string.
20. Write a program to extract the first n characters of a string.
21. Write a program that creates a list of 10 random integers. Then create two lists by name **odd_list** and **even_list** that have all odd and even values of the list respectively.
22. Write a program to sort the elements in ascending order using insertion sort.
23. Write a Python program to use binary search to find the key element in the list.
24. Make a list of the first eight letters of the alphabet, then using the slice operation do the following operations:
    a. Print the first three letters of the alphabet.
    b. Print any three letters from the middle.
    c. Print the letters from any particular index to the end of the list.
25. Write a program to sort the elements in ascending order using selection sort.
26. Write a program that prints the maximum value of the second half of the list.
27. Write a program that creates a list of numbers 1–100 that are either divisible by 5 or 6.
28. Write a function that prompts the user to enter five numbers, then invoke a function to find the GCD of these numbers.
29. Write a function named addfruit, which is passed with a set of fruit names and their prices and returns a dictionary containing the entered information and raises a **ValueError** exception if the fruit is already present.
30. Write a function to add the air quality index as the value and the date as the key; create the dictionary for the entered information.
31. Create a dictionary that contains usernames as the key and passwords as the associated values. Make up the data for five dictionary entries and demonstrate the use of clear and **fromkeys()** methods.
32. Write Pythonic code to create a dictionary that accepts a country name as a key and its capital city as the value. Display the details in sorted order.
33. Write a program that has the dictionary of your friends' names as keys and phone numbers as its values. Print the dictionary in a sorted order. Prompt the user to enter the name and check if it is present in the dictionary. If the name is not present, then enter the details in the dictionary.
34. Write a program to create a dictionary containing the author name as the keys and ISBN number as the value. Make up the data for five dictionary entries and demonstrate the use of **clear()** and **fromkeys()** methods.
35. Create a list containing three elements, and then create a tuple from that list.

36. Write a program to unpack a tuple to several variables.
37. Write a program to check whether an item exists within a tuple.
38. Write a program to unzip a list of tuples into individual lists.
39. Write a program to create an intersection, union, set difference, and symmetric
40. difference of sets.
41. Write a program to demonstrate the use of **issubset()** and **issuperset()** methods.
42. Write a program that takes a range and creates a list of tuples within that range with the first element as the number and the second element as the square of the number.
43. Write a program to clear a set.
44. Write a program to find the length of the set.
45. Write a program to store the latitude and longitude of your house as a tuple and display it.
46. Write a program that prompts the user to enter a text file, reads words from the file, and displays all the non-duplicate words in ascending order.
47. Write a program to get the file size of a plain text file.
48. Write a program that prompts the user to enter a text filename and displays the number of vowels and consonants in the file.
49. Write a program to read the first n lines of a file. Prompt the user to enter the value for n.
50. Write a program that reads the contents of the file and counts the occurrences of each letter. Prompt the user to enter the filename.
51. Write a program to read the last n lines of a file. Prompt the user to enter the value for n.
52. Write a program to combine each line from the first file with the corresponding line in the second file.
53. Write a program to remove newline characters from a file.
54. Write a program to read the random line from a file.
55. Write a program to read and write the contents from one csv file to another.
56. Write a Python program that matches a word containing 'z'.
57. Write a Python program to remove all leading zeros' from an IP address
58. Write a Python program to search the numbers (0–9) of length between 1 to 3 in a given string.
59. Write a Python program to find the substrings within a string
60. Write a Python program to extract year, month and date from an url.
61. Write a Python program to read a file and to convert a date of yyyy-mm-dd format to dd-mm-yyyy format.
62. Write a Python program to abbreviate 'Street' as 'St.' in a given string.
63. Write a Python program to find all five characters long word in a string.
64. Create a class named quadratic, where a, b, c are data attributes and the methods are
    a. __init__() to initialize the data attributes
    b. roots() to compute the quadratic equation
65. Define a class called student. Display the marks details of top five students using inheritance.
66. Create a class called library with data attributes like acc_number, publisher, title and author. The methods of the class should include
    a. read() – acc_number, title, author.
    b. compute() - to accept the number of days late, calculate and display the fine charged at the rate of $1.50 per day.
    c. display the data.
67. Create two base classes named clock and calendar. Based on these two classes define a class calendarclock, which inherits from both the classes which displays month details, date and time.
68. Write a program to add two polynomials using classes.
69. Define JSON. Construct a simple JSON document and write Pythonic code to parse JSON document.
70. Elaborate on the differences between XML and JSON.
71. Define XML. Construct a simple XML document and write Python code to loop through XML nodes in the document.

72. Explain NumPy array creation functions with examples.
73. Explain NumPy integer indexing, array indexing, Boolean array indexing and slicing with examples.
74. Write Python program to create and display a one-dimensional array-like object containing an array of data using pandas library.
75. Write Python program to add, subtract, multiply and divide two Pandas Series.
76. Write Python program to create and display a Data Frame from a dictionary data which has the index labels.

Note: ***The questions provided are illustrative samples. Students are encouraged to practice additional exercises aligned with the prescribed syllabus topics for comprehensive understanding.***

Theory: Linear Algebra & Statistical Methods
Credits - 03, Contact hours - 45.

Linear Algebra	
Vectors Introduction, Vector in Rn, vector addition and scalar multiplication, Dot (Inner) product, orthogonality, Cauchy-Schwarz inequality, spatial vectors(R3). Definition of matrices, elementary operations on matrices (addition, scalar multiplication, matrix multiplication, transpose), different types of matrices, determinants and their properties, matrix inverse. Definition of linear transformations (Linear mappings), properties, kernel and image, matrix representation of linear transformations. Representation of systems of linear equations in matrix form, consistent and inconsistent systems, Gaussian elimination, Gauss-Jordan method, echelon matrices, row canonical form, row equivalence. Definition of vector spaces, examples, spanning sets, linear span, linear dependence and independence, basis and dimension, rank, change of basis	**15 hours**
Eigenvalues and Eigenvectors Definition, characteristic equation, computing eigenvalues and eigenvectors, diagonalizing matrices. Applications in computer graphics and machine learning: Principal Component Analysis (PCA), Singular Value Decomposition (SVD) for dimensionality reduction and data analysis.	**5 hours**
Statistical methods	
Descriptive Statistics & Probability basics Data representation and summarization: Types of data (quantitative, qualitative), measurement scales (nominal, ordinal, interval, ratio), frequency distributions, graphical representation (histograms, frequency polygons), stem and leaf plots. Measures of central tendency: mean, median, mode. Measures of dispersion: range, variance, standard deviation. Moments, skewness, and kurtosis: Characteristics of data distribution. Moment-generating functions of random variables.	**5 hours**

Probability distributions 10 hours Common distributions: Binomial, Poisson, Normal, Uniform, Exponential. Sampling distributions of mean and variance.	**10 hours**
Statistical inference Point and interval estimation, estimating population parameters from sample data. Hypothesis testing, formulating hypotheses, Type I and Type II errors, significance levels, various tests (t-tests, chi-square tests, ANOVA.	**5 hours**
Regression and Correlation Definition of correlation, Karl Pearson coefficient of correlation, Spearman's rank correlation coefficient. Linear regression, least squares method, simple linear regression, fitting of polynomials and exponential curves.	**5 hours**

References:

1. Introduction to Linear Algebra, Gilbert Strang, 6th ed.
2. Linear Algebra and Its Applications, 5th Edition - Pearson by David C. Lay.
3. Linear Algebra: Step By Step, by Kuldeep Singh, Oxford University Press.
4. Linear Algebra, G. Hadley
5. Probability and Statistical Inference, by Robert V. Hogg, Elliot Tanis, Dale Zimmerman, 10th ed.
6. Fundamental Of Mathematical Statistics, S C Gupta & V K Kapoor, Sultan Chand and sons.
7. Introduction to Probability and Statistics for Engineers and Scientists, by Sheldon M. Ross, Academic Pr; 5th edition.
8. Statistical Methods, N G Das, Tata McGraw Hill.
9. Statistical Methods An Introductory Text, by Jyotiprasad Medhi, New Age International (P) Limited, Wiley.

Practical: Linear Algebra and Statistical methods Lab.
Credits - 01, Contact hours - 30.

1. Implementing machine learning algorithms, understanding their underlying mechanisms by using linear algebra concepts like vectors, matrices, eigenvalues, and eigenvectors

2. Solving systems of linear equations: Gaussian elimination, Gauss-Jordan elimination, and LU decomposition

3. Statistical methods like analysing datasets by utilizing descriptive statistics by mean, median, standard deviation

4. Data visualization techniques (histograms, boxplots) to analyse and interpret datasets containing information like heights and weights of individuals, identify outliers, and assess normality

5. Designing experiments and analysing results by applying principles of experimental design, like one-way and two-way ANOVA and for tasks such as testing the efficacy of different pain management methods

6. Predicting trends and classifying data by building regression models to predict outcomes based on given data and employing classification techniques to categorize data into different classes

7. Understanding and evaluating machine learning algorithms like hypothesis testing and confidence intervals which are vital in evaluating the performance and reliability of machine learning models

(These are suggestive applications of the theory portion. The software mentioned above can be used in the applications).

Tools/Simulator/Programming Language – Python Programming language with **libraries like NumPy, Sage Math, or Scilab.**