



Deccan Education Society's

**Fergusson College
(Autonomous) Pune**

Learning Outcomes-Based Curriculum

for 2 years M.Sc.

Program as per the

guidelines of

NEP-2020

for

F. Y. M. Sc. (Computer Application)

With effect from the Academic Year

2026-2027

Program Outcomes (POs) for M.Sc.	
PO1	Disciplinary Knowledge: Demonstrate comprehensive knowledge of the discipline that forms a part of a postgraduate programme. Execute strong theoretical and practical understanding generated from the specific programme in the area of work.
PO2	Critical Thinking and Problem solving: Exhibit the skill of critical thinking and understand scientific texts and place scientific statements and themes in contexts and also evaluate them in terms of generic conventions. Identify the problem by observing the situation closely, take actions and apply lateral thinking and analytical skills to design the solutions.
PO3	Social competence: Exhibit thoughts and ideas effectively in writing and orally; communicate with others using appropriate media, build effective interactive and presenting skills to meet global competencies. Elicit views of others, present complex information in a clear and concise and help reach conclusion in group settings.
PO4	Research-related skills and Scientific temper: Infer scientific literature, build sense of enquiry and able to formulate, test, analyse, interpret and establish hypothesis and research questions; and to identify and consult relevant sources to find answers. Plan and write a research paper/project while emphasizing on academics and research ethics, scientific conduct and creating awareness about intellectual property rights and issues of plagiarism.
PO5	Trans-disciplinary knowledge: Create new conceptual, theoretical and methodological understanding that integrates and transcends beyond discipline-specific approaches to address a common problem.
PO6	Personal and professional competence: Perform independently and also collaboratively as a part of a team to meet defined objectives and carry out work across interdisciplinary fields. Execute interpersonal relationships, self-motivation and adaptability skills and commit to professional ethics.
PO7	Effective Citizenship and Ethics: Demonstrate empathetic social concern and equity centered national development, and ability to act with an informed awareness of moral and ethical issues and commit to professional ethics and responsibility.
PO8	Environment and Sustainability: Understand the impact of the scientific solutions in societal and environmental contexts and demonstrate the knowledge of and need for sustainable development.
PO9	Self-directed and Life-long learning: Acquire the ability to engage in independent and life-long learning in the broadest context of socio-technological changes.

PSO No.	Program Specific Outcomes (PSOs) Upon completion of this programme the student will be able to
PSO1	Academic competence: Identify, formulate and solve real world computing system problems of various domains by understanding and applying the principles of mathematics, computing techniques and business concepts.
PSO2	Personal and Professional Competence: Analyse, design, test, develop and maintain desktop, web, mobile and cross platform software applications using modern tools and technologies in economical and affordable way.
PSO3	Research Competence: Identify the potentiality to design experiments for complex problems, collect data, interpret data, develop, implement computer programs and use knowledge in various domains to identify research gaps and hence to provide solutions to new ideas and innovations.
PSO4	Entrepreneurial and Social competence: Analyse and understand the evolutionary changes in computing by inculcating employability and entrepreneur skills among the lifelong learners with moral values and ethics. Develop skills required for social interaction.

F. Y. M.Sc. (Computer Application) (NEP 2020 Pattern)
from Academic Year 2026-2027

Semester		Paper Code	Paper Title	Credits
I	MC	CSA-531	Python Programming	4
		CSA-532	Software Engineering and Management	4
		CSA-541	Practical – I (Based on CSA-531)	2
		CSA-542	Practical – II (Based on CSA-533 or CSA- 534)	2
	Elective	CSA-533 OR	Data Structure	4
		CSA-534	Mathematics for Computing	
	RM	CSA-511	Research Methodology	4
		Total Semester Credits		20
II	MC	CSA-581	Java Programming	4
		CSA-582	Database Management System	4
		CSA-591	Practical – III (Based on CSA-581 and CSA-582)	2
		CSA-592	Practical – IV (Project)	2
	Elective	CSA-583	Operating System	4
		CSA-593	Practical -V (Based on CSA-583)	
			OR	
		CSA-584	WEB Development using MERN	
		CSA-594	Practical -VI (Based on CSA-584)	
	OJT/FP	CSA-561	OJT/FP	4
		Total Semester Credits		20
		Total PG-II Credits		40

Teaching and Evaluation (Only for FORMAL education courses)

Course Credits	No. of Hours per Semester Theory/Practical	No. of Hours per Week Theory/Practical	Maximum Marks	CE 40 %	ESE 60%
1	15 / 30	1 / 2	25	10	15
2	30 / 60	2 / 4	50	20	30
3	45 / 90	3 / 6	75	30	45
4	60 / 120	4 / 8	100	40	60

F.Y. M.Sc. Semester I		
CSA-531	Python Programming	Credits: 4 Hours: 60
Course Outcomes (COs) On completion of the course, the students will be able to:		
CO1	Understand core Python programming concepts.	
CO2	Apply Python for solving computational problems.	
CO3	Develop object-oriented programs using Python.	
CO4	Create file-based and database-driven applications.	
CO5	Design simple web applications using Python frameworks.	
CO6	Develop a complete mini-project using Python.	

Unit	Contents	No. of hours
I	Introduction to Python 1.1 Features of Python 1.2 Installation and Environment setup 1.3 Variables and Data Types 1.4 Operators and Expressions 1.5 Input and Output Functions 1.6 Conditional Statements 1.7 Looping Statements 1.8 Basic Programs	05
II	Functions and Data Structures 2.1 Recursion 2.2 Lambda Functions 2.3 Strings and String Operations 2.4 Lists and List Methods 2.5 Tuples 2.6 Sets 2.7 Dictionaries	08
III	Object-Oriented Programming 3.1 Classes and Objects 3.2 Constructor and Destructor 3.3 Instance and Class Variables 3.4 Inheritance 3.5 Method Overriding 3.6 Polymorphism 3.7 Encapsulation 3.8 Exception Handling	06

IV	File Handling and Modules 4.1 File Operations (Read, Write, Append) 4.2 Working with Text Files 4.3 Working with CSV Files 4.4 Introduction to JSON 4.5 Modules and Packages 4.6 Creating User-defined Modules	05
V	Working with Libraries: - 5.1. Overview of popular libraries (NumPy, Pandas, Matplotlib) 5.2. Basic usage of a data manipulation library 5.3. Overview of GUI libraries 5.4. Basic GUI application development	08
VI	Database Programming with Python 6.1. Introduction to SQLite 6.2. Connecting Python with SQLite 6.3. Creating Tables 6.4. Insert, Update, Delete, Select Operations 6.5. Simple CRUD Application 6.6. Introduction to MySQL connectivity	06
VII	Web Application Development using Python 7.1. Introduction to Web Applications 7.2. Basics of Flask Framework 7.3. Routing 7.4. Templates 7.5. Forms Handling 7.6. Connecting Web App with Database 7.7. Simple Web Application deployment	08
VIII	Testing in Python 8.1 Introduction to Unit Testing 8.2 Python <i>unittest</i> Module 8.3 Writing Test Cases 8.4 Assertion Methods 8.5 Test Fixtures 8.6 Running Tests 8.7 Best Practices in Unit Testing	02

References:

1. Lutz, M. (2013). *Learning Python* (5th ed.). O'Reilly Media.
2. Sweigart, A. (2019). *Automate the Boring Stuff with Python: Practical Programming for Total Beginners* (2nd ed.). No Starch Press.

3. Beazley, D., & Jones, B. K. (2013). *Python Cookbook* (3rd ed.). O'Reilly Media.
4. Matthes, E. (2023). *Python Crash Course* (3rd ed.). No Starch Press.
5. Ramalho, L. (2015). *Fluent Python: Clear, Concise, and Effective Programming*. O'Reilly Media.
6. Guttag, J. V. (2016). *Introduction to Computation and Programming Using Python* (2nd ed.). MIT Press.

e-resources:

1. <https://docs.python.org/3/>
2. <https://flask.palletsprojects.com/>
3. <https://www.w3schools.com/python/>
4. <https://www.tutorialspoint.com/python/index.htm>
5. <https://www.edx.org/learn/python>
6. <https://www.udemy.com/topic/python/>
7. <https://docs.python.org/3/library/unittest.html>

F.Y.M.Sc. Semester I		
CSA-532	Software Engineering and Project Management	Credits: 4 Hours: 60
Course Outcome (COs) On completion of the course, the students will be able to:		
CO1	Identify the factors that direct the Software Project and outline the role of process model in a Project.	
CO2	Exemplify, SRS to develop use cases for the project.	
CO3	Carry out life cycle activities like Analysis, Design, Implementation, Testing and Maintenance. Apply the knowledge, techniques, and skills in the development of a software product.	
CO4	Appraise, various processes used in all the phases of the product	
CO5	Evaluate the background and driving strengths for taking an Agile approach to Software Development	
CO6	Design real time applications using Object Oriented Analysis and Design Methodology.	

Unit	Contents	No.of lectures
I	Introduction 1.1. Introduction to Software Engineering, 1.2. Software Components, Software Characteristics, 1.3. Software Engineering Processes, 1.4. Software Quality Attributes. 1.5. Software Development Life Cycle (SDLC) Model: Waterfall Model	05
II	Software Requirement Specifications (SRS) 2.1. Requirement Engineering Process: Elicitation, Analysis, Documentation, Feasibility Study, Information Modeling 2.2. Entity Relationship Diagrams, 2.3. SRS Document, IEEE Standards for SRS. 2.4. Software Quality Assurance (SQA): Verification and Validation, 2.5. SQA Plans, Software Quality Frameworks, 2.6. ISO 9000 Models, SEI-CMM Model.	08
III	Software Design. 3.1. Design Process, Design concepts, 3.2. Software Architecture, Architectural Styles, and Design	07

	3.3. Assessing alternative architectural designs 3.4. Designing Class-Based Components 3.5. Component-level design for Web Apps 3.6. Designing Traditional Components – User Interface Design.	
IV	Agile Model 4.1. Introduction 4.2. Agile Principles and Concepts 4.3. Agile Methodologies: Scrum, Extreme programming, Kanban 4.4. Advantages and Challenges of Agile 4.5. Agile vs Traditional Models	06
V	Software Testing 5.1 Testing Objectives, 5.2 Unit Testing, Integration Testing, Acceptance Testing, Regression Testing 5.3 Testing, Testing for Functionality and Testing for Performance, 5.4 Top-Down and Bottom-Up 5.5 Testing Strategies: Test Drivers and Test Stubs, 5.6 Structural Testing (White Box Testing), 5.7 Functional Testing (Black Box Testing),	04
VI	Project Management 6.1 Project Management concepts, 6.2 Process Framework, 6.3 Project Planning Software Life Cycle Models, 6.4 Artifacts of the Project Management Process. 6.5 Cost and Scheduling Estimation Models: Various Levels of COCOMO for Cost , Effort, Schedule and Productivity Estimation. 6.6 Approaches to Effort, Cost Estimation, and Schedule Estimation factors through COCOMO II.	08
VII	Project Management Techniques: 7.1 Project Organizations and Responsibilities, 7.2 Establishing Project Environment, 7.3 Risk Management Process, 7.4 Project Tracking and Control Defect Tracking Concepts such as Process monitoring and audit, Reviews, Inspections and walkthroughs. 7.5 Activity Planning including CPM and PERT: Network planning model 7.6 Activity-on-arrow network; Precedence network; Forward pass; Backward pass; Critical path; Slack and float.	10

References:

1. Roger S Pressman, “Software Engineering”, Tata McGraw-Hill Publications

2. I. Sommerville, "Software Engineering", Addison Wesley
3. Clean Code: A Handbook of Agile Software Craftsmanship, Robert C. Martin, PHI
4. R. Fairley, Software Engineering Concepts, Tata McGraw-Hill Edition.
5. Software Metrics for Project Management and process improvement Robert B. Grady Prentice hall

E- BOOKS :

1. <http://www.ddegjust.ac.in/studymaterial/mca-3/ms-12.pdf>

MOOC :

1. <https://www.classcentral.com/course/swayam-software-engineering-14293>
2. <https://archive.nptel.ac.in/courses/106/101/106101061/>

F.Y.M.Sc. Semester I		
CSA- 541	Practical -I (Based on CSA-531: Python programming)	Credits: 2 Hours: 60
Course Outcomes (COs) On completion of the course, the students will be able to:		
CO1	Understand core Python programming concepts.	
CO2	Apply Python for solving computational problems.	
CO3	Develop object-oriented programs using Python.	
CO4	Create file-based and database-driven applications.	
CO5	Design simple web applications using Python frameworks.	
CO6	Develop a complete mini-project using Python.	

Lab Course in Python Programming

Note: - Each Assignment will be based on following concepts

Expt No	Title of the Experiment
1	Python Basics- 1
2	Python Basics- 2
3	String processing system
4	List and Dictionary
5	Functions and recursion
6	Object-Oriented Programming
7	Inheritance and Polymorphism
8	Exception Handling
9	File Handling – Text file
10	Working with CSV File processing and JSON Data Processing
11	Working with data visualization
12	SQLite Database Application
13	Flask Web Application
14	REST API using Flask
15	Working with Libraries

F.Y. M.Sc. Semester I		
CSA-542	Practical – II (CSA-533 OR CSA-534)	Credits: 2 Hours: 30
Course Outcomes (COs) On completion of the course, the students will be able to:		
CO1	Apply requirement analysis techniques and design ER models, relational schemas, and implement database operations using SQL and MongoDB.	
CO2	Develop and execute SQL queries using joins, nested queries, views, stored procedures, cursors, and triggers to manage and manipulate database systems efficiently.	
CO3	Implement fundamental data structures such as arrays, structures, pointers, stacks, queues, linked lists, trees, and graphs using appropriate programming techniques.	
CO4	Apply searching and sorting algorithms to solve computational problems and analyze their behavior in practical applications.	
CO5	Implement programs based on discrete mathematical concepts, including logic, set theory, relations, functions, combinatorics, and graph traversal.	
CO6	Develop programs to solve numerical and mathematical problems using modular arithmetic, matrix operations, interpolation, numerical integration, and root-finding methods.	

Expt. No.	Title of the Experiment
Based on CSA-503	
1	Array, Structure, pointer
2	searching, sorting
3	Stack
4	Queue
5	Link list
6	Tree
7	Graph
Based on CSA-504	
1	Assignment 1: Logical Operations and Truth Tables

	Programs: • Program to implement logical operations (AND, OR, NOT, XOR). • Program to generate truth tables for logical expressions. • Program to verify logical equivalence.
2	Assignment 2: Set Theory and Relations Programs: • Program to perform set operations (Union, Intersection, Difference). • Program to compute the Cartesian product of sets. • Program to represent relations and check properties.
3	Assignment 3: Functions and Combinatorics Programs: • Program to check one-to-one and onto functions. • Program to compute permutations. • Program to compute combinations. • Program to implement the Binomial theorem.
4	Assignment 4: Graph Representation and Traversal Programs: • Program to represent a graph using an adjacency matrix. • Program to implement Breadth First Search (BFS). • Program to implement Depth First Search (DFS).
5	Assignment 5: Modular Arithmetic, Cryptography, and Matrix Operations Programs: • Program to perform modular arithmetic operations. • Program to implement simple cryptography using modular arithmetic. • Program for matrix operations (addition, multiplication, transpose, inverse).
6	Assignment 6: System of Linear Equations and Root Finding Methods Programs: • Program to solve systems of linear equations. • Program to implement Bisection method. • Program to implement Newton–Raphson method.
7	Assignment 7: Numerical Integration and Interpolation Programs: • Program for numerical integration using Trapezoidal rule. • Program for numerical integration using Simpson's 1/3 rule. • Program for numerical integration using Simpson's 3/8 rule.

	• Program to implement Lagrange interpolation. • Program to implement Newton's forward interpolation. • Program to implement Newton's backward interpolation.
--	---

References:

1. Hsieh, W. K., Pletcher, J., & Pollice, G. (2017). *Practical SQL: A beginner's guide to storytelling with data*. No Starch Press.
2. MongoDB, Inc. (2021). *MongoDB: The definitive guide: Powerful and scalable data storage* (4th ed.). O'Reilly Media.

F.Y. M.Sc. Semester I		
CSA-533	Data Structure (Elective)	Credits: 4 Hours: 60
Course Outcomes (COs) On completion of the course, the students will be able to:		
CO1	Apply programming concepts such as arrays, pointers, and structures for implementing data structures.	
CO2	Analyze problems and design efficient algorithms using appropriate problem-solving strategies and complexity analysis.	
CO3	Implement and apply linear data structures such as stacks, queues, and linked lists to solve computational problems.	
CO4	Implement tree data structures including Binary Search Trees, AVL trees, and heaps, and perform traversal operations.	
CO5	Represent graphs using suitable methods and apply graph traversal algorithms such as BFS and DFS.	
CO6	Apply and compare different sorting techniques based on their performance and suitability for given problems.	

Unit	Contents	No. of hours
I	Programming Foundations for Data Structures 1.1 Introduction to Data Structures • Definition and importance of data structures. ADT • Role of data structures in efficient problem solving 1.2 Review of C Programming Concepts • Basic syntax and program structure • Functions and memory concepts 1.3 Arrays • One-dimensional arrays • Two-dimensional arrays • Memory representation of arrays • Basic array operations 1.4 Pointers • Pointer fundamentals • Pointer arithmetic • Relationship between pointers and arrays • Pointers with functions 1.5 Structures	7

	• Definition and declaration of structures • Array of structures • Pointer to structure • Relationship between arrays, pointers, and structures	
II	Algorithmic Thinking and Design 2.1 Introduction to algorithms and their characteristics. 2.2 Problem definition and algorithm formulation. 2.3 Steps involved in algorithmic problem solving. 2.4 Problem-solving techniques • brute force method, • divide and conquer strategy, • greedy approach, • dynamic programming (conceptual understanding), • and backtracking (conceptual understanding). 2.5 Algorithm analysis • time complexity • space complexity • Best case, average case, and worst-case analysis. 2.6 Asymptotic notations • Big-O, Big-Ω, and Big-Θ.	6
III	Sorting Techniques 3.1 Introduction to Sorting • Definition and importance • Applications of sorting 3.2 Bubble Sort • Algorithm • Working and example 3.3 Selection Sort • Algorithm • Working and example 3.4 Insertion Sort • Algorithm • Working and example 3.5 Merge Sort • Algorithm • Working and example 3.6 Comparison of Sorting Techniques • Performance comparison • Time complexity analysis • Selection of an appropriate sorting method	6

IV	Linear Data Structures – Stack and Queue as ADT 4.1 Stack Data Structure • Concept and characteristics of stack • Stack operations 4.2 Stack Implementation • Stack using an array • Stack using a linked list 4.3 Applications of Stack • Expression conversion • Expression evaluation • Recursion 4.4 Queue Data Structure • Concept and characteristics of a queue • Queue operations 4.5 Types of Queues • Linear queue • Circular queue • Priority queue • Deque 4.6 Queue Implementation • Queue using an array • Queue using linked list (After Unit IV)	7
V	Linked Lists 5.1 Introduction to Linked Lists • Definition and basic concepts • Advantages and disadvantages 5.2 Singly Linked List • Representation • Insertion operations • Deletion operations • Traversal 5.3 Doubly Linked List • Representation • Operations 5.4 Circular Linked List • Representation • Operations 5.5 Applications of Linked Lists	10

	• Real-world applications • Dynamic memory management	
VI	Trees and Graphs 6.1 Introduction to Trees • Definition and terminology • Tree representation 6.2 Binary Trees • Representation of binary trees • Tree traversal techniques • Inorder traversal • Preorder traversal • Postorder traversal 6.3 Binary Search Tree (BST) • BST concept and properties • Insertion operation • Deletion operation • Searching operation 6.4 AVL Trees • Height-balanced tree concept • Rotations • Insertion and deletion 6.5 Heap Data Structure • Heap concept • Max heap and min heap • Heap operations • Heap representation 6.6 Graph Data Structures • Introduction and terminology • Graph representation using adjacency matrix • Graph representation using adjacency list 6.7 Graph Traversal Algorithms • Breadth First Search (BFS) • Depth First Search (DFS) 6.8 Shortest Path Algorithms • Concept of shortest path in graphs • Dijkstra's Algorithm • Applications of shortest path algorithms	12

References

1. Mark Allen Weiss, Data Structures and Algorithm Analysis in C, Pearson Education.

2. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, Introduction to Algorithms, MIT Press.
3. Reema Thareja, Data Structures Using C, Oxford University Press.
4. Ellis Horowitz, Sartaj Sahni, and Susan Anderson-Freed, Fundamentals of Data Structures in C, Universities Press.
5. Seymour Lipschutz, Data Structures (Schaum's Outline Series), McGraw-Hill.

F.Y.M.Sc. Semester I		
CSA-534	Mathematics for Computing (Elective)	Credits: 4 Hours: 60
Course Outcomes (COs) On completion of the course, the students will be able to:		
CO1	Apply propositional and predicate logic in computing problems.	
CO2	Solve problems using set theory and relations.	
CO3	Apply functions and combinatorics in algorithm analysis.	
CO4	Use graph theory in computer science applications.	
CO5	Apply algebraic structures in computing and cryptography.	
CO6	Understand numerical method and computing	

Unit	Contents	No. of hours
I	Propositional Logic and Predicate Logic 1.1 Propositions and logical operators 1.2 Truth tables and logical equivalence 1.3 Tautology, contradiction, and contingency 1.4 Laws of logic 1.5 Predicate logic and quantifiers 1.6 Applications in computing (Boolean logic, program conditions)	5
II	Set Theory and Relations 2.1 Sets and set operations 2.2 Venn diagrams and applications	3

	2.3 Cartesian product 2.4 Relations and properties 2.5 Equivalence relations 2.6 Partial ordering and applications	
III	Functions and Combinatorics 3.1 Functions and types of functions 3.2 One-to-one and onto functions 3.3 Composition of functions	2
IV	Graph Theory 4.1 Graphs and types of graphs 4.2 Directed and undirected graphs 4.3 Graph representation 4.4 Graph traversal concepts (BFS and DFS) 4.5 Trees and binary trees 4.6 Applications of graphs in computing	8
V	Mathematical Foundations 5.1 Matrices and vectors 5.2 Matrix operations and properties 5.3 Systems of linear equations 5.4 Eigenvalues and Eigenvectors 5.5 Multivariable functions: partial derivatives, maxima and minima	20
VI	Numerical Methods 6.1 Errors in numerical computation 6.2 Bisection method 6.3 Newton–Raphson method 6.3 Trapezoidal rule 6.4 Simpson’s 1/3 rule 6.5 Simpson’s 3/8 rule	5
VII	Interpolation and Curve Fitting (6 Hours) 7.1 Lagrange interpolation 7.2 Newton’s forward interpolation 7.3 Newton’s backward interpolation 7.4 Newton’s divided difference formula 7.5 Curve fitting using least squares method	5

References:

1. Kenneth Rosen – Discrete Mathematics and Its Applications
2. Tremblay and Manohar – Discrete Mathematical Structures
3. Susanna Epp – Discrete Mathematics with Applications
4. Kolman, Busby, Ross – Discrete Mathematical Structures

5. James Stewart – Multivariable Calculus, Cengage Learning
6. B. S. Grewal – Numerical Methods, Khanna Publishers
7. Gilbert Strang – Linear Algebra and Its Applications, Cengage Learning

F.Y. M.Sc. Semester I		
CSA-511	CSA-510 Research Methodology	Credits: 4 Hours: 60

Instructions:

- A student will choose 4 suitable modules depending on his/her faculty. The contact hours will be around 60 based on assignments and examinations. The examination for each module will be separately performed.

Mode of examination-

- The internal examination for each module will be separately conducted. The examination mode is decided by the instructor of that module.
- The external examination will be conducted at the time of 4th half yearly progress review. The student's implementation of various aspects in research methodologies will be checked.

Unit	Contents	No. of hours
I	History of research. Indian, Egyptian, Greek ideas, methodologies, and research in agriculture, chemistry, metallurgy, medical. Ancient Indian research methodology applications.	6
II	Statistical analysis and its significance. , Exploratory and confirmatory research, Planned and ad-hoc methods of data collection, non-response, and methods of recovering the missing response, Various software for statistical analysis. The module will consist of case studies of the research performed in various subjects using statistical methods, Error and noise analysis, curve fitting.	12
III	Creating a questionnaire. Data analysis from answers, Selection of research topic (case study-based). Selection of research topic (case study-based)	8
IV	Literature search, selection of research topic (case study-based), maintaining laboratory records (case study-based). Safety in Laboratories, Ethical considerations, effective verbal and non- verbal communication, field data collection, safety in the field.	12
V	Writing a research paper and/or thesis, making a presentation, writing a research proposal, and patents in science and technology.	11

VI	Databases and Research Metrics Databases (i) Indexing databases. (ii) Citation databases: Web of Science, Scopus, etc. Research Metrics (i) Impact Factor of the journal as per Journal Citation Report, SNIP, SJR, IPP, Cite Score. (ii) Metrics: h- index, g-index, i10-index, altimetric	5
VII	Research Methods for Computer Science Formal Methods: Formal Specification, Algorithm, and Complexity; Building Artefacts: Proof of Performance, Proof of Concept, and Proof of Existence; Process Methodology: Methods for Software Engineering and Human-Computer Interaction, Cognitive Processes, Interactive Games, Social Networks, and Web Analytics	6

References:

1. 'History of the Scientific Methods' by Martin Shuttleworth, <https://explorable.com/history-of-the-scientific-method>.
2. The Statistical Analysis of Experimental Data' by, John Mandel, ISBN: 0486646661, ISBN13: 9780486646664

F.Y. M.Sc. Semester II		
CSA-581	Java Programming	Credits: 4 Hours: 60
Course Outcomes (COs) On completion of the course, the students will be able to:		
CO1	List object-oriented programming concepts.	
CO2	Illustrate the programming constructs of Java.	
CO3	Examine different classes for the effective use of file handling operations.	
CO4	Analyze the usage of interfaces, packages and exceptional handling.	
CO5	Determine the use of different Java libraries and JDBC tools.	
CO6	Write complex Java programs by specifying the appropriate standard API library of Java. Write Web applications for client-server architecture.	
Unit	Contents	No. of hours
I	Introduction to Java Language 1.1 History and Evolution of Java, 1.2 JDK Environment, 1.3 Java Tools, 1.4 Java Byte Code 1.5 OOP Principles, 1.6 Java Programming – Keywords, Data Types, Variables, Operators, Naming Conventions, Type Casting, Control Statements Arrays	03
II	Object Oriented Concepts of Java 2.1 Introducing classes and objects Constructors (All types), 2.2 Garbage Collection and finalize () method 2.3 Inheritance Basics, Types of Inheritance , 2.4 Implementation of polymorphism: Method Overloading and Method Overriding Nested and Inner classes , 2.5 Modifiers and Access Control Specifiers 2.6 Final variables, methods and classes Abstract methods and classes 2.7 Interfaces, Creating and Importing Packages	06

III	Exception Handling 3.1 Dealing Errors, 3.2 Exception class, Checked and Unchecked, exception, 3.3 Catching exception and exception handling, 3.4 Creating user defined exceptions	04
IV	Multithreaded Programming 4.1 The Java Thread Model, 4.2 The Main Thread, Creating a Thread, 4.3 Creating Multiple Threads, 4.4 Inter-thread Communication Suspending, Resuming and Stopping Threads	04
V	Collections and File Handling 5.1 Introduction to the Collection framework (Interfaces, Implementation, and algorithms) 5.2 Interfaces Collection classes: Set, List, Queue and Map Set: HashSet, TreeSet, and LinkedHashSet 5.3 Interfaces such as Lists, Set, Stack, LinkedList, Comparator, Iterator, Hash table. 5.4 Reading and Writing Files, 5.5 Automatically Closing a File, 5.6 The I/O Classes and Interfaces, File handling in Java	05
VI	GUI in Java 6.1 Delegation Event Model, MVC (Model View Controller) Architecture for Swing, 6.2 Event Handling: Event Handling Mechanism, Event Classes, Event Listener Interfaces, Adapter Classes, Layout Managers Windows Fundamentals, 6.3 Working with Frame Windows, Working with Fonts Swing Features, Components & Containers, Swing Components 6.4 Introduction to JavaFX 6.5 Building user interfaces and layouts, Event Handling	07
VII	Database Programming 7.1 The design of JDBC 7.2 JDBC configuration 7.3 Types of drivers Executing SQL statements, 7.4 Query execution, Batch execution 7.5 Scrollable and updatable result sets Rowset,	04

	7.6 Metadata, transactions, 7.7 Swing application using database	
VIII	Web development using JSP 8.1 Introduction to JSP 8.2 JSP Architecture 8.3 JSP Directives, JSP scripting elements Default objects in JSP , JSP Actions 8.4 JSP with Database 8.5 Error handling in JSP 8.6 Session tracking techniques in JSP, JSON object	05
IX	Spring Framework and Introduction to Spring Boot 9.1 Spring 9.1.1 Introduction of Spring framework, Bean 9.1.2 Spring Applications 9.1.3 Spring – MVC framework 9.2 Spring Core, Spring AOP, 9.3 Spring JDBC, 9.4 Spring Boot– Rapid Application Development 9.5 Introduction to Components of Hibernate 9.6 Introduction to Struts and framework 9.7 Introduction to Maven framework, MOJO, POJO	10

References

1. E-Book : Cay S. Horstmann, Gary Cornell, Core Java Volume-II-Advanced Features, Thirteenth Edition, Pearson, 2024.
2. Herbert Schildt (13th edition), Complete reference Java, 2024
3. Mastering Spring Boot 2.0 O'reilly Publication, 2018

F.Y. M.Sc. Semester II		
CSC-582	Database Management System	Credits: 4 Hours: 60
Course Outcomes (COs) On completion of the course, the students will be able to:		
CO1	Recall fundamental concepts of database systems, including data models, DBMS architecture, components, and database users.	
CO2	Explain relational data model concepts and integrity constraints	
CO3	Design ER models and convert them into normalized relational schemas for enterprise applications	
CO4	Construct and execute advanced SQL queries, stored procedures, and triggers to support business operations and analytical reporting.	
CO5	Analyze transaction-processing concepts including ACID properties, serializability, concurrency control mechanisms, and deadlock handling.	
CO6	Compare relational, distributed and NoSQL database models and justify their suitability for different application environments.	

Unit	Contents	No. of hours
I	Introduction to Database Systems 1.1 Database system concepts 1.2 File system vs DBMS 1.3 Database architecture (Three-schema architecture) 1.4 Data models: Hierarchical, Network, Relational, Object-oriented 1.5 Database users, administrators, roles in database systems	05
II	Database Design for Real Applications 2.1 Requirement analysis from real scenarios 2.2 ER diagram creation using tools 2.3 Mapping ER to relational schema 2.4 Case study-based database modeling	06
III	SQL for Business Applications 3.1 Schema creation for real applications 3.2 CRUD operations in enterprise systems 3.3 Simple queries, Complex joins for reporting 3.4 Subqueries & views for dashboards 3.5 Stored procedures for automation 3.6 Triggers for audit logging	15

IV	Transactions & Multi-User Systems 4.1 Transaction concepts 4.2 ACID properties 4.3 Serializability 4.4 Concurrency control techniques 4.5 Deadlocks 4.6 Recovery mechanisms 4.7 Case study- Banking system, Online ticket booking system	12
V	NoSQL & Modern Data Applications 5.1 Limitations of relational databases in big data 5.2 MongoDB for document storage 5.3 Use cases: social media, IoT, Healthcare records 5.4 Comparison: MySQL vs MongoDB 5.5 Schema-less design	05
VI	Introduction to NoSQL Database 6.1 Introduction to NOSQL DB 6.2 Advantages of NOSQL DB 6.3 Type of NOSQL DB, 6.4 Installation (mlab – mongoose connection), 6.5 Data modelling 6.6 Creating Schemas with Mongoose, Create Database, Drop Database, CRUD Operations	05

References:

1. Silberschatz, A., Korth, H. F., & Sudarshan, S. (2019). *Database system concepts* (7th ed.). McGraw-Hill.
2. Rob, P., & Coronel, C. (2018). *Database systems: Design, implementation, & management* (13th ed.). Cengage Learning.
3. Hsieh, W. K., Pletcher, J., & Pollice, G. (2017). *Practical SQL: A beginner's guide to storytelling with data*. No Starch Press.
4. MongoDB, Inc. (2021). *MongoDB: The definitive guide: Powerful and scalable data storage* (4th ed.). O'Reilly Media.

e-resources:

1. MySQL. (n.d.). *MySQL reference manual*. Oracle. <https://dev.mysql.com/doc/>
2. MongoDB, Inc. (n.d.). *MongoDB manual*. <https://docs.mongodb.com/>
3. Microsoft Learn. (n.d.). *Introduction to Azure SQL*. Microsoft. <https://learn.microsoft.com/azure/azure-sql/>

F.Y.M.Sc. Semester II		
CSA-583	Operating Systems	Credits:02 Hours: 30
CO1	Explain the role, structure, and functions of operating systems.	
CO2	Describe process management and understand how applications execute in operating systems.	
CO3	Explain memory management concepts and their impact on application performance.	
CO4	Describe file system organization and storage management in operating systems.	
CO5	Understand the role of operating systems in Linux environments, cloud computing, and application deployment.	
CO6	Understand shell Scripting	

Unit	Contents	No. of lectures
I	Introduction to Operating Systems and Linux 1.1 Definition and objectives of operating systems 1.2 Functions of operating systems 1.3 Types of operating systems: Single-user, Multi-user, Batch, Time-sharing 1.4 Overview of Linux operating system 1.5 Linux directory structure: /home, /bin, /etc, /var 1.6 Interaction between applications and operating systems	5
II	Process Management and Application Execution 2.1 Process concept and process lifecycle 2.2 Process states and state transitions 2.3 Threads and multithreading concept 2.4 Application execution in operating systems 2.5 Process monitoring and control concept 2.6 Role of operating systems in multitasking environments	7
III	Memory Management and Application Performance 3.1 Memory management concept and objectives 3.2 Logical and physical memory 3.3 Virtual memory concept 3.4 Memory allocation and application execution 3.5 Impact of memory on application performance 3.6 Memory management in modern operating systems	7
IV	File System and Storage Management 4.1 File system concept and objectives 4.2 File operations: Create, read, write, delete 4.3 Directory structure and organization 4.4 Storage management in operating systems 4.5 Application data storage and file handling 4.6 File systems in the Linux operating system	5

V	Operating Systems for Developers and Cloud Computing 5.1 Role of operating systems in application deployment 5.2 Operating systems in cloud computing 5.3 Virtual machines and operating systems 5.4 Containers and operating system support 5.5 Linux as development and deployment environment	6

References:

1. William Stallings, Operating Systems: Internals and Design Principles, Pearson.
2. Achyut S. Godbole, Operating Systems, Tata McGraw-Hill.
3. Kaiwan N Billimoria ,Linux Kernel Programming, Packt Publishing.
4. Rajib Mall, Operating Systems, Pearson.
5. Silberschatz, Abraham, Peter Baer Galvin, and Greg Gagne
Operating System Concepts, 10th Edition, Wiley.
6. Sumitabha Das ,Unix and Shell Programming, Tata McGraw-Hill
7. Andrew S. Tanenbaum and Herbert Bos ,Modern Operating Systems, 4th Edition, Pearson.
8. Brian Ward, How Linux Works: What Every Superuser Should Know, 3rd Edition

F.Y. M.Sc. Semester II		
CSA-584	WEB development using MERN Stack	Credits: 02 Hours :30
Course Outcome (COs) On completion of the course, the students will be able to:		
CO1	Describe web development using different technologies like HTML, CSS, and JavaScript	
CO2	Understand MERN stack development, and understand the fundamentals of databases.	
CO3	Apply various technologies to develop server-side and client-side applications. To create expressive and dynamic websites	
CO4	Plan web development by implementing various techniques of React JS, NodeJS, Expres.JS etc. Plan database designs for various case studies.	
CO5	Justify various web designs and development techniques.	
CO6	Express database designs using MongoDB	

Unit	Contents	No. of hours
I	Induction for Full Stack Web Developer - MERN Stack 1.1. Introduction to Full Stack Web Developer. 1.2. Different approaches to web development 1.3. Introduction to different types of WEB technologies.	02
II	Fundamentals of web development 2.1 Agile and Scrum methodologies, 2.2 HTML, CSS, and JavaScript , 2.3 GIT to manage version control systems	04
III	Develop frontend with REACT 3.1 Introduction to front and back-end web development 3.2 Introduction to REACT 3.3 Introduction to Component-based Programming 3.4 Virtual DOM , JSX & Rendering Elements,	10

	3.5 Components and Props, 3.6 State Management , Event Management, Routing & Redux 3.7 Introduction to React Native 3.8 View, State, Props, and Style 3.9 React Native Apps VS Platform-Specific Apps	
IV	Introduction to Node.js. 4.1 Introduction Node.js, 4.2 Features of Node.js, 4.3 Node.js Installation & configuration, 4.4 Where to use Node, 4.5 Server-side JavaScript, Asynchronous events vs. threads, Performance, 4.6 Server utilization	08
V	Introduction to Express.js 5.1 Routing in Express.js, 5.2 Middleware in Express.js, 5.3 Templating Engines with Express.js 5.4 Handling Static Files, 5.5 Error Handling, RESTful API Development, 5.6 Authentication and Authorization, 5.7 Database Integration, Sessions and Cookies, 5.8 Express.js Best Practices	06

References:

- 1) Amos Q. Haviv, Mean Web Development
- 2) MERN Projects for Beginners: Create Five Social Web Apps Using MongoDB, Express.js, React, and Node
1st ed. Edition by Nabendu Biswas
- 3) Beginning MERN Stack: Build and Deploy a Full Stack MongoDB, Express, React, Node.js App by Greg Lim

Web References

- 1) MongoDB Docs: <https://www.mongodb.com/docs/manual/introduction> mongoose Docs: <https://mongoosejs.com/docs/>
- 2) React Docs : <https://reactjs.org/tutorial/tutorial.html>
- 3) Github Docs : <https://docs.github.com/en>
- 4) <https://pepa.holla.cz/wp-content/uploads/2016/12/Learning-React-Native.pdf>

F.Y. M.Sc. Semester II		
CSA-591	Practical – III (Based on CSA-581, CSA582)	Credits: 2 Hours: 30
Course Outcomes (COs) On completion of the course, the students will be able to:		
CO1	Identify different packages and interfaces of Java to implement concepts of file streams and object-oriented programming.	
CO2	Illustrate test cases and validations required while developing Java-based applications.	
CO3	Apply JDBC to provide a program-level interface for communicating with databases using Java programming.	
CO4	Explain the Java collection API and the Java standard class library.	
CO5	Review different multithreading programming aspects.	
CO6	Write swing-based GUI applications and different web-based JSP applications, including Spring Boot concepts.	

Expt. No.	Title of the Experiment
1	Assignment based on class basics
2	Assignment based on packages, inheritance, interfaces
3	Assignment based on exception handling, multithreading
4	Assignment based on collections, file handling
5	Assignment based on Swings
6	Assignment based on Event handling
7	Assignment based on JDBC, Spring
8	Assignment based on JSP
9	Requirement Analysis and ER Modelling
10	Relational Schema, Table designing, and CRUD operation implementation
11	SQL queries, Joins, and Nested Queries
12	SQL Views, Stored Functions and Procedures
13	Cursors, Triggers
14	Assignment on NoSQL

15	Mini project
----	--------------

References:

1. Cay S. Horstmann, Gary Cornell, Core Java Volume-II-Advanced Features, Eighth Edition, Prentice Hall, Sun Microsystems Press, 2007.
2. Ivan Bayross, Commercial web development using java 2.0, BPB, 2007.
3. Steven Horlzner , Java 2 programming black books, 2007.
4. Herbert Schildt(11th edition), Complete reference Java, 2017
5. Mastering Spring Boot 2.0 O'reilly Publication, 2018

F.Y. M.Sc. Semester II		
CSA-592	Prcatical -IV Project	Credits: 02
Course Outcome (COs) On completion of the course, the students will be able to:		
CO1	Describe a systematic understanding of project contents.	
CO2	Illustrate knowledge of computing and mathematics appropriate to the discipline.	
CO3	Implement project management knowledge, processes, tools and techniques to achieve project success.	
CO4	Design, implement, and evaluate a computational system to meet desired needs within realistic constraints.	
CO5	Integrate the identified modules using techniques and tools.	
CO6	Evaluate the generated modules using evaluation techniques and tools.	

The objective of field project is to make the students understand Requirement analysis, design and implementation cycle. Any open problem statement can be taken for implementation. The system can be designed in any programming language implemented on any platform.

Sr.	Date	Details of Project work	Remark	Project guide sign

F.Y.M.Sc. Semester II		
CSA-593	Practical -V (Based on CSA-583: OS)	Credits: 2 Hours: 60
Course Outcomes (COs) On completion of the course, the students will be able to:		
Expt No	Title of the Experiment	
1	Linux Installation and Environment Setup	
2	Getting familiar with Linux commands and windows commands	
3	CPU Scheduling Simulation using C / Java	
4	CPU Scheduling Simulation using C / Java	
5	Memory Management Simulation – Page Replacement using C / Java	
6	File System Simulation using C / Java	
7	Linux Commands – File and Directory Management	
8	Linux Commands – System and Process Management	
9	Linux Commands – User, Permissions, and Environment	
10	Shell Script – File Management Automation	
11	Shell Script – System Monitoring Tool	
12	Shell Script – User Management Automation	
13	Shell Script – Backup and Restore System	
14	Shell Script – Log File Analyzer	
15	Mini Project – Linux Shell Script-Based System Tool	

F.Y. M.Sc. Semester II		
CSA- 594	Practical- VI (Based on CSA- 584: MERN)	Credits : 02 Hours : 60
Course Outcomes (COs) On completion of the course, the students will be able to:		
CO1	Apply MERN Technologies to develop Web Applications.	
CO2	Understand how MongoDB can be used for data handling and validation in a web application.	
CO3	Design and Develop web applications using various libraries and frameworks	
CO4	Understand functional components, state components, lifecycle, and routing in React.	
CO5	Apply various Express JS and React JS methods to create an interactive web application.	
CO6	Analyze outputs of web-based applications and their performance	

Expt. No.	Title of the Experiment
1	Assignment based on HTML, CSS
2	Assignment based on JavaScript
3	Assignment based on JavaScript
4	Assignment based on Express JS
5	Assignment based on Express JS
6	Assignment based on Express JS
7	Assignment based on Node JS
8	Assignment based on Node JS
9	Assignment based on Node JS
10	Assignment based on React
11	Assignment based on React

12	Assignment based on React
13	Case Study: User Authentication System
14	Case Study: Notes App
15	Case Study: E-Commerce System

References:

1. Amos Q. Haviv, Mean Web Development
2. MERN Projects for Beginners: Create Five Social Web Apps Using MongoDB, Express.js, React, and Node 1st ed. Edition by Nabendu Biswas
3. Beginning MERN Stack: Build and Deploy a Full Stack MongoDB, Express, React, Node.js App by Greg Lim

F.Y. M.Sc. Semester II		
CSA- 651	OJT/FP	Credits : 04 Hours : 120
Course Outcomes (COs) On completion of the course, the students will be able to:		
CO1	Apply knowledge acquired during the M.Sc. Computer Applications programme to real-world computing problems.	
CO2	Analyse requirements and identify appropriate technologies, tools and methodologies for solving practical problems.	
CO3	Develop and implement an appropriate software, computing or research-oriented solution	
CO4	Demonstrate professional skills including communication, teamwork, time management, documentation and workplace ethics.	
CO5	Analyse challenges encountered during the OJT/Field Project and propose appropriate solutions	
CO6	Document, demonstrate and effectively present the work carried out during the OJT/Field Project.	

Course Description

CSA-561 is an experiential learning component designed to provide students with an opportunity to apply the knowledge and technical skills acquired during the M.Sc. Computer Applications programme to real-world problems and professional situations.

Students may complete the course through either of the following approved pathways:

- Pathway A – On-the-Job Training (OJT): Structured training/work experience in an approved organization.
- Pathway B – Field Project (FP): An approved practical, industry-oriented, institutional, research-oriented, or community-based computing project.

The work shall be undertaken individually under the guidance of a Faculty Guide and, wherever applicable, an Industry/Organization Supervisor.

Course Objectives

- Provide students with practical exposure to real-world computing environments.
- Enable students to apply theoretical concepts and technical skills to practical problems.
- Develop professional, analytical, problem-solving and communication skills.
- Familiarize students with professional practices, teamwork, documentation and workplace ethics.
- Develop the ability to identify, analyse and solve computing problems independently.
- Provide experience in planning, executing, documenting and presenting a practical project or professional assignment.

- Enhance students' employability and readiness for professional or research careers.

Pathway A – On-the-Job Training (OJT) The student shall undertake training in an approved IT/software company, startup, government or semi-government organization, research organization, educational/research institution, technology organization, industry/enterprise, or other organization approved by the Department. The OJT may involve Software Development, Web Development, Mobile Application Development, Database Management, Data Analytics, Artificial Intelligence/Machine Learning, Cloud Computing, Cyber Security, Software Testing, DevOps, Networking, Business Intelligence, or other relevant computing areas OJT Requirements • OJT shall be undertaken individually. • The organization and nature of work shall be approved by the Department before commencement. • The student shall work under an Industry/Organization Supervisor wherever applicable. • A Faculty Guide shall monitor the student's progress. • The student shall maintain an OJT Diary/Logbook. • The student shall maintain regular attendance as prescribed by the Department/host organization. • The student shall follow the rules, regulations, professional ethics and confidentiality requirements of the host organization. • The student shall submit an OJT Completion/Performance Certificate from the organization, wherever applicable. • The student shall submit an OJT Report and appear for presentation/viva-voce.	Pathway B – Field Project (FP) The Field Project shall involve the identification and systematic solution of a real-world computing problem. It may be undertaken in collaboration with an industry, educational institution, government organization, NGO, startup, research organization, community/field organization, college/department, or other approved organization. A Field Project may also be undertaken without association with a specific external organization, provided the project is approved by the Department and has a clearly defined practical problem and outcome. Suggested areas include Web/Mobile Application Development, Software Development, Data Analytics, AI/ML, Database Applications, Cloud Applications, Business Intelligence, Cyber Security, Automation, Image Processing, Software Testing, IoT Applications, Educational Technology, Social/Community Computing, and Research-oriented Computing Applications. 6. Field Project Process Problem Identification → Literature/Background Study → Requirement Analysis → Planning → Design → Implementation → Testing/Validation → Documentation → Presentation/Demonstration The project should have a clearly defined problem statement, objectives, scope, methodology, appropriate technology/tools, expected and achieved outcomes, and proper documentation.
---	--

Faculty Guide and Industry Supervisor

Faculty Guide

- Approve the proposed OJT/FP.
- Guide the student in defining objectives and scope.
- Monitor progress periodically.
- Review the logbook/project work.
- Guide report preparation.
- Evaluate the student's academic and professional progress.

Industry/Organization Supervisor

- Assign appropriate work.
- Monitor the student's performance.
- Provide professional/technical guidance.
- Verify attendance/work undertaken.
- Provide feedback/evaluation at completion of OJT/FP.

Documentation Guideline

For OJT: Joining/permission letter, attendance record, OJT diary/logbook, periodic progress report, supervisor feedback, completion certificate and final OJT report.

For Field Project: Project proposal, approval of Faculty Guide, progress diary, requirement specification, design/architecture, implementation record, testing/results, final project report, demonstration and presentation.

Suggested Report Structure

OJT Report	Field Project Report
• Title Page • Certificate • Declaration • Acknowledgement • Table of Contents • Organization Profile • Introduction • Objectives of OJT • Roles and Responsibilities • Technologies/Tools Used • Work Undertaken • Methodology/Process • Problems Encountered and Solutions • Skills and Learning Outcomes • Contribution to the Organization • Conclusion • Future Scope • References • Annexures	• Title Page • Certificate • Declaration • Acknowledgement • Abstract • Table of Contents • Introduction • Problem Statement • Objectives • Scope • Literature/Background Study • Requirement Analysis • Methodology • System Design/Architecture • Implementation • Testing and Results • Discussion • Learning Outcomes • Conclusion • Future Scope • References • Annexures