Bansilal Ramnath Agarwal Charitable Trust's

Vishwakarma Institute of Technology

(An Autonomous Institute affiliated to Savitribai Phule Pune University)

NEP Compliant Structure & Syllabus of

Department of Computer Science and Engineering – Data Science Pattern

'A-24'

S.Y.B. Tech.

Effective from Academic Year 2026-27

Prepared by: - Board of Studies in Computer Science and Engineering – Data Science

Approved by:-Academic Board, Vishwakarma Institute of Technology,

Dr. Deepa Abin
Chairman-BoS

Dr. Parikshit Mahalle
Dean- Academics

Dr. R. M. Jalnekar
Chairman-Academic Board

Bansilal Ramnath Agarwal Charitable Trust's
VISHWAKARMA INSTITUTE OF TECHNOLOGY–PUNE
(Autonomous Institute affiliated to Savitribai Phule Pune University)

Vision & Mission Statement

Institute

Vision:

Excellence in Technical Education with Holistic Development

Mission:

- To make the industry ready graduating engineers with high human values
- To impart technical, social, innovative, and entrepreneurial skills of the highest standards.
- To prepare passing graduates for higher studies and high quality research.

Department of CSE (Data Science)

Vision:

Excellence in Data Science to empower the future of technology with holistic development

Mission:

- To equip students for industry, academia, research, and entrepreneurship by offering excellent education in emerging data science techniques.
- To empower students with ethical values and ensure their positive impact on society and beyond.
- To envision students with inter disciplinary skill sets, fully equipped to address the evolving needs of society.

Program Outcomes

PO1: Engineering Knowledge: Apply knowledge of mathematics, natural science, computing, engineering fundamentals and an engineering specialization as specified in WK1 to WK4 respectively to develop to the solution of complex engineering problems.

PO2: Problem Analysis: Identify, formulate, review research literature and analyze complex engineering problems reaching substantiated conclusions with consideration for sustainable development. (WK1 to WK4)

PO3: Design/Development of Solutions: Design creative solutions for complex engineering problems and design/develop systems/components/processes to meet identified needs with consideration for the public health and safety, whole-life cost, net zero carbon, culture, society and environment as required. (WK5)

PO4: Conduct Investigations of Complex Problems: Conduct investigations of complex engineering problems using research-based knowledge including design of experiments, modelling, analysis & interpretation of data to provide valid conclusions. (WK8).

PO5: Engineering Tool Usage: Create, select and apply appropriate techniques, resources and modern engineering & IT tools, including prediction and modelling recognizing their limitations to solve complex engineering problems. (WK2 and WK6)

PO6: The Engineer and The World: Analyze and evaluate societal and environmental aspects while solving complex engineering problems for its impact on sustainability with reference to economy, health, safety, legal framework, culture and environment. (WK1, WK5, and WK7).

PO7: Ethics: Apply ethical principles and commit to professional ethics, human values, diversity and inclusion; adhere to national & international laws. (WK9)

PO8: Individual and Collaborative Team work: Function effectively as an individual, and as a member or leader in diverse/multi-disciplinary teams.

PO9: Communication: Communicate effectively and inclusively within the engineering community and society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations considering cultural, language, and learning differences

PO10: Project Management and Finance: Apply knowledge and understanding of engineering management principles and economic decision-making and apply these to one's own work, as a member and leader in a team, and to manage projects and in multidisciplinary environments.

PO11: Life-Long Learning: Recognize the need for, and have the preparation and ability for i) independent and life-long learning ii) adaptability to new and emerging technologies and iii) critical thinking in the broadest context of technological change

Academic Information–Please visit www.vit.edu

PSOs-Program Specific Outcomes

PSO-1: Apply Data Science techniques to extract valuable insights from real-world data for informed decision-making.

PSO-2: Apply Machine Learning and AI concepts for predictive modeling and intelligent system development.

PEOs-Program Educational Objectives

PEO-1: To excel in a professional career in artificial intelligence, data science engineering and related interdisciplinary domains by applying advanced knowledge and practical skills.

PEO-2: To demonstrate a strong foundational understanding that supports higher education, research endeavors, and innovation in emerging areas of technology.

PEO-3: To embody professional ethics, a commitment to lifelong learning, leadership qualities, and a collaborative spirit aimed at contributing meaningfully to society and promoting environmental sustainability.

Nomenclature for Teaching and Examination Assessment SchemeAY2026-27

Sr No.	Category	Head of Teaching/Assessment	Abbreviation used
1	Teaching	Theory	Th
2	Teaching	Laboratory	Lab
3	Teaching	Tutorial	Tut
4	Teaching	Open Elective	OE
5	Teaching	Multi-Disciplinary	MD
6	Teaching	Computer Science	CS
7	Assessment	Laboratory Continuous Assessment	CA
8	Assessment	Mid Semester Assessment	MSA
9	Assessment	End Semester Assessment	ESE
10	Assessment	Home Assignment	HA
11	Assessment	Course Project	CP
12	Assessment	Group Discussion	GD
13	Assessment	Power Point Presentation	PPT
14	Assessment	ClassTest–1	CT1
15	Assessment	ClassTest–2	CT2
16	Assessment	Mid Semester Examination	MSE
17	Assessment	End Semester Examination	ESE
18	Assessment	Written Examination	WRT
19	Assessment	Multiple Choice Questions	MCQ
20	Assessment	Laboratory	LAB

Contents

FF No.: 653

Structure of Second Year CSE-DS for Academic Year 2026-27

Course Code	Course Name	Type	Teaching Learning Scheme				
			Th	Tut	Lab	Hrs. / Week	Credits
Module 3 - Semester 1							
DS2003	FUNDAMENTALS OF DATA STRUCTURE AND ALGORITHMS	**DC**	3	0	2	5	4
DS2004	FUNDAMENTALS OF DATA SCIENCE	**DC**	2	0	2	4	3
DS2005	OBJECT ORIENTED PROGRAMMING	**DC**	2	0	2	4	3
DS2006	COMPUTER NETWORKS	**DC**	2	0	2	4	3
MM0402	PROBABILITY AND STATISTICS	**IEL**	2	1	0	3	3
HS2002	FROM CAMPUS TO CORPORATE - 1	**IEL**	2	0	0	2	2
HS2001	REASONING AND APTITUDE DEVELOPMENT - 3	**DC**	1	0	0	1	1
DS2001	DESIGN THINKING - 3	**SM**	0	1	0	1	1
DS2002	ENGINEERING DESIGN AND INNOVATION - III	**SM**	0	0	2	2	2

Assessment scheme

Course Code	Course Name	ESE TH (W)	ESE (R)	CVV	CP	LAB	GD/PPT/HA	MSE (W)	MSE (R)	ESE	PRACT + CVV	Total
DS2003	FUNDAMENTALS OF DATA STRUCTURE AND ALGORITHMS				30	10					40 + 20	100
DS2004	FUNDAMENTALS OF DATA SCIENCE	40		20	30	10						100
DS2005	OBJECT ORIENTED PROGRAMMING	40		20	30	10						100
DS2006	COMPUTER NETWORKS			20	30	10		40				100
MM0402	PROBABILITY AND STATISTICS	40			30		20+ 10(HA)					100
HS2002	FROM CAMPUS TO CORPORATE - 1		50						50			100
HS2001	REASONING AND APTITUDE DEVELOPMENT - 3									100		100
DS2001	DESIGN THINKING - 3									100		100
DS2002	ENGINEERING DESIGN AND INNOVATION - III		70						30			100

Dr. Deepa Abin
Chairman-BoS

Dr. Parikshit Mahalle
Dean Academics

FF No. : 654

DS2003 : Fundamentals of Data Structure and Algorithms

Course Prerequisites: Introduction to Programming, Discrete Mathematics, Basic Problem-Solving Skills

Course Objectives:

1. Understand basic data structures and algorithmic complexity.
2. Implement searching, sorting, and recursion algorithms.
3. Apply advanced structures like trees, graphs, and hashing.
4. Design solutions using stacks, queues, and linked lists.
5. Compare and optimize algorithm performance.
6. Use tools to develop and test algorithmic solutions

Credits: 4 **Teaching Scheme Theory:** 3 Hours/Week

Lab: 2 Hours/Week

Course Relevance:. This subject builds essential skills in organizing data and designing efficient algorithms, enabling effective problem-solving and optimized software development.

SECTION-1

Unit-1: Introduction to Data Structures and Complexity: Abstract Data Types, need of data structures, classification of data structures, complexity analysis of algorithms using Big-O, Big-Ω and Big-Θ notations, time-space trade-off, recursion basics and applications.

(7 Hrs)

Unit-2: Arrays and Searching & Sorting Techniques: 1D and 2D arrays, sparse matrix representation, Simple and Fast Transpose, linear search, binary search, bubble sort, selection sort, insertion sort, merge sort, quick sort, time complexity of all techniques.

(7 Hrs)

Unit-3: Stacks and Queues: Stack operations and applications like expression evaluation, infix to postfix conversion, prefix conversion, implicit and explicit stack, recursion using stack; queue operations, circular queue, priority queue, deque, applications of queues in real-world problems.

(7 Hrs)

SECTION-2

Unit-4: Linked Lists: Singly linked list, doubly linked list, circular linked list, operations (insert, delete, traverse, search), applications such as polynomial operations, memory management using dynamic allocation.

(7 Hrs)

Unit-5: Trees: Tree terminologies, binary trees, binary search trees (BST), tree traversals (inorder, preorder, postorder), AVL trees, expression trees, heap trees, applications in decision making and file systems.

(7 Hrs)

Unit-6: Graphs and Hashing: Graph representation using adjacency list and matrix, BFS, DFS, applications of graphs in networking and social media, hash tables, collision resolution techniques: chaining and open addressing.

(7 Hrs)

List of Practical's: (Any Ten)

1. Design a recursive program to compute the factorial of a given number, illustrating how function calls build up and unwind like a stack.
2. Simulate bed allocation in a hospital ward using a sparse matrix where only a few beds are occupied, and display the occupied vs. available status efficiently.
3. Develop a phonebook application where users can search for contacts using both linear and binary search methods. Compare the search performance for small vs. large contact lists.
4. Build a shopping cart system where products can be sorted by price or ratings using various sorting algorithms. Measure and compare the time efficiency of each method for large orders.
5. Create a calculator that validates and evaluates arithmetic expressions using stacks. Implement infix to postfix conversion and postfix evaluation to simulate real-time computation.
6. Design a queue-based ticketing system (like in banks or hospitals) where customers are issued tokens and served in their order of arrival. Use a circular queue to manage overflow efficiently.
7. Simulate how train coaches (bogies) are connected using different linked lists. Each coach is a node; use singly, doubly, and circular lists to represent and manipulate various train configurations such as adding, removing, and navigating between coaches.
8. Create a contact management system where names are stored in a Binary Search Tree (BST). Implement insertion, deletion, and search functionalities to maintain a well- organized directory.

9. Design a city navigation simulator using graphs where roads between landmarks are represented as edges. Use BFS and DFS to find paths from one place (e.g., home) to another (e.g., hospital or restaurant).
10. Implement a phone directory system using hash tables where each contact is stored by name. Handle collisions using separate chaining and provide options to add, delete, and search for contacts efficiently.
11. Develop a system to manage a library shelf where book titles are stored and sorted alphabetically. Use sorting techniques like insertion and quick sort and compare their performance.
12. Simulate an undo-redo feature of a text editor using two stacks. Enable users to type, undo recent changes, and redo actions as seen in common writing tools.
13. Create a task scheduling system where each task has a priority level. Use a priority queue to ensure higher priority tasks (e.g., emergency alerts) are served before others.
14. Simulate a memory management module where memory blocks are allocated and freed using linked lists. Implement dynamic memory allocation.
15. Build a laundry basket simulator where clothes are added one by one and removed in reverse order using a stack. Display the order in which clothes are removed for washing.
16. Develop an inventory management system for a small retail store where products are stored in a hash table using their IDs as keys. Implement features to add, search, update, and remove products efficiently while resolving collisions using open addressing.

List of Projects:

1. Smart Hospital Management System: A hospital wants to efficiently manage patient flow, prioritize emergency cases, and allocate resources like beds and doctors. Students need to perform:
   • Design data structures to manage patient queues with priority handling
   • Implement searching techniques for quick patient lookup
   • Use sparse matrix concepts for bed allocation representation
   • Optimize operations for faster emergency handling
2. E-Commerce Cart and Inventory System: An online shopping platform needs to manage customer carts, product sorting, and inventory tracking efficiently. Students need to perform:
   • Design structures for cart management with dynamic updates
   • Implement sorting algorithms based on price or ratings
   • Maintain inventory using appropriate data structures
   • Implement undo/redo functionality using stack concepts
3. City Route Planner and Navigation Assistant: A city navigation system needs to find optimal routes between locations and guide users efficiently. Students need to

perform:
- Represent the city map using graph data structures
- Implement BFS/DFS for route finding
- Analyze shortest or feasible paths
- Optimize traversal for better performance

4. Digital Library Book Manager: A digital library wants to manage book records, search efficiently, and track borrowing activities. Students need to perform:
   - Store and manage book data using appropriate structures
   - Implement searching algorithms
   - Apply sorting techniques for organizing books
   - Track borrow/return operations efficiently
5. Student Academic Dashboard: A university wants to maintain student records including marks, attendance, and performance analytics. Students need to perform:
   - Design data structures to store student information
   - Implement operations for updating and retrieving records
   - Analyze performance using basic algorithms
   - Generate summaries and reports efficiently
6. Login and Credential Manager: A system needs to securely manage user credentials and track login attempts. Students need to perform:
   - Implement hashing techniques for storing passwords
   - Design search mechanisms for user authentication
   - Track login attempts and manage user sessions
   - Ensure efficient lookup and security handling
7. Supermarket Billing System: A retail store wants to automate billing, manage inventory, and apply discounts. Students need to perform:
   - Design structures for product storage and billing
   - Implement algorithms for subtotal and discount calculation
   - Update inventory dynamically after each transaction
   - Optimize billing process for efficiency
8. To-Do List Manager: A task management system is required to help users organize and track daily activities. Students need to perform:
   - Implement list-based structures for task management
   - Add, delete, and update tasks dynamically
   - Prioritize tasks using suitable data structures
   - Implement undo functionality using stacks
9. Expression Solver and Visualizer: A system is needed to evaluate mathematical expressions and show step-by-step computation. Students need to perform:
   - Convert infix expressions to postfix notation
   - Evaluate postfix expressions using stacks
   - Display step-by-step evaluation process
   - Optimize expression handling
10. Railway Coach Management Simulator: A railway system needs to manage coach connections dynamically. Students need to perform:
    - Represent coaches using linked list structures

- Implement insertion and deletion of coaches
- Traverse and display train structure
- Handle dynamic updates efficiently

Assessment Scheme:

CP	LAB	PRACT	CVV
30	10	40	20

CP - Course Project
LAB - Continuous Assessment
PRACT – External Practical Exam
CVV – Comprehensive Viva

Text Books:

1. E. Horowitz, S. Sahni, D. Mehta; Fundamentals of Data Structures in C, 2nd Edition, Universities Press, 2017.
2. Y. Langsam, M. Augenstein, A. Tannenbaum; Data Structures Using C and C++, 2nd Edition, Pearson Education, 2006.

Reference Books:

1. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein; Introduction to Algorithms, 4th Edition, MIT Press, 2022. https://mitpress.mit.edu/9780262033848/
2. Brad Miller, David Ranum; Problem Solving with Algorithms and Data Structures using Python; Open Book Project; 3rd Edition, 2023. https://runestone.academy/ns/books/published/pythonds/index.html
3. NarasimhaKarumanch, Data Structures and Algorithms Made Easy, Publisher CareerMonk Publications, 2024

Moocs Links and additional reading material:

1. Prof. Naveen Garg; Data Structures and Algorithms; NPTEL – IIT Delhi; https://nptel.ac.in/courses/106/102/106102064.
2. UC San Diego, National Research University Higher School of Economics; Data Structures; Coursera; https://www.coursera.org/learn/data-structures

Course Outcomes:

Student should be able to

1. Analyze and compute time and space complexity of algorithms.
2. Apply searching and sorting algorithms to solve computational problems.
3. Apply stack and queue operations in applications.
4. Develop and use various types of linked lists to manage dynamic data.
5. Apply tree data structures for hierarchical data representation and operations.
6. Demonstrate understanding of graph and hashing techniques and their applications.

CO PO Map

CO/PO	PO1	PO 2	PO 3	PO4	PO5	PO 6	PO 7	PO 8	PO 9	PO10	PO11	PSO1	PSO2
CO:1	3	3	1	2	2	-	-	-	-	-	2	1	1
CO:2	3	3	2	1	2	-	-	-	-	-	1	2	1
CO:3	3	2	2	1	2	-	-	-	-	-	1	1	2
CO:4	3	2	3	1	2	-	-	-	-	-	1	2	1
CO:5	3	3	3	2	2	-	-	-	-	-	2	2	2
CO:6	3	3	3	2	3	-	-	-	1	-	2	2	3

CO Attainment levels

CO	Attainment level
CO .1	
CO .2	
CO .3	
CO .4	
CO .5	
CO .6	

Future Courses Mapping:

Advanced Algorithms, Database Systems, Operating Systems, Software Engineering, Artificial Intelligence, Machine Learning, Computer Networks.

Job Mapping:

Software Developer, Data Scientist, System Analyst, Backend Engineer, AI/ML Engineer, Database Administrator, Network Engineer, Tech Consultant.

FF No. : 654

DS2004: Fundamentals of Data Science

Course Prerequisites: Python programming, fundamental concepts in statistics, basic problem-solving skills

Course Objectives:

1. Understand the evolution, scope, lifecycle, and real-world applications of data science.
2. Learn data collection methods, preprocessing techniques, and feature extraction from various data sources.
3. Perform exploratory data analysis and create effective data visualizations using Python and visualization tools.
4. Develop basic machine learning models for regression and classification problems.

Credits: 3 Hours/Week

Teaching Scheme Theory: 2 Hours/Week
Lab: 2 Hours/Week

Course Relevance: This course builds foundational skills in data science, enabling students to analyze real-world data, derive insights, and develop predictive models preparing them for advanced courses and careers in analytics, machine learning, and AI-driven decision-making.

SECTION-1

Unit-1: Introduction to Data Science

Overview, Evolution and Scope, Components: Data, Algorithms, Insights, Data Science Life Cycle: Data acquisition, Analysis, Action, Applications of Data Science: Healthcare, Finance, Retail & E-commerce, Manufacturing, Smart Cities & IoT & Social Media & Marketing, Case Study: Predictive Maintenance in Manufacturing – to identify the type of data requirement and to sketch a basic DS workflow.

(6 Hrs)

Unit-2: Data Collection, Cleaning, and Feature Extraction

Data Sources and Types: Types of data: structured, unstructured, semi-structured, Data formats: numerical, categorical, time-series, text. Sources: databases, APIs, IoT sensors, web, surveys. Data Collection Techniques: Manual vs automated collection methods, Web scraping requests, Data Cleaning and Preprocessing: Handling missing values, Removing duplicates and irrelevant data, Standardizing and converting data types, Detecting and treating outliers (Z-score, IQR – theory only), Encoding categorical variables: label encoding, one-hot encoding, Feature Extraction: Understanding features, Feature selection, Feature engineering, Converting non-numeric to numeric data for models.
Case Study: Social Media Sentiment Analysis- to process and extract features from user-generated text data to support sentiment classification.

(8 Hrs)

SECTION-2

Unit-3: Exploratory Data Analysis using Python

Exploratory Data Analysis (EDA) Concepts, Importance of EDA in Data Science projects, Basic statistics: Mean, Median, Mode, Standard Deviation; understanding distributions: Normal, Skewed, Correlation analysis (Pearson, heatmaps).Visualization Tools and Techniques: Line plots, bar charts, histograms, scatter plots, funnel chart, heatmap, waterfall chart; matplotlib, tableau, seaborn, Plotly, D3.js, Power BI, Qlik Sense.
Emerging AI tools: Datawrapper, Flourish, Gafrana.
Case Study: Indian rainfall data (2010 to 2020)-to analyze trends across different regions.

(8 Hrs)

Unit-4: Model Building and Evaluation

Model Building: Introduction to model training workflow methods. Building simple models: Linear Regression, K-Nearest Neighbors (KNN), K-means clustering. Model Evaluation Techniques Metrics for Regression: RMSE, MAE, R^2 Score. Metrics for Classification: Accuracy, Precision, Recall, F1-score, Confusion Matrix, Cross-validation.
Case Study – Student Performance Prediction- machine learning model that predicts whether a student is likely to pass or fail based on personal, academic, and family-related data.

(6 Hrs)

List of Practicals:

1) Explore a sample dataset and summarize its structure- including column names, data types, potential use cases; data cleaning and pre-processing, and analyse the distribution and central tendencies of variables and detect any skewness or irregularities.

2) A multi-specialty hospital wants to improve patient care and reduce treatment delays. Currently, they rely on manual records and experience difficulty in identifying disease trends and patient risk factors. Analyze how Data Science can transform the hospital system by:

- Explaining the evolution of Data Science in healthcare
- Identifying its scope and benefits
- Highlighting real-world applications such as disease prediction and patient monitoring

3) An online retail company is facing inconsistent sales performance across regions and seasons. Management wants to understand the entire data flow from raw data to actionable insights. Design a complete Data Science lifecycle for analyzing sales data:

- Define how data is collected (transactions, user behavior)
- Describe cleaning, EDA, modeling, and deployment stages
- Suggest how insights can improve business decisions

4) A smart city project collects data from multiple sources like traffic sensors, weather APIs, and public transport systems. However, the data is scattered and inconsistent. Develop a system to:

- Collect data from at least two sources (CSV + API)
- Integrate and merge datasets
- Prepare a unified dataset for analysis

5) A banking system has transaction data with missing values, duplicate entries, and inconsistent formats. These issues are affecting fraud detection models. Perform data preprocessing by:

- Handling missing values appropriately
- Removing duplicate and inconsistent records
- Detecting and treating outliers
- Applying normalization and encoding techniques

6) A retail company wants to understand customer buying behavior but the existing dataset lacks meaningful features for analysis. Enhance the dataset by:

- Creating new features (e.g., total spending, frequency of purchase)
- Performing feature scaling and selection
- Identifying the most relevant features for prediction

7) A university wants to analyze student performance to identify factors affecting academic success. Perform EDA to:

- Analyze distribution of marks
- Identify correlations between subjects
- Detect outliers and unusual patterns

8) A company's management team needs a dashboard to quickly understand sales trends, profits, and regional performance. Design visualizations that:

- Represent sales trends over time
- Compare region-wise performance
- Highlight key insights using charts and graphs

9) A real estate company wants to predict house prices based on features like location, size, and number of rooms. Build a regression model to:

- Train on historical housing data
- Predict prices for new houses
- Evaluate performance using error metrics

10) A bank wants to automate loan approval decisions based on applicant details such as income, credit score, and employment status. Develop a classification model to:

- Predict whether a loan should be approved or rejected
- Evaluate using accuracy and confusion matrix

11) A startup wants to leverage data to improve its decision-making but lacks a structured analytics pipeline. Perform a complete Data Science workflow:

- Data collection
- Data cleaning and preprocessing
- EDA and visualization
- Model building (regression or classification)
- Present insights and recommendations

Assessment Scheme:

CP	LAB	ESE(W)	CVV
30	10	40	20

CP - Course Project
LAB - Continuous Assessment
ESE (W)– End Sem Exam (Written)
CVV – Comprehensive Viva

Text Books:

1. C. O'Neil, R. Schutt; Doing Data Science; 1st Edition; O'Reilly Media; 2013

2. J. Grus; Data Science from Scratch: First Principles with Python; 2nd Edition; O'Reilly Media; 2019
3. W. McKinney; *Python for Data Analysis*; 2nd Edition; O'Reilly Media; 2018

Reference Books:

1. C. O'Neil, R. Schutt; Doing Data Science; 1st Edition; O'Reilly Media; 2013. Accessed: Jun. 17, 2025. [Online]. Available: https://www.oreilly.com/library/view/doing-data-science/9781449363871/

2. J. Grus; Data Science from Scratch: First Principles with Python; 2nd Edition; O'Reilly Media; 2019. Accessed: Jun. 17, 2025. [Online]. Available: https://www.oreilly.com/library/view/data-science-from/9781492041122/

3. W. McKinney; Python for Data Analysis; 2nd Edition; O'Reilly Media; 2018. Accessed: Jun. 17, 2025. [Online]. Available: https://www.oreilly.com/library/view/python-for-data/9781491957653/

4. C. Müller, S. Guido; Introduction to Machine Learning with Python; 1st Edition; O'Reilly Media; 2016. Accessed: Jun. 17, 2025. [Online]. Available: https://www.oreilly.com/library/view/introduction-to-machine/9781449369880/

Moocs Links and additional reading material:

1. J. Grus; 'Data Science from Scratch'; O'Reilly Learning; https://learning.oreilly.com/library/view/data-science-from/9781492041122/;

2. C. Brooks; 'Python Data Structures'; Coursera; https://www.coursera.org/learn/python-data;

3. R. Tibshirani, T. Hastie; 'Statistical Learning'; Stanford Online; https://online.stanford.edu/courses/sohs-ystatslearning-statistical-learning

4. A. Ng; 'Machine Learning'; Coursera; https://www.coursera.org/learn/machine learning

Course Outcomes:

Course Outcomes:

Student should be able to

1. Explore the fundamentals, applications, lifecycle, and ethical aspects of data science in various real-world domains.
2. Apply data pre-processing and feature extraction techniques to prepare data for analysis.
3. Analyse data using statistical summaries and visualize relationships and patterns through suitable graphical representations.
4. Build and evaluate predictive models using appropriate techniques for classification and regression.

CO PO Map

CO/PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO:1	3	3		2	2						2	1	1
CO:2	3	3	2		2						1	2	1
CO:3	3	2	3		3							1	2
CO:4	3	2	3		3							2	1

CO attainment levels

CO	Attainment level
CO .1	
CO .2	
CO .3	
CO .4	

Future Courses Mapping:

This course serves as a foundation for advanced courses such as Machine Learning, Deep Learning, Big Data Analytics, Artificial Intelligence, Business Intelligence, and Data Visualization.

Job Mapping:

Data Analyst, Junior Data Scientist, Business Analyst, Machine Learning Engineer (entry-level), and Data Visualization Specialist.

FF No. : 654

DS2005: Object Oriented Programming

Course Prerequisites: Introduction to Programming in C language, Basic Problem-Solving Skills

Course Objectives:

1. To explain the fundamental concepts of object-oriented programming and apply them to develop modular programs using classes and objects in C++.
2. To implement and analyze object-oriented features such as constructors, destructors, and operator overloading for developing reusable and efficient components.
3. To examine and evaluate the use of inheritance, polymorphism, and virtual functions in designing extensible and maintainable software systems.
4. To apply exception handling and file input/output mechanisms to design and develop robust and persistent C++ applications.

Credits: 3 **Teaching Scheme Theory:** 2 Hours/Week
Lab: 2 Hours/Week

Course Relevance: This subject develops core competencies in object-oriented programming, equipping students with the ability to design modular, reusable, and efficient software solutions. Through mastery of C++ concepts like classes, inheritance, polymorphism, and templates, learners build a strong foundation for system-level programming, competitive coding, and real-world application development.

SECTION-I

Unit I: Introduction to Object Oriented Programming

Fundamentals of Object-Oriented Programming (OOP) in C++: Variables, Data Types, and Operators, Control Structures, Loops and Iteration, Functions and Modular Programming, Basics of Console Input and Output Class, Dynamic Memory Allocation.
Overview of OOPs Principles, Classes & Objects, Creation & destruction of objects, Data Members, Member Functions, Access Specifier, this Pointer, Constructor & Destructor, Static class member, Friend class and functions, Function Overloading, Introduction to Operator Overloading.

(8 Hrs)

UNIT-II: Object Oriented Programming Principles

Inheritance, Types of Inheritance, Base and Derived class Constructors, Down casting and upcasting, Function overriding, Virtual functions, Polymorphism, Pure virtual functions, Virtual Base Class, C++ Class Hierarchy

(6 Hrs)

SECTION-II

UNIT-III: Templates, Operator Overloading and Standard Template Library

Function Templates, Class Templates, Operator Overloading, Overloading << and >> operators, Namespace usage and scope resolution, Standard Template Library (STL) components: vector, list, map, set, stack, queue.

(8 Hrs)

UNIT-IV: Exception and File Handling in C++

Exception Handling: Benefits, Try and catch block, throw statement, pre-defined exceptions in C++, writing custom Exception class, Stack Unwinding, File Handling: Streams, File Input/Output, Text File Handling, Binary File Handling, Error Handling during file operations, Use Cases

(6 Hrs)

List of Practical's:

1. **Fundamentals of C++**

a) Design an electricity billing system to calculate total cost based on unit slabs and taxes.
b) Develop a temperature monitoring module that converts Celsius readings to Fahrenheit for reporting.
c) Create a land measurement system to compute area of rectangular plots for municipal records.

2. **Decision Making and Loops**

a) Develop a voter eligibility system that processes multiple users and generates a report of eligible voters.
b) Create a student result analysis system that categorizes students into pass/fail/distinction using conditions and loops.
c) Simulate population growth using Fibonacci logic for a given number of years.

3. **Functions and Modular Programming**

a) Build a scientific calculator module implementing factorial (recursion) and prime number checking.
b) Develop a reusable function-based system to generate Fibonacci series and analyze number patterns.
c) Create a modular application for validating user inputs (e.g., password strength or numeric constraints).

4. **Classes and Objects**

a) Design a geometry tool using a Rectangle class for area and perimeter calculations in construction planning.
b) Develop a product management system using classes to store and display product details.
c) Implement a student information system using Python (class and constructor __init__).

5. **OOP Concepts (Static, this pointer, Friend Function)**

a) Develop a banking system using static members to track total accounts created.
b) Create an application demonstrating use of this pointer to resolve ambiguity in employee records.
c) Develop a class-based application using Python demonstrating constructor (__init__) and destructor (__del__).

6. Operator Overloading

a) Implement arithmetic operator overloading for a Complex number or financial transaction system.
b) Overload comparison operators to compare objects like students (marks) or products (price).
c) Design a system where objects can be added or compared using intuitive operators**.**

7. Inheritance and Polymorphism

a) Build a shape hierarchy (Circle, Rectangle, Triangle) using inheritance and virtual functions for area calculation.
b) Extend a banking system with Savings and Current accounts using runtime polymorphism.
c) Develop an employee management system using Python demonstrating inheritance, constructor chaining, and method overriding.

8. Templates and STL

a) Implement a generic function template to find maximum values in datasets (marks, salaries).
b) Develop a system using STL vector and list to manage dynamic collections (e.g., student records).
c) Use map and set to create a key-value based application like employee database or dictionary.
d) Develop an application using Python data structures (list, tuple, set, dictionary) to perform operations similar to STL containers.

9. Exception Handling

a) Design an ATM system that handles errors like insufficient balance and invalid inputs using try-catch.
b) Create a program with user-defined exceptions for invalid transactions or data entries.
c) Implement exception handling using Python (try-except and user-defined exceptions).

10. File Handling

a) Develop a student record system using text files for storing and retrieving data.
b) Implement a payroll system using binary files for efficient storage of employee records.
c) Develop a file handling application using Python for storing and retrieving records using classes.

List of Projects:

1.Multi-Player Game Engine Framework (Using OOP + Polymorphism + STL)
Description:
Design a modular game engine that supports multiple games like Tic Tac Toe, Snake, and Chess. Use inheritance and polymorphism for game rules, STL containers to manage player profiles and scores, and templates for generic game logic.

2. University ERP Simulator (Object-Oriented + File Handling + Exception Handling)
Description:
Simulate a mini-ERP with modules for student info, staff, timetable, fee tracking, and attendance. Use classes and virtual functions, file storage for persistent records, and exception handling for login errors, data corruption, etc.

3. Compiler Intermediate Code Generator (Advanced OOP + Stack + STL Map)
Description:
Parse simplified mathematical expressions and generate three-address code (TAC) or postfix

code. Use stack, maps for symbol tables, and template functions for evaluation logic.

4. Airline Reservation & Flight Management System (Linked Lists + Operator Overloading + File Streams)

Description:

Allow seat reservation, cancellation, and route planning. Overload operators for seat comparisons, use linked lists for dynamic seat mapping, and manage historical booking data via files.

5. Blockchain-based Voting System (File Handling + Hashing + STL)

Description:

Simulate a simple blockchain to securely record votes. Use file handling to persist blocks, hash functions for verification, and vectors/queues/maps from STL to manage users, blocks, and votes.

6. Dynamic Code Snippet Organizer (Templates + File Parsing + OOP)

Description:

A tool to manage C++/Python snippets. Use template classes to generalize storage, file handling to import/export snippets, and OOP to manage languages, tags, and categories.

7. AI-Powered Chatbot Framework (Polymorphism + STL + File Handling)

Description:

Design a rules-based chatbot with polymorphic response models (default, emotion-based, logic-based), maps and vectors for pattern matching, and text file storage for chat history and knowledge base.

8. Social Media Post Scheduler (Priority Queue + Templates + Exception Handling)

Description:

Simulate scheduling of posts with time-based priority. Use STL priority queues, template-based scheduling logic, and exception handling for invalid scheduling formats.

9. Memory Management Visualizer (Pointer Arithmetic + Custom Allocators)

Description:

Create a tool that visualizes memory allocation in arrays, objects, and pointers. Implement a custom memory allocator, simulate new/delete, and show fragmentation via console visualization.

10. Trading Bot Simulator for Stock Market (Templates + STL + File I/O + Inheritance)

Description:

Design a simulated trading bot that makes buy/sell decisions. Use template-based data structures for financial instruments, inheritance for bot strategies, and files to import/export market data.

Assessment Scheme:

CP	LAB	CVV	ESE(W)
30	10	20	40

CP - Course Project
LAB - Continuous Assessment
GD/PPT – Group Discussion/PowerPoint Presentation
TH(O) – Online Examination

Reference Books:

1. Behrouz A. Forouzan, Richard F. Gilberg, "COMPUTER SCIENCE – A Structured Programming approach using C", Indian Edition, Thomson, 3rd edition, 2007, ISBN: 9788131517888.
2. Bjarne Stroustrup, ― The C++ Programming language‖, Third edition, Pearson Education.1997, ISBN 9780201889543. 3. Kernighan, Ritchie, "The C Programming Language", Prentice Hall of India, 1988. ISBN 0-13-110362-8.
3. Robert Lafore, ―Object-Oriented Programming in C++, fourth edition, Sams Publishing, 2002, ISBN:0672323087 (ISBN 13: 9780672323089)
4. Herbert Schildt, ―C++ The complete reference, Eighth Edition, McGraw Hill Professional, 2011, ISBN:978-00-72226805
5. E. Balagurusamy- Object-oriented programming with C++, fourth edition, Mc Hill Professional,2008, ISBN 978-0-07-066907-9

MOOCS Links and Additional Reading Material:

1. Prof. Partha Pratim DasProgramming in C++, IIT Kharagpurhttps://onlinecourses.nptel.ac.in/noc21_cs02/preview
2. Programming in C++: A Hands-on Introduction Specialization https://www.coursera.org/specializations/hands-on-cpp

Course Outcomes:

Student should be able to

1. Demonstrate understanding of object-oriented programming fundamentals and implement modular programs using classes and objects in C++.
2. Apply constructors, destructors, operator overloading, and analyze class relationships to design solutions using inheritance and polymorphism
3. Analyze templates and STL components to develop generic software modules.
4. Implement exception-handling and file I/O techniques and evaluate their effectiveness in developing robust and persistent applications.

CO- PO Mapping:

CO/PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO:1	3	3	2	2	3	0	0	0	2	2	2	3	3
CO:2	3	3	2	3	3	0	0	0	2	2	2	3	3
CO:3	3	3	3	3	3	0	0	0	2	2	3	3	3

CO:4	3	3	2	3	3	0	0	0	2	2	2	3	3

CO attainment levels

CO	**Attainment level**
CO .1	
CO .2	
CO .3	
CO .4	

Future Courses Mapping:
Algorithm Analysis, Database Systems, Operating Systems, Data structures Artificial Intelligence, Machine Learning, Computer Networks.

Job Mapping:
Software Developer / Software Engineer, Systems Programmer, Embedded Systems Engineer Game Developer, Backend Developer, Machine Learning Engineer, AI Developer, Database Developer, Cybersecurity Analyst

FF No. : 654

DS2006: Computer Networks

Course Prerequisites: Basics of Computer organization and Digital electronics.

Course Objectives:

1. Describe basic concepts of switching, network topologies, and modules.
2. Understand different Transmission medias and differentiate Guided and Unguided medias.
3. Apply the concept of Data Link and Network layer to simulate and analyze different networks.
4. Differentiate the role of Transport and Application layer in OSI and TCP/IP network models.

Credits: 3 **Teaching Scheme Theory:** 2 Hour/Week
Lab: 2 Hours/Week

Course Relevance: This course builds foundational knowledge of computer networks, protocols, and models. It prepares students to design, configure, and troubleshoot networks using tools and real-world scenarios. The concepts are essential for careers in networking, cybersecurity, and IoT.

SECTION-1

Unit-1: Networking Fundamentals and Addressing Techniques

Basics of Computer Networks: Switching (Circuit & Packet), Network Topologies architecture and components, Network types, OSI/TCP-IP models, Introduction to Subnets, IP Addressing and its types (in detail Ex. types of addressing for classful and classless subnets), Basic Protocol stack.

Unit-2: Physical Layer

Transmission media: Guided and Unguided Standard terminologies. Introduction to the Physical Layer and its role in the OSI model, Protocols.

(16 Hrs)

SECTION-2

Unit-3: Data Link and Network Layer Protocols

Data Link Layer protocols, MAC addressing, Error detection and correction techniques, Hop-to-hop delivery and framing, Routing (Static and Dynamic), Router configuration basics, Network Layer Protocols: IPv4 and IPv6. Subnetting (Classful/Classless)

Unit-4: Transport and Application Layers

Transport Layer responsibilities, TCP vs UDP, Socket Programming basics using TCP/IP, Application Layer protocols and services, Session and Presentation layer concepts, Overview of well-known application protocols (HTTP, FTP, DNS, SMTP), DHCP and DNS server configuration using simulation tools.

Case Study: Data Centre Network Architecture

(12 Hrs)

List of Practical's: (Any 6)

1. **Basic LAN Design Using Star and Bus Topologies**
   Design simple LANs using switches and hubs.
   Observe device interconnection and test communication.
2. **Simulating OSI Layer Communication with Ping and Traceroute**
   Use ping and tracert to visualize data flow through OSI/TCP-IP layers.
   Understand encapsulation and address resolution.
3. **Static Routing Between Two Networks**
   Configure routers with static routes to enable communication between different subnets.
   Test with ping and verify routing table entries.
4. **Class-full Subnetting and IP Address Allocation**
   Use class-full addressing to efficiently divide a network.
   Assign IPs to devices and verify connectivity.
5. **Classless Subnetting and IP Address Allocation**
   Use VLSM/CIDR to efficiently divide a network.
   Assign IPs to devices and verify connectivity.
6. **DNS/ DHCP Server Configuration**
   Set up DNS and DHCP servers in a network.
   Verify dynamic IP assignment and name resolution.
7. **ACL-Based Packet Filtering and Access Control**
   Apply standard ACLs to permit or deny specific traffic (e.g., ICMP, Telnet) between subnets.
   Observe restricted and allowed paths.
8. **Firewall Simulation Using Extended ACLs**
   Use extended ACLs to simulate firewall behavior (e.g., block HTTP, allow DNS/FTP).
   Apply on router interfaces to control traffic.

Assessment Scheme:

CVV	CP	LAB	MSE (W)
20	30	10	40

CVV - Comprehensive Viva Voce
CP - Course Project
LAB - Continuous Assessment
MSE(W) – Mid Semester Examination (Written)

Reference Books:

1. Andrew S. Tanenbaum ,"Computer Networks", Pearson, 1994, ISBN-13: 978-0-13-212695-3
2. Stallings William., "Data and Computer Communications", Sixth Edition, Prentice Hall of India, 2014, 10th edition, ISBN 978-0-133-50648-8
3. Fourouzan B., "Data Communications and Networking", 5th edition, McGraw- Hill Publications, Fifth Edition - 1 July 2017. ISBN-13: 978-1259064753 ISBN-10: 1259064751.
4. Atul Kahate, "Cryptography and Network Security", McGraw Hill Publication, 2nd Edition, 2008, ISBN : 978-0-07-064823-4.

Moocs Links and additional reading material:

1. CCNA Module : https://www.netacad.com/courses/ccna-introduction-networks?courseLang=en US
2. CCNA Module : https://www.netacad.com/courses/ccna-switching-routing-wireless essentials?courseLang=en-US
3. Introduction to Networking, Instructor: NVIDIA Training https://www.coursera.org/learn/introduction-to-networking-nvidia
4. Computer Networks and Internet Protocol, by Prof. Soumya Kanti Ghosh, Prof. Sandip Chakraborty IIT Kharagpur https://onlinecourses.nptel.ac.in/noc22_cs19/preview

Course Outcomes:

Student should be able to

1. **Discuss** the fundamental concepts of networks, network topologies, and modules**.**
2. **Understand** the different Transmission medias and basics of physical layer protocols.
3. **Use** the concept of Data Link and Network layer to simulate and analyze different networks
4. **Differentiate** the role of Transport and Application layer in OSI and TCP/IP network models

CO PO Map

CO/PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO1
CO:1	2												
CO:2	2												
CO:3	2	2	2		2								
CO:4	2												

CO attainment levels

CO	Attainment level
CO .1	
CO .2	
CO .3	
CO .4	

Future Courses Mapping:
Advance Computer Networks, Introduction to IoT.

Job Mapping:
Software Developer / Network Engineer, Network Programmer, Embedded Systems Engineer Network Executive, Network Administrator, Backend Developer, Cybersecurity Analyst

FF No. : 654

MM0404: Probability and Statistics

Course Prerequisites: Basic algebra, Calculus, Logical reasoning and basic problem-solving skills

Course Objectives:

- To introduce students to basic statistical concepts and probability theory.
- To enable students to model real-world uncertainty using probability distributions.
- To train students in inferential techniques such as estimation and hypothesis testing.
- To build foundational skills in analyzing relationships between variables using correlation and regression.

Credits: 3

Teaching Scheme Theory: 2 Hours/Week
Tutorial: 1 Hour/Week

Course Relevance: This course provides the mathematical foundation needed to analyze data, quantify uncertainty, and make evidence-based engineering decisions. It enables students to work effectively in fields such as data science, quality control, machine learning, reliability engineering, and research.

SECTION-1

Unit-1: Descriptive Statistics & Fundamentals of Probability: Types of data: qualitative, quantitative, discrete, continuous, Frequency distribution; visualizations (histogram, bar chart, box plot), Measures of central tendency: mean, median, mode, Measures of dispersion: variance, standard deviation, coefficient of variation, Basic probability rules: axioms, complement rule, addition rule, Conditional probability & independence, Bayes' theorem (simple engineering examples)

(**7 Hrs)**

Unit-2: Random Variables & Common Distributions: Random variables: discrete & continuous, Probability Mass Function (PMF), Probability Density Function (PDF), Cumulative Distribution Function (CDF), Expectation, variance, and properties, Discrete distributions: Bernoulli, Binomial, Poisson, Continuous distributions: Uniform and Exponential, Applications in engineering.

(**7 Hrs)**

SECTION-2

Unit-3: Normal Distribution & Sampling Theory: Normal distribution: standard normal, Z-scores, Applications of normal distribution (tolerances, quality control), Normal approximation to binomial & Poisson, Sampling: random sampling, sample mean, sample variance, Sampling distribution of sample mean, Standard error, Central Limit Theorem (CLT) – intuitive explanation with examples.

(**7 Hrs)**

Unit-4: Estimation, Hypothesis Testing, Correlation & Regression: Estimation: Point estimation: methods (Maximum Likelihood, Method of Moments), properties of estimators (unbiasedness, consistency, efficiency); Interval estimation: confidence intervals for mean (σ known/unknown) Hypothesis Testing: Test statistic, null and alternative hypotheses, level of significance, p-value, Type I & II errors, one-tailed and two-tailed tests; One-sample Z-test and t-test Chi-Square (χ^2) Tests: Chi-square distribution, degrees of freedom, test for goodness of fit, test for independence of attributes, applications Analysis of Variance (ANOVA): Assumptions, ANOVA table, one-way ANOVA, F-test, interpretation, engineering applications Correlation & Regression: Karl Pearson's correlation coefficient, Spearman's rank correlation, simple linear regression, regression coefficients, applications.

(**7 Hrs)**

List of Tutorial: (Any 8)

1. Descriptive Statistics for Factory Production A small factory records the daily production output of its machines for one week. Using the given data, students calculate the mean, median, and mode to understand the typical production level. Measures such as variance, standard deviation, and coefficient of variation are also computed to study how much the production varies from day to day and whether the process is stable.
2. Probability in Electrical Component Failure An electrical system uses components that may fail during operation. The probability of failure under normal and overloaded conditions is provided. Students solve numerical problems using basic probability rules and conditional probability to determine the chances of system failure. Bayes' theorem is used to update probabilities when additional information about operating conditions is known.
3. Modeling Call Arrivals Using Discrete Distributions A customer support center receives a fixed number of calls during office hours. Students use Binomial and Poisson distributions to calculate the probability of receiving a certain number of calls in a given time period. The results are interpreted to understand workload planning and staffing requirements.
4. Reliability Analysis Using Exponential Distribution The lifetime of a machine part follows an exponential distribution. Students calculate the probability that the part fails within a specific time and find the expected life of the component. The numerical results are used to explain maintenance scheduling and replacement planning in engineering systems.
5. Normal Distribution in Quality Control The weights of packaged products follow a normal distribution with a known mean and standard deviation. Students calculate Z-scores and determine the percentage of products that fall outside acceptable limits. Based on the numerical results, they analyze whether the packaging process meets quality standards.
6. Hypothesis Testing for Process Verification A company claims that the average time taken to complete a manufacturing task meets a given standard. Using sample data, students frame the null and alternative hypotheses and perform a one-sample Z-test or t-test. The decision is made based on the calculated test statistic and significance level, and the result is interpreted in a practical context.
7. Chi-Square Test for Inspection Data (Tool-Based) A quality inspection department records the number of defective and non-defective items produced by different machines. Students use a statistical tool such as Excel or Python to perform the chi-square test for independence. The output from the tool is analyzed to decide whether product quality depends on the machine used.
8. One-Way ANOVA for Comparing Manufacturing Methods (Tool-Based) Three different manufacturing methods are tested to compare their average output. Students apply one-way ANOVA using Excel or Python to analyze the data. Based on the F-value and p-value obtained from the tool, they conclude whether there is a significant difference between the methods.

9. A smart commercial building records hourly energy consumption (in kWh) for different days of the week. Students analyze the data using descriptive statistics (mean, median, variance, standard deviation) to understand typical energy usage and variability. Probability concepts are applied to estimate the likelihood of peak-load conditions, and the results are interpreted to suggest energy-saving strategies.
10. In a computer network, data packets are transmitted through a router. Each packet has a certain probability of being successfully transmitted or lost due to congestion. Students apply conditional probability and Bayes' theorem to determine the probability that packet loss occurred due to congestion given that a transmission error is detected. The analysis helps in understanding network reliability and performance tuning.
11. The waiting time of customers at a service counter (bank or hospital reception) follows a probability distribution. Students model the waiting time using exponential and normal distributions, calculate expected waiting time, and determine the probability that waiting time exceeds a given limit. The outcomes are used to recommend improvements in service efficiency.
12. An engineering team tests three different design configurations of a product and records performance measurements. Students use statistical inference tools (correlation, regression, hypothesis testing, and one-way ANOVA) in Excel or Python to analyze the data. Based on statistical results, they identify which design performs best and justify their conclusion scientifically.

Text Books:

1. J. L. Devore, "Probability and Statistics for Engineering and the Sciences", 9th ed., Boston, MA, USA: Cengage Learning, 2016.
2. D. P. Bertsekas and J. N. Tsitsiklis, "Introduction to Probability", 2nd ed., Belmont, MA, USA: Athena Scientific, 2008, ISBN: 978-1886529236.
3. S. M. Ross, "Introduction to Probability and Statistics for Engineers and Scientists", 5th ed., Amsterdam, Netherlands: Academic Press (Elsevier), 2014.
4. R. A. Johnson and G. K. Bhattacharyya, Miller and Freund's, "Probability and Statistics for Engineers", 9th ed., Harlow, U.K.: Pearson, 2017.

Reference Books:

1. R. E. Walpole, R. H. Myers, S. L. Myers and K. Ye, Probability and Statistics for Engineers and Scientists, 9th ed., Boston, MA, USA: Pearson, 2012.
2. C. R. Kothari and Gaurav Garg, *Research Methodology: Methods and Techniques*, 5th ed., New Delhi, India: New Age International Publishers, 2023.
3. W. W. Hines, D. C. Montgomery, D. M. Goldsman and C. M. Borror, Probability and Statistics in Engineering, 4th ed., Hoboken, NJ, USA: Wiley, 2003.
4. V. K. Rohatgi and A. K. Md. E. Saleh, An Introduction to Probability and Statistics, 2nd ed., New York, NY, USA: Wiley, 2001.
5. S. C. Gupta and V. K. Kapoor, Fundamentals of Mathematical Statistics, New Delhi, India: Sultan Chand & Sons, latest ed.

Moocs Links and additional reading material:

1. Probability and Statistics – IIT Kharagpur: https://onlinecourses.nptel.ac.in/noc23_ma83/preview
2. Engineering Statistics – IIT Bombay: https://onlinecourses.nptel.ac.in/noc23_ge25/preview?utm_source=chatgpt.com
3. Introduction to Probability – IIT Madras: https://onlinecourses.nptel.ac.in/noc25_mg35/preview?utm_source=chatgpt.com

Course Outcomes:

Student should be able to

1. Explain descriptive statistics, probability concepts, and random variables.
2. Apply discrete and continuous probability distributions to engineering problems.
3. Evaluate data using statistical inference methods, including estimation and hypothesis testing.
4. Analyze and interpret relationships in data using correlation and regression.

CO PO Map

co/po	po1	po2	po3	po4	po5	po6	po7	po8	po9	po10	po11	Pso 1	Pso2
CO:1	3	2			1						1	1	1
CO:2	3	3	1	1	1						1	2	2
CO:3	3	3	1	2	1				1		1	3	2
CO:4	3	3	2	2	2	1		1	1	2	3	3	3

CO attainment levels

CO	Attainment level
CO .1	
CO .2	
CO .3	
CO .4	

Future Courses Mapping:
Data Science, Machine Learning, AI & Neural Networks

Job Mapping:
Data Analyst, Quality Engineer, Machine Learning Engineer, Business Analyst, etc.