



ANDHRA KESARI UNIVERSITY :: ONGOLE

Programme: BCA Honours Data Science (Major)
(w.e.f. Academic Year 2025-26)

COURSESTRUCTURE

Year	semester	Course Code	Title of the Course	No.of Hours	Credits	
I	I	1	Computer Fundamentals and Office Automation	3	3	
			Computer Fundamentals and Office Automation Lab	2	1	
		2	Problem Solving Using C	3	3	
			Problem Solving Using C Lab	2	1	
	II	3	Python Programming and Data Structures	3	3	
			Python programming and Data Structures lab	2	1	
		4	Statistical Foundations for Data Science	3	3	
			Statistical Foundations for Data Science lab	2	1	
	II	III	5	Database Management Systems	3	3
				Database Management Systems Lab	2	1
6			Data Science with R	3	3	
			Data Science With R lab	2	1	
7			Object oriented programming using Java	3	3	
			Object oriented programming using Java Lab	2	1	
IV		8	Data Mining & Data warehousing	3	3	
			Data Mining & Data warehousing Lab	2	1	
		9	Exploratory Data Analysis and Visualization	3	3	
			Exploratory Data Analysis and Visualization lab	2	1	
		10	Software Engineering	3	3	
Software Engineering Lab			2	1		
V		11	Business Intelligence Tools	3	3	
			Business Intelligence Tools Lab	2	1	

SEA: Skill Elective Track A (Application Development)						
III	V-Track A	12	SEA1: Web Design & Development Fundamentals	3	3	
			SEAP1: Web Design & Development Fundamentals Lab	2	1	
		13	SEA2: Web Application Development using PHP & MySQL	3	3	
			SEAP2: Web Application Development using 1 PHP & MySQL Lab	2	1	
	V-Track A	14	SEA3: Mobile Application Development	3	3	
			SEAP3: Mobile Application Development Lab	2	1	
		15	SEA4: MERN Stack	3	3	
			SEAP4: MERN Stack Lab	2	1	
	SEB: Skill Elective Track B (AIML & Data Engineering)					
	V-Track B	12	SEB1: Artificial Intelligence	3	3	
			SEBP1: Artificial Intelligence Lab	2	1	
		13	SEB2: Big Data Technologies	3	3	
SEBP2: Big Data Technologies lab			2	1		
V-Track B	14	SEB3: Machine Learning	3	3		
		SEBP3: Machine Learning Lab	2	1		
	15	SEB4: Cloud computing for Data Science	3	3		
		SEBP4: Cloud computing for Data Science Lab	2	1		

Semester-I

C1: Computer fundamentals and Office Automation

Course Objectives

1. Understand foundational computing concepts, including number systems, the evolution of computers, block diagrams, and generational progress.
2. Develop knowledge of computer architecture, focusing on system organization and networking fundamentals.
3. Acquire practical skills in document creation, formatting, and digital presentations using word processing tools.
4. Gain proficiency in spreadsheet operations, such as data entry, formulas, functions, and charting techniques.
5. Introduce data visualization and basic modelling principles, fostering analytical thinking in structuring and interpreting data sets.

Course Outcomes

1. At the End of the Course, The Students will be able to explain different number systems, the historical evolution of computers, and identify key components in a block diagram.
2. Learners will demonstrate basic blocks of a computer and fundamental networking knowledge.

3. Learners will create professional-level documents and **design visually appealing presentations** using word processing software and presentation software.
4. Learners will manipulate data within spreadsheets, apply formulas, and **generate accurate summaries and visualizations.**
5. Learners will apply data modelling techniques to **analyze, organize, and represent data effectively** in various scenarios.

Unit 1. Number Systems, Evolution , Block Diagram and Generations:

Number Systems: Binary, Decimal, Octal, Hexadecimal; conversions between number systems.

Evolution of Computers: History from early mechanical devices to modern-day systems.

Block Diagram of a Computer: Components like Input Unit, Output Unit, Memory, CPU (ALU + CU).

Generations of Computers: First to Fifth Generation – technologies, characteristics, examples.

Unit 2. Basic organization and N/W fundamentals:

Computer Organization: Functional components – Input/Output devices, Storage types, Memory hierarchy.

Types of Computers: Micro, Mini, Mainframe, and Supercomputers.

Networking Fundamentals: Definition, need for networks, types (LAN, WAN, MAN), topology (Star, Ring, Bus).

Internet Basics: IP Address, Domain Name, Web Browser, Email, WWW.

Unit 3. Word Processing and presentations:

Word Processing Basics: Using MS Word/Google Docs – formatting, styles, tables, mail merge.

Presentation Tools: Using PowerPoint/Google Slides – slide design, animations, transitions.

Applications: Creating resumes, reports, brochures, and presentations.

Keyboard Shortcuts

Unit 4. Spreadsheet Basics:

Spreadsheet Concepts: Understanding rows, columns, cells in tools like MS Excel/Google Sheets, cell referencing.

Functions and Formulae: SUM, AVERAGE, IF, COUNT.

Charts and Graphs: Creating visual representations

Data Handling: Sorting, filtering, conditional formatting.

Text Functions: LEFT, RIGHT, MID, LEN, TRIM, CONCAT, TEXTJOIN

Advanced Functions: Logical: IF, AND, OR, IFERROR, **Lookup:** VLOOKUP, HLOOKUP, XLOOKUP, INDEX, MATCH

Unit 5. Data Analysis and Visualization:

Conditional Formatting: Custom rules, Color scales, Icon sets, Data bars

Data Analysis Tools: Pivot Tables and Pivot Charts, Data Validation (Drop-downs, Input Messages, Error Alerts), What-If Analysis: Goal Seek, Scenario Manager, Data Tables

Charts and Dashboards: Creating Interactive Dashboards, Using slicers with Pivot Tables, Combo Charts and Sparklines

Productivity Tips: Using Named Ranges, Freeze Panes, Split View

Textbooks:

1. **Fundamentals of Computers**, Reema Thareja, Oxford University Press, Second Edition
2. **Fundamentals of Computers**, V. Rajaraman – PHI Learning
3. **Introduction to Computers** by Peter Norton – McGraw Hill
4. **Microsoft Office 365 In Practice** by Randy Nordell – McGraw Hill Education

References:

1. **Excel 2021 Bible** by Michael Alexander, Richard Kusleika – Wiley
2. **Networking All-in-One For Dummies** by Doug Lowe – Wiley
3. **Microsoft Official Docs and Training:** <https://learn.microsoft.com>
4. **Google Workspace Learning Center:** <https://support.google.com/a/users/>

Activities:

Outcome: At the End of the Course, The Students will be able to **explain different number systems**, the historical evolution of computers, and identify key components in a block diagram.

Activity: Create a digital poster or infographic comparing number systems (binary, decimal, octal, hexadecimal) and illustrating the timeline of computer generations with key innovations.

Evaluation Method: Rubric-based assessment of the poster presentation on a 10-point scale focusing on:

- Accuracy of number system conversions
- Correct identification of block diagram components
- Visual organization and creativity

Outcome: Learners will demonstrate **basic blocks of a computer and fundamental networking knowledge**.

Activity: Design a concept map showing the internal architecture of a computer and types of networks (LAN, WAN, MAN), including devices and topologies.

Evaluation Method: Checklist-based peer review and instructor validation:

- Completeness of the map
- Correctness of networking concepts
- Use of appropriate terminology
- Logical flow and structure of the map

Outcome: Learners will create professional-level documents and **design visually appealing presentations** using word processing software and presentation software.

Activity: Prepare a formal report (e.g., project proposal) in a word processor and present it using a slide deck with transitions, embedded media, and design elements.

Evaluation Method: Performance-based evaluation using a 10-point scoring scale:

- Formatting and structure of the document
- Presentation aesthetics and clarity
- Communication skills during presentation

Outcome: Learners will manipulate data within spreadsheets, apply formulas, and **generate accurate summaries and visualizations**.

Activity: Analyze a dataset (e.g., student scores or sales data) using spreadsheet software. Apply formulas (SUM, AVERAGE, IF, VLOOKUP) and create relevant charts.

Evaluation Method: Practical test with a rubric:

- Correct use of formulas
- Accuracy of data summaries

Outcome: Learners will apply data modelling techniques to **analyze, organize, and represent data effectively** in various scenarios.

Activity: Prepare an interactive dashboard for a given data set using EXCEL.

Evaluation Method: Evaluation of the dashboard on a 10-point scoring scale:

- Presentation aesthetics and clarity
- Interactiveness
- Communication skills during presentation

C1P: Computer fundamentals and Office Automation Lab

List of Experiments:

1. Demonstration of Assembling and Dessembling of Computer Systems.
2. Identify and prepare notes on the type of Network topology of your institution.
3. Prepare your resume in Word.
4. Using Word, write a letter to your higher official seeking 10-days leave.
5. Prepare a presentation that contains text, audio and video.
6. Using a spreadsheet, prepare your class Time Table.
7. Using a Spreadsheet, calculate the Gross and Net salary of employees (Min 5) considering all the allowances.
8. Generate the class-wise and subject-wise results for a class of 20 students. Also generate the highest and lowest marks in each subject.
9. Using IF, AND, OR, and IFERROR to Automate Grade Evaluation.
 - a. Create a table of student scores in different subjects.
 - b. Use IF to assign grades (A/B/C/Fail).
 - c. Use IFERROR to handle missing scores or invalid data.



10. Employee Database Search Using VLOOKUP, HLOOKUP, XLOOKUP, INDEX, and MATCH

- a. Create a database of employees (Name, ID, Department, Salary).
- b. Implement VLOOKUP to search by employee ID.
- c. Use HLOOKUP to extract department heads by role.
- d. Apply XLOOKUP for more flexible searches.
- e. Use INDEX + MATCH as an alternative to VLOOKUP.

11. Sales Report Analysis Using Pivot Tables and Charts

- a. Use a dataset of product sales (Product, Region, Date, Quantity, Revenue).
- b. Create Pivot Tables to summarize data by region/product.
- c. Insert Pivot Charts for visual analysis (e.g., bar, line).
- d. Add slicers to make the dashboard interactive.

12. Designing a Data Entry Form with Drop-downs and Input Rules


- a. Create a student registration form.
- b. Add drop-down lists for course selection using Data Validation.
- c. Add input messages to guide users.
- d. Add error alerts for wrong entries.

13. Monthly Budget Planning using Goal Seek and Scenario Manager

- a. Create a simple personal budget (income, expenses, savings).
- b. Use Goal Seek to determine income needed to save a desired amount.
- c. Use Scenario Manager to compare different budgeting scenarios (best/ worst/ realistic case).
- d. Create a one-variable Data Table to analyze how different expenses affect savings.

14. Dashboard Creation Using Combo Charts, Sparklines & Slicers

- a. Use existing sales or attendance data.
- b. Insert combo charts (e.g., column + line).
- c. Add sparklines to show trends.
- d. Use slicers with Pivot Tables to control dashboard elements.
- e. Finalize and format for interactivity.



C2: Problem Solving using C

Course Objectives:

1. Understand the fundamentals of computer programming, Apply structured problem-solving approaches using algorithms, flowcharts, and C programming constructs.
2. Develop efficient logic using decision-making, loop, and jump control statements.
3. Utilize derived data types like arrays and strings for modular program design.
4. Design and implement modular solutions using functions, recursive logic, pointer operations, and dynamic memory management.
5. Handle complex data structures including structures, unions, and text file operations.

Course Outcomes:

At the end of the course, students will be able to:

1. Understand basic computing concepts, programming paradigms and write structured C programs.
2. Apply control flow statements to solve logical and repetitive tasks in C.
3. Implement arrays and string operations to manage and manipulate data efficiently.
4. Design modular code using functions, recursion, and appropriate parameter passing.
5. Utilize pointers and memory operations for effective data handling. Demonstrate competence in dynamic memory allocation and text file processing.

Unit 1. Introduction to computer programming:

Introduction, Types of software, Compiler and interpreter, Concepts of Machine level, Assembly level and high-level programming, Flowcharts and Algorithms, Fundamentals of C: History of C, Features of C, C Tokens-variables and keywords and identifiers, constants and Data types, Rules for constructing variable names, Operators, Structure of C program, Input /output statements in C-Formatted and Unformatted I/O



Unit 2. Control statements:

Decision making statements: if, if else, else if ladder, switch statements. Loop control statements: while loop, for loop and do-while loop. Jump Control statements: break, continue and goto.

Unit 3. Derived data types in C:

Arrays: One Dimensional arrays - Declaration, Initialization and Memory representation; Two Dimensional arrays - Declaration, Initialization and Memory representation. Strings: Declaring & Initializing string variables; String handling functions, Character handling functions

Unit 4. Functions:

Pointers: Pointer data type, Pointer declaration, initialization, accessing values using pointers. Pointer arithmetic, Pointers and arrays.

Function Prototype, definition and calling. Return statement. Nesting of functions. Categories of functions. Recursion (Basic Concept only). Parameter Passing by address & by value. Local and Global variables. Storage classes: automatic, external, static and register.

Unit 5. Dynamic Memory Management:

Introduction, Functions-malloc, calloc, realloc, free Structures: Basics of structure, structure members, accessing structure members, nested structures, array of structures, structure and functions, structures and pointers. Unions - Union definition; difference between Structures and Unions. Working with text files - modes: opening, reading, writing and closing text files.

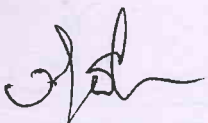
Text Books:

1. Programming in ANSI C, E. Balagurusamy, Tata McGraw Hill, 6 th Edn,
2. Computer fundamentals and programming in C, Reema Theraja, Oxford University Press

Reference Books:

1. Let us C, Y Kanetkar, BPB publications
2. Head First C: A Brain-Friendly Guide, David Griffiths, Dawn Griffiths

Activities:



Outcome: Understand basic computing concepts, programming paradigms and write structured C programs.

Activity: Create a concept map of computing fundamentals and programming paradigms (procedural, structured, object-oriented). Then, they write a structured C program (e.g., a calculator or student grade system) using proper syntax, indentation, and modular design.

Evaluation Method: Rubric-based Code Review & Viva to check the

- The correctness of the concept map
- Correct use of structure (main + functions)
- Identification of paradigm used
- Code readability and documentation

Outcome: Apply control flow statements to solve logical and repetitive tasks in C.

Activity: Implement a program that solves a logic puzzle (e.g., number guessing game, pattern generation, or prime number finder) using if, switch, for, while, and do-while.

Evaluation Method: Automated Test Cases + Peer Review to check the

- Correct use of control statements
- Logical correctness of output
- Efficiency and edge case handling
- Peer feedback on clarity and logic

Outcome: Implement arrays and string operations to manage and manipulate data efficiently.

Activity: Build a program that stores and arranges student marks in ascending and descending order using arrays and performs string operations like concatenation, comparing, and formatting names.

Evaluation Method: Functional Demonstration + Code Walkthrough to check the

- Correct array and string usage
- Memory efficiency
- Handling of invalid inputs
- Explanation of sorting/searching logic

Activity:

- **Recursive Problem Solver**

Students write a modular program to solve a recursive problem (e.g., factorial, Fibonacci, or Tower of Hanoi) using functions with parameters and return values.



Evaluation Method:

- Code Trace + Written Quiz
 - Correct function decomposition
 - Proper parameter passing (by value/reference)
 - Recursion depth and base case handling
 - Quiz on tracing recursive calls

Outcome: Utilize pointers and memory operations for effective data handling. Demonstrate competence in dynamic memory allocation and text file processing.

Activity: Create a program that dynamically stores user input (e.g., survey responses) using pointers and writes/reads the data to/from a text file.

Evaluation Method: Memory Debugging + File I/O Assessment to check the

- Proper use of malloc, calloc, realloc, and free
- Pointer arithmetic and dereferencing
- File creation, reading, writing, and error handling
- Use of tools like Valgrind or manual memory trace (Optional for Unix flavours)

C2P: Problem Solving using C Lab

List of Experiments:

1. Write a program to check whether the given number is Armstrong or not.
2. Write a program to find the sum of individual digits of a positive integer.
3. Write a program to generate the first n terms of the Fibonacci sequence.
4. Write a program to find both the largest and smallest number in a list of integer values
5. Write a program to demonstrate change in parameter values while swapping two integer variables using Call by Value & Call by Address
6. Write a program to perform various string operations.
7. Write a program to search an element in a given list of values.
8. Write a program that uses functions to add two matrices.
9. Write a program to calculate factorial of given integer value using recursive functions
10. Write a program for multiplication of two N X N matrices.
11. Write a program to sort a given list of integers in ascending order.



12. Write a program to calculate the salaries of all employees using the Employee (ID, Name, Designation, Basic Pay, DA, HRA, Gross Salary, Deduction, Net Salary) structure.
- DA is 30 % of Basic Pay
 - HRA is 15% of Basic Pay
 - Deduction is 10% of (Basic Pay + DA)
 - Gross Salary = Basic Pay + DA + HRA
 - Net Salary = Gross Salary - Deduction
13. Write a program to read / write the data from / to a file.
14. Write a program to reverse the contents of a file and store in another file.
15. Write a program to create Book (ISBN, Title, Author, Price, Pages, Publisher) structure and store book details in a file and perform the following operations
- Add book details
 - Search a book details for a given ISBN and display book details, if available
 - Update a book details using ISBN
 - Delete book details for a given ISBN and display list of remaining Books



Semester-II

C3: Python Programming and Data Structures

Course Objectives

- To introduce the fundamentals of Python programming, including environment setup, syntax, and core concepts.
- To develop problem-solving skills using control flow, functions, and modules.
- To provide knowledge of Python data structures, file handling, and exception handling for effective programming.
- To impart object-oriented programming concepts and GUI development skills for building applications.

Course Outcomes (COs)

After successful completion of the course, students will be able to:

1. Explain the basic features, syntax, data types, and operators of Python programming.
2. Apply control flow constructs, functions, and modules to develop structured Python programs.
3. Demonstrate the use of sequences, sets, and dictionaries for effective data handling and manipulation.
4. Implement file handling techniques and apply exception handling mechanisms for robust applications.
5. Develop object-oriented and GUI-based applications using Python.

Unit 1. Basics of Python Programming:

Introduction to Python, Features of Python, Programming Modes - Interactive Mode & Script Mode, Identifiers, Naming Conventions, Keywords (Reserved Words), Built-in Data Types, Literals - Integer, Float, Complex, Boolean, String, Variables, Operators, Expressions, Assignment Statements, Input/Output Statements, Python Syntax (Lines, Comments, Indentation)

Operators & Operands, Classification of Operators - Arithmetic Operators, Relational Operators, Logical Operators, Bitwise Operators, Assignment, Augmented Assignment, Identity Operators, Expressions & Precedence Rules

Unit 2. Control Flow, Functions & Modules:

Control Flow - if Statement, if-else, if-elif-else. Iterative Statements – while, for, Nested Loops, Loop Control Statements – break, continue, pass; else with loops

Need for Functions, Defining & Invoking User-defined Functions, Return Statement, Function Input/Output Cases, Scope of Variables - Local, Global, Nested Functions, Function Arguments - Required, Positional, Default, Variable-length, main() Function, Documentation Strings, Recursive Functions, Anonymous Functions (Lambda), Library Functions

Modules - Import, from..import, Creating & Using Modules, Namespaces

Unit 3. Sequence, Set, Mapping Types:

Strings- Representation, Indexing, Slicing, Immutability, String Operators, Traversal, Accumulation, Formatting & Methods

Lists - Overview, Indexing, Slicing, Methods, Mutability, List Operations - Add, Update, Delete, Search, Copy, Traverse, Comprehension

Tuples - Operations, Immutability, Tuple Assignment, Arrays & Operations

Sets - Overview, Methods, Mathematical Operations, Frozenset, Comprehension

Dictionaries - Overview, Methods, Operations, Traversal, Comparison

Unit 4. File Handling, Exception Handling & Object Oriented Programming:

File Handling - Types, Paths, Basic Operations on Files - Open/Close, Read/Write, CSV Files, OS/Pathlib

Error & Exception Handling - Syntax Errors, Built-in Exceptions, Catching and Handling

Exceptions: try-except, raise, User-defined Exceptions, Assertions

OOP Concepts: Classes, Objects, Attributes, Methods, Constructor and Destructors

Encapsulation: Private and Public Members

Inheritance: Single, Multilevel, Multiple, Method Overriding

Unit 5: Abstract Data Structures and GUI Programming

Abstract Data Structures (ADTs): Concepts and Importance

Linked List: Definition, Types- Singly, Doubly, Circular; Node Structure, Insertion, Deletion, Traversal (Single Linked list implementation only)

Stacks: LIFO Principle, Implementation using List, Applications

Queues: FIFO Principle, Implementation using List, Priority Queues

GUI Programming with Tkinter: Widgets (Label, Button, Entry, Menu, Listbox, Canvas etc.), Event Handling, Building Simple GUI Apps

Textbooks:

1. Python Programming-An Object Oriented approach, Anita Goel, Universities Press
2. Python Programming using Problem Solving Approach Reema Thareja Oxford University Press 2020

3. Exploring Python, Budd T A, McGraw-Hill Education, 1st Edition, 2011.

Reference Book:

1. Python: The Complete Reference, Martin C. Brown, Mc Graw-Hill, 2018
2. Fundamentals of Python, Kenneth A. Lambert. (2019), First Programs, 2nd Edition, CENGAGE Publication.

Activities:

Outcome: Explain the basic features, syntax, data types, and operators of Python programming.

Activity: Conduct a "Python Basics Lab" where students write small programs to demonstrate literals, variables, data types, and operators (e.g., swapping numbers, simple calculator).

Evaluation Method:

- Lab performance checklist (execution of 3 mini tasks)
- Short quiz with multiple-choice and fill-in-the-blanks on syntax, data types, and operators

Outcome: Apply control flow constructs, functions, and modules to develop structured Python programs.

Activity: Group activity - "Python Problem Solving Challenge": Students solve real-life problems (e.g., finding prime numbers, grade calculator, menu-driven calculator) using control structures, functions, and importing standard modules.

Evaluation Method:

- Code submission with proper use of functions/modules (20%)
- Viva-voce to explain logic and flow of control (40%)
- Unit test with scenario-based programming questions (40%)

Outcome: Demonstrate the use of sequences, sets, and dictionaries for effective data handling and manipulation.

Activity: Hands-on mini project – "Student Data Manager": Students create a program using lists, tuples, sets, and dictionaries to store and manipulate student records (e.g., marks, courses, hobbies).

Evaluation Method:

- Practical demo of program with at least 5 data operations (add, search, delete, update, traverse)
 - Evaluation rubric for correctness, efficiency, and use of appropriate data structure
-

Outcome: Implement file handling techniques and apply exception handling mechanisms for robust applications.

Activity: Individual assignment - "File-Based Address Book": Students create a program to store, update, and retrieve data from files, with exception handling for invalid inputs or missing files.

Evaluation Method:

- Assessment of program correctness (file read/write, append, delete, exception handling)
- Short quiz with error-tracing and debugging questions (given code with errors, students identify and correct)

Outcome: Develop object-oriented and GUI-based applications using Python.

Activity: Mini Project – "Student Information System with GUI": Students design a simple Tkinter-based application with classes/objects for handling student data, including basic GUI widgets (Entry, Button, Listbox).

Evaluation Method:

- Project demo and presentation (50%)
- Rubric-based evaluation for OOP concepts (classes, inheritance, encapsulation) and GUI design (widgets, event handling) (30%)
- Peer review/feedback on usability (20%)

C3P: Python Programming and Data Structures Lab:

1. Basic Python Programs:

- a. Write a program to display basic details (name, roll number, department) using print() and demonstrate different literal types (int, float, string, boolean, complex).
- b. Write a program to perform arithmetic, relational, logical, bitwise, and assignment operations on given inputs.

2. Control Flow Practice

- a. Write a program to find the largest of three numbers using if-elif-else.
- b. Write a program to check whether a number is prime or not using loops.
- c. Write a program to illustrate the use of loop control statements (break, continue, pass).

3. Functions and Recursion

- a. Write a program to define a function to calculate factorial of a number (using recursion).
 - b. Write a program to demonstrate different types of function arguments (default, positional, keyword, variable-length).
4. Write a program to illustrate string slicing, concatenation, repetition, and built-in methods.
 5. Write a program to create a list of numbers, perform insertion, deletion, searching, sorting, and list comprehension.
 6. Write a program to demonstrate tuple packing, unpacking, and immutability.
 7. Write a program to implement set operations (union, intersection, difference, subset, superset).
 8. Write a program to create a dictionary of student roll numbers and marks, and perform add, update, delete, and traversal operations.
 9. Write a program to read and display count of vowels, consonants, digits, and spaces of a text file.
 10. Write a program to copy the contents of one file into another file.
 11. Write a program to read and process student marks from a CSV file (calculate average, highest, lowest).
 12. Write a program to demonstrate exception handling using try-except-finally.
 13. Write a program to create a class Student with attributes and methods to display details.
 14. Write a program to demonstrate single and multilevel inheritance.
 15. Implement stack (LIFO) and queue (FIFO) using lists and linked lists.
 16. Implement singly linked lists: node creation, insertion, deletion, traversal.
 17. Write a Tkinter program with Label, Entry, and Button widgets to take user input and display it.
 18. Write a Tkinter program to create a simple calculator application.

C4: Statistical Foundations for Data Science

Course Objectives

3. Functions and Recursion

- a. Write a program to define a function to calculate factorial of a number (using recursion).
 - b. Write a program to demonstrate different types of function arguments (default, positional, keyword, variable-length).
4. Write a program to illustrate string slicing, concatenation, repetition, and built-in methods.
 5. Write a program to create a list of numbers, perform insertion, deletion, searching, sorting, and list comprehension.
 6. Write a program to demonstrate tuple packing, unpacking, and immutability.
 7. Write a program to implement set operations (union, intersection, difference, subset, superset).
 8. Write a program to create a dictionary of student roll numbers and marks, and perform add, update, delete, and traversal operations.
 9. Write a program to read and display count of vowels, consonants, digits, and spaces of a text file.
 10. Write a program to copy the contents of one file into another file.
 11. Write a program to read and process student marks from a CSV file (calculate average, highest, lowest).
 12. Write a program to demonstrate exception handling using try-except-finally.
 13. Write a program to create a class Student with attributes and methods to display details.
 14. Write a program to demonstrate single and multilevel inheritance.
 15. Implement stack (LIFO) and queue (FIFO) using lists and linked lists.
 16. Implement singly linked lists: node creation, insertion, deletion, traversal.
 17. Write a Tkinter program with Label, Entry, and Button widgets to take user input and display it.
 18. Write a Tkinter program to create a simple calculator application.

C4: Statistical Foundations for Data Science

Course Objectives

Unit 1: Fundamentals of Probability & Basic Statistics

Probability: Concept of Uncertainty, Axioms and rules of probability, Conditional probability and independence, Law of total probability and Bayes' theorem

Measures of central tendency: Mean, Median, Mode

Measures of dispersion: range, interquartile range, variance, standard deviation

Introduction to correlation and covariance

Data representation: histograms, bar charts, scatter plots

Unit 2: Random Variables, Expectation, and Variance

Random variables: definition, types (discrete & continuous), and properties, Probability mass function (PMF) and probability density function (PDF), Cumulative distribution function (CDF), Mathematical expectation (mean), variance, and standard deviation, Moments and moment-generating functions

Unit 3: Probability Distributions

Discrete distributions: Binomial, Poisson, Geometric, Negative Binomial distributions - definitions, properties, and examples

Continuous distributions: Uniform, Normal (Gaussian), Exponential, Gamma distributions - definitions, properties, and applications

Joint, marginal, and conditional distributions, Introduction to Central Limit Theorem

Unit 4: Correlation and Regression

Bivariate data and scatter plots

Correlation: Pearson and Spearman coefficients, interpretation

Simple linear regression: model, estimation, properties, and analysis of variance

Multiple linear regression basics (conceptual understanding)

Residuals and goodness of fit

Unit 5: Statistical Inference, Estimation, and Hypothesis Testing

Population and sample, parameters and statistics, Sampling distributions, Point and interval estimation (confidence intervals), Tests of significance: z-test, t-test, chi-square test, and F-test, p-values and errors (Type I & II), Power of a statistical test

Textbooks:

1. Probability and Statistics for Engineers and Scientists, Ronald E. Walpole, Wiley.
2. Sheldon M. Ross, Introduction to Probability and Statistics for Engineers and Scientists
3. Douglas C. Montgomery & George C. Runger, Applied Statistics and Probability for Engineers

Reference Books:

1. D.C. Agarwal, Statistics for Data Science and AI
2. Larry J. Stephens, Excel Data Analysis: Your visual blueprint for analyzing data, statistics, and AI

Activities:

Outcome: Apply probability rules, conditional probability, and Bayes' theorem

Activity: Classroom Quiz (MCQs & short problems on probability, conditional probability, Bayes).

Evaluation Method: Individual quiz marks (accuracy of solutions).

Outcome: Compute and interpret descriptive statistics & visualize data

Activity: Poster Presentation – students prepare posters with a dataset summary (mean, median, variance, histograms, scatter plot).

Evaluation Method: Rubric-based evaluation (clarity, correctness, creativity, interpretation).

Outcome: Analyze random variables and probability distributions

Activity: Seminar/Presentation – each student (or group) explains one distribution (Binomial, Poisson, Normal, Exponential, etc.) with real-life examples and graphs.

Evaluation Method: Seminar grading (content accuracy, explanation, use of examples, presentation skills).

Outcome: Perform correlation and regression analysis

Activity: Case Study Assignment – students are given real-world data (e.g., sales vs advertising, study hours vs marks) to compute correlation/regression and interpret.

Evaluation Method: Submission of case study report + viva (correctness of analysis and interpretation).

Outcome: Conduct statistical inference (estimation & hypothesis testing)

Activity: Group Discussion/Debate – teams test a given hypothesis (e.g., “Male and Female students score equally”) using sample data and statistical tests, then defend their conclusion.

Evaluation Method: Marks based on correct application of test, interpretation of p-value, and clarity in argument.

C5P: Statistical Foundations for Data Science Lab

Advanced Spreadsheets/Excel Lab/PSPP Open Source

1. Construct a contingency table from sales data and compute conditional probabilities. Verify independence of variables.
2. Apply Bayes' theorem on medical test data to compute the probability of disease given a positive result.
3. Calculate measures of central tendency (mean, median, mode) for student marks dataset.
4. Compute measures of dispersion (range, variance, standard deviation, IQR) for the same dataset and interpret variability.
5. Create a histogram of student marks and comment on the shape of the distribution.
6. Prepare bar charts of categorical data (e.g., Gender vs Section) and interpret group comparisons.
7. Generate scatter plots between Hours Studied and Exam Score, compute correlation and covariance, and interpret the relationship.
8. Random Variable Simulation: Simulate and visualize discrete/continuous random variables using Excel functions and Data Analysis Tool pack.
9. Expectation & Variance Calculation: Use Excel formulas to compute expected value, variance, and standard deviation from given datasets.