ZEAL COLLEGE OF ENGINEERING & RESEARCH, PUNE – 41

(An Autonomous Institute Affiliated to Savitribai Phule Pune University)

*NBA Accredited, NAAC Accredited with **A+** Grade / ISO 21001:2018*

DEPARTMENT OF COMPUTER ENGINEERING

Curriculum Structure and Syllabus of

S.Y. B. Tech. – COMPUTER ENGINEERING
(With effect from - Academic Year 2025- 26)

VISION OF THE INSTITUTE

To be a premier institute in technical education by imparting academic excellence, research, social and entrepreneurial attitude**.**

MISSION OF THE INSTITUTE

- To achieve academic excellence through innovative teaching and learning process.
- To imbibe the research culture for addressing industry and societal needs.
- To inculcate social attitude through community engagement initiatives.
- To provide conducive environment for building the entrepreneurial skills.

DEPARTMENT OF COMPUTER ENGINEERING

VISION:

To emerge as a department of repute in Computer Engineering through innovative teaching, research, social responsibility, and entrepreneurial skills, developing responsible IT professionals.

MISSION:

M1: To provide in depth technical education and hands-on experiences in Computer Engineering using modern tools and technologies.

M2: To endeavor innovative research culture to fulfill the needs of Industry and Society.

M3: To instill in students a deep sense of social responsibility.

M4: To strengthen collaboration between industry and academia, fostering the development of entrepreneurial skills among the students.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs):

PEO1: Graduate will apply knowledge of computer engineering to solve complex engineering problems, propose algorithmic solutions, thus establishing themselves as successful IT professional

PEO2: Graduate will exhibit leadership qualities and innovative thinking, contributing to the development of cutting edge solutions and Carrier advancement in the field of computer engineering through research, collaborative teamwork and entrepreneurial initiatives.

PEO 3: Graduate will maintain ethics, meet societal duties, and pursue life-long learning to stay updated and contribute meaningfully to their field and the society

PROGRAM OUTCOMES (POs):

PO1: **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2: **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3: **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4: **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5: **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6: **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7: **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8: **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9: **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10: **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11: **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12: **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs):

PSO1: Professional Skills-The ability to understand, analyze and develop computer programs in the areas related to algorithms, system software, multimedia, web design, big data analytics, and networking for efficient design of computer-based systems of varying complexities.

PSO2: Problem-Solving Skills- The ability to apply standard practices and strategies in software project development using open-ended programming environments to deliver a quality product for business success.

PSO3: Successful Career and Entrepreneurship- The ability to employ modern computer languages, environments and platforms in creating innovative career paths to be an entrepreneur and to have a zest for higher studies.

DEPARTMENT OF COMPUTER ENGINEERING

LIST OF ABBREVIATIONS

Abbreviation	Description
BSC	Basic Science Course
ESC	Engineering Science Course
PCC	Programme Core Course
PEC	Programme Elective Course
MDM	Multidisciplinary Minor
OE	Open Elective - Other than a particular program
VSEC	Vocational and Skill Enhancement Course
AEC	Ability Enhancement Course
ENTR	Entrepreneurship
EC	Economics
MC	Management Courses
IKS	Indian Knowledge System
VEC	Value Education Courses
RM	Research Methodology
CEP	Community Engagement Project
FP	Field Project
PROJ	Project
INT	Internship
OJT	On Job Training
CC	Co-curricular Courses
HSSM	Humanities Social Science and Management
ELC	Experiential Learning Course
B. Tech	Bachelor of Technology
L	Lecture
P	Practical
T	Tutorial
H	Hours
CR	Credits
CIE	Continuous Internal Evaluation
ETE	End Term Evaluation
TH	Theory
Tut	Tutorial
TW	Term Work
OR	Oral
PR	Project

DEPARTMENT OF COMPUTER ENGINEERING

Second Year B. Tech. – Computer Engineering: Semester – III

Course Code	Course Type	Course Name	Teaching Scheme (hrs/Week)							Evaluation Scheme					
			L	P	T	H	CR			CIE	ETE	TW	PR	OR	Total
							TH	PR/Tut	Total						
COPC302	PCC	Data Structures	3	4	-	7	3	2	5	40	60		25		125
COPC303	PCC	Processor Architecture and System Design	2	2	-	4	2	1	3	40	60	-	-	25	125
COPC304	PCC	Java Programming	2	-	-	2	2	-	2	40	60	-	-	-	100
COMD301	MDM	Discrete Mathematics	3	-	-	3	3	-	3	40	60	-	-	-	100
ALOE301	OE	Open Elective - I #	2	-	-	2	2	-	2	40	60	-	-	-	100
COMC301	HSSM-MC	Digital Marketing and Social Media	1	-	1	2	1	1	2	-	-	25	-	-	25
COVS302	VSEC	Java Programming Lab	-	4	-	4	-	2	2	-	-	25	25	-	50
COCE301	CEP	Project Based Learning	-	2	-	2		1	1	-	-	50	-	-	50
COIN302	ELC - INT	Internship - II	4 Weeks				-	2	2	-	-	25	-	-	25
Total			**13**	**12**	**1**	**26**	**13**	**9**	**22**	**200**	**300**	**125**	**50**	**25**	**700**

# - Select any one course from the given Open Elective Courses		
Course Code	**Course Type**	**Open Elective - I**
ALOE301A	OEC	Digital Literacy and Applications
ALOE301B		Environmental Studies
ALOE301C		Green Energy and Sustainability
ALOE301D		Basics of Consumer Electronics
ALOE301E		Renewable Energy Systems

BoS Chairman

ZES's Zeal College of Engineering & Research PU/PN/Engg./285/2007 Narhe, Pune-41

Director

ZES's Zeal College of Engineering & Research Narhe, Pune - 411041.

DEPARTMENT OF COMPUTER ENGINEERING

Second Year B. Tech. – Computer Engineering: Semester – IV

Course Code	Course Type	Course Name	Teaching Scheme (hrs/Week)							Evaluation Scheme					
			L	P	T	H	CR			CIE	ETE	TW	PR	OR	Total
							TH	PR/Tut	Total						
COPC405	PCC	Database Management Systems	3	2	-	5	3	1	4	40	60		25	-	125
COPC406	PCC	Computer Graphics & Visualization	3	2	-	5	3	1	4	40	60	-	-	25	125
COPC407	PCC	Software Engineering	2	-	-	2	2		2	40	60	-	-	-	100
COMD402	MDM	Probability & Statistics	3	-	-	3	3		3	40	60	25	-	-	125
ALOE402	OE	Open Elective -II #	2	-	-	2	2	-	2	40	60	-	-	-	100
COMC402	HSSM-MC	E-Commerce	1	-	1	2	1	1	2			25		-	25
COAE402	AEC	Quantitative Aptitude and Logical Reasoning	-	2	-	2	-	1	1			25		-	25
COVS403	VSEC	Python Programming Lab	-	4	-	4	-	2	2			25	25	-	50
COIN403	ELC - INT	Internship - III	4 Weeks				-	2	2			25		-	25
Total			**14**	**10**	**1**	**25**	**14**	**8**	**22**	**200**	**300**	**125**	**50**	**25**	**700**

# - Select any one course from the given Open Elective Courses		
Course Code	**Course Type**	**Open Elective - II**
ALOE402A	OEC	Cyber Security and Laws
ALOE402B		Sustainability and Climate Change
ALOE402C		Energy Audit and Electrical Safety
ALOE402D		Digital Marketing
ALOE402E		Entrepreneurship and Innovations

BoS Chairman

ZES's Zeal College of Engineering & Research PU/PN/Engg./285/2007 Narhe, Pune-41

Director

ZES's Zeal College of Engineering & Research Narhe, Pune - 411041.

DEPARTMENT OF COMPUTER ENGINEERING

INDEX

SYLLABUS
SEMESTER-III

DEPARTMENT OF COMPUTER ENGINEERING

Program: B. Tech. (Computer Engineering)						**Semester:** III			
Course: Data Structures						**Code:** COPC302			
Teaching Scheme				**Evaluation Scheme**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
3	4	-	5	40	60	-	-	25	125

Prerequisites:

Basic knowledge of C/C++ programming**,** basics of computer organization and memory representation**.**

Course Objectives:

1.Introduce the fundamental concepts of data structures, abstract data types, and memory management.
2.Develop the ability to implement and analyze searching, sorting, and hashing techniques for data manipulation.
3.Apply stack-based algorithms to solve computational and real-world problems.
4.Explore linked list structures for efficient dynamic memory-based data operations.
5.Implement queue variations to model scheduling and real-time applications.
6.Design and manipulate tree structures for hierarchical data representation and searching.

Course Outcomes: On completion of the course, students will be able to -

CO1	Understand abstract data types and arrays for solving structured data storage problems.
CO2	Apply and analyze searching, sorting, and hashing for efficient data management.
CO3	Apply stack operations for expression handling and real-life applications.
CO4	Differentiate and analyze singly, circular, and doubly linked lists.
CO5	Evaluate queue structures for scheduling and resource management.
CO6	Analyze tree structures and traversals for hierarchical data management

Course Contents:

Unit	**Description**	**Duration (Hrs.)**
1.	**Unit I: Introduction to Data Structures and Arrays**: • Introduction to Data Structures: Need and importance of data structures in problem-solving. Characteristics of a good algorithm. Difference between data types and data structures • Abstract Data Types (ADT), Classification: Linear vs Non-linear, Static vs Dynamic, Applications in Computer Engineering • Memory Allocation: Static vs Dynamic Memory Allocation • Array as a Data Structure: One-dimensional and Two-dimensional Arrays, Array Representation in Memory, Advantages and Limitations, Operations on Arrays (Insert, Delete, Traverse), Real-world Examples Using Arrays • Case Study: Simple Address Book or Student Record Manager using Arrays	7
2.	**Unit II: Searching, Sorting, and Hashing:** • Searching Techniques: Binary Search, Linear Search	8

	• Sorting Algorithms: Bubble Sort, Selection Sort, Insertion Sort, Merge Sort, Quick Sort • Hashing**:** Hash Tables, Hash Functions, Collision Resolution (Chaining, Linear Probing) • Case Study**:** Real-world usage of searching and sorting in databases and data processing, Inventory Lookup System using Hashing	

3.	**Unit III: Stacks**
	• Stack Definition and Operations (Push, Pop, Peek)
	• Array and Linked Representation
	  ○ Expression Conversion**:** Infix to Postfix, Infix to Prefix
	  ○ Expression Evaluation: Postfix Evaluation
	• Case Study: Web Browser Navigation (Back/Forward Stack)
		6

4.	**Unit IV: Linked Lists** • Types of Linked Lists**:** Singly, Circular, Doubly Linked List • Operations: Create, Insert, Delete, Traverse, Search, Sort, Concatenate • Case Study: Undo-Redo Functionality in Text Editors	**7**
5.	**Unit V: Queues** • Queue Definition and Operations (Enqueue, Dequeue, Peek) • Types of Queues**:** Linear, Circular, Priority, Deque, Linked Queue • Applications in Real-time Systems • Case Study: Task Scheduler using Priority Queue	7
6.	**Unit VI: Trees Introduction to Trees and Binary Trees** • Tree Representations: Sequential and Linked • Tree Operations: Insert, Search, Delete • Recursive Tree Traversals: Inorder, Preorder, Postorder • Binary Search Tree (BST): Properties and Operations • Case Study: Product Category Hierarchy	7
	TOTAL	**42**

Suggested List of Experiments:

1. Write a program to manage student attendance records (number of days present). Implement the following:
   a. Compute the average attendance of all students.
   b. Find the student with highest and lowest attendance.
   c. Count the number of students with zero attendance.
   d. Display the most common attendance count (mode).
   e. Also, determine the time and space complexity of each operation.

2. A hospital maintains patient records using unique patient IDs stored in a list. Write a program to:
   a. Linear Search: Check if a particular patient ID exists in the list.
   b. Binary Search: Use binary search to quickly locate a patient ID.

Display an appropriate message if the ID is not found

3. An E-Commerce website has a range of products having Product id, name, manufacturer, price and quality rating out of 5. Write a C++/Java Program to display products as…
    a. In Increasing order of Product id (Use Bubble Sort)
    b. In Increasing order of Product price (Use Selection Sort)
    c. In decreasing order of Product Quality Rating (Use Insertion Sort).

4. Library Book Management using Linked List. Use a singly/doubly linked list to maintain the library catalog. Book Structure: BookID, Title, Author, Availability Status
    a. addBook (): Add a new book to the list.
    b. removeBook (bookID): Remove a book using ID.
    c. searchBook (title/author): Search for a book by title or author.
    d. updateBook (bookID): Update details like availability.
    e. sortBooksByTitle/BookID (): Sort and display book records.

5. The ticket booking system of Theater has to be implemented using C++ program. There are M rows and N seats in each row. Doubly circular linked list has to be maintained to keep track of free seats at rows. Assume some random booking to start with. Use an array to store pointers (Head pointer) to each row. On demand
    a. The list of available seats is to be displayed.
    b. The seats are to be booked.
    c. The booking can be cancelled.

6. Implement a real-time browser navigation system using two stacks to simulate back and forward navigation. The system should support the following operations:
    a. Visit New Page: Open a new webpage and update the current state.
    b. Back Navigation: Go back to the previous page and store the current one for forward use.
    c. Forward Navigation: Move forward to the next page (if back was used before).
    d. Display Current Page: Show the currently open webpage.
    e. Display Navigation History: Show the back and forward stack contents for debugging or display.

7. Design a print spooling system where multiple print jobs are queued and processed in FIFO order using a queue data structure. The system should support:
    a. Add Print Job**:** A new document is added to the print queue.
    b. Process Next Job**:** The first job in the queue is sent to the printer and removed.
    c. Cancel Print Job**:** Cancel a specific print job if it's still pending.
    d. Display Print Queue: List all jobs currently waiting to be printed.
    e. Display Processed Jobs: Maintain and display a list of already printed documents (optional – can use a second queue or stack).

8. A call center receives incoming calls, and each call is assigned a unique customer ID. The calls are answered in the order they are received. Your task is to simulate the call queue of a call center using a queue data structure.
    a. addCall(customerID, callTime): Add a call to the queue with the customer ID and the call time (in minutes).
    b. answerCall(): Answer and remove the first call from the queue.
    c. viewQueue(): View all calls currently in the queue without removing them.
    d. isQueueEmpty(): Check if the queue is empty.

9. Managing Data with Binary Search Tree Operations. Design a C++ program that implements a Binary Search Tree (BST) supporting the following operations:
    a. Inserting a new element.
    b. Deleting an existing element.
    c. Searching for a specific value.
    d. Displaying the entire tree in-order, pre-order, or post-order.

10. Storing City Data in a Binary Search Tree. Maintain a list of cities and their populations using a Binary Search Tree. Implement functionalities in C++ to:
    a. Add a new city.
    b. Delete a city.
    c. Update the population of a city.
    d. Display city names in ascending and descending order.
    e. Compute the maximum number of comparisons needed to find a city in the tree (worst-case search complexity).

Text Books:

1. Horowitz and Sahani, "Fundamentals of Data Structures in C++", University Press 2007, 2nd Edition, ISBN: 978-07-1678-292-6.
2. Data structures and algorithms in python by Michael T. Goodrich, ISBN-13: 978- 1118290279, ISBN-10: 1118290275, Publisher: Wiley; 1st edition (March 18, 2013).
3. Problem Solving with Algorithms and Data Structures Using Python by Bradley N Miller and David L. Ranum. ISBN-13: 978-1590282571, ISBN-10: 1590282574, Publisher: Franklin, Beedle & Associates; 2nd edition (August 22, 2011).

Reference Books:

1. Brassard & Bratley, "Fundamentals of Algorithmics", Prentice Hall India/Pearson Education 1996, ISBN: 978-8120311312.
2. R. Gilberg, B. Forouzan, "Data Structures: A Pseudocode approach with C++", Cengage Learn 2005, 2nd Edition, ISBN 978-8131503140.
3. M. Weiss, "Data Structures and Algorithm Analysis in C++", 2nd edition, Pearson Education, 2002, ISBN: 978-0201498400.

E-Resources:

1. NPTEL course on, "Programming, Data Structures And Algorithms Using Python", By Prof. Madhavan Mukund : https://onlinecourses.nptel.ac.in/noc22_cs26/preview.
2. IITBombayX: Foundations of Data Structures: https://www.edx.org/learn/data-structures/iitbombay-foundations-of-data-structures

DEPARTMENT OF COMPUTER ENGINEERING

Program: B. Tech. (Computer Engineering)						**Semester:** III			
Course: Processor Architecture and System Design						**Code:** COPC303			
Teaching Scheme				**Evaluation Scheme**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
2	2	-	3	40	60	-	25	-	125

Prerequisites:

Fundamentals of Computer Systems and Networking

Course Objectives:

1. To understand the fundamental concepts of computer architecture and arithmetic.
2. To study CPU organization, instruction formats, and addressing modes.
3. To analyze register transfers, micro-operations, and control unit design.
4. To evaluate different memory devices and cache organization techniques.
5. To study I/O communication techniques and multiprocessor architectures.
6. To explore applications of computer architecture in modern computing environments including HCI.

Course Outcomes: On completion of the course, students will be able to -

CO1	Understand various computer architectures, data representation formats, and arithmetic algorithms.
CO2	Analyze CPU organization, instruction formats, and addressing modes
CO3	Apply register transfer language and micro-operations to design simple control unit operations.
CO4	Evaluate memory hierarchy and cache mapping techniques with respect to performance.
CO5	Compare I/O techniques and multiprocessor communication mechanisms.
CO6	Apply architectural concepts to real-world computing environments (HCI, Cloud, AI).

Course Contents:

Unit	**Description**	**Duration (Hrs.)**
1.	**Fundamentals of Computer Architecture & Arithmetic:** Advanced Von Neumann and Harvard Architectures, Working Principle of Computer, System Software and Performance Measures, Multiprocessors vs Multicomputer. **Data representation**: Fixed-point, Floating-point, Error Detection & Correction Codes. **Computer Arithmetic:** Addition, Subtraction, Multiplication, Division Algorithms, Floating-point Arithmetic, Decimal Arithmetic	4
2.	**Instruction Set Architecture & CPU Design:** Instruction Codes and Formats, Computer Registers and Instruction Cycle, Timing and Control, Memory-Reference Instructions, Input-Output & Interrupts. **CPU Organization**: Registers, Stack Organization	5

	Addressing Modes, Data Transfer and Manipulation. Timing and Control :Hardwired vs Microprogrammed. Instruction Set Architectures: CISC, RISC, VLIW	
3.	**Register Transfer & Control Unit:** Register Transfer Language, Bus and Memory Transfers Arithmetic, Logic, and Shift Micro-operations Arithmetic Logic Shift Unit (ALSU) **Control Unit Design:** Microprogrammed Control, Control Memory, Address Sequencing, Example Microprogram.	5
4.	**Memory Organization:** Memory Hierarchy Design Principles, Semiconductor Memories: RAM, ROM, Flash **Cache Memory:** Mapping Techniques (Direct, Associative, Set-Associative). Performance Considerations: Hit ratio, Access time, Replacement policies.	5
5.	**I/O and Multiprocessors:** I/O Interfaces: Programmed I/O, Memory-Mapped I/O, Interrupt-Driven I/O Direct Memory Access (DMA) – design and working Multiprocessor Architectures: Characteristics, Interconnection Structures. Inter-Processor Arbitration, Communication, Synchronization. Cache Coherence and Consistency Models.	5
6.	**Applications of Computer Architecture:** Physical, Social, and Cognitive Computing Environments Introduction to Human Computer Interaction (HCI), Use Cases of HCI: Healthcare, Education, Business, Smart Devices Role of Modern Architecture in Cloud, Edge, and AI-driven environments.	4
	TOTAL	**28**

Suggested List of Experiments:

1. Write a program (using simulator/assembly) to implement integer addition, subtraction, multiplication, and division using basic CPU instructions.
2. Implement floating-point arithmetic operations (addition, subtraction) using software routines and verify results.
3. Execute programs to demonstrate data transfer, logical, and bitwise operations.
4. Implement subroutine calls and returns to demonstrate flow control and modular programming.
5. Simulation of instruction execution pipeline vs sequential execution, measure and compare performance.
6. Implement a simple microprogrammed control sequence using simulator/software tools.
7. Simulation to demonstrate cache memory operation (direct-mapped, associative, or set-associative).
8. Interrupt-driven I/O simulation using CPU programming (keyboard/console-based).
9. Simulation of Direct Memory Access (DMA) data transfer.
10. Demonstrate use of a HCI for object classification using MATLAB
11. Case study / experiment on inter-processor communication or synchronization techniques.

12. Mini-project: Application of Human-Computer Interaction (e.g., design a simple UI with usability evaluation).
Text Books:
1. M. Moris Mano (2006), Computer System Architecture, 3rd edition, Pearson/PHI, India. 2. William Stallings (2010), Computer Organization and Architecture- designing for performance, 8th edition, Prentice Hall, New Jersy.
Reference Books:
1. Carl Hamacher, Zvonks Vranesic, SafeaZaky (2002), Computer Organization, 5th edition, McGraw Hill, New Delhi, India. 2. Anrew S. Tanenbaum (2006), Structured Computer Organization, 5th edition, Pearson Education Inc. 3. John P. Hayes (1998), Computer Architecture and Organization, 3rd edition, Tata McGrawHill
E-Resources:
1. Mooc **:** https://www.coursera.org/

DEPARTMENT OF COMPUTER ENGINEERING

Program: B. Tech. (Computer Engineering)						**Semester:** III			
Course: Java Programming						**Code:** COPC304			
Teaching Scheme				**Evaluation Scheme**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
2	-	-	2	40	60	-	-	-	100

Prerequisites:

Fundamentals of programming in C / C++, Basic understanding of OOP concepts

Course Objectives:

1. To introduce the principles of object-oriented programming using Java.
2. To explain classes, objects, constructors, and memory management in Java.
3. To develop understanding of inheritance, polymorphism, and packages.
4. To explore interfaces and exception handling for robust program design.
5. To apply multithreading and Java collections for concurrent and efficient applications.
6. To implement file handling and GUI programming for real-world software solutions.

Course Outcomes: On completion of the course, students will be able to -

CO1	Understand Java evolution, environment setup, data types, control structures, and arrays
CO2	Apply concepts of classes, objects, constructors, and encapsulation to develop basic Java programs
CO3	Analyze inheritance, polymorphism, and packages for code reusability and modularity.
CO4	Demonstrate abstraction using interfaces and evaluate exception-handling mechanisms in Java.
CO5	Implement multithreaded applications and compare collection framework utilities for different problem contexts.
CO6	Develop programs using file handling and GUI components to design interactive applications

Course Contents:

Unit	**Description**	**Duration (Hrs.)**
1.	**Java Evolution and Overview:** Fundamentals of Object Oriented Programming, Java Features, How Java Differs from C, C++, Java Environment : JDK, JRE and JVM , Simple Java Program, Constant, Variables and Data Types, Operator and Expression, Java's Control Statements : Selection statement (if and switch), Iteration Statements(while, do-while, for , Nested Loops), Jump Statements (Using break ,continue, return), Arrays (One Dimensional and Multi Dimensional).	4
2.	**Introduction to Java Classes:** Encapsulation in Java and Access Specifiers : Defining a Class, data members, methods, Creating Objects, Accessing class members, Constructors, Method Overloading, Using objects as parameters, returning an object, this keyword	5

	(Instance variable hiding), memory management, garbage collection, finalize () method, Static members, static methods and static blocks, Exploring the String Class, Using Command-Line Arguments, Use of getter and setter methods, Wrapper classes and the concept of boxing and unboxing in Java, Enumerated types.	
3	**Inheritance and Packages:** **Inheritance:** Defining a Subclass, Subclass Constructor, Types of inheritance (Single, Multilevel, Hierarchical) and Visibility Control, Polymorphism: compile-time and runtime, Method overriding, dynamic method dispatch Method, Use of Super Keyword, Use of final keyword, Abstract classes, The Object Class. **Packages**: Defining a Package, Finding Packages and CLASSPATH, Access Protection, An Access Example, Importing Packages.	5
4	**Interfaces and Exception Handling:** **Interfaces:** Concept of abstraction using abstract classes and interfaces: Defining an Interface, Implementing Interfaces, Applying Interfaces, Variables in Interfaces, Interfaces Can Be Extended **Exception Handling:** Types of exceptions – checked and unchecked. Try-catch-finally block, throw and throws keyword, Multiple catch clauses, Nested try Statements, Custom exceptions (creating Your Own Exception Subclasses, Chained Exceptions.	5
5	**Multithreaded programming and Collection framework in Java:** **Multithreading in Java:** Thread class, Runnable interface, thread lifecycle, thread methods, synchronization. **Java Collections:** List, Set, Map, Iterator, generics.	5
6	**File Handling and GUI programming:** **File I/O:** File handling using packages – reading/writing characters, bytes, and buffered streams. **GUI programming:** GUI using AWT/Swing: Components, Layouts, Event handling, Adapter classes, simple GUI applications.	4
	TOTAL	**28**

Text Books:

1. Java: The Complete Reference (11th Edition or latest),Author: Herbert Schildt, Publisher: McGraw Hill Education
2. Programming with Java, Author: E. Balagurusamy, Publisher: McGraw Hill Education

Reference Books:

1. Introduction to Java Programming (Comprehensive Version), Daniel Liang, Seventh Edition, Pearson.
2. Programming in Java, Sachin Malhotra & Saurabh Chaudhary, Oxford University Press.
3. Murach's Beginning Java 2, Doug Lowe, Joel Murach and Andrea Steelman, SPD.
4. Core Java Volume-I Fundamentals, Eight Edition, Horstmann & Cornell, Pearson Education.
5. The Complete Reference, Java 2 (Fourth Edition), Herbert Schild, TMH.

6. Java Programming, D. S. Malik, Cengage Learning.

E-Resources:

1. https://nptel.ac.in/courses
2. https://docs.oracle.com/javase/tutorial/

DEPARTMENT OF COMPUTER ENGINEERING

Program: B. Tech. (Computer Engineering)						**Semester:** III			
Course: Discrete Mathematics						**Code:** COMD301			
Teaching Scheme				**Evaluation Scheme**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
3	-	-	3	40	60	-	-	-	100

Prerequisites:

Basic knowledge of Sets, Functions, and Relations

Course Objectives:

1. To learn the basics of logic and proofs for solving problems in Computer Engineering.
2. To understand induction, recursion, and recurrence relations as tools for analysing algorithms.
3. To use counting methods and combinatory in designing algorithms and checking their complexity.
4. To study graphs, paths, and circuits and see how they are applied in real computing problems.
5. To learn about tree structures and how they are used in algorithms, data compression and optimization
6. To understand the basics of group theory and how it is applied in coding and cryptography.

Course Outcomes: On completion of the course, learner will be able to -

CO1	Understand propositional logic, truth tables, and mathematical proofs to construct valid logical arguments
CO2	Apply mathematical induction and recurrence relations to solve recursive problems.
CO3	Analyze counting principles, algorithms, and combinatorial methods for solving computational problems.
CO4	Demonstrate graph theory concepts and evaluate algorithms for shortest path, circuits, and graph coloring
CO5	Implement tree algorithms such as Huffman coding and spanning trees for practical applications.
CO6	Analyze algebraic structures (groups, rings, fields) and their applications in coding theory

Course Contents:

Unit	**Description**	**Duration (Hrs.)**
1.	**Introduction to Logic:** Significance of Discrete Mathematics in Computer Engineering, Application areas in Computer Engineering. Compound Statements, Proofs in Mathematics, Truth Tables, The Algebra of Propositions, Logical Arguments.	7
2.	**Induction and Recursion:** Mathematical Induction, Recursively Defined Sequences, Solving Recurrence Relations; The Characteristic Polynomial, Solving Recurrence Relations; Generating Functions.	6

3	**Principles of Counting and Algorithms:** The Principle of Inclusion- Exclusion, The Addition and Multiplication Rules, The Pigeon-Hole Principle Algorithms, Complexity, Searching and Sorting, Enumeration of Permutations and Combinations	7
4	**Graphs, Paths and Circuits and Applications:** A Gentle Introduction, Definitions and Basic Properties, Isomorphism, Eulerian Circuits, Hamiltonian Cycles, The Adjacency Matrix, Shortest Path Algorithms Dijkstra's Algorithm, Planar Graphs, Graph Colouring, The Chinese Postman Problem, Digraphs, Tournaments, Scheduling Problems.	8
5	**Trees and Searching:** Introduction of Tree, Properties of Trees, Huffman Algorithm for Optimal Tree, Spanning Trees, Minimum Spanning Tree Algorithms, Acyclic Digraphs and Bellman's Algorithm, **Case Study:** Applications of Trees in Computer Engineering.	7
6	**Group Theory:** Basic Properties of Group, Semigroup & Monoid, Abelian group, Subgroup, Normal subgroup**,** Groups and Coding**,** Rings, Integral Domain and Field. **Case Study:** Application of Group Theory in Computer Engineering.	7
	TOTAL	**42**

Text Books:

1. E.G. Goodaire & M.M Parmenter- Discrete Mathematics with Graph Theory, 2nd Edition Pearson Education, New Delhi - 2002.
2. C. L. Liu, "Elements of Discrete Mathematics", Tata McGraw-Hill, 4th Edition, 2017, ISBN 978-1259006395.
3. Kenneth H. Rosen, "Discrete Mathematics and its Applications"‖, Tata McGraw-Hill, ISBN 978- 0-07-288008-3
4. B. Kolman et.al- Discrete mathematical Structures, 5th Edition, Pearson Education, New Delhi - 2004.

Reference Books:

1. K.H. Rosen – Discrete Mathematics and Its Applications – 4th Edition, Tata McGraw Hill, New Delhi - 2001
2. D.B. West – Introduction to Graph Theory, 2nd Edition, Pearson Education, New Delhi 2002.
3. N.Deo – Graph Theory with Application to Engineering and Computer Science, PHI, New Delhi-2004.

E-Resources:

1. https://www.ebookphp.com/discrete-mathematical-structures-6th-edition-epub-pdf/
2. http://discrete.openmathbooks.org/pdfs/dmoi-tablet.pdf

DEPARTMENT OF COMPUTER ENGINEERING

Program: B. Tech. (Computer Engineering)						**Semester:** III			
Course: Digital Marketing and Social Media						**Code:** COMC301			
Teaching Scheme				**Evaluation Scheme**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
1	-	1	2	-	-	25	-	-	25

Prerequisites:

Basic knowledge of internet usage and browsing, Familiarity with communication tools such as email, blogs, or websites

Course Objectives:

1. To provide fundamental knowledge of digital marketing strategies and tools
2. To familiarize students with social media platforms and their business applications.
3. To introduce concepts of SEO, SEM, email marketing, and analytics.
4. To enable the design and execution of basic digital marketing campaigns.
5. To explore ethical practices and emerging trends in online marketing.

Course Outcomes: Upon successful completion of the course, students will be able to:

CO1	Understand the core concepts of digital marketing and its advantages over traditional marketing
CO2	Apply SEO and SEM techniques to enhance online visibility
CO3	Design social media strategies for different platforms (Facebook, Instagram, LinkedIn, etc.)
CO4	Create and manage email marketing and content marketing campaigns
CO5	Use tools like Google Analytics to measure marketing effectiveness
CO6	Analyze case studies to understand trends and challenges in the digital space

Course Contents:

Unit	**Description**	**Duration (Hrs.)**
1.	**Introduction to Digital Marketing:** Definition, need and importance; Comparison with traditional marketing; Types of digital marketing; Digital marketing ecosystem; Buyer's journey and conversion funnel.	3
2.	**Search Engine Optimization (SEO):** On-page and off-page SEO; Keywords, backlinks, meta tags; Tools: Google Search Console, SEMrush, Moz; SEO performance metrics.	2
3.	**Search Engine Marketing (SEM) and Paid Ads:** Google Ads, Bing Ads; Ad formats, keyword research, bidding strategies, PPC campaigns; Landing page design and optimization.	3
4.	**Social Media Marketing (SMM):** Overview of platforms (Facebook, Instagram, Twitter, LinkedIn); Creating content calendars, running ad campaigns, influencer marketing; Engagement metrics and KPIs.	2
5.	**Email and Content Marketing:** Email campaign creation, tools like Mailchimp;	2

	Email list building, segmentation; Blogging, video marketing, storytelling; Content writing tips and tools.	
6.	**Analytics and Trends in Digital Marketing:** Web analytics, Google Analytics basics; Conversion tracking, ROI measurement; Current trends: AI in marketing, chatbots, voice search, privacy laws (GDPR, etc.).	2
	TOTAL	14

Suggested List of Assignments: -

1. **SEO Audit Using Free Tools**
   Perform an SEO audit of a website using tools like Uber suggests Screaming Frog, or Google Search Console. Identify technical issues, keywords, and backlink data.
2. **Keyword Research and Content Optimization**
   Use Google Keyword Planner or Uber suggest finding keywords for a blog topic. Optimize sample content using SEO-friendly titles, meta tags, and keywords.
3. **Create and Simulate Google Ads Campaign**
   Design a sample Google Ads search campaign using mock billing. Choose keywords, write ad copy, set budget, and identify target audience.
4. **Design and Run a Facebook/Instagram Ad (Simulation)**
   Use Meta Business Suite to design a mock or actual ad campaign. Define objective, target demographics, ad design, and expected reach.
5. **Create a Social Media Content Calendar**
   Plan one week of social media posts for a chosen brand. Include post types, platforms, visuals, and caption samples using tools like Canva or Buffer.
6. **Create and Send Email Campaign Using Mailchimp**
   Design and send a sample marketing email using Mailchimp or similar tools. Include subject line, body, image, call-to-action, and analyze open/click-through rate.
7. **Blog Setup and Optimization**
   Create a blog using Blogger or WordPress. Write an SEO-optimized post including internal/external links, headings, and media.
8. **Google Analytics Simulation and Reporting**
   Set up a Google Analytics demo account and generate basic reports. Interpret metrics like bounce rate, session duration, and traffic sources.
9. **Build a Business Page on LinkedIn/Facebook**
   Create a professional brand page for a business/service. Add cover image, description, services, and one sample post.
10. **Reputation Management Simulation**
    Track brand mentions online using Google Alerts or Brand24 (trial). Write a brief strategy to handle a mock reputation crisis.

11. **Case Study: Successful Digital Campaign Analysis** Analyze a famous digital campaign (e.g., Amul, Zomato, Swiggy, Nike). Present objectives, tools used, target audience, campaign creatives, and success factors.
12. **Mini Project: Design a Complete Digital Strategy** Choose a product/service and create a mini digital strategy. Include SEO plan, ad sample, social calendar, email draft, analytics dashboard (mock), and campaign goal. Submit final report with screenshots and justification of platform choices.
Text Books:
1. Seema Gupta, *Digital Marketing*, McGraw Hill Education 2. Ian Dodson, *The Art of Digital Marketing*, Wiley
Reference Books:
1. Philip Kotler, Hermawan Kartajaya, Iwan Setiawan, *Marketing 4.0: Moving from Traditional to Digital*, Wiley 2. Ryan Deiss & Russ Henneberry, *Digital Marketing for Dummies*, Wiley 3. Avinash Kaushik, *Web Analytics 2.0*, Wiley
E-Resources:
1. Google Digital Garage: https://learndigital.withgoogle.com 2. HubSpot Academy: https://academy.hubspot.com 3. Coursera – Digital Marketing Specialization: https://www.coursera.org/specializations/digital-marketing 4. Neil Patel Blog: https://neilpatel.com

DEPARTMENT OF COMPUTER ENGINEERING

Program: B. Tech. (Computer Engineering)								**Semester:** III	
Course: Java Programming Lab								**Code:** COVS303	
Teaching Scheme				**Evaluation Scheme**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
-	04	-	02	-	-	25	-	25	50

Prerequisites:

Basic understanding of C/C++, Knowledge of fundamental programming concepts such as loops, conditionals, and functions

Course Objectives:

1. To introduce object-oriented programming concepts using Java
2. To develop Java programs using classes, objects, inheritance, and interfaces
3. To implement exception handling, multithreading, and file I/O
4. To use GUI components for interactive programming
5. To apply object-oriented design in solving real-life problems

Course Outcomes: Upon successful completion of the course, students will be able to:

CO1	Demonstrate understanding of object-oriented principles using Java
CO2	Develop modular programs using classes, objects, and constructors
CO3	Implement inheritance, interfaces, polymorphism, and abstraction in Java
CO4	Apply exception handling, file I/O, and multithreading in Java programs
CO5	Design basic GUI applications using AWT and Swing
CO6	Build mini-projects using multiple Java concepts in real-world applications

Suggested List of Practicals:

1. Write a program to develop calculator for arithmetic operations (Use concept of switch case, do-while loop)
2. Write a program to count number of even and odd elements in an array and display the counts. Accept values of any array elements from the user. (Use concept of for loop, if -else statements, Single dimension array)
3. **Matrix operations**: Write a program to perform addition of two matrices. Accept values for the matrix elements from the user. Students can also perform matrix multiplication program. (Use concept of nested loop, Two dimensional array and you can use command line arguments to to provide values for matrix elements)
4. **ATM Machine Simulation (Custom Exception):** Develop a Java program to simulate an ATM machine. Implement operations like withdrawal, deposit, and balance inquiry. Throw a custom exception InsufficientBalanceException if the user tries to withdraw more than the balance. Demonstrate checked and unchecked exceptions. Also display the count of the users who used ATM to perform transaction. (Use concept of Encapsulation (Class, objects, methods) for creation of Account class, static data member, static block and static method).

5. **String based operations:** Develop a program to demonstrate the use of String class and Boxing concepts (Boxing and Unboxing).
6. **Vehicle Management System**: Design a Java application to manage different types of vehicles. Create a base class Vehicle with subclasses Car, Bike, and Truck. Demonstrate method overriding for displaying vehicle details. Use packages to organize the classes. Implement dynamic method dispatch while processing vehicles. (Use concept of Inheritance, Method overriding , Dynamic method dispatch and Abstract class and packages)
7. **Electricity Bill Generator**: Write a Java program to generate electricity bills for different types of customers – Domestic and Commercial. Define customer categories. Apply concept of Interfaces to calculate bills based on tariff rules. Handle exceptions for invalid unit entries. (Use concept of Interfaces and exception handling)
8. Write a program to create threads using Runnable interface and Thread class. Demonstrate the use of various methods of the Thread class (Methods for Thread priorities , join etc).
9. **Share market/ Stock Application or Movie ticket booking Application**: Develop a multithreaded application to purchase shares of a company for the multiple buyers. (Use concept of multithreading and synchronization)
10. **Library Management System**: Develop a library management application where users can add, delete, search, and update books. Use Vector or ArrayList and Hashmap from the collection framework to manage records. Accept required input/data from the user. (Use concept of collection framework)
11. Write a program to demonstrate various file I/O operations.
12. **Student Result Processing with GUI**: Design a GUI-based application where students can enter their marks through text fields. The program should calculate total marks, grade, CGPA, and rank among classmates and save report cards into a file. (Use concept of file handling and GUI)
13. **Mini Project:** Online Examination System: Develop a Java program to conduct a simple online test. Create classes for different types of questions (MCQ, True/False). Use inheritance and polymorphism to evaluate answers. Apply exception handling to manage invalid inputs during the test. Also provide GUI for better user interface.

Text Books:

1. Herbert Schildt, *Java:* The Complete Reference, McGraw Hill Education
2. E. Balagurusamy, Programming with Java, McGraw Hill Education

Reference Books:

1. Paul Deitel & Harvey Deitel, Java: How to Program, Pearson Education
2. Cay S. Horstmann, Core Java Volume I – Fundamentals, Pearson Education

E-Resources:

1. NPTEL Java Programming: https://nptel.ac.in/courses/106/105/106105191/
2. Oracle Java Tutorials: https://docs.oracle.com/javase/tutorial/
3. GeeksforGeeks – Java: https://www.geeksforgeeks.org/java/

DEPARTMENT OF COMPUTER ENGINEERING

Program: B. Tech. (Computer Engineering)								**Semester:** III	
Course: Project Based Learning								**Code:** COCE301	
Teaching Scheme				**Evaluation Scheme**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
-	02	-	01	-	-	50	-	-	50

Prerequisites:

Basic understanding of social and ethical responsibilities, Teamwork and communication skills, Familiarity with problem-solving methodologies and project planning, Conversation in local language

Course Objectives:

1. To sensitize students to local societal issues and the role of engineers in addressing them.
2. To expose students to real-life field experiences through community immersion.
3. To develop critical thinking, communication, teamwork, and empathy.
4. To encourage students to apply technical knowledge to solve or address real-world challenges.
5. To foster civic responsibility, ethical engagement, and environmental awareness.

Course Outcomes:

On completion of the course, learner will be able to -

CO1	Identify and understand key socio-environmental problems in local communities.
CO2	Engage with communities, authorities, or NGOs to plan meaningful interventions.
CO3	Implement an awareness, data-collection, or solution-building activity through teamwork.
CO4	Document activities, analyze impact, and reflect on the role of technology in society.
CO5	Demonstrate teamwork, communication skills, and social sensitivity through presentation.

Course Contents:

Implementation:

- A group of 3 to 4 students could be assigned for a particular habitation or village or municipal ward, as far as possible, in the near vicinity of their place of stay/college premise.
- Each group /practical batch is allotted to a faculty member of the department as a mentor.
- A division of 60 students can have 3 batches of minimum 20 students. Practical load of 4 hours to be allocated to each batch.
- The group of students will be associated with a government official / village authority/NGOs etc. concerned, allotted by the district administration, during the duration of the project.
- The Community Engagement Project should be different from the regular programmes of NSS/NCC/Green Club/Hobby Clubs, Special Interests Groups etc
- An activity book has to be maintained by each of the students to record the activities undertaken/ Involved and will be countersigned by the concerned mentor.
- Project report shall be submitted by each student/group of students.
- An internal evaluation shall also be conducted by a committee constituted by the HoD. Evaluation to be done based on the active participation of the student and marks could be awarded by the

mentor/HoD.

- Students groups can conduct an awareness programme on Health and Hygiene or in Organic Farming or in Fisheries or in advocating prohibition of liquor or about renewable energy, ewaste management or any other activity in an area of their studies and as per his/her aptitude.
- Oral Examination shall consist of presentation and demonstration of the project work carried out by the project groups.

Suggested list of topics:

The below lists are not exhaustive and open mentors to add, delete or modify. It is expected that the focus should be on specific local issues in their nearby areas. The students are expected to carry out these projects with involvement, commitment, responsibility and accountability. The mentors of a student/group of students shall:

A. **Digital Literacy & Safe Technology Use:**

1. **Mobile Misuse Awareness in Schools**
   - Conduct awareness sessions on mobile addiction and its impact.
   - Create a short Android app (using MIT App Inventor) for screen-time tracking and healthy usage tips
2. **Cyber Safety & Digital Hygiene for Homemakers and Seniors**
   - Hands-on workshop on how to use WhatsApp, UPI, and avoid online scams.
   - Design a pictorial handbook in local language on safe digital practices.
3. **Career Orientation of Rural Youth**
   - Organize seminars with role models, coding challenges, or aptitude quiz sessions in nearby schools.
   - Design a simple career guidance website in regional language.

B. **Health, Hygiene & Nutrition:**

4. **Hygiene Audit in Government Schools**
   - Use Google Forms or KoboToolbox to survey hygiene practices.
   - Make a report with charts and a presentation for school authorities.
5. **Awareness Campaign on Food Adulteration**
   - Demonstrate simple detection methods for milk, oil, spices, etc.
   - Create a video/tutorial and post it on social platforms for community sharing.
6. **Health Awareness for Housewives & Elders**
   - Conduct sessions on lifestyle diseases (diabetes, blood pressure).
   - Develop a local-language brochure with preventive tips.

C. **Environmental Awareness & Sustainability**

7. **Air/Water Pollution Survey**
   - Use mobile sensors or low-cost Arduino kits to measure PM2.5 / water TDS levels.
   - Develop a pollution dashboard using Google Sheets or a basic web app.
8. **Renewable Energy Awareness**
   - Organize demo sessions in housing societies on solar panel usage and cost-saving.
   - Create a cost-benefit calculator spreadsheet or website for common users.

9. Tree Plantation with IoT Monitoring

- Collaborate with local panchayat to plant saplings.
- Add moisture sensor + ESP8266-based IoT to monitor watering needs.

D. Nutrition & Local Practices:

10. Herbal Medicine Awareness (Home Remedies App)

- Interview elderly residents about herbal or natural remedies.
- Create a searchable mini-app or website with content and images.

11. Nutritional Awareness in Children

- Conduct "fun + food" camps to explain balanced diet.
- Build a quiz game (in Scratch or HTML/CSS/JS) to teach food groups.

E. Women Empowerment & Education

12. Basic Computer & Internet Training for Women

- Conduct classes on Google Forms, MS Word, Email creation.
- Help them set up resumes or e-commerce accounts (e.g., Meesho, Amazon).

13. Women in Tech Talks

- Organize motivational sessions featuring women in engineering, coding, or entrepreneurship.
- Record and publish it on YouTube or college website.

DEPARTMENT OF COMPUTER ENGINEERING

Program: B. Tech. (Computer Engineering)						**Semester:** III			
Course: Internship-II						**Code:** COIN302			
Teaching Scheme				**Evaluation Scheme**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
-	-	-	02	-	-	25	-	-	25

Preamble:

Internships serve as vital educational and career development experiences, offering practical exposure in a specific field. Employers seek individuals who possess the necessary skills and an understanding of industry environments, practices, and cultures. This internship is designed as a structured, short-term, supervised training program, often centered on specific tasks or projects with clear timelines. The primary goal is to immerse technical students in an industrial setting, providing experiences that cannot be replicated in the classroom. This exposure aims to develop competent professionals who understand the social, economic, and administrative factors influencing the operations of industrial organizations

Course Objectives:

1. Exposure to students to the industrial environment, which cannot be provided in the classroom and hence creating deployable professionals for the industry.
2. Learn to implement the technical knowledge in real industrial situations.

Course Outcomes: After completion of this course, students will be able to -

CO1	Gain exposure to industry practices and understand how academic concepts are applied in professional settings.
CO2	Develop and demonstrate effective communication and teamwork skills within a work environment.
CO3	Improve your problem-solving and time management skills by working in real-world industry settings.

Internship Requirements:

1. **Internship Duration:** It is mandatory for all students to undergo an internship after every semester during vacations for the duration of 4 weeks. Internships completed during this period will be considered for the assessment of Term Work (TW).
2. **Internship Opportunities**: Students can explore various opportunities for internships at: a. Industries b. Research labs or organizations c. Collegiate clubs d. In-house research projects e. Online internships
3. **Support and Assistance**: Students can seek assistance for securing internships from: a. The Training and Placement cell, along with departmental coordinators b. Department or institute faculty members c. Personal contacts d. Directly connecting with industries or organizations
4. **Request Letter**: Once an industry, research organization, or collegiate club is identified, students must obtain a request letter from the concerned department or placement office. This letter, in the standard format must be duly signed by the authority, should be addressed to the HR manager or relevant authority.

5. **Confirmation Letter:** Students must submit the confirmation letter from the industry, research organization, or collegiate club to the Internship Coordinator and the Head of Department (HOD) office.
6. **Joining Report:** Upon commencing the internship, students must submit the joining report, joining letter, or a copy of the confirmation email to the Internship Coordinator and the HOD office.
7. **Faculty Mentor**: A faculty member will be assigned as a mentor to a group of students. The mentor will be responsible for monitoring, evaluating, and assessing student internship activities. The faculty mentor is also required to visit the internship location and submit formal feedback to the Internship Coordinator.
8. **Faculty Visits**: Faculty members are advised to visit the internship site once or twice during the internship period to monitor progress.
9. **Progress Report:** Students must submit progress report fortnightly to their faculty guide and the final internship report to the Internship Coordinator and department office.
10. **Evaluation Report**: After the completion of the internship, the mentor, along with the assessment panel members, should submit the evaluation report of the students to the department office and the Internship Coordinator.
11. **Internship Certificate:** Students must receive the Internship Certificate from the industry and submit it to the Internship Coordinator and department office.
12. **Presentation and Assessment:** Students are required to give a presentation on their internship work as part of the term work. The internship diary and report will also be verified and assessed.

SYLLABUS
SEMESTER-IV

DEPARTMENT OF COMPUTER ENGINEERING

Program: B. Tech. (Computer Engineering)							**Semester:** IV		
Course: Database Management Systems							**Code:** COPC405		
Teaching Scheme				**Evaluation Scheme**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
3	2	-	4	40	60	-	-	25	125

Prerequisites:

Fundamentals of data representation and data structures, Basic programming concepts

Course Objectives:

1. Introduce fundamental concepts of databases and DBMS architecture.
2. Develop ability to design relational databases using ER/EER models and normalization.
3. Apply SQL to manage and query databases effectively.
4. Implement advanced SQL features, PL/SQL constructs, and database programming.
5. Explain transaction management, concurrency control, and recovery techniques.
6. Explore NoSQL databases, their applications, and relevance in big data environments.

Course Outcomes: On completion of the course, students will be able to -

CO1	Understand DBMS concepts, architectures, data models, and applications by comparing with traditional file systems.
CO2	Design and analyze relational databases using ER/EER models, keys, functional dependencies, and normalization up to BCNF.
CO3	Apply SQL commands (DDL, DML) and construct queries using selection, sorting, aggregate functions, and subqueries for data retrieval.
CO4	Develop and implement advanced SQL queries, PL/SQL blocks, procedures, functions, cursors, and triggers for database applications.
CO5	Examine and evaluate database transactions with ACID properties, concurrency control methods, and recovery techniques.
CO6	Differentiate and justify the use of NoSQL databases over traditional RDBMS for handling large-scale, unstructured, and graph-based data.

Course Contents:

Unit	**Description**	**Duration (Hrs.)**
1.	**Introduction to Database Management System:** **Database concepts**: Data, Information, Metadata, File system vs. DBMS, Characteristics of DBMS, Database users and administrators **DBMS architecture**: 2-tier, 3-tier **Data models**: Hierarchical, Network, Relational, Object-oriented DBMS components, Advantages and applications of DBMS **Case Study:** Library System – Compare how a library manages books, members, and borrowing records using a file system vs. a DBMS. Discuss issues like redundancy, consistency, and data security.	5

2.	**Relational Database Design:** **Relational model:** Attributes, tuples, relations, schemas **Keys**: Primary, Candidate, Super, Foreign **Entity-Relationship (ER) model**: Entities, relationships, attributes **Enhanced ER (EER) model:** Specialization, generalization, aggregation Mapping ER/EER to relational model, **Normalization**: 1NF, 2NF, 3NF, BCNF Functional dependencies **Case Study:** *Student Course Registration System* – Create an ER diagram for students, courses, and faculty. Convert it into relations and normalize it up to 3NF.	7
3.	**SQL Queries:** Introduction to SQL, Data types and table creation (DDL), Insertion, updation, deletion of records (DML) **Basic queries**: Select, Where, Order By **Aggregate functions**: Count, Sum, Avg, Min, Max, Simple subqueries **Case Study:** *Hospital Management* – Write SQL queries to list patients admitted in the last 7 days, count the number of doctors in each department, and find patients with bills greater than a given amount.	7
4.	**Advanced SQL Queries and PL/SQL:** **Advanced SQL:** • Joins (Inner, Outer, Self, Cross) • Group By And Having Clause • Set operations (Union, Intersect, Minus) • Views, Indexes • DCL (Grant, Revoke), TCL (Commit, Rollback, Savepoint) **PL/SQL:** • Introduction and advantages • Control structures (If, Loop, While) • Cursors • Procedures and Functions • Triggers **Case Study:** *E-Commerce System* – Write queries to find top 5 customers based on purchase amount, create a view for monthly sales, design a trigger to update stock after a sale, and implement a stored procedure to calculate discounts.	9
5.	**Database Transactions:** Concept of transaction, ACID properties, Transaction states, Schedules: Serial, non-serial, Concurrency control: Lock-based, timestamp-based, Deadlock handling and prevention, Recovery techniques: Log-based, checkpoints **Case Study:** *Banking System* – Analyze a money transfer transaction (withdrawal from one account and deposit into another). Identify how ACID properties ensure consistency. Show what happens if a failure occurs in the middle of the transaction.	7
6.	**NoSQL Databases:** • Introduction to NoSQL	7

	• Characteristics and need for NoSQL • Types: Key-value, Document, Column-oriented, Graph • SQL vs. NoSQL comparison • Applications (MongoDB, Cassandra, Neo4j) **Case Study:** *Social Media Platform* – Model user profiles, posts, and friend relationships using a graph database (Neo4j). Discuss why NoSQL is more suitable than relational DB for handling millions of posts and connections.	
	TOTAL	**42**

Suggested List of Experiments:

1. **Real-Time Application Design with ER Modeling**
   A university wants to computerize its **Library Management System** to handle books, members and transactions.
   - Identify entities, relationships, attributes, keys, and cardinalities.
   - Draw an ER diagram using a tool like **ERD Plus**.
   - Convert the ER model into relational tables and normalize them up to **3NF**

2. **DDL Operations in SQL**
   A company maintains employee records. Create a database CompanyDB with tables **EMPLOYEE** and **DEPARTMENT**.
   - Use **DDL commands** (Create, Alter, Drop, Rename, Truncate) to design the schema.
   - Insert data and apply constraints: Not Null, Unique, Primary Key, Foreign Key, Check.

3. **Data and Access Control with SQL**
   A bank manages **Customer Accounts**.
   - Use **DML commands** (Insert, Update, Delete) to maintain account records.
   - Implement **DCL commands** (Grant, Revoke, Role) for access control.
   - Apply **TCL commands** (Commit, Rollback, Savepoint) to handle transactions like deposits and withdrawals.

4. **SQL Computation and Functions**
   In a **Hospital Management System**, maintain patient records.
   - Write queries using arithmetic and logical operators.
   - Use SQL functions (e.g., Length, Substr, Concat, Round, Now).
   - Perform operations using In, Not In, And Pattern Matching (Like).

5. **Grouping and Views in SQL**
   A retail store maintains **SALES** records.
   - Use **GROUP BY** and **HAVING** to display monthly sales statistics.
   - Write queries with **EXISTS/NOT EXISTS**.
   - Create and use **Views** (e.g., high-value customers, daily sales summary).
6. **Joins and Subqueries in SQL**

In a **Student Course Registration System**, maintain STUDENT, COURSE, and ENROLLMENT tables.

- Perform **Inner Join, Outer Join, Self Join, Cross Join** queries.
- Write subqueries to list students enrolled in more than 3 courses.
- Demonstrate set operators (Union, Intersect, Minus).

7. **Stored Procedure and PL/SQL Block**
   A university maintains **Exam Results**.
   - Write a **Stored Procedure** proc_Grade to categorize students as *Distinction*, *First Class*, *Higher Second Class* based on marks.
   - Create a **PL/SQL block** to call this procedure using Stud_Marks(name, total_marks) and store results in Result(Roll, Name, Class).

8. **Triggers for Salary Tracking**
   A company maintains **Employee Salary Records**.
   - Implement **BEFORE** and **AFTER triggers** to track salary updates.
   - Maintain total employee count and expenses in COMPANY_INFO.
   - Log salary changes in EMP_LOG with timestamp and status.

9. **Parameterized Cursor for Merging Data**
   A college maintains **Roll Call Data** in two tables N_RollCall and O_RollCall.
   - Write a **PL/SQL block** using parameterized cursor to merge data from N_RollCall into O_RollCall.
   - Skip duplicates if the data already exists.

10. **CRUD Operations in MongoDB**
    An institute maintains **Teacher Records**.
    - Create a **teachers** collection with details: name, qualification, department, salary, joining date, expertise.
    - Perform **CRUD operations** (insert, update, delete, retrieve).
    - Use comparison and logical operators in queries.

11. **MongoDB Aggregation and Indexing**
    In a **Shopping Application**, maintain orders placed by customers.
    - Use **aggregation** to calculate total sales per customer.
    - Create **indexes** on customer names and order IDs for faster retrieval.

12.MongoDB Map-Reduce
In a **Social Media Platform**, maintain posts and likes.
- Implement a **Map-Reduce operation** to count the number of posts liked by each user.

13. **Java MySQL Connectivity**
    Develop a **Java program** to connect with StudentDB in MySQL.
    - Implement CRUD operations (add, delete, update, display).
    - Demonstrate database navigation using JDBC.

Text Books:

1. SilberschatzA.,KorthH.,SudarshanS.,"DatabaseSystemConcepts",McGrawHill Publishers,7th Edition, 2020 ISBN 978-0-07-802215-9.
2. Ivan Bayross, "SQL, PL/SQL the Programming Language of Oracle", BPB Publications, 2014 ISBN: 9788176569644.
3. Connally T, Begg C.,"DatabaseSystems-A Practical Approach to Design, Implementation and Management", Pearson Education, 5th Edition, 2010, ISBN 81-7808-861-4.
4. Pramod J. Sadalage and Martin Fowler, "NoSQL Distilled", Addison Wesley, ISBN 10: 0321826620, 2013, ISBN 13: 978-0321826626.

Reference Books:

1. C. J. Date, "An Introduction to Database Systems", Addison-Wesley, 8th Edition, 2004, ISBN 0321189566.
2. S. K. Singh, "Database Systems: Concepts, Design and Application", Pearson Education, 2009,ISBN 9788177585674.
3. Kristina Chodorow, Michael Dierolf, "MongoDB: The Definitive Guide", O"Reilly Publications, 3rd Edition, 2019 ISBN 9781491954461.
4. Kevin Roebuck, "Storing and Managing Big Data - NoSQL, HADOOP and More", Emereo Pty Limited, 2011,ISBN. 1743045743, 9781743045749.

E-Resources:

1. NPTEL – Database Management Systems: https://nptel.ac.in/courses/106/105/106105175/
2. GeeksforGeeks – DBMS: https://www.geeksforgeeks.org/dbms/
3. Oracle SQL Tutorial: https://docs.oracle.com/en/database/

DEPARTMENT OF COMPUTER ENGINEERING

Program: B. Tech. (Computer Engineering)						**Semester:** IV			
Course: Computer Graphics & Visualization						**Code:** COPC406			
Teaching Scheme				**Evaluation Scheme**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
3	2	-	4	40	60	-	25	-	125

Prerequisites:

Basic knowledge of programming, mathematics concepts, Familiarity with basic computer architecture

Course Objectives:

1. Introduce the fundamentals of computer graphics and its applications.
2. Explain algorithms for raster graphics and drawing.
3. Develop understanding of 2D and 3D transformations.
4. Demonstrate concepts of windowing, clipping, and projections.
5. Apply techniques of curves, lighting, shading, and rendering.
6. Explore visualization methods and recent trends in computer graphics.

Course Outcomes: On completion of the course, students will be able to -

CO1	Describe the basic concepts, file formats, graphics standards, and system architecture used in computer graphics.
CO2	Implement line, circle drawing, and polygon filling algorithms for raster graphics.
CO3	Apply and analyze 2D/3D transformations using matrix representation and homogeneous coordinates to manipulate graphical objects.
CO4	Demonstrate and analyze windowing, clipping algorithms, and projection techniques for graphical scenes.
CO5	Apply and evaluate curves, shading models, and basic rendering techniques to generate realistic graphics.
CO6	Examine and evaluate visualization techniques and emerging technologies like VR/AR, GPU rendering, and AI in graphics.

Course Contents:

Unit	**Description**	**Duration (Hrs.)**
1.	**Basics of Computer Graphics:** • Introduction & applications (engineering, gaming, multimedia). • Coordinate systems. • Graphics file formats: BMP, GIF, JPEG, TIFF, PCX (overview only). • Graphics functions & standards (modes, shapes, colors). • Graphics devices and pipeline (basic overview). *Case Study*: Graphics in CAD/CAM.	6

2.	**Raster Graphics and Drawing Algorithms:** • Output primitives: point, line, circle, polygon. • Line drawing: DDA, Bresenham. • Circle generation: Bresenham. • Polygon filling: Flood fill, Boundary fill, Scan-line. *Case Study*: Simple paint/drawing tool.	7
3.	**2D and 3D Transformations:** • Basic Transformations: Translation, Scaling, Rotation. • 2D transformations: translation, scaling, rotation, reflection, shear. • Homogeneous coordinates & matrix representation. • Composite transformations. • 3D transformations: translation, scaling, rotation, reflection, shear. • Viewing pipeline (basic). *Case Study*: Rotating cube.	8
4.	**Windowing, Clipping & Projections:** • Windowing concepts. • Line clipping: Cohen–Sutherland, Midpoint. • Polygon clipping: Sutherland–Hodgman. • Projections: parallel & perspective. *Case Study*: Viewport in graphics editor.	7
5.	**Curves, Lighting & Rendering:** • Curves: Bezier, B-spline. • Basics of lighting: ambient, diffuse, specular. • Shading: flat, Gouraud, Phong. • Texture mapping (intro). *Case Study*: Shaded 3D object.	7
6.	**Visualization & Emerging Trends:** • Data visualization: charts, heatmaps, dashboards. • Tools overview: Matplotlib, D3.js, Tableau. • Emerging trends: i. VR & AR ii. Game Engines (Unity, Unreal) iii. GPU & real-time rendering (ray tracing) iv. Web-based graphics (WebGL, Three.js) v. AI in graphics (generative art, image synthesis) *Case Study*: AR/VR in education & gaming.	7
	TOTAL	**42**

DEPARTMENT OF COMPUTER ENGINEERING

Suggested List of Experiments:

1. Write a program in C++ or Java to
   a) Draw a single pixel (a single pixel)
   b) Draw a simple straight line between two points.
   c) Draw a circle using basic graphics functions.
2. Line Drawing Algorithms
   - Implement DDA Line Drawing Algorithm.
   - Implement Bresenham's Line Drawing Algorithm.
3. Circle Drawing Algorithm
   - Implement Midpoint Circle Algorithm**.**
4. Polygon Filling
   - Implement Flood Fill OR Boundary Fill Algorithm**.**
5. 2D Transformations
   - Apply Translation, Scaling, and Rotation on a triangle/rectangle.
6. Clipping:
   - Implement Cohen–Sutherland Line Clipping Algorithm.
7. Curves
   - Implement a Bezier Curve.
8. Basic Animation (OpenGL or graphics.h)
   - Demonstrate a simple bouncing ball OR rotating fan**.**

Text Books:

1. S.Harrington, "Computer Graphics" 2nd Edition, McGraw-HillPublications, 1987,ISBN –07 – 472 – 6.
2. Donald D.Hearnand Baker,"Computer Graphics with OpenGL",4thEdition,ISBN-13: 9780136053583.
3. D.Rogers,"Procedural Elements for Computer Graphics",2nd Edition, Tata McGraw-Hill Publication, 2001, ISBN 0–07 –047371–4.
4. D. Hearn, M. Baker, "Computer Graphics – C Version", 2nd Edition, Pearson Education, 2002, ISBN81 – 7808 – 794 – 4
5. Grigore C. Burdea, Philippe Coiffet, "Virtual Reality Technology", second edition, Wiley India Edition, ISBN 81-265-0789-6

Reference Books:

1. J. Foley, V. Dam, S. Feiner, J. Hughes, "Computer Graphics Principles and Practice", 2nd Edition, Pearson Education, 2003, ISBN 81 – 7808 – 038 – 9.
2. Foley, "Computer Graphics: Principles & Practice in C", ISBN 9788131705056, Pearson Edu.
3. F.S. Hill JR, "Computer Graphics Using Open GL", Pearson Education

E-Resources:

1. https://open.umn.edu/opentextbooks/textbooks/introduction-to-computer-graphics
2. http://www2.cs.uidaho.edu/~jeffery/courses/324/lecture.html
3. https://nptel.ac.in/courses/106/106/106106090/
4. https://nptel.ac.in/courses/106/102/106102065/

DEPARTMENT OF COMPUTER ENGINEERING

Program: B. Tech. (Computer Engineering)						**Semester:** IV			
Course: Software Engineering						**Code:** COPC407			
Teaching Scheme				**Evaluation Scheme**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
2	-	-	2	40	60	-	-	-	100

Prerequisites:

Familiarity with software development fundamentals (coding, compiling, debugging).

Course Objectives: The course aims to:

1. Introduce software engineering principles and process models.
2. Explain requirement gathering, documentation, and modeling techniques.
3. Develop understanding of software design concepts and architectures.
4. Introduce software testing strategies and quality assurance practices.
5. Familiarize students with software project management and maintenance concepts.
6. Explore modern tools, methodologies, and ethical aspects in software engineering.

Course Outcomes: On completion of the course, students will be able to –

CO1	Understand the importance of software engineering
CO2	Understand Software Requirement Specification (SRS) and illustrate requirements using UML diagrams.
CO3	Apply design principles and construct system models
CO4	Apply appropriate testing techniques and *analyze* software quality using manual and automation methods.
CO5	Demonstrate project planning (Gantt/PERT charts, risk management, version control) and evaluate software maintenance strategies.
CO6	Understand modern SE tools (Agile, DevOps, CASE, Low-code) and evaluate their role in solving real-world software engineering problems.

Course Contents:

Unit	**Description**	**Duration (Hrs.)**
1.	**Introduction to Software Engineering & Process Models:** • Introduction to Software Engineering, Software Crisis, Software Myths. • Software Development Life Cycle (SDLC) Phases: Requirements, Design, Coding, Testing, Deployment, Maintenance. • Software Process Models: Waterfall, V-Model, Incremental, Spiral, Agile (Scrum). *Case Study:* Agile vs Waterfall in real-world project scenarios.	4
2.	**Requirements Engineering & System Modeling:** • Requirements: Functional & Non-functional. • Requirements Engineering process, Requirement Traceability Matrix (RTM). • Software Requirement Specification (SRS): Structure, IEEE format. • System Modeling: Use Case Diagrams, Activity Diagrams, Prototyping	4

	Case Study: Writing an SRS in IEEE format for Library Management System	
3.	**Software Design Principles and Architectures:** • Design Concepts: Abstraction, Modularity, Cohesion & Coupling. • Function-Oriented Design: Data Flow Diagram (DFD). • Object-Oriented Design: Class Diagram (basic UML). • Software Architecture styles: Layered, Client-Server, MVC. *Case Study:* Design of Library Management System (DFD + UML diagrams).	5
4.	**Software Testing & Quality Assurance:** • Types of Testing: Unit, Integration, System, Acceptance. • Testing Techniques: White-box, Black-box, Boundary value analysis. • Test Case Design basics. • Software Quality Assurance (intro), Manual vs Automation Testing *Case Study:* Testing an Online Food Delivery Application.	5
5.	**Software Project Management & Maintenance:** • Project Scheduling: Basics of Gantt Chart & PERT/CPM. • Risk Management • Version Control basics (Git/SVN overview). • Software Maintenance: Types (corrective, adaptive, perfective). • Software Re-engineering *Case Study:* Project Management of Online Examination System.	5
6.	**Advanced Topics & Tools in SE:** • DevOps Basics (CI/CD) • Agile Tools: Jira, Trello (intro) • CASE Tools (overview) • Future trends: AI in Software Engineering, Low-code/No-code platforms. • Ethical & Social Issues in Software Engineering *Case Study:* Building a Smart Campus Portal using Agile + DevOps + Low-Code platforms.	5
	TOTAL	**28**

Text Books:

1. Ian Sommerville – *Software Engineering*, Pearson. 10th Edition, 2015, ISBN-13: 978-0133943030
2. Pressman & Maxim – *Software Engineering: A Practitioner's Approach*, McGraw-Hill, 9th Edition,2020, ISBN-13: 978-1259872976

Reference Books:

1. Pankaj Jalote, "Software Engineering: A Precise Approach", Wiley India, ISBN: 9788-1265-2311- 5.
2. Rajib Mall, "Fundamentals of Software Engineering", Prentice Hall India, ISBN-13:9788-1203- 4898-1

E-Resources:

1. https://nptel.ac.in/courses/106105087,NPTEL:Software Engineering –Prof. Rajib Mall,IIT Kharagpur
2. https://www.atlassian.com/software/jira, Jira by Atlassian – Agile Project & Issue Tracking Tool

DEPARTMENT OF COMPUTER ENGINEERING

Program: B. Tech. (Computer Engineering)							**Semester:** IV		
Course: Probability and Statistics							**Code:** COS303		
Teaching Scheme				**Evaluation Scheme**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
03	-	-	03	40	60		-	-	100

Prerequisites:

Knowledge of basic calculus**.**

Course Objectives:

1. To understand the fundamental concepts of statistics and sampling theory, including their origins, definitions, and importance.
2. To learn about measures of central tendency and their relevance in describing data distributions.
3. To understand the concepts of dispersion, skewness, and kurtosis, and their significance in data analysis.
4. To gain a foundational understanding of probability theory, including basic definitions, laws, and concepts.
5. To understand the concepts of discrete and continuous random variables, probability mass and density functions, and distribution functions, and apply them to solve probability problems.
6. To introduce students to understand, explain, and apply the foundational mathematical concepts at the core of computer science.

Course Outcomes: On completion of the course, students will be able to -

CO1	Develop skills to apply various sampling methods and understand their implications on data analysis.
CO2	Acquire the ability to analyze frequency distributions, construct histograms and frequency polygons.
CO3	Develop proficiency in calculating and interpreting measures of dispersion
CO4	Apply different data visualization techniques to understand the data
CO5	Understand the concepts of discrete and continuous random variables, probability mass and density functions.
CO6	Formulate problems precisely, solve the problems, apply formal proof techniques and explain the reasoning clearly.

Course Contents:

Unit	**Description**	**Duration (Hrs.)**
1.	**Statistics and Sampling Theory:** Statistics: Introduction, Definition, Importance and Scope. Sampling, Introduction, Types of sampling, Random sampling, simple sampling, stratified sampling, parameter and statistics.	7
2.	**Descriptive Statistics: Measures of Central Tendency:** Frequency Distributions and Measures of central Tendency: Frequency Distribution, Continuous Frequency Distribution, Measures of Central Tendency,	7

	Arithmetic Mean, Median, Mode.	
3	**Descriptive Statistics: Measures of Dispersion:** Measures of Dispersion: Range, Quartile Deviation, Mean Deviation, Standard Deviation and Root Mean Square Deviation, Coefficient of Dispersion, Coefficient of Variation Skewness and Kurtosis.	7
4	**Introduction to Probability Theory:** Definition of a probability experiment and sample space and Outcomes, axiomatic approach to probability. Rules and Laws of probability: addition, multiplication and complement Rule. Conditional Probability, Bayes' theorem.	7
5	**Introduction to Random Variables:** Definition of Random Variable and it types. Probability Mass Function, Probability Density Function. Standard Probability distribution: Binomial, Poison and Normal.	7
6	**Statistical Inference:** Definition and importance of statistical inference, Concepts of population and sample, Estimation and testing as two branches of inference. Basics of Hypothesis Testing, Null Hypothesis, Alternative Hypothesis, Types of errors:Type I Error, Type II Error. Applications of Hypothesis Testing: t-test, F-test.	7
	TOTAL	**42**

Text Books:

1. Hossein Pishro-Nik, "Introduction to Probability, Statistics, and Random Processes," Pearson, 2014.
2. Morris H. DeGroot and Mark J. Schervish, "Probability and Statistics," Addison-Wesley, 2012.
3. R. Johnson, "Probability and Statistics for Engineers", Prentice India Ltd, 8 Edition, ISBN 13:978- 8120342132.

Reference Books:

1. Geoffrey R. Grimmett and David R. Stirzaker, "Probability and Random Processes," Oxford University Press, 2001.
2. Roy D. Yates and David J. Goodman, "Probability and Stochastic Processes: A Friendly Introduction for Electrical and Computer Engineers," Wiley, 2005.
3. George Casella and Roger L. Berger, "Statistical Inference," Cengage Learning, 2002.
4. S.P. Gupta, "Statistical Methods", Paperback publication, 43 edition, ISBN: 9788180549892, 8180549895

E-Resources:

1. An Introduction to Statistical Learning by Gareth James
https://www.ime.unicamp.br/~dias/Intoduction%20to%20Statistical%20Learning.pf
2. NPTEL Course: "Introduction To Probability Theory And Statistics"
https://onlinecourses.nptel.ac.in/noc22_ma81/preview

DEPARTMENT OF COMPUTER ENGINEERING

Program: B. Tech. (Computer Engineering)						**Semester:** IV			
Course: E-Commerce						**Code:** COMC402			
Teaching Scheme				**Evaluation Scheme**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
1	-	1	2	-	-	25	-	-	25

Prerequisites:

Basic knowledge of the Internet and the World Wide Web, Familiarity with business or commerce principles, Interest in digital platforms and technologies

Course Objectives:

1. To provide an understanding of fundamental concepts and models in E-Commerce.
2. To familiarize students with digital business strategies and payment systems.
3. To develop awareness of legal, ethical, and security issues in E-Commerce.
4. To give exposure to digital marketing and E-Commerce tools.

Course Outcomes: On completion of the course, students will be able to -

CO1	Understand the foundations and evolution of E-Commerce and its business models.
CO2	Analyze various payment systems, technologies, and infrastructure of E-Commerce.
CO3	Examine security and legal frameworks applicable to E-Commerce.
CO4	Evaluate the ethical, social, and regulatory concerns in digital commerce.
CO5	Apply digital marketing strategies to online businesses
CO6	Develop and demonstrate a basic E-Commerce prototype or campaign.

Course Contents:

Unit	**Description**	**Duration (Hrs.)**
1.	Introduction to E-Commerce: Definition, history, and evolution; traditional vs. electronic commerce; E-Commerce framework and business impact; features and limitations of E-Commerce.	4
2.	E-Commerce Business Models: B2B, B2C, C2C, C2B, G2C; revenue models; case studies (Amazon, Flipkart, Meesho, etc.); advantages and challenges.	4
3	E-Commerce Infrastructure: Internet and WWW, web hosting, domain registration, client-server architecture; overview of E-Commerce platforms (Shopify, Wix, WooCommerce).	5
4	Electronic Payment Systems & Security: Payment methods: credit/debit cards, wallets, UPI, net banking; digital signatures; SSL; common threats and prevention; e-fraud case studies.	5
5	Legal, Ethical & Regulatory Issues: Overview of IT Act 2000, data protection laws (GDPR); privacy concerns; ethical and IPR issues in digital business	5

6	Digital Marketing & Strategy: Introduction to SEO, SEM, affiliate marketing, email campaigns, and social media marketing; use of Google Ads, Meta Ads, CRM tools.	5
	TOTAL	**28**

Text Books:

1. P.T. Joseph, "E-Commerce: An Indian Perspective", PHI Learning
2. Kenneth C. Laudon & Carol Guercio Traver, "E-Commerce 2023", Pearson

Reference Books:

1. S.J. Joseph, "E-Commerce: A Managerial Perspective", Prentice Hall
2. Elias M. Awad, "Electronic Commerce: From Vision to Fulfilment", Pearson

E-Resources:

1. NPTEL: https://nptel.ac.in
2. Google Digital Garage: https://learndigital.withgoogle.com
3. HubSpot Academy: https://academy.hubspot.com

DEPARTMENT OF COMPUTER ENGINEERING

Program: B. Tech. (Computer Engineering)						**Semester:** IV			
Course: Quantitative Aptitude & Numerical Analysis						**Code:** COAE402			
Teaching Scheme				**Evaluation Scheme**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
-	02	-	02	-	-	25	-	-	25

Prerequisites:

Basic knowledge of Mathematics at Higher Secondary Level, Familiaty with basic logical reasoning and problem-solving skills

Course Objectives:

1. To develop and enhance quantitative aptitude and numerical problem-solving skills.
2. To train students in basic numerical techniques useful for competitive exams and placements.
3. To encourage analytical thinking and logical reasoning

Course Outcomes: On completion of the course, students will be able to -

CO1	Solve basic and advanced quantitative aptitude problems.
CO2	Apply shortcut methods for fast calculations in aptitude questions.
CO3	Develop logical reasoning skills for competitive exams.
CO4	Use numerical techniques to solve real-life problems.
CO5	Improve accuracy and speed in solving numerical analysis problems.
CO6	Demonstrate confidence in attempting placement and entrance tests.

Guidelines for Instructor's Manual

The instructor's manual should contain detailed solutions and clear explanations for each assignment included in the lab syllabus. It should also mention alternative methods for solving problems where applicable and provide additional practice questions to help students strengthen their understanding. The manual must support instructors in conducting the practical sessions smoothly and effectively.

Guidelines for Student's Lab Journal

Each student must maintain a well-organized lab journal, which should include the problem statement, a detailed step-by-step solution, and the final answer for every assignment. Students should write their observations and conclusions after completing each practical. The journal should reflect neatness, clarity, and completeness, ensuring it serves as a valuable reference for revision.

Guidelines for Lab /TW Assessment

Term work should be evaluated continuously throughout the semester. Marks should be awarded based on the regularity and punctuality of submissions, the correctness and completeness of solutions, and the student's understanding of the methods used. Viva voce examinations should also be conducted periodically to assess the student's conceptual clarity and problem-solving approach.

Guidelines for Laboratory Conduction

Practical sessions should begin with an explanation of the relevant concepts and shortcut methods, followed by solving a few examples as a demonstration. Instructors should encourage students to

discuss alternative approaches and share problem-solving tips among peers. Ample practice problems should be given during lab hours to reinforce learning and improve speed and accuracy.

Suggested list of Assignments:

1. To solve problems based on number systems including even-odd numbers, divisibility rules, and digit sums using shortcut methods.
2. To find the Highest Common Factor (HCF) and Least Common Multiple (LCM) of numbers and apply shortcut techniques for fast calculations.
3. To solve various percentage-based problems related to profit, loss, discounts, and marked price efficiently using quick methods.
4. To apply profit and loss formulas to practical business scenarios and calculate gains or losses using fast techniques.
5. To solve problems on ratio, proportion, and partnership using direct and inverse methods and learn practical applications.
6. To analyze time and work problems involving efficiency, multiple workers, and pipes & cisterns using various shortcut approaches.
7. To calculate time, speed, and distance for different situations including trains, boats & streams, and relative speed problems.
8. To compute simple and compound interest for different principal amounts, rates, and time periods and understand installment calculations.
9. To interpret and analyze data using tables, bar graphs, pie charts, and line graphs and solve related questions accurately.
10. To solve problems on permutations and combinations and apply counting principles for arrangements and selections.
11. To understand and solve basic probability problems including independent and dependent events and real-life applications.
12. To apply the properties of logarithms to simplify and solve exponential equations and real-life numerical problems.

Text Books:

1. R.S. Aggarwal, *Quantitative Aptitude for Competitive Examinations*, S. Chand Publishing.

Reference Books:

1. Arun Sharma, *How to Prepare for Quantitative Aptitude for CAT*, McGraw Hill.
2. Abhijit Guha, *Quantitative Aptitude for Competitive Examinations*, Tata McGraw Hill.

E-Resources:

1. www.indiabix.com
2. www.lofoya.com
3. YouTube Channels: *Unacademy*, *Gradeup*

DEPARTMENT OF COMPUTER ENGINEERING

Program: B. Tech. (Computer Engineering)						**Semester:** IV			
Course: Python Programming Lab						**Code:** COVS404			
Teaching Scheme				**Evaluation Scheme**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
-	04	-	02	-	-	25	-	25	50

Prerequisites:

Basic understanding of programming logic, Knowledge of flowcharts, algorithms, and data types

Course Objectives:

1. To introduce Python programming language for general-purpose and application development.
2. To develop the ability to write simple to intermediate Python programs.
3. To implement object-oriented and modular programming in Python.
4. To utilize Python libraries for real-world problems including data handling.

Course Outcomes: Upon successful completion of the course, students will be able to:

CO1	Understand Python syntax, data types, and control structures
CO2	Develop Python programs using functions, modules, and file handling.
CO3	Implement object-oriented programming concepts in Python.
CO4	Use libraries like NumPy and Pandas for data processing.
CO5	Solve real-world problems using Python scripting.
CO6	Analyze and debug Python programs to improve code efficiency and readability.

Course Contents:

Sr. No	**Description**	**Duration (Hrs.)**
1.	**Introduction to Python:** Overview of Python, features, applications, Python interpreter, comments, indentation, basic syntax, variables, keywords, operators, input/output, basic data types: int, float, string, Boolean, Python Casting	4
2.	**Control Structures and Data Structures:** Conditional statements (if, elif, else), loops (for, while), loop control statements (break, continue). Data structures – lists, tuples, sets, dictionaries – creation, operations, and methods.	4
3.	**Functions, Modules and Date-Time:** Defining and calling functions, argument types, return values, recursion. Lambda functions. Modules – built-in and user-defined. Packages and __main__,datetime, date, time, timedelta, Formatting dates using strftime / strptime	4
4.	**Object Oriented Programming in Python:** Classes and objects, constructor, instance and class variables, inheritance, method overriding, polymorphism, encapsulation.	4

5.	**File Handling, Exception Handling and Regular Expressions:** Opening, reading, writing files, file modes. Exception handling – try, except, else, finally, custom exceptions, re module, Searching, matching, replacing	4
6.	**Data Processing with Libraries:** Introduction to NumPy – arrays, Diamensions in array, Indexing in Numpy, Numpy functions, vectorized operations. Pandas – Series, DataFrame, file I/O (CSV, Excel), basic operations. Simple data visualization using Matplotlib.	4
	TOTAL	24

Guidelines for Instructor's Manual

The instructor's manual should briefly cover the university/program background, course objectives, syllabus, lab conduction methods, and assessment criteria. It should provide topic-wise concepts, expected outcomes, and a list of practical assignments using Python programming language. Sample code structures, algorithmic logic, and references should be included to support effective lab delivery. Standard evaluation rubrics, viva questions, and clarity on CO (Course Outcome) mapping must be provided. The manual is intended to ensure consistency in lab sessions and support instructors in aligning activities with the desired learning outcomes.

Guidelines for Student's Lab Journal

The student's lab journal should follow a uniform structure containing the experiment title, objective, problem statement, algorithm or pseudocode, program code, sample input/output, and conclusion. It should reflect individual work and understanding. Each journal must include a content index and a summary of weekly progress. Code documentation and neat presentation are expected. This record serves as the student's learning evidence and must be duly signed and verified each week by the lab instructor.

Guidelines for Lab /TW Assessment

The term work should be continuously assessed throughout the semester. Marks should be allotted based on timely submissions, code functionality, understanding of concepts, journal completion, and viva performance. Experiments should be graded using standardized rubrics focusing on logic design, correctness, output, and documentation. Bonus weightage may be given for case study analysis and the mini-project. The goal is to promote regular engagement, practical proficiency, and overall conceptual clarity.

Guidelines for Practical Examination

The practical examination should assess the student's coding ability, problem-solving skills, and conceptual understanding. It should consist of one programming problem from the performed experiments and an oral viva. Students must write and execute the solution independently within the given time frame. Evaluation should be based on execution correctness, code structure, and oral response, using standard rubrics. Both internal and external examiners must align on evaluation methods.

Guidelines for Laboratory Conduction

Laboratory sessions should begin with a brief explanation of the topic, objectives, and expected outcomes. Instructors should encourage active participation, individual coding practice, and peer learning. Faculty should monitor progress during the session and resolve student doubts. Weekly viva or discussion should be conducted for conceptual reinforcement. A lab timetable, attendance record,

and assessment log should be maintained. The aim is to build consistent technical skills and analytical thinking through hands-on practice.

Suggested List of Assignments:

1. Write a Python program to accept a number from the user and check whether it is even or odd. Then, use a for loop to display the multiplication table of that number from 1 to 10.
2. Create a list of five student names and display each using a loop. Then create a dictionary to store the names and their marks, and display the highest marks using a dictionary method.
3. Write a function to calculate the simple interest using parameters: principal, rate, and time. Also write a recursive function to find the factorial of a number entered by the user.
4. Create a user-defined module math_utils.py with functions for square() and cube(). Write a main Python program that imports this module and uses these functions for user input. Also, ask the user for their birthdate (in DD-MM-YYYY format) and display: a) Their age in years b) Day of the week they were born c) Days left until their next birthday
5. Create a class Student with attributes name and roll number. Add a method to display student details. Create an object of the class and call the method.
6. Write a program that accepts a string from the user and writes it into a file data.txt. Then, read the content back and display it. Handle exceptions like file not found and input errors.
7. Write a Python program to: a) Validate an email address using regular expressions b) Extract all mobile numbers from a given multiline string c) Replace all whitespace with a hyphen in a given sentence
8. Create a NumPy array of 5 elements and display the array. Perform operations like array addition, mean, and sorting.
9. Create a DataFrame from a dictionary containing employee names and salaries. Display the DataFrame and calculate the average salary using Pandas.
10. Write a Python program using Matplotlib to draw a bar chart showing the marks of 5 subjects. Add proper labels and title.
11. Write a program that accepts a list of filenames from the user and checks which of them exist in the current directory using the os module.
12. Create a simple student record keeper that stores name and marks using a dictionary. Allow the user to add, search, and delete records using functions.

Text Books:

1. Reema Thareja, Python Programming, Oxford University Press
2. E. Balagurusamy, Introduction to Computing and Problem Solving with Python, McGraw-Hill

Reference Books:

1. Mark Lutz, "Learning Python", O'Reilly
2. Allen B. Downey, "Think Python", O'Reilly

E-Resources:

1. NPTEL Online Courses: https://nptel.ac.in
2. W3Schools: https://www.w3schools.com
3. GeeksforGeeks: https://www.geeksforgeeks.org
4. Coursera: https://www.coursera.org

DEPARTMENT OF COMPUTER ENGINEERING

Program: B. Tech. (Computer Engineering)						**Semester:** IV			
Course: Internship – III						**Code:** COIN403			
Teaching Scheme				**Evaluation Scheme**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
-	-	-	02	-	-	25	-	-	25

Preamble:

Internships serve as vital educational and career development experiences, offering practical exposure in a specific field. Employers seek individuals who possess the necessary skills and an understanding of industry environments, practices, and cultures. This internship is designed as a structured, short-term, supervised training program, often centered on specific tasks or projects with clear timelines. The primary goal is to immerse technical students in an industrial setting, providing experiences that cannot be replicated in the classroom. This exposure aims to develop competent professionals who understand the social, economic, and administrative factors influencing the operations of industrial organizations.

Course Objectives:

1. Exposure to students to the industrial environment, which cannot be provided in the classroom and hence creating deployable professionals for the industry.
2. Learn to implement the technical knowledge in real industrial situations.

Course Outcomes: After completion of this course, students will be able to -

CO1	Gain exposure to industry practices and understand how academic concepts are applied in professional settings.
CO2	Develop and demonstrate effective communication and teamwork skills within a work environment.
CO3	Improve your problem-solving and time management skills by working in real-world industry settings.

Internship Requirements:

1. **Internship Duration**: It is mandatory for all students to undergo an internship after every semester during vacations for the duration of 3 to 5 weeks. Internships completed during this period will be considered for the assessment of Term Work (TW).
2. **Internship Opportunities:** Students can explore various opportunities for internships at:
    a. Industries
    b. Research labs or organizations
    c. Collegiate clubs
    d. In-house research projects
    e. Online internships
3. **Support and Assistance**: Students can seek assistance for securing internships from:
    a. The Training and Placement cell, along with departmental coordinators
    b. Department or institute faculty members
    c. Personal contacts
    d. Directly connecting with industries or organizations

4. **Request Letter**: Once an industry, research organization, or collegiate club is identified, students must obtain a request letter from the concerned department or placement office. This letter, in the standard format must be duly signed by the authority, should be addressed to the HR manager or relevant authority.
5. **Confirmation Letter:** Students must submit the confirmation letter from the industry, research organization, or collegiate club to the Internship Coordinator and the Head of Department (HOD) office
6. **Joining Report**: Upon commencing the internship, students must submit the joining report, joining letter, or a copy of the confirmation email to the Internship Coordinator and the HOD office
7. **Faculty Mentor**: A faculty member will be assigned as a mentor to a group of students. The mentor will be responsible for monitoring, evaluating, and assessing student internship activities. The faculty mentor is also required to visit the internship location and submit formal feedback to the Internship Coordinator.
8. **Faculty Visits**: Faculty members are advised to visit the internship site once or twice during the internship period to monitor progress.
9. **Progress Report**: Students must submit progress report fortnightly to their faculty guide and the final internship report to the Internship Coordinator and department office.
10. **Evaluation Report**: After the completion of the internship, the mentor, along with the assessment panel members, should submit the evaluation report of the students to the department office and the Internship Coordinator.
11. **Internship Certificate**: Students must receive the Internship Certificate from the industry and submit it to the Internship Coordinator and department office.
12. **Presentation and Assessment**: Students are required to give a presentation on their internship work as part of the term work. The internship diary and report will also be verified and assessed.