



A CMI INSTITUTION

CHRIST COLLEGE OF ENGINEERING (AUTONOMOUS)

MASTER OF COMPUTER APPLICATIONS

CURRICULUM AND SYLLABUS

2026 REGULATION



A CMI INSTITUTION

CHRIST COLLEGE OF ENGINEERING (AUTONOMOUS)

MASTER OF COMPUTER APPLICATIONS

2026 REGULATION (SEMESTER 1)

Approved by BoS meeting held on: 20/06/2026

Approved by Academic Council meeting held on: 23/06/2026

BoS Chairperson

Dr. Dincy R Arikkat

Director of Academics

Dr. Manoj George



Principal

Dr. Soloman P. A.

FIRST SEMESTER														
Sl.No	Slot	Course Code	Course Type	Course Category	Course Title (Course Name)	Credit Structure				SS	Total Marks		Credits	Hrs./Week
						L	T	P	R		CAE	ESE		
THEORY														
1	A	CA26MFT101	BS	GC	Mathematical Foundations for Computing	3	1	0	0	5	40	60	4	4
2	B	CA26OST102	PC	PC	Operating Systems	3	1	0	0	5	40	60	4	4
3	C	CA26DST103	PC	PC	Advanced Data Structures	3	1	0	0	5	40	60	4	4
4	D	CA26SET104	PC	PC	Software Engineering and DevOps	3	1	0	0	5	40	60	4	4
PRACTICALS														
5	R	CA26PYP105	PC	PC	Python Programming Lab	0	1	3	0	2	50	50	2	4
6	S	CA26WPP106	PC	PC	Web Programming Lab	0	1	3	0	2	50	50	2	4
7	T	CA26DSP107	PC	PC	Data Structures using C Lab	0	1	3	0	2	50	50	2	4
Total										26			22	28

SEMESTER S1**Mathematical Foundations for Computing**

Course Code	CA26MFT101	CAE Marks	40
Teaching Hours/Week (L: T:P: R)	3:1:0:0	ESE Marks	60
Credits	4	Exam Hours	3 Hrs.
Prerequisites (if any)	Basic knowledge in set, statistics	Course Type	Theory

Preamble: This course introduces students to some basic mathematical ideas and tools which are at the core of MCA course. It introduces the concepts of linear algebra, graph theory and statistics.

Prerequisite: A basic course in set theory and statistics.

Course Objectives:

The course aims to:

1. To develop a strong foundation in linear algebra concepts and techniques for solving systems of linear equations and analyzing matrix properties.
2. To introduce fundamental concepts of probability theory and probability distributions used in modelling uncertainty.
3. To provide an understanding of multivariable calculus and optimization methods used in computational and data-driven applications.
4. To familiarize students with graph theory concepts and their applications in modeling and solving computing and network-related problems.
5. To equip students with mathematical tools and analytical skills required for advanced studies in computer science, data science, and artificial intelligence.

Course Outcomes (COs)

At the end of the course students should be able to:

Course Outcome		Bloom's Knowledge Level (KL)
CO1	Apply matrix operations, Gaussian elimination, row echelon forms, rank, eigenvalues, eigenvectors, and diagonalization techniques to solve systems of linear equations and related computational problems.	K3

CO2	Analyse uncertain events using probability concepts, conditional probability, Bayes theorem, and common discrete and continuous probability distributions.	K4
CO3	Apply concepts of multivariable calculus, including partial derivatives, gradients, Hessians, and chain rule, to solve optimization problems in computational contexts.	K3
CO4	Analyse and evaluate optimization problems using convexity concepts, gradient descent algorithms, and their variants for finding optimal solutions.	K5
CO5	Analyse graph structures and apply graph-theoretic concepts such as connectivity, graph representations, shortest paths, Euler and Hamilton paths, and planar graphs to model and solve computing problems.	K4

Note: K1- Remember; K2- Understand, K3- Apply, K4- Analyse, K5- Evaluate, K6- Create

CO-PO Mapping Table:

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PSO1	PSO2
CO1	3	2	2	2				1	3	2
CO2	3	3	2	3			2	1	2	3
CO3	3	2	2	2				1	2	1
CO4	3	3	3	3		1		2	3	2
CO5	3	3	2	2	2	3		1	3	2

SYLLABUS

Module No.	Syllabus Description	Contact Hours
1	Linear system of equations, coefficient matrix, augmented matrix, Gauss elimination method and back substitution, elementary row operations, row equivalent systems, Gauss elimination- three possible cases, Row Echelon form and information from it, Linear independence- rank of a matrix. Solution of linear system, fundamental theorem of non- homogeneous linear system (without proof). Homogeneous linear system (theory only), Matrix eigen value problem- determination of eigen values and eigen vectors, Basis of eigen vectors- diagonalization of matrix- Quadratic form principle axis theorem (without proof).	12

2	Basics of probability theory, joint, marginal and conditional probability, Bayes theorem, discrete probability distributions (Binomial and Poisson distributions), and continuous probability distributions (Normal and Exponential distributions).	11
3	Functions of one and several variables, limits and continuity, partial derivatives, gradients and Hessians, chain rule, optimization of multivariable functions, convex and non-convex functions, gradient descent algorithm and its variants.	11
4	Graphs and Graph Models, Graph Terminology and Special Types of Graphs, Representing Graphs and Graph Isomorphism, Connectivity, Euler and Hamilton Paths, Shortest-Path Problems, Planar Graphs, Directed Graph, Multigraph, Connected graph, Euler circuit and trail, Planar and Non-planar Graphs.	11

Course Assessment Method

(CAE: 40 Marks, ESE: 60 Marks)

Attendance	Assignment/ Quiz/ Course Project	Continuous Evaluation Test -I (Written)	Continuous Evaluation Test -II (Written)	Total
8	12	50	50	
		A total of 100 marks will be scaled to 20		40

End Semester Examination Marks (ESE)

In Part A, all questions need to be answered and in Part B, each student can choose any one full question out of two questions.

Part A	Part B	Total
2 questions from each module. - Total of 8 questions, each carrying 3 marks (8 x 3 = 24 Marks)	- Each question carries 9 marks. - Two questions will be given from each module, out of which 1 question should be answered. - Each question can have a maximum of 3 sub divisions. (4 x 9 = 36 Marks)	60

Text Books

Sl. No.	Title of the Book	Name of the Author(s)	Name of the Publisher	Edition and Year
1	Elementary Number Theory	David M. Burton	McGraw-Hill	7 th Edition, 2012
2	Discrete and Computational Mathematics: An applied introduction	Ralph P Grimaldi	Pearson Education	5 th Edition, 2007
3	Mathematics for Machine Learning	Marc Peter Deisenroth, A. Aldo Faisal, Cheng Soon Ong	Cambridge University Press	1 st Edition, 2020
4	Advanced Engineering Mathematics	Kreyszig, E.	John Wiley & Sons	10 th Edition, 2014
5	Fundamentals of Mathematical Statistics	Gupta S.C and Kapoor V .K	Sultan Chand and Sons	11 th Edition

Reference Books

Sl. No	Title of the Book	Name of the Author/s	Name of the Publisher	Edition and Year
1	Mathematics for Machine Learning: Hands-On for the Curious Data Scientist	Davani, J.	Packt Publishing	1 st Edition, 2020
2	Discrete Mathematics and Its Applications	Rosen, K. H.	McGraw-Hill Education	8 th Edition, 2019
3	Introduction to the Practice of Statistics	David S. Moore, George P. McCabe	W.H. Freeman & Company	5 th Edition, 2005
4	Introduction to Probability and Statistics Using R	G. Jay Kerns	Chapman & Hall	2010
5	Introduction to Linear Algebra	Strang G.	Wellesley-Cambridge Press	5 th Edition, 2016

Video Links (NPTEL, SWAYAM...)

Sl. No.	Source	URL
1	Linear Algebra	https://onlinecourses.nptel.ac.in/e-learning/preview/noc26_ma100
2	Mathematics for Machine Learning	https://onlinecourses.nptel.ac.in/e-learning/preview/noc26_ma129

SEMESTER S1

Operating Systems

Course Code	CA26OST102	CAE Marks	40
Teaching Hours/Week (L: T:P: R)	3:1:0:0	ESE Marks	60
Credits	4	Exam Hours	3 Hrs.
Prerequisites (if any)	None	Course Type	Theory

Preamble:

This course introduces the fundamental concepts of Operating Systems, including process management, memory management, file systems, synchronization, and security. It provides essential knowledge required to understand how modern computer systems operate and manage resources efficiently.

Prerequisites: None

Course Objectives:

The course aims to:

1. Understand the fundamental concepts of operating systems and types of advanced operating systems.
2. Gain knowledge of process management concepts and process synchronization using semaphores and monitors, deadlock handling techniques.
3. Understand, distributed operating systems, virtualization, and cloud computing concepts.
4. Understand and apply memory management techniques.
5. Understand file system organization and operating system security mechanisms.

Course Outcomes (COs)

At the end of the course students should be able to:

Course Outcome		Bloom's Knowledge Level (KL)
CO1	Understand operating system concepts, functions, design approaches, multiprogramming, multiprocessing, and process fundamentals.	K2
CO2	Apply CPU scheduling algorithms, process synchronization techniques using semaphores and monitors, inter-process communication mechanisms, and deadlock handling methods.	K3
CO3	Analyse distributed operating system concepts, cloud computing service and deployment models, and virtualization technologies.	K4

CO4	Apply memory management techniques such as paging, segmentation, demand paging, and page replacement algorithms.	K3
CO5	Apply file system and security concepts, including directory structures, file allocation methods, file system implementation, and access control models.	K3

Note: K1- Remember, K2- Understand, K3- Apply, K4- Analyse, K5- Evaluate, K6- Create

CO-PO Mapping Table:

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PSO1	PSO2
CO1	3	2	1	1				1	2	
CO2	2	3	2	2	1			1	3	1
CO3	2	3	2	3				2	3	2
CO4	2	3	2	2			3	1	3	1
CO5	2	2	2	2				2	3	2

SYLLABUS

Module No.	Syllabus Description	Contact Hours
1	Introduction to Operating Systems, functions of Operating System, design approaches and types of operating systems. Dual-mode operation, concept of multiprogramming and multiprocessing. Concept of processes and threads, process states and process state transition diagram, Process Control Block, process context. CPU scheduling and process scheduling. The Critical Section Problem. Process synchronization using semaphores and monitors. Inter-Process Communication (IPC): pipes, message passing, shared memory, sockets.	12
2	Deadlock: characterization, necessary conditions, prevention, avoidance, detection and recovery. Distributed Operating Systems: issues in distributed operating systems. Cloud Computing: service models and deployment models (overview only). Virtualization: concept of virtualization, hypervisors (Type 1 and Type 2), OS-level virtualization (containers overview), virtual machines vs containers.	11

3	Memory Management - Types of memory, Memory organization, Address binding, Memory Partitioning, Dynamic memory Partitioning, buddy system, Paging, Demand Paging, Segmentation, Page replacement algorithms. I/O Management: Basic concepts of I/O hardware, I/O system organization. Disk Scheduling Algorithms: FCFS, SSTF, SCAN, C-SCAN.	10
4	File System - Directory structure - single level, two-level, tree, acyclic graph, general graph; File system mounting, Implementing File System: File system structure - Layered file system, file attributes, File control block; Directory Implementation, Allocation Methods. Security and Protection - Goals, Principles in normal OS for security, Access Control models and methods.	12

Course Assessment Method

(CAE: 40 Marks, ESE: 60 Marks)

Attendance	Assignment/ Quiz/ Course Project	Continuous Evaluation Test -I (Written)	Continuous Evaluation Test -II (Written)	Total
8	12	50	50	
		A total of 100 marks will be scaled to 20		40

End Semester Examination Marks (ESE)

In Part A, all questions need to be answered and in Part B, each student can choose any one full question out of two questions.

Part A	Part B	Total
2 questions from each module. - Total of 8 questions, each carrying 3 marks (8 x 3 = 24 Marks)	- Each question carries 9 marks. - Two questions will be given from each module, out of which 1 question should be answered. - Each question can have a maximum of 3 sub divisions. (4 x 9 = 36 Marks)	60

Text Books

Sl. No.	Title of the Book	Name of the Author(s)	Name of the Publisher	Edition and Year
1	Operating System Concepts	Abraham Silberschatz, Peter Baer Galvin, Greg Gagne	Wiley India	9th Edition, 2015
2	Modern Operating Systems	Andrew S. Tanenbaum, Herbert Bos	Pearson	5 th Edition, 2022

Reference Books

Sl. No.	Title of the Book	Name of the Author(s)	Name of the Publisher	Edition and Year
1	An Introduction to Operating Systems: Concepts and Practice	P. C. P. Bhatt	Prentice Hall of India	5 th Edition, 2019
2	Operating Systems: Internals and Design Principles	William Stallings	Pearson	7 th Edition, 2012
3	Operating Systems: Three Easy Pieces	Remzi H. Arpaci-Dusseu, Andrea C. Arpaci-Dusseu	Arpaci-Dusseu Books	1 st Edition, 2023

Video Links (NPTEL, SWAYAM...)

Sl. No.	Source	URL
1	Operating Systems	nptel.ac.in/courses/106106144
2	Operating System Fundamentals	nptel.ac.in/courses/106105214

SEMESTER S1

Advanced Data Structures

Course Code	CA26DST103	CAE Marks	40
Teaching Hours/Week (L: T:P: R)	3:1:0:0	ESE Marks	60
Credits	4	Exam Hours	3 Hrs.
Prerequisites (if any)	Basic data structures.	Course Type	Theory

Preamble: Advanced Data Structures is an extension of the fundamental concepts of data structures and algorithms studied at the undergraduate level. The course introduces modern and efficient data structures and associated algorithms for solving complex computational problems. It focuses on the design, analysis, and implementation of advanced structures that support efficient storage, retrieval, and processing of data in real-world applications.

Prerequisite: Basic Data Structures

Course Objectives:

The course aims to:

1. Develop a strong foundation in the analysis of algorithms, asymptotic complexity, hashing techniques, amortized analysis, and disjoint-set data structures.
2. Understand and apply advanced tree-based data structures, including AVL trees, Red-Black trees, B-Trees, Splay trees, and Suffix trees for efficient data storage and retrieval.
3. Study priority queues and advanced heap structures, including Binary Heaps, Binomial Heaps, and Fibonacci Heaps, and analyze their performance characteristics.
4. Develop problem-solving skills using graph representations and graph algorithms.
5. Introduce advanced data structures for efficient query processing and enable students to select appropriate data structures for solving complex computing problems.

Course Outcomes (COs)

At the end of the course students should be able to:

Course Outcome		Bloom's Knowledge Level (KL)
CO1	Explain the concepts of algorithm analysis, hashing, amortized analysis, and disjoint-set data structures.	K2
CO2	Apply advanced tree structures such as AVL trees, Red-Black trees, B-Trees, B+ Tree, and Splay trees for efficient data storage and retrieval.	K3

CO3	Analyse the performance and operations of Binary Heaps, Binomial Heaps, and Fibonacci Heaps used in priority queue implementations.	K4
CO4	Analyse graph algorithms for traversal, connectivity, minimum spanning tree construction, and shortest path computation.	K4
CO5	Apply advanced data structures and graph algorithms to design efficient solutions for real-world computing and optimization problems.	K3

Note: K1- Remember, K2- Understand, K3- Apply, K4- Analyse, K5- Evaluate, K6- Create

CO-PO Mapping Table:

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PSO1	PSO2
CO1	3	2	1	2				1	2	
CO2	2	3	3	2	1			1	3	1
CO3	2	3	2	2				2	3	1
CO4	2	3	3	2	1			2	3	2
CO5	2	3	3	3	2	1		2	3	2

SYLLABUS

Module No.	Syllabus Description	Contact Hours
1	Review of basic data structures- Array, Linked List and its variants, Stack, Queue and Trees. Introduction to algorithm analysis, asymptotic notations and complexity analysis. Hash tables, hash functions, separate chaining, open addressing and rehashing. Amortized analysis using aggregate, accounting and potential methods. Disjoint sets - representations, Union, Find algorithms. Introduction to Trie (basic structure and applications)	12
2	Binary Search Trees, AVL trees, tree rotations, insertion and deletion operations. Red-Black trees- Properties of Red Black trees, Rotations, Insertion, Deletion. B-Trees- Insertion and Deletion. B+ Trees – basic concept and database indexing. Introduction to Splay Trees and Suffix Trees.	12
3	Binary heaps and priority queues. Mergeable Heaps and operations on Mergeable Heaps. Binomial Heaps, Binomial Heap operations and Analysis, Fibonacci Heaps, Fibonacci Heap operations and Analysis.	11

4	Advanced Graph Structures: Representation of graphs, Depth First and Breadth First Traversals, Topological Sorting, strongly connected Components and Biconnected Components Minimum Cost Spanning Tree algorithms- Prim's Algorithm, Kruskal's Algorithm, Shortest Path Finding algorithms – Dijkstra's single source shortest paths algorithm. Range Query Processing: Segment Trees.	10
---	---	----

Course Assessment Method

(CAE: 40 Marks, ESE: 60 Marks)

Attendance	Assignment/ Quiz/ Course Project	Continuous Evaluation Test -I (Written)	Continuous Evaluation Test -II (Written)	Total
8	12	50	50	
		A total of 100 marks will be scaled to 20		40

End Semester Examination Marks (ESE)

In Part A, all questions need to be answered and in Part B, each student can choose any one full question out of two questions.

Part A	Part B	Total
2 questions from each module. - Total of 8 questions, each carrying 3 marks (8 x 3 = 24 Marks)	- Each question carries 9 marks. - Two questions will be given from each module, out of which 1 question should be answered. - Each question can have a maximum of 3 sub divisions. (4 x 9 = 36 Marks)	60

Text Books

Sl. No.	Title of the Book	Name of the Author(s)	Name of the Publisher	Edition and Year
1	Data Structures Using C	Reema Thareja	Reema Thareja	2 nd Edition, 2014
2	Introduction to Algorithms	Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein	MIT Press	4 th Edition, 2022
3	Data Structures and Algorithm Analysis in C	Mark Allen Weiss	Addison-Wesley	2 nd Edition, 1997

Reference Books

Sl. No.	Title of the Book	Name of the Author(s)	Name of the Publisher	Edition and Year
1	Data Structures and Algorithms in Python	Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser	Wiley	1 st Edition, 2013
2	Data Structures and Algorithm Analysis in C++	Mark A. Weiss	Pearson Education	4 th Edition, 2014
3	Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology	Dan Gusfield	Cambridge University Press	1st Edition, 1997
5	Algorithm Design	Jon Kleinberg, Eva Tardos	Pearson Education India	1st Edition, 2006
6	Data Structures and Algorithms	Alfred V. Aho, John E. Hopcroft, Jeffrey D. Ullman	Addison-Wesley	1st Edition, 1983

Video Links (NPTEL, SWAYAM...)

Sl. No.	Source	URL
1	Data Structures and Algorithms	nptel.ac.in/courses/106102064
2	Data Structures and Algorithms Design	https://onlinecourses.nptel.ac.in/e-learning/preview/noc25_cs81
3	Data Structure and Algorithms using Java	nptel.ac.in/courses/106105225

SEMESTER S1

Software Engineering and DevOps

Course Code	CA26SET104	CAE Marks	40
Teaching Hours/Week (L: T:P: R)	3:1:0:0	ESE Marks	60
Credits	4	Exam Hours	3 Hrs.
Prerequisites (if any)	Knowledge of programming language concepts	Course Type	Theory

Preamble:

This course introduces the fundamental principles of software engineering and DevOps for the systematic development, testing, deployment, and maintenance of software systems. It emphasizes modern development practices, quality assurance, collaboration, and automation techniques used in contemporary software industries.

Prerequisite: Basic knowledge of programming concepts and proficiency in at least one programming language.

Course Objectives:

The course aims to:

1. Provide an understanding of software engineering principles, software development life cycle models, agile methodologies, and software project management techniques for developing quality software systems.
2. Develop the ability to elicit, analyze, specify, and design software requirements using appropriate modeling and design techniques.
3. Introduce software quality assurance practices, software testing methodologies, and version control systems for ensuring software reliability and maintainability.
4. Familiarize students with DevOps culture, continuous integration, continuous delivery, and automation practices used in modern software development environments.
5. Enable students to apply software engineering and DevOps tools, techniques, and best practices for the efficient development, deployment, and maintenance of software systems.

Course Outcomes (COs)

At the end of the course students should be able to:

Course Outcome		Bloom's Knowledge Level (KL)
CO1	Explain software development life cycle models, agile practices, and software project planning techniques.	K2

CO2	Apply requirements analysis and software design methodologies to model software systems.	K3
CO3	Analyse software quality assurance activities, testing strategies, and version control workflows.	K4
CO4	Analyse DevOps workflows and Continuous Integration/Continuous Delivery (CI/CD) practices for software delivery automation.	K4
CO5	Develop and manage software systems through a mini/micro project through a mini/micro project by applying software engineering and DevOps practices, tools, and collaborative workflows.	K6

Note: K1- Remember, K2- Understand, K3- Apply, K4- Analyse, K5- Evaluate, K6- Create

CO-PO Mapping Table:

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PSO1	PSO2
CO1	2	2	1	1	2	3	1	2	2	
CO2	2	3	3	2	1	1			3	1
CO3	1	3	2	3	2	1	2	2	3	1
CO4	1	2	3	3	2	2	1	2	3	1
CO5	2	3	3	3	3	3	2	3	3	2

SYLLABUS

Module No.	Syllabus Description	Contact Hours
1	Introduction to Software Engineering, Characteristics of Software, Software Development Life Cycle (SDLC), Waterfall Model, Incremental Model, Spiral Model, Agile Model, Agile Manifesto, Agile Principles, Scrum Framework, Sprint Planning, Agile Estimation Techniques and Velocity-Based Planning, Product Backlog, Sprint Backlog, Software Project Planning, Project Scope, Software Metrics, Software Cost Estimation, and COCOMO Model.	12
2	Requirements Engineering, Requirements Elicitation, Software Requirements Specification (SRS), Requirements Modelling, Use Cases, Software Design Concepts, Architectural Design, UML Diagrams, Object-Oriented Design Principles, Design Patterns, Software Documentation, and Coding Standards.	11

3	Software Quality Assurance, Software Reviews, Walkthroughs, Inspections, Software Testing Principles, Black Box Testing, White Box Testing, Unit Testing, Regression Testing, Defect Life Cycle, Software Configuration Management, Build Management, Version Control Systems, Distributed Version Control Systems, Git Concepts, GitHub Repository Management, Branching, Merging, and Conflict Resolution.	10
4	Introduction to DevOps, DevOps Lifecycle, Collaboration and Automation, Dependency Management, Continuous Integration Concepts, CI Pipelines, Automated Build Process, Automated Testing, Introduction to Jenkins and CI Tools, DevSecOps Concepts and Security in CI/CD Pipelines, Continuous Delivery and Continuous Deployment Concepts, Infrastructure as Code, Configuration Management, Introduction to Ansible, Test Automation Concepts, Test Automation Tools and Frameworks, Containerization using Docker, Monitoring and Logging Concepts, DevOps Toolchains, and Industry Applications.	12

Course Assessment Method

(CAE: 40 Marks, ESE: 60 Marks)

Attendance	Assignment/ Quiz/ Course Project	Continuous Evaluation Test -I (Written)	Continuous Evaluation Test -II (Written)	Total
8	12	50	50	
		A total of 100 marks will be scaled to 20		40

End Semester Examination Marks (ESE)

In Part A, all questions need to be answered and in Part B, each student can choose any one full question out of two questions.

Part A	Part B	Total
2 questions from each module. - Total of 8 questions, each carrying 3 marks (8 x 3 = 24 Marks)	- Each question carries 9 marks. - Two questions will be given from each module, out of which 1 question should be answered. - Each question can have a maximum of 3 sub divisions. (4 x 9 = 36 Marks)	60

Text Books

Sl. No.	Title of the Book	Name of the Author(s)	Name of the Publisher	Edition and Year
1	Software Engineering	Ian Sommerville	Pearson Education	10 th Edition
2	The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations	Gene Kim, Jez Humble, Patrick Debois, John Willis	IT Revolution Press	2 nd Edition, 2021
3	Software Engineering: A Practitioner's Approach	Roger S. Pressman, Bruce R. Maxim	McGraw-Hill Education	8 th Edition, 2014

Reference Books

Sl. No.	Title of the Book	Name of the Author(s)	Name of the Publisher	Edition and Year
1	Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation	Jez Humble, David Farley	Addison-Wesley	1 st Edition, 2010
2	Pro Git	Scott Chacon, Ben Straub	Apress	2 nd Edition, 2014
3	Clean Code: A Handbook of Agile Software Craftsmanship	Robert C. Martin	Addison-Wesley Professional	1 st Edition, 2008

Video Links (NPTEL, SWAYAM...)

Sl. No.	Source	URL
1	Software Engineering	https://onlinecourses.nptel.ac.in/e-learning/preview/noc25_cs108
2	Software Testing & Quality Assurance	nptel.ac.in/courses/106105192
3	Introduction to DevOps Coursera	https://www.coursera.org/learn/intro-to-devops?msocid=05007000b0196e2111f657fb1b46f62

SEMESTER S1

Python Programming Lab

Course Code	CA26PYP105	CAE Marks	50
Teaching Hours/Week (L: T:P: R)	0:1:3:0	ESE Marks	50
Credits	2	Exam Hours	3 Hrs.
Prerequisites (if any)	None	Course Type	Practical

Preamble: This course introduces the fundamentals of programming and develops logical thinking and problem-solving skills using the Python programming language. It enables students to write, test, and debug Python programs effectively.

Prerequisite: None

Course Objectives:

The course aims to:

1. Introduce fundamental programming concepts using Python for problem-solving.
2. Develop skills in implementing control structures, functions, and data structures in Python.
3. Enable students to design modular and object-oriented programs using Python.
4. Provide hands-on experience in file handling, exception handling, and data validation.
5. Apply Python programming tools and libraries to perform data analysis, visualization, and real-world problem solving.

Course Outcomes (COs)

At the end of the course students should be able to:

Course Outcome		Bloom's Knowledge Level (KL)
CO1	Apply basic Python programming constructs such as variables, operators, decision-making, and looping statements to solve simple computational problems.	K3
CO2	Analyse and implement Python data structures such as lists, tuples, dictionaries, and comprehensions for problem-solving.	K4

CO3	Design and implement modular and object-oriented programs using classes, objects, and functions to solve real-world problems.	K6
CO4	Apply file handling and exception handling techniques to develop robust Python applications with proper error handling and data storage.	K4
CO5	Apply Python libraries such as NumPy, Pandas, Matplotlib, and external APIs to perform data analysis, visualization, web interaction, and automation tasks.	K3

Note: K1- Remember, K2- Understand, K3- Apply, K4- Analyse, K5- Evaluate, K6- Create

CO-PO Mapping Table:

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PSO1	PSO2
CO1	3	3	2	3	1			2	2	1
CO2	3	3	2	3	2			2	2	2
CO3	3	3	3	3	2	1		2	3	2
CO4	3	3	2	3	1		2	2	2	1
CO5	3	3	3	3	2	1	2	3	3	3

SYLLABUS

Module No.	Syllabus Description	Contact Hours
1	Introduction to Python programming, data types, variables, operators, input and output operations. Control structures including decision-making and looping statements.	10
2	Built-in data structures such as lists, tuples, dictionaries, and sets. Operations including searching, sorting, updating, and filtering. String processing techniques and list comprehensions.	10
3	Introduction to object-oriented programming concepts including classes, objects, constructors, and methods. File handling operations such as reading, writing, and updating files. Exception handling using try, except, else, and finally blocks.	10

4	Working with external data formats and libraries for data handling, analysis, and visualization. Introduction to network-based data handling, web interaction, and automation concepts. Development of applications involving structured data processing and visualization.	15
---	---	----

List of Laboratory Experiments (Minimum 15)

1. Develop a Python program to solve basic computational problems by accepting user input and performing arithmetic and logical operations such as finding the area of a circle, checking leap years, and determining the largest of three numbers.
2. Develop a Python program to generate the factorial of a given number using looping statements.
3. Develop a Python program to generate Fibonacci series up to n terms using looping statements.
4. Develop a Python program to check whether a given number is a prime number or not using loops.
5. Write Python programs to perform string processing operations, including character replacement, string reversal, checking palindromes, and counting character and word frequencies.
6. Develop Python programs using lists, list comprehensions, and built-in list methods to create, search, update, filter, and manipulate collections of data.
7. Write Python programs to perform operations on dictionaries and sets, including sorting, merging, frequency counting, and identifying common or unique elements.
8. Write a program which accepts a sequence of comma-separated numbers from console and generate a list and a tuple which contains every number. Suppose the following input is supplied to the program: 34, 67, 55, 33, 12, 98. Then, the output should be: ['34', '67', '55', '33', '12', '98'] ('34','67', '55', '33', '12', '98'). Write a Python program that accepts a sequence of comma-separated numbers from the console and generates a list and a tuple containing every number. Display the generated list and tuple, the total number of elements, the first and last elements, and the list in reverse order.
9. Develop a modular Python program for a Student Performance Analysis System using user-defined functions and lambda expressions. The program should accept marks of students, compute the total, average, highest, and lowest marks using user-defined functions, determine the grade, and use lambda expressions to sort students based on their total marks.
10. Write a Python program to create a BankAccount class with attributes such as account number, account holder name, account type, and balance. Use a constructor to initialize the attributes and implement methods to deposit money, withdraw money, and display the account details and current balance. Create one or more objects to demonstrate the functionality.

11. Develop a Python program to maintain a library catalogue using text files. The program should store book details in a text file, allow new books to be added, search for books by title or author, update the catalogue by appending new entries, and generate reports such as the total number of books or books by a specific author.
12. Develop a Python program to calculate the electricity bill based on the number of units consumed. Handle exceptions such as invalid input and negative unit values using try, except, else and finally.
13. Develop a Python program using NumPy to simulate basic image processing operations by representing an image as a matrix of pixel values. The program should perform operations such as reshaping the image matrix, slicing specific regions, applying brightness adjustments using broadcasting, and performing matrix multiplication for transformation effects.
14. Develop a Python program to process weather information obtained in sample JSON dataset or public weather API. The program should parse JSON data to extract temperature, humidity, and location details, convert nested JSON structures into a tabular format using Python, and display summarized weather reports for multiple cities.
15. Develop a Python program for an input validation system used in online registration portals where the program validates user email addresses, detects valid phone numbers, and extracts transaction IDs from logs using regular expressions. The system should ensure proper data format validation for secure and clean data entry.
16. Develop a Python program using the requests library to access the VirusTotal REST API. The program should accept a SHA-256 file hash as input, send an authenticated HTTP GET request using an API key, parse the JSON response, and display the number of malicious detections (last_analysis_stats) for the specified file.
17. Develop a Python program to extract and analyse real-time news headlines from a publicly available news website using the BeautifulSoup library. The program should scrape headline text, publication date, and category information from HTML pages, store the extracted data in a CSV file.
18. Develop a Python program using Selenium WebDriver to automate data entry in the DemoQA Text Box page (<https://demoqa.com/text-box>). The program should launch a web browser, navigate to the web page, enter the user's full name, email address, current address, and permanent address, submit the form, verify that the entered details are displayed correctly, and close the browser.
19. Develop a Python program to analyse student academic performance using a CSV dataset from web or a sample dataset. The program should use Pandas to filter subject-wise marks, NumPy to compute mean, median, and standard deviation, and Matplotlib to generate a bar chart comparing average marks across subjects.
20. Develop a Python program using Pandas and Matplotlib to analyse monthly temperature data of a city. The program should read a CSV file, extract monthly temperature values, and generate a line chart to show temperature trends over a year. The program should also identify the hottest and coldest months.

Course Assessment Method**(CAE: 50 Marks, ESE: 50 Marks)**

Attendance	Maintenance of daily lab record and GitHub management	Regular class viva	Timely completion of day-to-day tasks	Tests/Evaluation	Total
10	10	5	10	15	50

End Semester Examination Marks (ESE)

Maximum Marks: 50			
Verification of Daily program record and Git Repository			5 Marks
Viva			10 Marks
Problem solving (Based on difficulty level, one or more questions may be given)	Flowchart / Algorithm / Structured description of problem to explain how the problem can be solved / Interface Design	15	35 Marks
	Program correctness	50	
	Code efficiency	15	
	Formatted output and pushing to remote Git repository	20	
Total Marks			50 Marks

Text Books

Sl. No.	Title of the Book	Name of the Author(s)	Name of the Publisher	Edition and Year
1	Python Programming: A Modern Approach	Vamsi Kurama	Pearson	2017
2	Core Python Applications Programming	Wesley J. Chun	Pearson Education	3rd Edition, 2012

Reference Books

Sl. No.	Title of the Book	Name of the Author(s)	Name of the Publisher	Edition and Year
1	Introduction to Python Programming	Gowrishankar S, Veena A	CRC Press	2018
2	Think Python	Allen B. Downey	O'Reilly Media	2nd Edition, 2015
3	Python Crash Course	Eric Matthes	No Starch Press	2nd Edition, 2019
4	Programming in Python 3	Mark Summerfield	Addison-Wesley	2nd Edition, 2010

Video Links (NPTEL, SWAYAM...)

Sl. No.	Source	URL
1	The Joy of Computing using Python	nptel.ac.in/courses/106106182
2	Scientific Computing using Python	https://www.coursera.org/specializations/python

SEMESTER S1

Web Programming Lab

Course Code	CA26WPP106	CAE Marks	50
Teaching Hours/Week (L: T:P: R)	0:1:3:0	ESE Marks	50
Credits	2	Exam Hours	3 Hrs.
Prerequisites (if any)	Basic knowledge in programming	Course Type	Practical

Preamble: This course introduces the fundamentals of web development and provides practical experience in building interactive web applications. It covers HTML5, CSS3, JavaScript, Flask framework, REST API integration, database connectivity, and Git/GitHub for version control. It enables students to design responsive interfaces, develop dynamic web applications, and deploy full-stack solutions.

Prerequisite: Basic knowledge of programming concepts and familiarity with computer fundamentals.

Course Objectives:

The course aims to:

1. Understand the fundamentals of web development including HTML5, CSS3, and responsive design principles.
2. Develop interactive web applications using JavaScript and DOM manipulation techniques.
3. Integrate external data sources using REST APIs and asynchronous programming concepts.
4. Design and implement server-side applications using Flask framework and database systems.
5. Apply modern web development practices such as version control using Git and deployment of web applications.

Course Outcomes (COs)

At the end of the course students should be able to:

Course Outcome		Bloom's Knowledge Level (KL)
CO1	Apply HTML5, CSS3, and responsive design principles to develop static web pages.	K3

CO2	Develop interactive web applications using JavaScript, DOM manipulation, and event handling.	K3
CO3	Integrate and process data using REST APIs and asynchronous JavaScript techniques.	K4
CO4	Design and implement server-side web applications using Flask and database connectivity.	K5
CO5	Apply Git, GitHub, and deployment tools for version control and hosting web applications.	K3

Note: K1- Remember, K2- Understand, K3- Apply, K4- Analyse, K5- Evaluate, K6- Create

CO-PO Mapping Table:

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PSO1	PSO2
CO1	2	2	2	3	1			1	2	1
CO2	2	3	2	3	2			2	3	2
CO3	2	3	3	3	2			2	3	3
CO4	3	3	3	3	2	1	1	2	3	3
CO5	1	1	2	3	2	3	3	3	3	2

SYLLABUS

Module No.	Syllabus Description	Contact Hours
1	Introduction to web development concepts, HTML5 structure, CSS3 styling, responsive web design principles, and layout techniques using Flexbox and Grid. Development of static web pages such as a responsive homepage, multi-page website, landing page, and portfolio website with proper navigation and styling consistency across pages.	10
2	JavaScript fundamentals for interactive web development including DOM manipulation, event handling, and form validation. Development of interactive applications such as to-do list, digital clock, calendar, quiz application, image gallery, and product catalogue with search and filter functionality.	10
3	Introduction to REST APIs and asynchronous programming using Fetch/AJAX. JSON data handling and dynamic content rendering without page reload. Development of web applications such as weather information system, news feed application, and live search functionality using external APIs.	10

4	Introduction to Flask framework, routing, templates (Jinja2), form handling, session management, and database integration using SQLite/MySQL. Development of web applications such as contact form system, user authentication system, CRUD-based application, and simple blogging platform. Integration of frontend with Flask backend using REST APIs. Introduction to Git and GitHub for version control and deployment of web applications on cloud platforms such as Render or PythonAnywhere.	15
---	---	----

List of Laboratory Experiments (Minimum 15)

1. Design a responsive company homepage using HTML5, CSS3, and Flexbox/Grid layout. The webpage should include a header section with logo and navigation menu, a hero section with banner image and tagline, a services section arranged using grid layout, and a footer with contact details. The layout should be made responsive so that it adapts properly to both desktop and mobile screen sizes using CSS media queries.
2. Develop a multi-page business website consisting of Home, Services, About, and Contact pages. Each page should be created as a separate HTML file and linked using a navigation menu. The website should maintain a consistent header and footer across all pages, and CSS should be used to ensure uniform styling and layout consistency throughout the site.
3. Create a mobile-friendly product landing page for a startup or product using HTML5 and CSS3. The page should include a product introduction section, features section, and a call-to-action button such as “Buy Now” or “Sign Up”. Responsive design techniques such as media queries should be applied to ensure proper display across different devices.
4. Design a personal portfolio website that showcases projects and skills. The website should include a profile section with image and biography, a projects section displaying project cards with images and descriptions, and links to project demos or GitHub repositories. A contact section with email or social media links should also be included.
5. Develop an interactive image gallery using HTML, CSS, and JavaScript. The gallery should display multiple images in a grid format with hover effects such as zoom or shadow. JavaScript should be used to implement slideshow functionality with next and previous navigation, along with optional auto-play feature.
6. Create a dynamic product catalogue web page using JavaScript. The page should display a list of products with details such as name, price, and category. The application should allow users to search products using a search bar and filter them based on categories. The results should update dynamically without reloading the page.
7. Develop a simple to-do list web application using JavaScript. The application should allow users to add new tasks, delete existing tasks, and mark tasks as completed. The task

- list should be dynamically updated using JavaScript arrays or local storage for temporary data persistence.
8. Build a digital clock and calendar web application using JavaScript. The application should display the current time with real-time updates and also show the current date, day, and a simple monthly calendar view. The clock should update automatically every second using JavaScript timing functions.
 9. Implement client-side form validation for registration and login pages using JavaScript. The validation should ensure that required fields are filled, email addresses follow proper format, and passwords meet basic strength requirements. Error messages should be displayed dynamically to guide the user during input.
 10. Design a simple online quiz application using JavaScript. The application should display multiple-choice questions and allow users to select answers. After submission, the system should calculate and display the final score along with a result summary indicating correct and incorrect answers.
 11. Create a weather information web page using a public weather API. The application should accept a city name from the user, fetch real-time weather data using Fetch API, and display details such as temperature, humidity, and weather conditions. The system should also handle invalid inputs or unavailable data gracefully.
 12. Develop a news feed web page using a REST API. The application should retrieve news articles dynamically and display information such as title, image, description, source, and publication date. The content should be loaded dynamically without page refresh.
 13. Build a live search web application using AJAX or Fetch API. The system should allow users to search data dynamically, and results should be updated instantly without reloading the page.
 14. Develop a contact form web application using Flask. The application should accept user inputs such as name, email, and message through an HTML form and send the data to a Flask backend. The backend should store the information in a database such as SQLite or MySQL and display a confirmation message after successful submission.
 15. Create a user authentication system using Flask. The system should allow users to register and log in using a secure form. User credentials should be stored in a database, and session management should be used to maintain login state. Proper validation and secure password handling should be implemented.
 16. Develop a CRUD (Create, Read, Update, Delete) web application using Flask and a database. The system should allow users to add new records, view existing records, update data, and delete records.

17. Design an employee management system using Flask and a database. The system should allow adding employee details, updating information, deleting records, and searching employees based on criteria such as name or ID. The application should provide a structured interface for managing employee data.
18. Develop a simple blogging platform using Flask where users can create, edit, and display blog posts. Each post should be stored in a database and displayed dynamically on the homepage. The system should allow viewing individual blog posts with full content.
19. Use Git and GitHub for version control of a web project. The experiment should include initializing a Git repository, committing project changes with proper messages, and pushing the project to a GitHub repository. A README file should be created to describe the project structure and usage.

Course Assessment Method

(CAE: 50 Marks, ESE: 50 Marks)

Attendance	Maintenance of daily lab record and GitHub management	Regular class viva	Timely completion of day to day tasks	Tests/Evaluation	Total
10	10	5	10	15	50

End Semester Examination Marks (ESE)

Maximum Marks: 50			
Verification of Daily program record and Git Repository			5 Marks
Viva			10 Marks
Problem solving (Based on difficulty level, one or more questions may be given)	Flowchart / Algorithm / Structured description of problem to explain how the problem can be solved / Interface Design	15	35 Marks
	Program correctness	50	
	Code efficiency	15	
	Formatted output and Pushing to remote Git repository	20	
Total Marks			50 Marks

Text Books

Sl. No.	Title of the Book	Name of the Author(s)	Name of the Publisher	Edition and Year
1	JavaScript: The Definitive Guide”,	David Flanagan	O'Reilly Media	6 th Edition
2	Web Design with HTML, CSS, JavaScript and jQuery Set	Jon Duckett	Wiley	1 st Edition, 2014
3	Flask Web Development: Developing Web Applications with Python	Miguel Grinberg	O'Reilly Media	2 nd Edition, 2018
4	REST API Design Rulebook	Mark Masse	O'Reilly Media	2011

Reference Books

Sl. No.	Title of the Book	Name of the Author(s)	Name of the Publisher	Edition and Year
1	The Internet Book: Everything You Need to Know About Computer Networking and How the Internet Works	Douglas E Comer	Chapman & Hall	5 th Edition, 2018
2	Internet and World Wide Web - How To Program	Harvey Deitel and Abbey Deitel	Pearson	5 th Edition, 2012

SEMESTER S1

Data Structures using C Lab

Course Code	CA26DSP107	CAE Marks	50
Teaching Hours/Week (L: T:P: R)	0:1:3:0	ESE Marks	50
Credits	2	Exam Hours	3 Hrs.
Prerequisites (if any)	Basic Data Structures	Course Type	Practical

Preamble: This course introduces advanced data structures and algorithmic problem-solving techniques. It develops students' ability to design and analyze efficient data structures such as trees, graphs, heaps, hashing techniques, and disjoint sets. The course also emphasizes algorithmic thinking, performance analysis, and application of data structures to real-world computing problems.

Prerequisite: Basic Data Structures

Course Objectives:

The course aims to:

1. Understand and implement advanced data structures such as trees, heaps, graphs, and disjoint sets.
2. Analyze the performance of algorithms using time and space complexity concepts.
3. Apply appropriate data structures to solve real-world computational problems efficiently.
4. Implement graph algorithms such as BFS, DFS, shortest path, and minimum spanning tree algorithms.
5. Design optimized solutions using advanced data structures for problem-solving and system design.

Course Outcomes (COs)

At the end of the course students should be able to:

Course Outcome		Bloom's Knowledge Level (KL)
CO1	Implement hashing and disjoint set data structures for efficient data organization and manipulation.	K3
CO2	Apply advanced tree structures such as AVL, Red-Black, Splay, and B-Trees for efficient searching and storage.	K3

CO3	Analyse and implement heap-based data structures including Binary Heap, Binomial Heap, and Fibonacci Heap.	K4
CO4	Apply graph algorithms such as BFS, DFS, Dijkstra's, Kruskal's, and Prim's algorithms for problem solving.	K3
CO5	Design and implement advanced data structure-based solutions such as segment trees and optimized graph/heap applications	K5

Note: K1- Remember, K2- Understand, K3- Apply, K4- Analyse, K5- Evaluate, K6- Create

CO-PO Mapping Table:

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PSO1	PSO2
CO1	3	3	2	2	1			1	3	1
CO2	3	3	2	2	1			1	3	1
CO3	3	3	2	3	1			2	3	1
CO4	3	3	3	3	2			2	3	2
CO5	3	3	3	3	2	1		2	3	2

SYLLABUS

Based on the syllabus of CA26DST103 Advanced Data Structures.

List of Experiments (Minimum 15)

1. Implement Hash Table using separate chaining and open addressing techniques for efficient storage and retrieval of key-value pairs.
2. Implement Disjoint Set Union (Union-Find) with path compression and union by rank for dynamic connectivity problems.
3. Implement Breadth First Search (BFS) traversal of a graph and apply it to find shortest paths in an unweighted graph.
4. Implement Depth First Search (DFS) traversal of a graph and use it for cycle detection and connectivity checking.
5. Implement Kruskal's Algorithm for Minimum Spanning Tree using Disjoint Set data structure.
6. Implement Prim's Algorithm for Minimum Spanning Tree using a priority queue (min-heap).
7. Implement Dijkstra's Algorithm for Single Source Shortest Path using a min-heap priority queue.

8. Implement Binary Heap (Min-Heap and Max-Heap) with insertion, deletion, and heapify operations.
9. Implement AVL Tree with insertion, deletion, and rotation operations to maintain balance in BST.
10. Implement Red-Black Tree operations (insertion and deletion) with emphasis on balancing rules.
11. Implement B-Tree of order m with insertion and traversal for multi-level indexing structures.
12. Implement Trie (Prefix Tree) for fast string search and autocomplete functionality.
13. Implement Segment Tree for range sum and range minimum query operations.
14. Implement Graph representation using adjacency list and adjacency matrix with basic operations.
15. Implement Topological Sorting of a Directed Acyclic Graph (DAG) using DFS and BFS (Kahn's Algorithm).

Course Assessment Method

(CAE: 50 Marks, ESE: 50 Marks)

Attendance	Maintenance of daily lab record and GitHub management	Regular class viva	Timely completion of day to day tasks	Tests/Evaluation	Total
10	10	5	10	15	50

End Semester Examination Marks (ESE)

Maximum Marks: 50			
Verification of Daily program record and Git Repository			5 Marks
Viva			10 Marks
Problem solving (Based on difficulty level, one or more questions may be given)	Flowchart / Algorithm / Structured description of problem to explain how the problem can be solved / Interface Design	15	35 Marks
	Program correctness	50	
	Code efficiency	15	

	Formatted output and Pushing to remote Git repository	20	
Total Marks			50 Marks

Text Books

Sl. No.	Title of the Book	Name of the Author(s)	Name of the Publisher	Edition and Year
1	Data Structures Using C	Reema Thareja	Reema Thareja	2 nd Edition, 2014
2	Data Structures and Algorithm Analysis in C	Mark Allen Weiss	Addison-Wesley	2 nd Edition, 1997

Reference Books

Sl. No.	Title of the Book	Name of the Author(s)	Name of the Publisher	Edition and Year
1	Algorithm Design	Jon Kleinberg, Eva Tardos	Pearson Education	2006
2	Data Structures and Algorithms	Alfred V. Aho, John E. Hopcroft, Jeffrey D. Ullman	Pearson Education	1983
3	Introduction to Algorithms	Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein	MIT Press	4 th Edition, 2022

Video Links (NPTEL, SWAYAM...)

Sl. No.	Source	URL
1	Data Structures and Algorithms	nptel.ac.in/courses/106102064
2	Data Structures and Algorithms Design	https://onlinecourses.nptel.ac.in/e-learning/preview/noc25_cs81