Zeal Education Society's

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 INFORMATION TECHNOLOGY

Curriculum Structure and Syllabus of

S.Y. B. Tech. – Information Technology

(With effect from - Academic Year 2025 - 26)
(2024 Pattern)

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 INFORMATION TECHNOLOGY

VISION:

To nurture the wisdom of young minds through modern, qualitative and Interdisciplinary research-oriented education to become a successful IT Professional.

MISSION:

M1: Advancing knowledge through fundamental and applied research.
M2: To encourage students for Innovative development and higher studies.
M3: To motivate the crux of learners towards real time solutions.
M4: To prepare skillful engineers for industry to cater best in IT Enabled Services.
M5: To nourish a student's leadership skills by inculcating personal touch and respect in professional relationships.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs):

PEO1: Graduates will develop a strong foundation in mathematics, science, and engineering Principles, enabling them to solve real-world technological problems in the field of Information Technology.
PEO2: Graduates will acquire the knowledge and technical skills necessary to design and Implement innovative IT solutions, fostering adaptability, analytical thinking, and entrepreneurial spirit.
PEO3: Graduates will demonstrate ethical responsibility, effective communication, teamwork and a commitment to continuous learning, enabling them to contribute meaningfully to Society and perform effectively in a global professional environment.

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: Apply theoretical and practical knowledge of Information Technology to design, develop, and manage efficient information systems and interdisciplinary applications

PSO2: Analyze real-world problems, identify system requirements, and design appropriate IT infrastructure and solutions, particularly for large-scale and enterprise computing systems.

PSO3: Demonstrate understanding of professional, ethical, legal, and societal responsibilities in IT practices, along with business and security processes.

PSO4: Exhibit effective communication, teamwork, and decision-making skills using modern tools and technologies to perform responsibly in professional environments.

LIST OF ABBREVIATIONS

Abbreviation	Description
BSC	Basic Science Course
ESC	Engineering Science Course
PCC	Program Core Course
PEC	Program 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
TW	Term Work
OR	Oral
PR	Project

DEPARTMENT OF INFORMATION TECHNOLOGY

Second Year B. Tech. – Information Technology: 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						
ITPC302	PCC	Data Structures	3	4	-	7	3	2	5	40	60	-	25	-	125
ITPC303	PCC	Operating Systems	2	2	-	4	2	1	3	40	60	-	-	25	125
ITPC304	PCC	Object Oriented Programming	2	-	-	2	2	-	2	40	60	-	-	-	100
ITMD301	MDM	Applied Mathematics	3	-	-	3	3	-	3	40	60	-	-	-	100
ALOE301	OE	Open Elective - I[#]	2	-	-	2	2	-	2	40	60	-	-	-	100
ITMC301	HSSM-MC	Digital Marketing and social media	1	-	1	2	1	1	2	-	-	25	-	-	25
ITVS303	VSEC	Programming Lab – I	-	4	-	4	-	2	2	-	-	25	25	-	50
ITCE301	CEP	Mini Project	-	2	-	2	-	1	1	-	-	50	-	-	50
ITIN302	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 Narhe, Pune-41 PU/PN/Engg./285/2007

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

DEPARTMENT OF INFORMATION TECHNOLOGY

Second Year B. Tech. – Information Technology: 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						
ITPC405	PCC	Database Management Systems	3	2	-	5	3	1	4	40	60	-	25	-	125
ITPC406	PCC	Software Engineering	3	-	-	3	3	-	3	40	60	-	-	-	100
ITPC407	PCC	Computer Organization	2	2	-	4	2	1	3	40	60	-	-	25	125
ITMD402	MDM	Discrete Mathematics	2	-	1	3	2	1	3	40	60	25	-	-	125
ALOE402	OE	Open Elective - II[#]	2	-	-	2		2	2	40	60		-	-	100
ITMC402	HSSM-MC	E-Commerce	1	-	1	2	1	1	2	-	-	25	-	-	25
ITAE402	AEC	Quantitative Reasoning & Analysis	-	2	-	2	-	1	1	-	-	25	-	-	25
ITVS404	VSEC	Programming Lab – II	-	4	-	4	-	2	2	-	-	25	25	-	50
ITIN403	ELC - INT	Internship - III	4 Weeks				-	2	2	-	-	-	-	25	
Total			**13**	**10**	**2**	**25**	**11**	**11**	**22**	**200**	**300**	**100**	**50**	**50**	**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 Narhe, Pune-41 PU/PN/Engg./285/2007

Director
ZES's Zeal College of Engineering & Research Narhe, Pune - 41 1041

DEPARTMENT OF INFORMATION TECHNOLOGY

INDEX

SYLLABUS

SEMESTER - III

DEPARTMENT OF INFORMATION TECHNOLOGY

Program: B. Tech. (Information Technology)							**Semester:** III		
Course: Data Structures							**Code:** ITPC302		
Teaching Scheme (Hrs/week)				**Evaluation Scheme (Marks)**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
03	04	-	04	40	60	-	-	25	125

Prerequisites:

Basic knowledge of programming (preferably in C/C++), Understanding of control structures and functions

Course Objectives:

1. To understand and implement fundamental data structures and algorithms.
2. To develop problem-solving skills and algorithmic thinking.
3. To analyze time and space complexity for efficient code design.
4. To apply appropriate data structures for real-life applications.

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

CO1	Understand the basic concepts of data structures and algorithm design.
CO2	Implement linear and nonlinear data structures using programming techniques.
CO3	Apply stack, queue, linked list, and tree structures in various applications.
CO4	Analyze and compare various searching and sorting algorithms.
CO5	Design efficient algorithms using appropriate data structures.
CO6	Evaluate the performance and complexity of algorithms.

Course Contents:

Unit	**Description**	**Duration (Hrs.)**
1.	**Introduction to Data Structures and Algorithms:** Need and importance of data structures in problem-solving. Characteristics of a good algorithm. Difference between data types and data structures. **Classification of data structures** – linear and non-linear. Static vs dynamic data structures. **Overview of Abstract Data Types (ADT)** – List, Stack, Queue. Comparison between recursive and iterative approaches. Common issues faced in programming without efficient data structures.	07
2.	**Arrays and Linked Lists:** **Concept and representation of arrays, operations** – insertion, deletion, traversal, and searching. Static vs dynamic memory allocation. **Linked Lists** – Singly, Doubly, and Circular. Implementation of linked lists using pointers. Applications of linked lists – polynomial addition, dynamic memory usage.	07

3.	**Stacks and Queues:** **Stack definition and operations** – push, pop, peek. Implementation using arrays and linked lists. Applications – expression conversion, expression evaluation, and recursion handling. **Queue definition and types** – Simple Queue, Circular Queue, Priority Queue, Double-Ended Queue (Deque). Queue operations and their real-life applications like CPU scheduling and buffering.	07
4.	**Trees:** **Basic terminology and representation of trees. Binary Tree** – creation and traversals (inorder, preorder, postorder). Binary Search Tree – insertion, deletion, and search. **Introduction to AVL Trees** – need and rotations. Heap – Min Heap, Max Heap and Heap Sort. Applications of trees – expression trees, Huffman coding.	07
5.	**Graphs:** **Introduction to graphs** – definitions and terminologies. Representations – adjacency matrix and adjacency list. Graph traversal techniques – BFS and DFS. **Minimum Spanning Tree** – Prim's and Kruskal's algorithms. Dijkstra's algorithm for shortest path. Applications of graphs – routing, social networks, topological sorting.	07
6.	**Introduction to Algorithms:** **Searching techniques** – Linear and Binary Search (recursive and iterative). **Sorting techniques** – Bubble, Selection, Insertion, Merge, Quick, and Heap Sort. Comparison of sorting algorithms based on time and space complexity. Real-world usage of searching and sorting in databases and data processing.	07
	TOTAL	**42**

List of Assignments: - (Perform Any 10 Practical's)

1. Implement insertion, deletion, searching, and traversal on one-dimensional and two-dimensional arrays using C, C++, or Java. Accept dynamic input and provide user-friendly menus. Analyze time complexity for each operation.
2. Develop programs for singly, doubly, and circular linked lists. Perform insertion and deletion at various positions. Display elements and handle edge cases like underflow and overflow.
3. Implement stack using arrays and linked lists. Perform infix to postfix conversion and evaluate postfix expressions. Use stack to check balanced parentheses in expressions.
4. Implement simple queue, circular queue, and priority queue using arrays. Include operations for enqueue, dequeue, peek, and display. Simulate a basic scheduling system using queue.
5. Construct a BST and perform insertion, deletion, and traversal (inorder, preorder, postorder). Display the tree in a structured format. Highlight applications in searching and sorting systems.
6. Implement max heap and min heap. Perform heap sort using heapify and demonstrate sorting steps. Explain the role of heaps in priority-based task handling.
7. Use adjacency matrix or list to represent graphs. Implement BFS and DFS traversal. Provide input for directed and undirected graphs and display the visited sequence clearly.
8. Implement Dijkstra's algorithm to find the shortest path in a weighted graph. Accept graph input

from the user and display step-wise output with distance table.

9. Implement and compare sorting algorithms: Bubble, Insertion, Merge, Quick, and Heap sort. Analyze and tabulate their performance based on time and number of comparisons for various inputs.
10. Implement linear and binary search algorithms (iterative and recursive). Perform complexity analysis. Accept unsorted and sorted input lists and demonstrate difference in performance.
11. **Choose any one case:**
    a) Railway reservation system using linked list and queue
    b) Syntax checker using stack
    c) Graph-based social network connection map
    Apply one or more data structures to build a simplified prototype and document usage with code and explanation.
12. **Mini Project**

Text Books:

1. Ellis Horowitz, Sartaj Sahni, "Fundamentals of Data Structures", Galgotia Publications.
2. Reema Thareja, "Data Structures using C", Oxford University Press.

Reference Books:

1. Mark Allen Weiss, "Data Structures and Algorithm Analysis in C", Pearson Education.
2. Seymour Lipschutz, "Data Structures", Schaum's Outline Series.

E-Resources:

1. NPTEL Online Courses: https://nptel.ac.in
2. GeeksforGeeks: https://www.geeksforgeeks.org
3. Coursera – Data Structures: https://www.coursera.org

DEPARTMENT OF INFORMATION TECHNOLOGY

Program: B. Tech. (Information Technology)								**Semester:** III	
Course: Operating Systems								**Code:** ITPC303	
Teaching Scheme (Hrs/week)				**Evaluation Scheme (Marks)**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
02	02	-	03	40	60	-	25	-	125

Prerequisites:

1. Fundamentals of Computer Systems
2. Basic knowledge of C programming
3. Familiarity with command-line interfaces (optional but desirable)

Course Objectives:

1. To understand the fundamental concepts and design principles of operating systems.
2. To apply OS concepts practically through the Linux operating system.
3. To develop skills in Linux commands, shell scripting, and process handling.
4. To explore memory management, file systems, and system calls using Linux.
5. To gain hands-on experience with real-world Linux environments.

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

CO1	Describe the structure and functionalities of modern operating systems.
CO2	Use Linux commands and shell scripts for performing system-level tasks.
CO3	Apply concepts of process management and inter-process communication in Linux.
CO4	Analyze and apply memory and file management strategies using Linux tools.
CO5	Develop simple system-level programs using Linux APIs and system calls.
CO6	Demonstrate practical skills in Linux system administration and automation.

Course Contents:

Unit	**Description**	**Duration (Hrs.)**
1.	**Introduction to Operating Systems and Linux:** Types of operating systems and structures, Functions of an Operating System, Overview of Linux and its architecture, Linux installation, booting process, Basic Linux shell and command-line interface System Calls, Types of System Calls	04
2.	**Commands and Shell Scripting:** File handling, process commands, filters, Bash shell scripting: variables, loops, functions, Cron jobs, script automation, Text processing with awk, sed, grep	04

3.	**Process Management:** Process Concept, Process states, Process control, Threads, Uniprocessor Scheduling: Types of scheduling: Pre-emptive, Non pre-emptive, Scheduling algorithms: FCFS, SJF, RR, Priority.	05
4.	**Memory Management:** **Memory partitioning:** Fixed and Variable Partitioning, Memory Allocation: Allocation Strategies (First Fit, Best Fit, and Worst Fit), Fragmentation, Swapping, Virtual Memory, Paging. Segmentation, Demand paging and Page replacement policies. **Deadlock:** Principles of deadlock, Deadlock Prevention, Deadlock Avoidance, Deadlock Detection, Dining Philosopher problem File Systems and I/O Management	05
5.	**File System:** Concept of a file, access methods, directory structure, file system mounting, file sharing, protection. File system implementation: file system structure, file system implementation, directory implementation, allocation methods, free-space management, efficiency and performance, comparison of UNIX and windows.	05
6.	**Input/output Management:** Principles of I/O Hardware: I/O devices, Device controllers, Direct memory access. Principles of I/O Software: Goals of I/O Software, Interrupt handlers, Device drivers. Disks: Disk Hardware, RAID, Disk Arm Scheduling Algorithm.	05
	TOTAL	**28**

List of Assignments: - (Perform Any 10 Practical's)

1. Execute and demonstrate at least 20 basic Linux commands related to file handling, directory navigation, and user interaction. Include commands like ls, pwd, cd, mkdir, rm, cat, touch, chmod, chown, whoami, passwd, df, and du.
2. Explore and analyze the Linux file system hierarchy. Use ls -l, chmod, and stat to inspect and modify file permissions. Demonstrate effects for user, group, and others with examples.
3. Write shell scripts to:
   a) Calculate the factorial of a number using loops
   b) Display system information such as hostname, uptime, logged-in users, and disk usage
4. Create a shell script to back up a directory into a .tar.gz file. Schedule the task using cron jobs. Validate scheduled execution with timestamps or logs.
5. Use awk, sed, and grep for text processing:
   a) Extract valid email IDs from a file
   b) Replace specific text
   c) Filter logs with keywords like "error" or "warning"
6. Write a C program using fork (), exec(), and wait() to create a child process that runs a different program. Display parent and child process information.

7. Develop a multithreaded C program using pthread to simulate a producer-consumer problem with shared buffer. Use mutex locks and condition variables.
8. Write a C program to set up a custom signal handler for SIGINT or SIGTERM. On signal receipt, display a custom message and exit gracefully.
9. Write a program using shared memory (shmget(), shmat(), shmdt()) or mmap() to allow data exchange between processes. Demonstrate both read and write.
10. Analyze memory usage using /proc/meminfo, top, free, and vmstat. Capture observations under different system conditions.
11. Write a C program to copy file contents using system calls (open (), read (), write (), close ()). Also show how to modify file permissions and create symbolic and hard links.
12. Write automation shell scripts for basic system admin tasks:
    a) Create multiple users and groups
    b) Install packages using apt or rpm
    c) Monitor disk usage and services using df, systemctl, journalctl

Text Books:

1. Abraham Silberschatz, Peter B. Galvin, and Greg Gagne, "Operating System Concepts", Wiley
2. Richard Blum and Christine Bresnahan, "Linux Command Line and Shell Scripting Bible", Wiley

Reference Books:

1. Andrew S. Tanenbaum, "Modern Operating Systems", Pearson
2. Daniel P. Bovet and Marco Cesati, "Understanding the Linux Kernel", O'Reilly
3. W. Richard Stevens, "Advanced Programming in the UNIX Environment", Addison-Wesley

E-Resources:

1. https://linuxjourney.com – Free Linux learning platform
2. https://explainshell.com – Shell command explanations
3. https://man7.org – Linux manual and system programming guides
4. YouTube: "Linux Essentials" by The Linux Foundation
5. NPTEL Course: "Operating System Fundamentals" – by Prof. P. K. Biswas (IIT Kharagpur)

DEPARTMENT OF INFORMATION TECHNOLOGY

Program: B. Tech. (Information Technology)						**Semester:** III			
Course: Object Oriented Programming						**Code:** ITPC304			
Teaching Scheme (Hrs/week)				**Evaluation Scheme (Marks)**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
03	-	-	03	40	60	-	-	-	100

Prerequisites:

Basic understanding of programming concepts, Familiarity with C or C++ programming languages

Course Objectives:

1. To understand the fundamental concepts of Object-Oriented Programming (OOP) and Java syntax
2. To develop object-oriented applications using Java classes, inheritance, and polymorphism
3. To implement exception handling, file I/O, and collections in Java
4. To demonstrate the use of threads, GUI programming, and event handling
5. To promote good design practices and modular programming

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

CO1	Explain the principles and pillars of object-oriented programming in the context of Java
CO2	Develop Java programs using classes, objects, constructors, and methods
CO3	Apply concepts of inheritance, interfaces, and polymorphism to create reusable code
CO4	Implement exception handling, multithreading, and file I/O in Java programs
CO5	Utilize collections, generics, and packages for efficient program structure
CO6	Design basic GUI applications using Java AWT and Swing components

Course Contents:

Unit	**Description**	**Duration (Hrs.)**
1.	**Introduction to OOP and Java Basics:** Concept of object-oriented programming: abstraction, encapsulation, inheritance, and polymorphism. Introduction to Java: features, JVM, JDK, JRE. Java program structure, data types, variables, operators, control structures, arrays, and string handling.	07
2.	**Encapsulation, Abstraction, and Boxing Concepts:** Access specifiers and encapsulation in Java. Use of getter and setter methods. Concept of abstraction using abstract classes and interfaces. Differences between abstraction and encapsulation. Wrapper classes and the concept of boxing and unboxing in Java. Autoboxing with collections and method calls.	07
3.	**Classes, Objects, and Constructors:** Creating and using classes and objects. Method definition and overloading. Constructors and constructor overloading. Use of 'this' keyword, static members, static blocks and methods, memory management, garbage collection, finalize ().	07

4.	**Inheritance and Polymorphism:** Types of inheritance (single, multilevel, hierarchical), method overriding, super keyword, dynamic method dispatch, use of final keyword. Interfaces and their implementation. Abstract classes vs interfaces. Polymorphism: compile-time and runtime.	07
5.	**Exception Handling and File I/O:** Types of exceptions – checked and unchecked. Try-catch-finally block, throw and throws keyword, custom exceptions. File handling using java.io and java.nio packages – reading/writing characters, bytes, and buffered streams.	07
6.	**Multithreading, Collections, and GUI Programming:** Multithreading: Thread class, Runnable interface, thread lifecycle, thread methods, synchronization. Java Collections: List, Set, Map, Iterator, generics. GUI using AWT/Swing: Components, Layouts, Event handling, Adapter classes, simple GUI applications.	07
	TOTAL	**42**

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/java/

DEPARTMENT OF INFORMATION TECHNOLOGY

Program: S.Y. B.Tech. (Information Technology)							**Semester:** III		
Course: Applied Mathematics							**Code:** ITMD301		
Teaching Scheme (Hrs/week)				**Evaluation Scheme (Marks)**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
03	-	-	03	40	60	-	-	-	100

Prerequisites:

Calculus (differentiation, integration), basic ordinary differential equations of first order, fundamental algebra including an introductory probability and statistics concepts.

Course Objectives:

1. To make the students familiarize with concepts and techniques in Ordinary differential equations, Statistical techniques, Probability theory and transforms.
2. To equip them with the techniques to understand advanced level mathematics and its applications that would enhance analytical thinking power, useful in their disciplines.

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

CO1	Solve higher order linear differential equation using appropriate techniques for modeling and analyzing the real-world problems raised in information Technology.
CO2	Apply Fourier transform in image processing.
CO3	Analyze the discrete signals using z-transform transform.
CO4	Summarize and analyze the data using statistical tools like Moments, Skewness and Kurtosis, Correlation, Regression and Curve fitting.
CO5	Use the concepts of Probability theory for analysis and prediction of a given data as applied to machine intelligence.
CO6	Test the assumption using sampling theory and hypothesis testing.

Course Contents:

Unit	**Description**	**Duration (Hrs.)**
1.	**Linear Differential Equations (LDE):** Linear differential equation of nth order with constant coefficients, Method of Variation of Parameters, Applications of differential equations in Information Technology.	08
2.	**Fourier Transform (FT):** Complex exponential form of Fourier series, Fourier integral theorem, Fourier Sine & Cosine integrals, Fourier transform, Fourier Sine and Cosine transforms and their Inverses.	07
3.	**Z - Transform (ZT):** Introduction, Definition, Standard properties, ZT of standard sequences and their inverses. Solution of difference equations.	07

4.	**Statistics:** Measures of central tendency, Standard deviation, Coefficient of variation, Moments, Skewness and Kurtosis, Correlation and Regression, Curve fitting by the method of least squares	07
5.	**Probability:** Definition and Theorems on Probability, Conditional Probability and Baye's Theorem, Mathematical Expectation, Discrete and Continuous probability distributions: Binomial, Poisson distribution, Normal (Gaussian) distribution	07
6.	**Sampling Theory and Hypothesis Testing:** Population and Sample, Sampling methods, Population parameters, Sample statistic, Central limit theorem, point estimate and Interval Estimates, confidence interval, standard error, Mean error. Hypothesis testing: Null and Alternate hypothesis, test of hypothesis, Type I and Type II errors, level of significance, p-value, chi- square test, t- test.	06
	TOTAL	**42**

Text Books:

1. B.S. Grewal, "Higher engineering Mathematics", Khanna publishers, Delhi (40th edition).
2. P. N. Wartikar & J. N. Wartikar, "Applied Mathematics". Volume I and II, Pune Vidyarthi Griha Prakashan, Pune.
3. B.V. Ramana, "Engineering Mathematics", Tata McGraw-Hill.
4. H. K. Das, "Higher Engineering Mathematics", S. Chand Publication.

Reference Books:

1. Erwin Kreyszig, "Advanced Engineering Mathematics", 10th edition, Wiley Publications, 2015.
2. Peter V. O'Neil, "Advanced Engineering Mathematics", Cengage Learning.
3. Sheldon M. Ross, "Introduction to Probability and Statistics for Engineers and Scientists", Elsevier Academic Press
4. Miller & Freund's Probability and Statistics for Engineers by Richard A. Johnson.

E-Resources:

NPTEL Online Courses:

1. https://onlinecourses.nptel.ac.in/noc25_ma85
2. https://onlinecourses.nptel.ac.in/noc25_ma90

DEPARTMENT OF INFORMATION TECHNOLOGY

Program: B. Tech. (Information Technology)				**Semester:** III					
Course: Digital Marketing and Social Media				**Code:** ITMC301					
Teaching Scheme (Hrs/week)				**Evaluation Scheme (Marks)**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
01	-	01	02	-	-	25	-	-	25

Prerequisites:

1. Basic knowledge of internet usage and browsing.
2. Familiarity with communication tools such as email, blogs, or websites.
3. Interest in business, branding, or marketing principles.

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: After completion of this course, students will 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.	03
2.	**Search Engine Optimization (SEO):** On-page and off-page SEO; Keywords, backlinks, meta tags; Tools: Google Search Console, SEMrush, Moz; SEO performance metrics.	02
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.	03

4.	**Social Media Marketing (SMM):** Overview of platforms (Facebook, Instagram, Twitter, LinkedIn); Creating content calendars, running ad campaigns, influencer marketing; Engagement metrics and KPIs.	02
5.	**Email and Content Marketing:** Email campaign creation, tools like Mailchimp; Email list building, segmentation; Blogging, video marketing, storytelling; Content writing tips and tools.	02
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.).	02
	TOTAL	**14**

List of Assignments: - (Perform Any 10 Practical's)

1. **SEO Audit Using Free Tools:** Perform an SEO audit of a website using tools like Uber suggest, Screaming Frog, or Google Search Console. Identify technical issues, keywords, and backlink data.
2. **Keyword Research and Content Optimization: Use** Google Keyword Planner or Ubersuggest to find 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 INFORMATION TECHNOLOGY

Program: B. Tech. (Information Technology)								**Semester:** III	
Course: Programming Lab – I								**Code:** ITVS303	
Teaching Scheme (Hrs/week)				**Evaluation Scheme (Marks)**					
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: After completion of this course, students will 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.

Guidelines for Instructor's Manual

The instructor's manual should include course objectives, unit-wise lab plans, expected outcomes, and model solutions. It must contain a detailed explanation of each experiment, code samples, evaluation rubrics, and CO mapping. The manual should help maintain consistency and outcome alignment across batches.

Guidelines for Student's Lab Journal

The journal should include a title page, table of contents, and properly formatted entries for each practical: aim, problem statement, algorithm, code, sample output, and conclusion. Each entry must be handwritten or neatly typed and signed weekly by the instructor.

Guidelines for Lab /TW Assessment

Term work will be assessed continuously based on timely submissions, quality of code, output, innovation, and viva performance. Each practical will carry marks based on correctness, documentation, and CO mapping. Bonus marks may be awarded for the mini-project.

DEPARTMENT OF INFORMATION TECHNOLOGY

Guidelines for Practical Examination

The practical exam will include any one program from the syllabus along with a viva. Evaluation will focus on logic, code correctness, output, and oral responses. External examiners must align with institutional rubrics.

Guidelines for Laboratory Conduction

Begin each lab session with a brief explanation of the concept. Encourage students to practice coding individually. Doubt sessions and frequent evaluation should be conducted. Maintain batch-wise attendance and performance records.

List of Assignments: - (Perform Any 10 Practical's)

1. Write a Java program to demonstrate variable declaration, data types and type conversion, operators, and control statements.
2. Create a class with data members and member methods. Write Java programs for object instantiation and accessing class methods.
3. Implement constructors (default and parameterized), constructor overloading, and method overloading in a Java program.
4. Write a Java program demonstrating single, multilevel inheritance, and method overriding using the super keyword.
5. Create an abstract class and interface to demonstrate abstraction and interface implementation in Java.
6. Demonstrate encapsulation using private data members and getter/setter methods. Use all four access specifiers in a Java program.
7. Write a Java program using try-catch blocks, multiple catch, nested try, finally, throw, and custom exceptions.
8. Implement file reading and writing using File Reader, FileWriter, BufferedReader, and Buffered Writer.
9. Demonstrate multithreading by extending the Thread class and implementing the Runnable interface. Include thread priority and synchronization.
10. Write a Java program to use Array List, HashMap, and TreeSet with generics to manage a list of student records.
11. Create a simple GUI application using AWT and Swing components such as buttons, labels, text fields, and layout managers.
12. **Case Study:** Document and analyze the OOP design of a real-world system (e.g., ATM simulator, Hospital Management). Include class structure, relationships (inheritance/interfaces), and design diagram.
13. **Mini Project**

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 INFORMATION TECHNOLOGY

Program: B. Tech. (Information Technology)						**Semester:** III			
Course: Mini Project						**Code:** ITCE301			
Teaching Scheme (Hrs/week)				**Evaluation Scheme (Marks)**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
-	02	-	01	-	-	50	-	-	50

Preamble:

Project Based Learning (PBL) is a dynamic teaching methodology that fosters critical thinking, collaboration, and personalized learning experiences. In this approach, students work in groups to explore topics of personal interest through meaningful inquiry. Projects are rooted in real-world problems, aligned with the curriculum, and often span multiple disciplines. Students take ownership of their learning by determining how to approach the problem and selecting appropriate methods and activities. They gather information from diverse sources, analyze and synthesize data, and derive meaningful insights. The real-life relevance of PBL enhances student motivation and lends authenticity to their work, while also building essential life skills like teamwork and self-reflection. Each group is supported by a faculty member, known as a mentor, who provides guidance throughout the process. Technology plays a vital role in facilitating collaboration and supporting each phase of the project. At the conclusion of the project, students present their outcomes, demonstrating both their knowledge and communication skills. They also engage in self-assessment to reflect on their progress and learning. The mentor's role is to support and advise, rather than to control or manage, encouraging student autonomy and deeper engagement.

Companion Course:

Online courses relevant to the project, along with expert lecture on Intellectual property rights, patents and software engineering.

Course Objectives:

1. To understand the key processes and methodologies involved in Project-Based Learning.
2. To enhance students' critical thinking abilities and their skills in solving real-world engineering problems.
3. To illustrate the roles and responsibilities of IT engineers in addressing engineering challenges within social, environmental, and economic contexts.
4. To prepare students with the knowledge and practical skills needed to develop effective solutions for problems typically presented in hackathons.

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

CO1	Develop innovative solutions for real-life problems by applying analytical thinking and collaborative problem-solving.
CO2	Demonstrate the ability to learn through hands-on experiences, fostering habits of lifelong learning.
CO3	Address and resolve technical challenges in real-world scenarios through effective teamwork.

CO4	Participate actively in interdisciplinary collaborations to enhance knowledge integration and practical understanding.

Course Contents:

Group Structure

Students will work collaboratively in **mentor-guided teams** to plan, manage, and complete projects or activities that address defined real-world problems.

1. Each team should consist of **3 to 6 students**, working cohesively toward a common goal.
2. A **dedicated mentor** will be assigned to each group to support their learning and development throughout the PBL journey.

Project/Problem Selection

1. Projects may originate from any domain**,** though topics with an IT or technical focus are encouraged**.**
2. Projects initiated in the first year of engineering may be extended if they show potential and relevance.
3. Preference should be given to projects that involve conceptual modelling and the application of software tools**.**
4. Multiple solution approaches — such as theoretical analysis, practical implementation, working models, simulations**,** or software-based solutions — are encouraged.
5. Projects that require a multidisciplinary perspective are highly desirable.
6. Problems should demand in-depth exploration of practical, scientific, or technical areas.
7. The project methodology should incorporate hands-on activities**,** organizational and field visits**,** interaction with research institutes, and expert consultations to expose students to the latest technologies and practices.

Assessment

The department shall ensure rigorous monitoring and assessment of both student progress and the impact of their proposed solutions.

- Weekly reviews will be conducted to track the project's progress.
- Assessment will include both individual and group performance, facilitated by the assigned mentor.
- Students are expected to maintain a culture of authentic collaboration, self-motivation, and peer learning**.**
- Departments must provide guidance, orientation, and access to resources to support the project process.
- Students will showcase their learning through public products, reports, or presentations**.**

Components of Assessment:

1. **Individual Evaluation** – Assessment of personal involvement, contribution, and understanding.
2. **Group Evaluation** – Review of team dynamics, role distribution, communication, and cohesion.
3. **Presentation and Documentation** – Evaluation of reporting quality, organization, and communication of outcomes.

DEPARTMENT OF INFORMATION TECHNOLOGY

Evaluation and Continuous Assessment
Students and mentors are required to maintain a PBL workbook that documents progress, deliverables, and reflections. • The workbook will serve as a record of accountability, punctuality, technical writing skills**,** and project workflow. • A Continuous Assessment Sheet (CAS) will be maintained by mentors and the department. ***Suggested Assessment Criteria and Weightage:*** 1. Idea Inception – 5% 2. Solution Quality / Problem Solving / Final Output – 40% 3. Documentation (requirements, design, tools, report, etc.) – 25% 4. Patentability Potential – 10% 5. Demonstration (UI, usability, presentation) – 10% 6. Participation in Competitions / Publications – 5% 7. Consideration of Environmental, Ethical, Legal, or Social Aspects – 5% Rubrics for evaluation should be designed based on these criteria and shared with students for transparency and guidance.
Faculty / Mentor is expected to perform following activities
Faculty mentors are expected to actively support student learning by: 1. Revising PBL concepts and assessing initial student skills 2. Forming diverse and balanced teams 3. Sharing knowledge on IPR (patents, copyrights, publications) 4. Discussing case studies to provide practical insight 5. Designing and explaining the rubrics for evaluation 6. Conducting weekly assessments of deliverables (presentations, reports, concept maps, logbooks) 7. Providing continuous guidance (scaffolding) 8. Engaging in both formative and summative assessment practices
Reference Books:
1. www.schoology.com 2. www.wikipedia.org 3. www.howstuffworks.com

DEPARTMENT OF INFORMATION TECHNOLOGY

Program: B. Tech. (Information Technology)								**Semester:** III	
Course: Internship – II								**Code:** ITIN302	
Teaching Scheme (Hrs/week)				**Evaluation Scheme (Marks)**					
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 INFORMATION TECHNOLOGY

Program: B. Tech. (Information Technology)							**Semester:** IV		
Course: Database Management System							**Code:** ITPC405		
Teaching Scheme (Hrs/week)				**Evaluation Scheme (Marks)**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
03	02	-	04	40	60	25	-	25	150

Prerequisites:

1. Basic knowledge of data structures and programming
2. Understanding of file systems and operating system fundamentals

Course Objectives:

1. To introduce the fundamental concepts of database systems
2. To design relational databases using E-R models and normalization techniques
3. To develop skills in SQL for managing and querying relational databases
4. To understand transaction processing, concurrency control, and recovery
5. To explore indexing, query optimization, and database architectures
6. To gain insight into emerging trends like NoSQL and distributed databases

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

CO1	Understand basic database concepts and architecture
CO2	Design relational databases using E-R models and normalization
CO3	Write SQL queries to manage, retrieve, and manipulate data
CO4	Apply transaction management, concurrency control, and recovery techniques
CO5	Analyze indexing strategies and query optimization methods
CO6	Explore advanced topics such as NoSQL, distributed, and cloud-based databases

Course Contents:

Unit	**Description**	**Duration (Hrs.)**
1.	**Introduction to Database Systems:** Purpose of databases, file system vs DBMS, advantages of DBMS, database users and administrators, DBMS architecture, data models (hierarchical, network, relational), schema and instance, data independence, DBMS languages (DDL, DML).	07
2.	**Entity-Relationship (ER) Modeling and Relational Model:** ER model: entity types, attributes, relationships, generalization, specialization, aggregation. Converting ER to relational schema. Relational model: structure, keys, integrity constraints, relational algebra operations.	07
3	**Structured Query Language (SQL):** Basic SQL commands (SELECT, INSERT, DELETE, UPDATE), constraints, aggregate functions, group by, order by, joins (inner, outer, self), subqueries,	07

	views, indexes, and stored procedures.	
4.	**Normalization and Functional Dependencies:** Functional dependencies, decomposition, lossless-join and dependency preservation, normal forms (1NF, 2NF, 3NF, BCNF), multivalued dependencies, join dependencies, denormalization.	07
5.	**Transaction Management and Concurrency Control:** Transaction properties (ACID), transaction states, serializability, conflict and view serializability, concurrency control protocols: locking, timestamp ordering, deadlock handling, recovery concepts and techniques.	07
6.	**Query Optimization and Advanced Topics Query:** Processing overview, evaluation strategies, indexing (B+ trees, hash-based), and introduction to NoSQL, distributed databases, cloud-based databases, and emerging trends in database systems.	07
	TOTAL	**42**

List of Assignments: - (Perform Any 10 Practical's)

1. Execute SQL commands to create, insert, update, delete, and retrieve data from a single table. Apply integrity constraints (primary key, not null, unique, default, check).
2. Perform queries involving two or more tables using INNER JOIN, LEFT JOIN, RIGHT JOIN, and FULL OUTER JOIN. Retrieve complex data using aliasing and filtering.
3. Write SQL queries using aggregate functions (SUM, AVG, COUNT, MIN, MAX) along with GROUP BY, HAVING, and ORDER BY.
4. Use nested subqueries in SELECT, FROM, and WHERE clauses. Demonstrate use of correlated subqueries and EXISTS operator.
5. Create and use views to simplify complex queries. Create indexes on selected columns to improve performance. Analyze execution plans if supported by the DBMS.
6. Design an ER model for a real-world scenario (e.g., Library or Hospital). Convert ER into relational schema and implement the schema using SQL CREATE TABLE.
7. Given a data set or set of attributes, normalize it into 1NF, 2NF, 3NF, and BCNF. Identify and explain functional dependencies and keys.
8. Create stored procedures and functions using PL/SQL or equivalent in MySQL/PostgreSQL. Demonstrate passing parameters and returning values.
9. Create BEFORE and AFTER triggers for insert/update/delete operations. Write cursors to iterate through query results and perform row-wise operations.
10. Simulate multiple users performing transactions. Demonstrate commit, rollback, and save point. Handle concurrency using isolation levels and locking.
11. **Case Study:** Choose a real-world system (e.g., Student Portal, Inventory, Hospital). Model ER diagram, create schema, write at least 5–6 queries, and explain relationships and normalization.
12. **Mini Project**

Text Books:

1. Korth, Silberschatz & Sudarshan, "Database System Concepts", McGraw Hill Education.
2. Ramez Elmasri and Shamkant B. Navathe, "Fundamentals of Database Systems", Pearson Education.

Reference Books:

1. C.J. Date, "An Introduction to Database Systems", Addison-Wesley.
2. Raghu Ramakrishnan and Johannes Gehrke, "Database Management Systems", McGraw Hill.
3. Rob and Coronel, "Database Systems: Design, Implementation, and Management", Cengage Learning.

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 INFORMATION TECHNOLOGY

Program: B. Tech. (Information Technology)								**Semester:** IV	
Course: Software Engineering								**Code:** ITPC406	
Teaching Scheme (Hrs/week)				**Evaluation Scheme (Marks)**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
03	-	-	03	40	60	-	-	-	100

Prerequisites:

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

Course Objectives:

1. Understand the role of software engineering in system development.
2. Learn different software development life cycle models.
3. Apply design methodologies, testing techniques, and quality assurance.
4. Use project management tools and principles.
5. Explore advanced practices like Agile, DevOps, and software security.

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

CO1	Define and explain the foundational principles of software engineering.
CO2	Identify and apply appropriate software process models.
CO3	Analyze and document software requirements using industry-standard techniques.
CO4	Design software using modeling tools and design patterns.
CO5	Apply software testing strategies and evaluate software quality.
CO6	Utilize modern engineering practices such as Agile, DevOps, and CI/CD pipelines.

Course Contents:

Unit	**Description**	**Duration (Hrs.)**
1.	**Introduction to Software Engineering:** Introduction to SE, Myths & Principle, Software Development Life Cycle (SDLC) Process Models: Waterfall, Incremental, Spiral, V-Model, Software Engineering Ethics (ACM/IEEE Code), Software vs. Program, Software Characteristics Introduction to CASE Tools. **Practical Task:** Compare SDLC models and identify scenarios for each	07
2.	**Requirements Engineering & Feasibility Study:** Requirement Engineering Phases, Functional & Non-functional Requirements SRS: Structure and IEEE Standard Format, Feasibility Analysis: Technical, Operational, Economic, Use Case Modeling and Requirement Tools **Practical Task:** Create SRS for a mini-project using IEEE template	07

3.	**Software Design & Modeling:** Principles of Design: Modularity, Abstraction, Coupling, Cohesion, Design Techniques: Structured and Object-Oriented, UML Diagrams: Use Case, Class, Sequence, Activity, Introduction to Design Patterns: Singleton, MVC **Practical Task:** Design UML diagrams using online tools like StarUML / draw.io	07
4.	**Software Testing and Maintenance:** Testing Lifecycle, Types of Testing (Unit, Integration, System), Testing Techniques: BVA, Equivalence Partitioning, Path Testing, White Box & Black Box Testing, Test Case Design & Test Plan Preparation, Software Maintenance, Configuration Management **Practical Task:** Develop test cases for a real-world app feature	07
5.	**Software Project Management:** Project Planning, Scheduling, Risk Management, Work Breakdown Structure (WBS), Gantt Charts, Cost Estimation Techniques (COCOMO, FP), Software Metrics: LOC, Cyclomatic Complexity, Tools: JIRA, Trello, GitHub Projects **Practical Task:** Create a project schedule using Gantt Project or similar tool	07
6.	**Advanced Trends in Software Engineering:** Agile Software Development: Scrum, Kanban, DevOps Lifecycle, CI/CD Pipelines, Software Reengineering and Reverse Engineering, Software Security: OWASP Basics **Case Study:** Modern Software Stack in Use (e.g., Netflix, Spotify)	07
	TOTAL	**42**

Text Books:

1. Roger S. Pressman, "Software Engineering: A Practitioner's Approach", McGraw Hill
2. Ian Sommerville, "Software Engineering", Pearson Education

Reference Books:

1. Pankaj Jalote, "An Integrated Approach to Software Engineering", Narosa
2. Bob Hughes, Mike Cotterell, "Software Project Management", McGraw Hill
3. Rajib Mall, "Fundamentals of Software Engineering", PHI

Digital References:

1. NPTEL – Software Engineering (IIT KGP): https://nptel.ac.in/courses/106105087
2. Coursera – Agile with Atlassian Jira (Atlassian**):** https://www.coursera.org/learn/agile-with-atlassian-jira
3. GeeksforGeeks – Software Engineering Portal**:** https://www.geeksforgeeks.org/software-engineering/
4. GitHub Education – Student Developer Pack: https://education.github.com/pack

DEPARTMENT OF INFORMATION TECHNOLOGY

Program: B. Tech. (Information Technology)									Semester: IV
Course: Computer Organization									Code: ITPC407
Teaching Scheme (Hrs/week)				**Evaluation Scheme (Marks)**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
02	02	-	03	40	60	-	25	-	125

Prerequisites:

Basic knowledge of Digital Systems Design and Architecture, Basics of Computer systems

Course Objectives:

1. Describe CPU, memory, I/O devices, and system buses.
2. Explain ALU, control units, registers, and instruction cycles.
3. Analyze instruction formats, addressing modes, interrupts, and pipelines
4. Illustrate memory hierarchy, cache mapping, and DMA techniques.
5. Differentiate I/O mechanisms and demonstrate peripheral device operations.
6. Compare ARM and RISC architectures and assess embedded applications.

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

CO1	Describe computer organization basics, CPU components, and differentiate between Von Neumann and Harvard architectures.
CO2	Explain processor structure including ALU, registers, control unit, and the instruction cycle.
CO3	Interpret machine instructions, addressing modes, interrupts, exceptions, and pipelining techniques.
CO4	Explain memory organization, cache mapping, coherence mechanisms, and DMA operations.
CO5	Compare different I/O mechanisms and demonstrate the functioning of standard peripheral devices.
CO6	Describe ARM architecture, RISC concepts, and their applications in embedded systems.

Course Contents:

Unit	**Description**	**Duration (Hrs.)**
1.	**Introduction To Computer Organization:** Basics of computer organization and architecture, CPU, Memory, I/O unit's overview System buses: Address, Data, and Control, On Neumann vs. Harvard architecture, Introduction to RISC and CISC concepts	04
2.	**Design andArchitecture of a Processor:** CISC vs RISC: basic comparison, Processor components: ALU, Registers, Control Unit, Instruction cycle: Fetch, Decode, Execute, Basics of pipelining and CPU performance, Single bus CPU organization and key register types (PC, MAR, MBR, IR), Micro-operations during instruction execution	05

3.	**Processor Instructions & Advancements:** Instruction structure, formats (0–3 address), and mnemonics Addressing modes and basic instruction types, Interrupts: types and handling (ISR, exceptions), Introduction to instruction pipelining, Basics of multiprocessor systems: MIMD and multicore overview	06
4.	**Memory Organization:** Memory hierarchy: main, cache, and registers, Memory types: SRAM, DRAM, ROM, Cache memory: mapping techniques, replacement basics, Basics of memory access and DMA (Direct Memory Access)	05
5.	**Input / Output Systems:** I/O Module, Memory-mapped I/O vs Isolated I/O Programmed I/O, Interrupt Driven I/O, Working mechanisms of peripherals: keyboard, video displays, touch screen panel, printers	04
6.	**Advanced Trends:** ARM and RISC design philosophy, Introduction to ARM processor & its versions, Features & advantages of ARM processor, Suitability of ARM processor in embedded applications, ARM 7 dataflow model, Programmers model. CPSR & SPSR registers, Modes of operation, Difference between PIC and ARM.	04
	TOTAL	**28**

List of Experiments:

Perform following assignments using virtual lab simulator (Proteus or Logisim Evolution, ARM Keil µVision)

1. Simulate basic CPU components such as ALU, registers, and control unit.
2. Simulate the fetch-decode-execute cycle of a processor.
3. Simulate instruction pipelining with stages and show data/control hazards.
4. Implement and simulate different instruction formats and addressing modes.
5. Simulate direct-mapped, associative, and set-associative cache
6. Simulate software and hardware interrupts and priorities.
7. Simulate memory-mapped vs. isolated I/O; programmed vs. interrupt-driven I/O
8. Demonstrate memory access mechanisms and control signals.
9. Explore ARM processor architecture, dataflow, and CPSR/SPSR registers.

Text Books:

1. William Stallings, "Computer Organization and Architecture: Designing for Performance", 7th Edition, Pearson Prentice Hall Publication.
2. C. Hamacher, V. Zvonko, S. Zaky, "Computer Organization", 5th Edition, Tata McGraw Hill Publication.
3. Kai Hwang, "Advanced Computer Architecture", Tata McGraw-Hill.

Reference Books:

1. Hwang and Briggs, "Computer Architecture and Parallel Processing", Tata McGraw Hill Publication.
2. A. Tanenbaum, "Structured Computer Organization", Prentice Hall Publication.
3. David A. Patterson & John L. Hennessy "Computer Organization and Design: The Hardware/Software Interface".

E-Resources:

1. NPTEL Video Lectures: www.nptelvideos.in
2. GeeksforGeeks – Computer Organization: https://www.geeksforgeeks.org/computer-organization-and-architecture
3. Udemy – Computer Organization & Architecture Courses: https://www.udemy.com

DEPARTMENT OF INFORMATION TECHNOLOGY

Program: B. Tech. (Information Technology)								**Semester:** IV	
Course: Discrete Mathematics								**Code:** ITMD402	
Teaching Scheme (Hrs/week)				**Evaluation Scheme (Marks)**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
02	-	01	03	40	60	25	-	-	125

Prerequisites:

Basic Mathematics

Course Objectives:

1. Equip students with foundational concepts of discrete mathematics essential for computing.
2. Provide understanding of sets, logic, functions, and relations to model real-world problems.
3. Develop problem-solving skills in combinatorics, graph theory, and tree structures relevant to algorithm design and analysis.
4. Foster mathematical thinking to support learning in data structures, databases, and theoretical computer science.

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

CO1	Apply the fundamentals of set theory, propositional logic, and mathematical induction in computing.
CO2	Analyze and classify different types of relations and functions with their properties and applications.
CO3	Solve counting problems using permutations, combinations, and pigeonhole principle.
CO4	Illustrate and solve problems using graph theory concepts and algorithms.
CO5	Apply algorithms on graphs for shortest path, coloring, and identify special graph types.
CO6	Explain properties of trees and apply them in decision-making and real-life computing applications.

Course Contents:

Unit	**Description**	**Duration (Hrs.)**
1.	**Set Theory and Logic:** Introduction and significance of Discrete Mathematics, **Sets**–Introduction to Set, Set Operations, Cardinality of set, Principle of inclusion and exclusion. **Propositional Logic**- logic, Propositional Equivalences, Application of Propositional Logic Translating English Sentences **Mathematical Induction**- Mathematical Induction. and Strong Mathematical Induction and Examples	04

2.	**Relations and Functions:** **Relations** – Relations and their Properties, Representing relations, Closures of relations, Equivalence relations, Partial orderings, Partitions, Hasse diagram, Lattices, Chains and Anti-Chains. Functions and its types**,** The Pigeonhole Principle. **Recurrence Relations:** Recurrence Relation, Linear Recurrence Relations with Constant Coefficients, Total Solutions, Applications of Relations and Functions	04
3.	**Counting Principles:** The Basics of Counting, rule of Sum and Product, Permutations and Combinations, Binomial Coefficients and Identities, Generalized Permutations and Combinations, Algorithms for generating Permutations and Combinations.	05
4.	**Graph Theory:** Graph Terminology and Special Types of Graphs, Matrix Representing Graphs and handshaking lemma, Graph Isomorphism, Connectivity	05
5.	**Algebra of Graphs:** Euler and Hamilton Paths, Single source shortest path- Dijkstra's Algorithm, Planar Graphs, Graph Colouring	05
6.	**Trees:** Introduction to Trees and Properties decision tree, prefix codes and Huffman coding, Decision Trees and their Applications in Machine Learning, Applications of Trees in File Systems, cut sets, The Max flow- Min Cut Theorem in Transport network.	05
	TOTAL	**28**

Text Books:

1. Kenneth H. Rosen, "Discrete Mathematics and Its Applications", McGraw-Hill Education.
2. C. L. Liu, "Elements of Discrete Mathematics", McGraw-Hill

Reference Books:

1. R.K. Jain and S.R.K. Iyengar, "Advanced Engineering Mathematics", Narosa Publishing House.
2. Tremblay J.P. and Manohar, "Discrete Mathematical Structures with Applications to Computer Science", McGraw Hill.
3. Norman L. Biggs, "Discrete Mathematics", Oxford University Press.

E-Resources:

1. NPTEL: Discrete Mathematics – IIT Ropar (Prof. Sudarshan Iyengar) :https://nptel.ac.in/courses/106106183
2. Discrete Mathematical Structures – IIT Madras : https://nptel.ac.in/courses/106106094
3. SWAYAM – Discrete Mathematics : https://swayam.gov.in/nd1_noc24_cs56/preview

DEPARTMENT OF INFORMATION TECHNOLOGY

Program: B. Tech. (Information Technology)							**Semester:** IV		
Course: E-Commerce							**Code:** ITMC402		
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
01	01	-	02	-	-	25	-	-	25

Prerequisites:

1. Basic knowledge of the Internet and the World Wide Web
2. Familiarity with business or commerce principles
3. 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: After completion of this course, students will 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.	02
2.	**E-Commerce Business Models:** B2B, B2C, C2C, C2B, G2C; revenue models; case studies (Amazon, Flipkart, Meesho, etc.); advantages and challenges.	02
3.	**E-Commerce Infrastructure:** Internet and WWW, web hosting, domain registration, client-server architecture; overview of E-Commerce platforms (Shopify, Wix, WooCommerce).	03
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.	02
5.	**Legal, Ethical & Regulatory Issues:** Overview of IT Act 2000, data protection laws (GDPR); privacy concerns; ethical	03

	and IPR issues in digital business.	
6.	**Digital Marketing & Strategy:** Introduction to SEO, SEM, affiliate marketing, email campaigns, and social media marketing; use of Google Ads, Meta Ads, CRM tools.	02
	TOTAL	**14**

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 INFORMATION TECHNOLOGY

Program: B. Tech. (Information Technology)								**Semester:** IV	
Course: Quantitative Aptitude & Numerical Analysis								**Code:** ITAE402	
Teaching Scheme (Hrs/week)				**Evaluation Scheme (Marks)**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
-	02	-	01	-	-	25	-	-	25

Prerequisites:

1. Basic knowledge of Mathematics at Higher Secondary Level
2. Familiarity 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: After completion of this course, students will 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 Practical Examination

The practical examination should test the student's ability to solve given numerical and aptitude problems accurately within a limited time frame. Along with the problem-solving part, an oral examination should be conducted to check the student's understanding of the methods and logic

applied during the practical's. The exam should ensure that students can confidently apply the techniques learned during the course.

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.

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

DEPARTMENT OF INFORMATION TECHNOLOGY

Program: B. Tech. (Information Technology)								**Semester:** IV	
Course: Programming Lab – II (Python Programming)								**Code:** ITVS404	
Teaching Scheme (Hrs/week)				**Evaluation Scheme (Marks)**					
Lecture	**Practical**	**Tutorial**	**Credit**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**Total**
-	04	-	02	-	-	25	-	25	50

Prerequisites:

Basic understanding of programming logic (preferably C/C++), 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: After completion of this course, students will 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:

Unit	**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	08
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.	08
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	10

4.	**Object Oriented Programming in Python:** Classes and objects, constructor, instance and class variables, inheritance, method overriding, polymorphism, encapsulation.	10
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	10
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.	10
	TOTAL	**56**

List of Assignments: - (Perform Any 10 Practical's)

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 INFORMATION TECHNOLOGY

Program: B. Tech. (Information Technology)								**Semester:** IV	
Course: Internship – III								**Code:** ITIN403	
Teaching Scheme (Hrs/week)				**Evaluation Scheme (Marks)**					
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.