Syllabi and Scheme of Examinations
for
Minor Courses for Under Graduate Programs
(Single Major / Multidisciplinary Programs) of Computer Science

(Based on Curriculum and Credit Framework for UG Programs under NEP)

With Effect from the Session 2024-25

MAHARSHI DAYANAND UNIVERSITY ROHTAK (HARYANA)

SYLLABI AND SCHEME OF EXAMINATIONS FOR MINORCOURSES FOR

UNDER GRADUATE (SINGLE MAJOR/MULTIDISCIPLINARY PROGRAMS/ SINGLE MAJOR PROGRAM AFTER 2nd SEMESTER OF MULTIDISCIPLINARY PROGRAM

Multidisciplinary Course (MDC)	**TYPE OF PROGRAM**				**Credits Distribution**			**Total Credits**	**Workload**			**Total Workload**	**Marks**			
	SINGLE MAJOR PROGRAM	**MULTIDISCIPLINARY PROGRAM / SINGLE MAJOR PROGRAM AFTER 2nd SEMESTER OF MULTIDISCIPLINARY PROGRAM**	**Nomenclature of Course**	**Course Code**	**L**	**T**	**P**		**L**	**T**	**P**		**Theory**		**Practical**	**Total Marks**
	SEMESTER	**SEMESTER**											**External**	**Internal**		
MIC 1 @ 4 credits	1	1	Fundamentals of Computing and Problem Solving Using C	24CSC401MI01	2	0	2	4	2	0	4	6	35	15	50	100
MIC 2 @ 4 credits	2	3	Internet and Web Design	24CSC402MI01	3	0	1	4	3	0	2	5	50	25	25	100
MIC 3 @ 4 credits	3	5 (6)	Object Oriented Concepts and C++ Programming	25CSC403MI01	3	0	1	4	3	0	2	5	50	25	25	100
MIC 4 (VOC) @ 4 credits	4	4	Java Programming	25CSC404MV01	2	0	2	4	2	0	4	6	35	15	50	100
MIC 5 (VOC) @ 4 credits	5	5	Database Design and MySQL	26CSC405MV01	3	0	1	4	3	0	2	5	50	25	25	100
MIC 6 (VOC) @ 4 credits	6	6	Python Programming	26CSC406MV01	2	0	2	4	2	0	4	6	35	15	50	100
MIC 7 (VOC) @ 4 credits	7	7	MERN Stack Development	27CSC407MV01	2	0	2	4	2	0	4	6	35	15	50	100

MIC 8 (VOC) @ 4 credits	8	8	Data Analytics using R (Option-1)	27CSC408MV01	2	0	2	4	2	0	4	6	35	15	50	100
			OR													
			Machine Learning and Generative AI(Option-II)	27CSC408MV01	3	0	1	4	3	0	2	5	50	25	25	100

L: Lecture; T: Tutorial; P: Practical

Note:

1. The Syllabi and Scheme of Examinations (SOE) for Minor (Vocational) Courses for UG Semester 7 and Semester 8 will be same as applicable for Vocational Course in Post Graduate semester 1 and semester 2 respectively.
2. Course coding of Minor courses for Single Major Programs will be applicable for Multidisciplinary Programs/ Multidisciplinary Programs after 2nd semester irrespective of their offering in any semester.
3. The student who select any Minor Course (MIC) of any discipline in first semester should study the Minor courses (MIC) in the same discipline in the subsequent semesters. However, while exercising the option for choosing Minor Vocational Course MIC (VOC), the student may opt the discipline either related to the discipline of Minor Course or the discipline of Major Course or any other discipline as per his/her choice.

Syllabi for Minor Courses for Under Graduate Programs (Single Major / Multidisciplinary Programs) of Computer Science

Academic Session: 2024-25
First Semester

Name of the Program	**Minor Courses (Single Major /Multidisciplinary Program)**	**Program Code**	**-----**
Name of the Course	**Fundamentals of Computing and Problem Solving using C**	**Course Code**	**24CSC401MI01**
Hours/Week	**2+0+4**	**Credits(L:T:P)**	**2:0:2**
Max. Marks.	**Theory: 50 (35+15) Practical: 50**	**Time of end term Examination**	**3 Hours**

Note: The examiner has to set nine questions in all by setting two questions from each Unit and Question No. 1 consisting of short-answer type questions covering the entire syllabus. Student will be required to attempt five questions in all by selecting one question from each Unit and Question No. 1, which is compulsory. All questions will carry equal marks.

Course Objective:

In this course, students will delve into the fundamental aspects of computing and gain a comprehensive understanding of computer systems and will acquire foundational programming skills. With a focus on the C programming language, students will explore program structure, variables, data types, control structures, pointers, input/output operations and modular programming principles to organize and optimize code structure. Additionally, they will adept at debugging and testing techniques to ensure program correctness. By the end of the course, students will have gained practical skills and knowledge applicable to real-world scenarios, equipping them to tackle a variety of programming challenges with confidence and proficiency.

Course Outcomes:

By the end of the course the students will be able to:

CO1: Gain knowledge of essential computing concepts and its applications in various fields.
CO2: Develop proficiency in writing, debugging, and executing C programs to efficiently solve computational problems.
CO3: Demonstrate an understanding of data types, control structures, functions, arrays, and pointers.
CO4: Cultivate problem-solving skills through algorithmic thinking and programming techniques in C.
CO5: Apply modular programming principles to effectively organize and structure code for improved maintainability, scalability, and code reuse in C programming projects.

Unit – I

Computing Fundamentals: Overview of computing principles and history, Generations of Computers, Block Diagram along with its components, Classification of computers, Applications of computers in various fields. Input/Output Devices, Memory: Concept of primary & secondary memory, Cache Memory, Secondary storage devices.

Basics of Networking & Operating System: Introduction to computer networking, Network types, Network topologies, Internet and its applications; Operating system and its functions.

Unit– II

Introduction to software development methodologies: Basics of algorithmic thinking and problem-solving strategies. Planning the Computer Program: Problem definition, Program design, Debugging, Types of errors in programming, Techniques of Problem Solving-Flowcharting, Algorithms

Introduction to the C programming language: History of C, Importance of C, Elements of C: C character set, identifiers and keywords, Data types, Constants and Variables, Assignment statement, Symbolic constant, Structure of a C Program, printf(), scanf()Functions, Operators & Expression, type casting and

conversion, operator hierarchy & associativity.

Unit–III

Decision making & Branching: Decision making with IF statement, IF-ELSE statement, Nested IF statement, ELSE-IF ladder, switch statement, go to statement. **Looping:** while, do-while and for loop, jumps in loops, break, continue statement, Nested loops.

Functions and modular programming concepts: Standard Mathematical functions, Input/output: Unformatted & formatted I/O function in C, Input functions, output functions, string manipulation functions. User defined functions: Introduction/Definition, function prototype, Local and global variables, passing parameters, recursion.

Unit– IV

Arrays & Pointers: Definition, types, initialization, processing an array, passing arrays to functions declaration and initialization of string, Input/output of string data, Introduction to pointers.**Advance Concepts of C Programming:** Pointers and memory management in C; File input/output operations in C; Dynamic memory allocation and deallocation.

Practical applications of C programming in software development: Algorithmic problem-solving using C programming constructs; Design and implementation of C programs; Debugging and testing techniques for C programs; Best practices and coding standards in C programming.

Suggested Readings:

1. Gottfried, Byron S.: Programming with C, Tata McGraw Hill
2. Gill Nasib Singh: Computing Fundamentals and Programming in C, Khanna Book Publishing Company (Private) Limited, New Delhi.
3. Balagurusamy, E.: Programming in ANSI C, Tata McGraw-Hill.
4. Gill, Nasib Singh: Handbook of Computer Fundamentals, Khanna Book Publishing Company (Pvt.) Limited, New Delhi.
5. Jeri R. Hanly& Elliot P. Koffman: Problem Solving and Program Design in C, Addison Wesley.
6. Yashwant Kanetker: Let us C, BPB.
7. Rajaraman, V.: Computer Programming in C, PHI.
8. Yashwant Kanetker: Working with C, BPB.
9. Any other book(s) covering the contents of the subject.

Note: Latest and additional good books may be suggested and added from time to time.

List of Practical/Programs/Lab Assignments

1. WAP to find factorial of a number
2. WAP to check whether number is Perfect Square or not
3. WAP to find SUM and AVERAGE of two numbers
4. WAP to convert temperature from Fahrenheit to Celsius and Celsius to Fahrenheit
5. WAP to read and print an employee's detail using structure
6. WAP to convert number from Decimal to Binary
7. WAP to check whether number is Palindrome or not
8. WAP to print "Hello World".
9. WAP to swap two numbers without using third variable.
10. WAP to check whether a number if Armstrong or not.
11. WAP to check whether a number if Even or Odd.
12. WAP to print all leap years from 1 to N.
13. WAP to calculate employee gross salary.
14. WAP to print tables of numbers from 1 to 20.
15. WAP to print star/pyramid series.
16. WAP to convert temperature from Celsius to Fahrenheit and vice versa.
17. WAP to convert number from Decimal to Binary.
18. WAP to convert number from Binary to Decimal.
19. WAP to print ASCII Table.
20. WAP to get and set current system date and time.
21. WAP to run dos command.
22. WAP to check whether a given number is palindrome or not using Bitwise Operator
23. WAP to count number of bits set to 1 in an Integer

24. WAP to check if all the bits of a given integer is one (1)
25. WAP to find the Highest Bit Set for any given Integer
26. WAP to Count the Number of Trailing Zeroes in an Integer
27. WAP to find the Biggest Number in an Array of Numbers using Recursion
28. WAP to accept Sorted Array and do Search using Binary Search
29. WAP to Cyclically Permute the Elements of an Array
30. WAP to print the sum and product of digits of an integer.
31. WAP to reverse a number.
32. WAP to compute the sum of the first n terms of the following series
    S = 1+1/2+1/3+1/4+……
33. WAP to compute the sum of the first n terms of the following series
    S =1-2+3-4+5…….
34. Write a function that checks whether a given string is Palindrome or not. Use this function to find whether the string entered by user is Palindrome or not.
35. Write a function to find whether a given no. is prime or not. Use the same to generate the prime numbers less than 100.
36. WAP to compute the factors of a given number.
37. Write a macro that swaps two numbers. WAP to use it.
38. WAP to print a triangle of stars as follows (take number of lines from user):
    *
    ***
    *****
    ****
    ****
39. WAP to perform following actions on an array entered by the user:
    i) Print the even-valued elements ii) Print the odd-valued elements
    iii) Calculate and print the sum and average of the elements of array
    iv) Print the maximum and minimum element of array
    v) Remove the duplicates from the array vi) Print the array in reverse order
    The program should present a menu to the user and ask for one of the options. The menu should also include options to re-enter array and to quit the program.
40. WAP that prints a table indicating the number of occurrences of each alphabet in the text entered as command line arguments.
41. Write a program that swaps two numbers using pointers.
42. Write a program in which a function is passed address of two variables and then alter its contents.
43. Write a program which takes the radius of a circle as input from the user, passes it to another function that computes the area and the circumference of the circle and displays the value of area and circumference from the main() function.
44. Write a program to find sum of n elements entered by the user. To write this program, allocate memory dynamically using malloc() / calloc() functions or new operator.
45. Write a menu driven program to perform following operations on strings:
    a. Show address of each character in string
    b. Concatenate two strings without using strcat function.
    c. Concatenate two strings using strcat function.
    d. Compare two strings
    e. Calculate length of the string (use pointers)
    f. Convert all lowercase characters to uppercase
    g. Convert all uppercase characters to lowercase
    h. Calculate number of vowels
    i. Reverse the string
46. Given two ordered arrays of integers, write a program to merge the two-arrays to get an ordered array.
47. WAP to display Fibonacci series (i)using recursion, (ii) using iteration
48. WAP to calculate Factorial of a number (i)using recursion, (ii) using iteration
49. WAP to calculate GCD of two numbers (i) with recursion (ii) without recursion.
50. WAP to design calculator with basic operations using switch.

Any other Programs/Lab Assignments given by the teachers.

Academic Session: 2024-25
Second Semester

Name of the Program	**Minor Courses (Single Major /Multidisciplinary Program)**	**Program Code**	**-----**
Name of the Course	**Internet and Web Design**	**Course Code**	**24CSC402MI01**
Hours/Week	**3+0+2**	**Credits(L:T:P)**	**3:0:1**
Max. Marks.	**Theory:75 (50+25)** **Practical: 25**	**Time of end term examination**	**3 Hours**

Note: The examiner has to set nine questions in all by setting two questions from each Unit and Question No. 1 consisting of short-answer type questions covering the entire syllabus. Student will be required to attempt five questions in all by selecting one question from each Unit and Question No. 1, which is compulsory. All questions will carry equal marks.

Course Objective:
The objective of this course is to acquire knowledge and Skills for creation of Web Sites. Also to acquire the knowledge regarding creation of Web applications using tools and techniques used in industry and how to design a basic web site using HTML & CSS to demonstrate responsive web design.

Course Outcomes:
By the end of the course the students will be able to:

CO1: Understand the fundamental concept to Internet and search engine.
CO2: Understand the concept of Web-casting techniques.
CO3: Understand the concept of website planning.
CO4: Get exposure of HTML.
CO5: Gain good exposure of CSS.

Unit–I

Introduction to Internet and World Wide Web: A brief Introduction to the Internet, Evolution of World Wide Web; Basic features; Web Browsers; Web Servers; Hypertext Transfer Protocol, URLs; Searching and Web-Casting Techniques; Search Engines and Search Tools, Domain Name System, Home Page, Web page and Website.

Unit–II

Web Publishing: Hosting your Site; Internet Service Provider; Phases of Planning and designing your Website; Steps for developing your Site; Choosing the contents;

Web Development: Introduction to HTML; Hypertext and HTML; HTML Document Features; HTML command Tags; Headers; Text styles; Text Structuring; Text colors and Background; Formatting text.

Unit–III

List: Definition and types of Lists - Ordered and Unordered, Table Creation and Layouts. Images; Inserting Graphics; Frame Creation and Layouts; Creating Links; Working with Forms and Menus; Working with Radio Buttons and Check Boxes; Text Boxes; Page layouts.

Unit–IV

Cascading Style Sheets (CSS): Basic Concepts, Properties, Creation of Style Sheets. Common Tasks with CSS: Text, Fonts, Margins, Links, Tables, Colors. Marquee. Mouse Overs. Filters and Transitions. Adding Links. Adding Tables. Adding Forms. Adding Image and Sound. Use of CSS in HTML Documents, Linking and Embedding of CSS in HTML.

Suggested Readings:

1. Raj Kamal: Internet and Web Technologies, Tata McGraw-Hill.
2. Ramesh Bangia: Multimedia and Web Technology, Firewall Media.
3. Thomas A. Powell: Web Design: The Complete Reference, 4/e, Tata McGraw-Hill

4. Wendy Willard: HTML Beginners Guide, Tata Mc Graw-Hill.
5. Deitel and Goldberg: Internet and World Wide Web, How to Program, PHI.
6. Any other book(s) covering the contents of the subject.

Note: Latest and additional good books may be suggested and added from time to time.

List of Programs/Lab Assignments

1. Write HTML code to display your education details in a tabular format.
2. Write HTML code to display your CV on a web page.
3. Write HTML code to create a Home page having three links: About Us, Our Services and Contact Us. Create separate web pages for the three links.
4. Write HTML code to create a login form. On submitting the form, the user should get navigated to a profile page.
5. Write HTML code to create a Registration Form. On submitting the form, the user should be asked to login with new credentials.
6. Write HTML code to create your Institute website, Department Website and Tutorial website for specific subject.
7. Write HTML code to illustrate the usage of the following:
8. Create the following: Ordered List, Unordered List and Definition List
9. Write HTML code to create a frameset having header, navigation and content sections.
10. Write HTML code to demonstrate the usage of inline CSS.
11. Write HTML code to demonstrate the usage of internal CSS.
12. Write HTML code to demonstrate the usage of external CSS.
13. Write HTML program to create a webpage to show different art forms of India, with appropriate title on the title bar. Use different heading tags for the headings, and list them using ordered list.
14. Write HTML program to create sections in the document using appropriate tags and apply different color as background to them. Use internal hyperlinks to move to different points within the page.
15. Write HTML program to insert a picture on the webpage, giving description for the picture in a paragraph. Use properties of height, width, hspace, vspace and align, with different values.
16. Write HTML Program, to create a profile of 2 pages, the First page containing the applicant's picture with personal details using unordered lists, and the second containing Educational details using tables. Use hyperlinks to move to the next page.

Any other Programs/Lab Assignments given by the teachers.

Academic Session: 2025-26
Third Semester

Name of the Program	**Minor Courses (Single Major /Multidisciplinary Program)**	**Program Code**	**-----**
Name of the Course	**Object-Oriented Concepts and C++ Programming**	**Course Code**	**25CSC403M101**
Hours/Week	**3+0+2**	**Credits (L: T:P)**	**3:0:1**
Max. Marks.	**Theory: 75 (50+25)** **Practical: 25**	**Time of end term examination**	**3 Hours**

Note: The examiner has to set nine questions in all by setting two questions from each Unit and Question No. 1 consisting of short-answer type questions covering the entire syllabus. Student will be required to attempt five questions in all by selecting one question from each Unit and Question No. 1, which is compulsory. All questions will carry equal marks.

Course Objective:

The objective of this course is to introduce the concepts of object-oriented programming (OOP) using C++, introducing OOPS concepts (objects, classes) and C++ syntax (variables, data types) to design well-structured and reusable code.

Course Outcomes:

By the end of the course the students will be able to:
CO1: Introduce the basic concepts of object-oriented programming.
CO2: Understand the structure of a C++ program and gain understanding about programming.
CO3: Allocate dynamic memory, access private members of class and the behaviour of inheritance and its implementation.
CO4: Introduce polymorphism, interface design and overloading of operator.
CO5: Handle backup system using file, general purpose template and handling of raised exception during programming.

Unit – I

Object Oriented Programming: Basic concept of OOP, Comparison of Procedural programming and OOP, Application of object-oriented programming. Characteristic of OOP: Objects, classes, Encapsulation, Data Abstraction, Inheritance, Polymorphism, Dynamic Binding, Message Passing.

Structure of C++ programming language: Basic syntax and structure of C++ programs, Data types, variables, and constants in C++, Control structures: decision making and looping constructs. Functions and parameter passing in C++, Arrays and strings in C++, Pointers.

Unit – II

Object Oriented Concepts: Class, Object, Memory allocation for objects, Member functions and data members, **Access specifiers**: public, private, protected, Encapsulation and data hiding, Constructors and destructors, Accessor and mutator functions, Friend functions.

Memory Management: Dynamic Memory allocation: new and delete, Static class members, Constructors, parameter Constructors and copy Constructors, Destructors.

Unit – III

Inheritance: Types of Inheritance, Overriding Base Class members in a derived class, Public, Protected and Private Inheritance, Constructors and destructors in derived classes, Virtual Inheritance.

Polymorphism: function overloading, Operator overloading, Overloading Unary and Binary Operators, Abstract classes and pure virtual functions, Dynamic polymorphism

Unit – IV

Exception handling: Try, Throw, Catch, Multiple catch, Re-Throwing an Exception specifications, Processing unexpected exceptions

Templates: Function Templates, Overloading Template function, Class Templates, namespaces and Overview of Standard Template Library.

Suggested Readings:

1. H. Schildt: C++-The Complete Reference, Tata McGraw Hill Publications.
2. 2.D. Samanta: Object Oriented Programming with C++ and Java, PHI
3. E. Balagurusamy: Object Oriented Programming with C++, TMH.
4. R. Rajaraman: Object Oriented Programming and C++ by, New Age International
5. R. Subburaj: Object Oriented Programming with C++, VIKAS.
6. Any other book(s) covering the contents of the subject.

Note: Latest and additional good books may be suggested and added from time to time.

List of Programs/Lab Assignments

1. Write a C++ program to define a class called `Rectangle` with attributes `length` and `width`, and display the area of the rectangle.
2. Create a C++ program to take two numbers as input from the user and display their sum.
3. Write a C++ program to demonstrate decision-making constructs like if-else and looping constructs like for and while loops.
4. Create a C++ program to demonstrate function overloading by defining multiple functions with the same name but different parameters.
5. Implement a C++ program to demonstrate the concept of inheritance by creating a base class `Shape` and derived class `Rectangle`. Display the area of the rectangle using inheritance.
6. Create a C++ program to define a class called `Student` with attributes `name` and `roll number`. Use member functions to input and display student details.
7. Write a C++ program to demonstrate the use of constructor and destructor in a class.
8. Implement a C++ program to showcase the use of access specifiers (`public`, `private`, `protected`) in a class.
9. Write a C++ program to demonstrate dynamic polymorphism using virtual functions.
10. Write a C++ program to demonstrate the working of friend function.
11. Implement a C++ program to demonstrate the use of pointers to objects. Define a class `Book` with attributes `title` and `author`, and use pointers to access and display book details.
12. Write a C++ program to perform file input and output operations, including opening, reading, writing, and closing files.
13. Implement error handling during file operations in a C++ program, by handling exceptions and error codes.
14. Write a C++ program to handle exceptions using `try-catch` blocks.
15. Create a C++ program to implement a simple template function to find the largest amomg two numbers. Test the function with different data types.

Any other Programs/Lab Assignments given by the teachers.

Academic Session: 2025-26
Fourth Semester

Name of Program	**Minor Courses (Single Major /Multidisciplinary Program)**	**Program Code**	
Name of Course	**Java Programming**	**Course Code**	**25CSC404MV01**
Hours/Week	**2+0+4**	**Credits (L: T:P)**	**2:0:2**
Max. Marks	**Theory: 50(35+15)** **Practical: 50**	**Time of Examination**	**3 Hours**

Note: The examiner has to set nine questions in all by setting two questions from each Unit and Question No. 1 consisting of short-answer type questions covering the entire syllabus. Student will be required to attempt five questions in all by selecting one question from each Unit and Question No. 1, which is compulsory. All questions will carry equal marks.

Course Outcomes:

This course is designed to provide students with foundational knowledge of Java programming, focusing on its environment, syntax, and structure. It introduces the key principles of object-oriented programming (OOP), including classes, inheritance, and interfaces. The course also emphasizes error handling, package creation, and graphical user interfaces to enable the development of robust Java applications.

Course Objectives:

By the end of the course, the students will be able to:

CO1: To introduce the basics features of Java.
CO2: Set up the Java development environment and demonstrate proficiency in fundamental programming constructs, including data types, control flow, and basic syntax.
CO3: Work with arrays, strings, and develop modular programs using classes, objects, and access modifiers.
CO4: Apply inheritance, polymorphism, and interfaces to build flexible and reusable programs.
CO5: Handle exceptions effectively, create packages, and design graphical user interfaces using Java frameworks.

Unit I

Overview of Java: Java Features, Setting Up the Java Development Environment, Writing, Compiling, and Running a Java Program, Java Virtual Machine (JVM) and Bytecode, Java Development Kit (JDK) and Java Runtime Environment (JRE)

Basic Language Elements:Java Syntax and Structure, Identifiers, Keywords, Literals, Comments, Operators Assignments, Data Types, Variables and its types, Constants, Expressions, Control Flow Statements: if-else, switch, loops (for, while, do-while).

Unit II

Array: Initializing, Single Dimensional Array, Operation on String, Mutable & Immutable String, Using Collection, Bases Loop for String, Creating Strings using StringBuffer.

Class Fundamentals: Object, Object Life cycle, Creating and Operating Objects, Constructor & initialization code block, Access Control, Inner Class, Abstract Class, Argument Passing Mechanism, Method Overloading, Recursion, Use of Access Modifiers with Classes & Methods.

Unit III

Introduction to Inheritance: Use and Benefits of Inheritance in OOPs, Types of Inheritance in Java, Inheriting Data members and Methods, Role of Constructors in inheritance, Overriding Super Class Methods, Use of "super", Polymorphism in Java.

Interface: Purpose of interface, defining an interface, implementing interfaces, Interface reference variables, Interface with variables, Extending interfaces

Unit IV

Exception Handling: Types of Errors in Java, Try-Catch Blocks and Finally Clause, Throw and Throws Keywords, Creating Custom Exceptions.

Packages: Package as Access Protection, Defining Package, CLASSPATH Setting for Packages, Import and Naming Convention for Packages.GUI programming.

Suggested Readings:

1. The Complete Reference JAVA, TMH Publication.
2. Beginning JAVA, Ivor Horton, WROX Public.
3. JAVA 2 UNLEASHED, Tech Media Publications.
4. JAVA 2(1.3) API Documentations.
5. Any other book(s) covering the contents of the subject.

Note: Latest and additional good books may be suggested and added from time to time.

List of Practical/Programs/Lab Assignments

1. Set up the Java Development Kit (JDK) and compile & execute a basic Java program.
2. Write a Java program to demonstrate variables, data types, and type casting.
3. Implement control flow structures (if-else, switch-case, loops) in Java.
4. Develop an array-based Java program to perform basic operations (insert, delete, search).
5. Write a program to manipulate strings using String and StringBuffer classes.
6. Implement a Java class with constructors and object instantiation.
7. Demonstrate the use of method overloading and recursion.
8. Create a simple calculator program using user-defined functions and access modifiers.
9. Implement single and multilevel inheritance using Java classes.
10. Implement polymorphism and method overriding in Java.
11. Develop an application that demonstrates the working of interfaces.
12. Write a program to handle exceptions using try-catch-finally blocks.
13. Create and handle user-defined exceptions in Java.
14. Write a Java program to define and use custom packages.
15. Implement file handling in Java (read/write operations using FileReader/FileWriter).
16. Develop a basic GUI-based Java program using AWT components (Button, Label, TextField).
17. Create an event-driven Java GUI application.
18. Build a basic student record management system using OOP concepts.
19. Implement collection framework classes such as ArrayList and HashMap.
20. Design a multithreaded Java application to perform parallel computation.

Any other Practical/Programs/Lab Assignments given by the teachers.

Academic Session: 2026-27
Fifth Semester

Name of the Program	**Minor Courses (Single Major /Multidisciplinary Program)**	**Program Code**	**-----**
Name of the Course	**Database Design and MySQL**	**Course Code**	**26CSC405MV01**
Hours/Week	**3+0+2**	**Credits (L:T:P)**	**3:0:1**
Max. Marks.	**Theory: 75 (50+25)** **Practical: 25**	**Time of end term examination**	**3 Hours**

Note: The examiner has to set nine questions in all by setting two questions from each Unit and Question No. 1 consisting of short-answer type questions covering the entire syllabus. Student will be required to attempt five questions in all by selecting one question from each Unit and Question No. 1, which is compulsory. All questions will carry equal marks.

Course Objective:

Introduce fundamental concepts of databases, data modelling, and relational database management systems (RDBMS). Develop proficiency in database design using ER modelling, normalization, and SQL. Equip students with practical skills to implement and manage databases using MySQL. Prepare students to optimize, secure, and deploy databases for real-world applications.

Course Outcomes:

By the end of the course the students will be able to:

CO1: Understand basic concepts of database and its relevance.
CO2: Explore core database concepts, architectures, and applications.
CO3: Design efficient databases using ER diagrams and normalization techniques.
CO4: Implement and query databases using SQL and MySQL.
CO5: Optimize and secure databases for scalability and integrity**.**

Unit – I

Database Fundamentals: Definition, purpose, and types of databases (flat-file vs. relational). Components of a DBMS:Storage engine, query processor, transaction manager. Applications of databases in industries (e.g., healthcare, e-commerce etc).
Relational Model Basics: Tables, rows, columns, keys (primary, foreign, candidate)**.** Integrity constraints:**E**ntity, referential, domain**. Introduction to MySQL:** Overview of MySQL as an RDBMS. MySQL architecture and installation.

Unit – II

Entity-Relationship (ER) Modelling: Entities, attributes, relationships (1:1, 1:N , M:N). ER diagram notation (Chen/UML) and case studies. Relational Schema Design: Converting ER diagrams to relational tables.
Normalization: Functional dependencies, anomalies (insertion, deletion, update). 1NF, 2NF, 3NF, and BCNF with practical examples.

Unit – III

Structured Query Language (SQL): DDL: CREATE, ALTER, DROP tables. DML: INSERT, UPDATE, DELETE. DQL: SELECT with clauses (WHERE, GROUP BY, JOIN, subqueries).

Advanced SQL in MySQL: Built-in functions (string, numeric, date/time). Views, indexes, and stored procedures.**Transactions and Concurrency:** ACID properties, COMMIT, ROLLBACK, isolation levels.

Unit – IV

Database Optimization: Query optimization using EXPLAIN, indexing strategies. Normalization vs. denormalization tradeoffs.**Security and Backup:** User management, roles, and privileges in MySQL. Data encryption, backups (mysqldump), and recovery.

Integration and Scalability: Connecting MySQL with applications. Introduction to cloud databases (e.g., AWS RDS, MySQL on Azure etc).

Suggested Readings:

1. Elmasri and Navathe, Fundamentals of Database Systems, Pearson Education.
2. Korth, Silberchatz, Sudarshan, Database System Concepts, McGraw-Hill.
3. Raghu Ramakrishnan, Johannes Gehrke, Database Management Systems, McGraw-Hill
4. Peter Rob and Coronel, Database Systems, Design, Implementation and Management, Thomson Learning.
5. C.J.Date, Longman, Introduction to Database Systems, Pearson Education.
6. Thomas Connolly,Carolyn Begg, Database Systems, Pearson Education.
7. Any other book(s) covering the contents of the subject.

Note: Latest and additional good books may be suggested and added from time to time.

List of Practical/Programs/Lab Assignments

1. Create a database and show table creation. Define tables with primary and foreign keys.
2. Basic SQL Queries – Retrieve, filter, and sort data using SELECT, WHERE, and ORDER BY.
3. Design an ER diagram for an E-Commerce system, and convert ER Diagram to Relational Schema. Translate entities and relationships into MySQL tables.
4. Show One-to-One, One-to-Many, Many-to-Many Relationships.
5. Normalization Steps (1NF, 2NF, 3NF, BCNF) – Demonstrate normalization of a database.
6. DDL Commands: CREATE, ALTER, DROP
7. DML Commands: INSERT, UPDATE, DELETE
8. DQL: Complex Queries Using GROUP BY and Aggregation
9. Joins: INNER JOIN, LEFT JOIN, RIGHT JOIN
10. Create a database to show Subqueries and Nested Queries.
11. Create a sql database to demonstrate Stored Procedures & Functions
12. Transactions & ACID Properties – Implement a banking transaction.
13. Query Optimization Using EXPLAIN.
14. Backup & Recovery Using mysqldump.

Any other Practical/Programs/Lab Assignments given by the teachers.

Academic Session: 2026-27
Sixth Semester

Name of the Program	**Minor Courses (Single Major /Multidisciplinary Program)**	**Program Code**	**-----**
Name of the Course	**Python Programming**	**Course Code**	**26CSC406MV01**
Hours/Week	**2+0+4**	**Credits (L:T:P)**	**2:0:2**
Max. Marks.	**Theory: 50 (35+15)** **Practical: 50**	**Time of Examination**	**3 Hours**

Note: The examiner has to set nine questions in all by setting two questions from each Unit and Question No. 1 consisting of short-answer type questions covering the entire syllabus. Student will be required to attempt five questions in all by selecting one question from each Unit and Question No. 1, which is compulsory. All questions will carry equal marks.

Course Objective:

The course is designed to impart knowledge of one of the latest and most powerful programming languages – Python. Python programming is intended for software engineers, system analysts, program managers and user support personnel who wish to learn the Python programming language.

Course Outcomes:

By the end of the course the students will be able to:

CO1:Develop problem-solving skills and critical thinking in Python using basic programming constructs including variables, operators and data types.
CO2: Will be able to learn the automating repetitive tasks using loop and conditional controlled statements.
CO3: Understand the complex data types including lists, tuples, dictionaries and Function packages.
CO4: Identify and use libraries for algorithmic thinking to implement various data structures.
CO5: Will be able to implement important context of database programming.

Unit-I

Introduction to Python: History and Features of Python Programming.

Basics of Python: Keywords, Variables, Operators, I/O Statements, Indentation, and Comments. Python Basic Data Types, Data Types Declaration, and Implementation.

Unit-II

Flow Control Statement: if statement, if-else statement,nested-if statement, if-else- if- else ladder, While loop, range() Function, For Loop, Nested Loops, Infinite Loop, Break Statement, Continue Statement, Pass Statement.

Unit-III

Python Complex data types: String Data Type, String Manipulation Methods and implementation using Python Programming, List and Dictionary Data Type, Declaration, and Implementation using Various built-in Functions and Libraries

Unit-IV

Python File Operations: Reading Files, Writing Files in Python, Understanding Read Functions: read(), readline(), readlines(), Understanding Write Functions: write() and writelines() Manipulating file pointer using seek Programming, using file operations.

Database Programming: Connecting to a Database, Creating Tables, INSERT, UPDATE, DELETE and READ operations, Transaction Control, Disconnecting from a database, and Exception Handling in Databases.

Suggested Readings:

1. Allen B. Downey: Think Python: How to Think Like a Computer Scientist, 2nd Edition, Green Tea Press.
2. Charles Dierbach: IntroductiontoComputerScienceUsingPython,1stEdition, Wiley India Pvt Ltd.
3. Wesley J Chun: Core Python Applications Programming, 3rd Edition, Pearson Education India
4. Roberto Tamassia, Michael H Goldwasser, Goodrich: Data Structures and Algorithms in Python, 1st Edition, Wiley India Pvt Ltd
5. Reema Thareja: Python Programming using problem solving approach, Oxford University press.
6. Charles R. Severance: Python for Everybody: Exploring Data Using Python3, 1st Edition, Shroff Publishers.
7. Any other book covering the contents of the subject.

Note: Latest and additional good books may be suggested and added from time to time.

List of Practical/Programs/Lab Assignments

1. Find the sum of given numbers using function with default arguments.
2. Demonstrate default constructor or no argument constructor.
3. Generate prime numbers between the given two numbers.
4. Perform arithmetic operations using Inline function.
5. Accept a three digit number and display it in words eg. 123 be printed out as One Two Three.
6. Swap two values using methods of passing arguments in function
7. Prepare a student Record using class and object.
8. Find the area of geometric shapes using function overloading.
9. Illustrate the use of Friend function.
10. Demonstrate parameterized constructor.
11. Demonstrate copy constructor with constructor overloading.
12. Demonstrate destructors and constructor using “this: pointer.
13. To Perform Addition Subtraction MultiplicationDivision
14. To Check Whether a Character is Alphabet or Not
15. To Check Whether a Character is Vowel or Not
16. To Count Zeros, Positive and Negative Numbers
17. To Remove all Non Alphabet Characters from a String
18. To Find Average of Numbers Using Arrays
19. To Find Size of Variables using sizeof Operator
20. To Check Leap Year.

Any other Practical/Programs/Lab Assignments given by the teachers.

Academic Session: 2026-27
Seventh Semester

Name of the Program	**Minor Courses (Single Major /Multidisciplinary Program)**	**Program Code**	
Name of the Course	**MERN Stack Development**	**Course Code**	**27CSC407MV01**
Hours/Week	**2+0+4**	**Credits (L:T:P)**	**2:0:2**
Max. Marks.	**Theory: 50 (35+15)** **Practical: 50**	**Time of end term examination**	**3 Hours**

Note: The examiner has to set nine questions in all by setting two questions from each Unit and Question No. 1 consisting of short-answer type questions covering the entire syllabus. Student will be required to attempt five questions in all by selecting one question from each Unit and Question No. 1, which is compulsory. All questions will carry equal marks.

Course Objectives:

This course introduces students to advanced concepts of MERN stack development. It emphasizes on the foundations of Web Development, JavaScript, frontend development using React.js, backend development with Node.js and Express.js. It aims to provide enough knowledge to the learners that they can design a small MERN application on their own.

Course Outcomes:

By the end of the course the students will be able to:
CO1: Understand the foundations of Web Development.
CO2: Apply JavaScript in the working of small applications.
CO3: Analyze and implement algorithms for frontend development using React.js.
CO4: Explore backend development with Node.js and Express.js.
CO5: Design a small application using the MERN technologies.

Unit – I

Foundations of Web Development: HTML5: Structure, forms, multimedia, and semantic tags CSS3: Selectors, layouts, animations, and media queries.

Advanced JavaScript ES6+: Arrow functions, de-structuring, promises, and async/await. Working with DOM and event listeners.**Version Control with Git:** Setting up a repository, branches, commits, and merging.

Unit – II

React Fundamentals: Basics of Front-End Development with React.js Components and JSX. Going from Data to UI, JSX can be anywhere, dealing with Arrays in the Context of JSX, working with events, listening and reacting to events, listening to Regular DOM Events, event handler, component lifecycle, lifecycle Methods, Hooks and Functional Components: useState, useEffect, and custom hooks.

Unit – III

Basics of Back-End Development with Node.js and Express.js Node.js Basics: Asynchronous programming and Node modules. Creating basic servers and event-driven programming. Express.js: Middleware, routing, and RESTful API design.

Unit – IV

MongoDB Basics: Documents, Collections, Dynamic Schemas, Naming, Databases, Getting and Starting MongoDB, MongoDB Shell, Running the Shell, MongoDB Client, Basic Operations with the Shell, querying, indexing, NoSQL basics, CRUD operations, and schema design.

Full-Stack Integration: Connecting React front-end to Node/Express back-end. Hosting MERN applications:

Managing environment variables and configurations.

Suggested Readings:

1. J. Duckett, HTML & CSS: Design and Build Websites. Indianapolis, IN: Wiley, 2011.
2. M. Haverbeke, Eloquent JavaScript, 3rd ed. San Francisco, CA: No Starch Press, 2018.
3. D. Flanagan, JavaScript: The Definitive Guide, 6th ed. Sebastopol, CA: O'Reilly Media, 2011.
4. S. Chacon, Pro Git, 2nd ed. Berkeley, CA: Apress, 2014.
5. M. Casciaro, Node.js Design Patterns. Birmingham, UK: Packt Publishing, 2016.
6. R. Wieruch, The Road to React. Self-published, 2020.
7. S. Stefanov, React - Up & Running. Sebastopol, CA: O'Reilly Media, 2016.
8. M. Schwarzmüller, React - The Complete Guide. Self-published, 2020.
9. Z. Gordon, React Explained. Self-published, 2019.
10. E. Brown, Web Development with Node and Express, 2nd ed. Sebastopol, CA: O'Reilly Media, 2019.
11. N. Dabit, Full Stack Serverless. Sebastopol, CA: O'Reilly Media, 2020.
12. M. Schwartz, Deploying Web Applications. Sebastopol, CA: O'Reilly Media, 2020.
13. S. Hoque, Full-Stack React Projects, 2nd ed. Birmingham, UK: Packt Publishing, 2020.
14. K. Chodorow, MongoDB: The Definitive Guide, 3rd ed. Sebastopol, CA: O'Reilly Media, 2019.
15. Any other relevant book as per the availability and content suitability

Note: Latest and additional good books may be suggested and added from time to time.

List of Practical/Programs/Lab Assignments

1. Build a React component that displays a greeting message based on user input.
2. Create a counter application with increment, decrement, and reset buttons.
3. Develop a React form for user registration with validation.
4. Build a React app with a navigation bar and multiple pages (Home, About, Contact).
5. Write a simple Node.js program to create a server and display a "Hello, World!" message.
6. Create an Express app with routes for /home, /about, and /contact.
7. Build a RESTful API in Express.js to manage a list of books (CRUD operations).
8. Perform CRUD operations on a MongoDB collection (e.g., student database).
9. Connect a React front-end with an Express.js back-end and display data from MongoDB.
10. Implement a simple login form in React and validate user credentials with Node.js.
11. Develop a weather app using useState and useEffect hooks to fetch weather data from an API.
12. Create a React app with nested routes and dynamic routing (e.g., product details based on ID).
13. Develop custom middleware in Express.js to log request details (e.g., method, URL, timestamp).
14. Perform aggregation operations in MongoDB to calculate total sales by category.
15. Build a Node.js app to handle file uploads and store them in MongoDB.

Any other Practical/Programs/Lab Assignments given by the teachers.

Academic Session: 2026-27
Eighth Semester

Name of the Program	**Minor Courses (Single Major /Multidisciplinary Program)**	**Program Code**	**-----**
Name of the Course	**Data Analytics using R**	**Course Code**	**27CSC408MV01**
Hours/Week	**2+0+4**	**Credits (L:T:P)**	**2:0:2**
Max. Marks.	**Theory: 50 (35+15)** **Practical: 50**	**Time of Examination**	**3 Hours**

Note: The examiner has to set nine questions in all by setting two questions from each Unit and Question No. 1 consisting of short-answer type questions covering the entire syllabus. Student will be required to attempt five questions in all by selecting one question from each Unit and Question No. 1, which is compulsory. All questions will carry equal marks.

Course Objective:

The objective of this course is to make the learners understand big data fundamentals and leverage big data processing technologies along with core Hadoop ecosystem and its tools for data transformation and analysis. It will enable them to utilize R for big data manipulation, mastering the R interpreter as well as understand data structures, control flow, functions and explore data manipulation packages available in R.

Course Outcomes:

By the end of the course the students will be able to:

CO1: Understand and design the data analytics and life cycle to be used for any real world problem.
CO2: Develop insights into the data and present results through visualization techniques for different types of selected problem statements.
CO3: Demonstrate the use of Hadoop and its ecosystem elements to analyze big data.
CO4: Demonstrate use of advanced FOSS computing environments for big health care data.
CO5: Exhibit skills in Hadoop, Yarn, Spark and R programming required for data analysis.

Unit-I

Basics of Big data: characteristics, types, sources, architectures, Data analysis process, Data analytics lifecycle, Pre-processing data, Market and Business Drivers for Big Data Analytics, Business Problems Suited to Big Data Analytics.

Unit-II

Technologies for Big Data Analytics: Distributed and Parallel Computing for Big Data, Cloud Computing and Big Data, In-Memory Computing Technology for Big Data, Introduction to Hadoop, HDFS, Map Reduce, YARN, HBase, Combining HDFS and HBase.

Unit-III

Hadoop Ecosystem: Sqoop, Impala, Apache Flume, Pig, Hive, Data transformation and analysis using Pig, Data analysis using Hive and Impala, Mahout, Oozie, Zookeeper.

Unit-IV

R Programming: R interpreter, Introduction to major R data structures like vectors, matrices, arrays, list and data frames, Control Structures.

Installing, Loading and using Packages: Read/write data from/in files, extracting data from web-sites, Clean data, Transform data by sorting, adding/removing new/existing columns, centering, scaling and normalizing the data values, converting types of values, using string in-built functions.

Suggested Readings:

1. DT Editorial Services: Big Data, Black Book: Covers Hadoop 2, MapReduce, Hive, YARN, Pig, R and Data Visualization.
2. David Dietrich, Barry Hiller: Data Science and Big Data Analytics, EMC education services, Wiley Publications.
3. Mohammed Guller: Big Data Analytics with Spark: A Practitioner's Guide to Using Spark for Large Scale Data Analysis.
4. David Loshin: Big Data Analytics From Strategic Planning to Enterprise Integration with Tools, Techniques, NoSQL, and Graph, Morgan Kaufmann.
5. VenkatAnkam, "Big Data Analytics", Packt Publishing.
6. Jenny Kim, Benjamin Bengfort: Data Analytics with Hadoop, O'Reilly Media, Inc.
7. Glenn J. Myatt: Making Sense of Data: A Practical Guide to Exploratory Data Analysis and Data Mining.
8. Any other book(s) covering the contents of the paper in more depth**.**

Note: Latest and additional good books may be suggested and added from time to time.

List of Practical/Programs/Lab Assignments

1. WAP to implement different arithmetic, logical and relational operations on scalar, complex and vectors.
2. WAP a program to print even number between 1-50 using if else statement.
3. WAP to print all alphabets from 5 to 15 using switch statement.
4. WAP to create Fibonacci series using while loop.
5. WAP to generate mathematical table entered by the user using do while loop.
6. Create a vector by specifying first element, last element with a step size of 4 and then implement the following operations on it:
   a) Print 2nd, 5th and 7th element, provided length of vector is 50
   b) Sort the given vector in increasing and decreasing order
   c) Replace the value of third element with 20
   d) If the value of any element is less than 4 then replace it by 8
7. Create a vector using inbuilt seq() function than print whether number is even or odd
8. Create a list of all the programs offered by your Department and running in your current semester and then implement the following operations on it:
   a) Access the value of an element present at 5th position
   b) Change the value present at 2nd position to "Ph.D"
   c) Print length of the list
   d) Check whether given element "MCA" present in the list or not
   e) Add new element into the list not at the end but at a given position
   f) Print all the elements of the list using for loop
9. Create a 3D array with dimensions 3x4x2 and fill it with numbers from 1 to 24 and implement the following operations on it:
   a) Display all the elements from the first row from first dimension
   b) Display all elements from second column
   c) Check whether 8 is present or not
   d) Print the size of an array
   e) Print all elements of an array using for loop
10. Create a matrix with 2 rows (subjects name and priority)and implement following operations on it:
    a) Print subject name with second priority
    b) Print all the elements of row 2
    c) Add one more subject and its priority
    d) Remove an element from second row first column
    e) Print the length of your matrix

f) Print all elements of matrix using for loop

11. Read an inbuilt CSV file in R studio and apply different operations on it:
    a) summary () b) head()
    c) tail() e) na()
    f) rm() g) isnull()
12. Read an inbuilt CSV file in R studio and apply different data cleaning operations on it:
    a) filling missing values
    b) create new column in dataset
    c) sort() d) arrange()
    e) select() f) filter()
    g) spread() h) mutate()
13. Download a CSV file from internet and read it in R studio. Explore the data using different data exploration functions and visualize it using different plots (Bar Plot, Box Plot and Scatter Plot).
14. Create your own data frame using different students records and apply different mathematical and statistical operations on it.
15. Download a file from internet and read it in R studio. Perform different operations on it:
    a) Data cleaning
    b) Data normalization
    c) Data standardization

Any other Practical/Programs/Lab Assignments given by the teachers.

Name of the Program	**Minor Courses (Single Major /Multidisciplinary Program)**	**Program Code**	**-----**
Name of the Course	**Machine Learning and Generative AI**	**Course Code**	**27CSC408MV01**
Hours/Week	**3+0+2**	**Credits (L:T:P)**	**3:0:1**
Max. Marks.	**Theory: 75 (50+25)** **Practical: 25**	**Time of end term examination**	**3 Hours**

Note: The examiner has to set nine questions in all by setting two questions from each Unit and Question No. 1 consisting of short-answer type questions covering the entire syllabus. Student will be required to attempt five questions in all by selecting one question from each Unit and Question No. 1, which is compulsory. All questions will carry equal marks.

Course Objectives:

Introduce core concepts of Machine Learning (ML) and generative AI, including algorithms, models, and ethical implications. Develop proficiency in designing, training, and evaluating ML models using Python and frameworks like TensorFlow/PyTorch. Equip students with the skills to implement generative AI systems (e.g., GANs, transformers) for real-world applications. Foster critical thinking about the societal impacts and limitations of AI technologies.

Course Outcomes:

By the end of the course the students will be able to:

CO1: Explain fundamental ML concepts, workflows, and mathematical foundations.
CO2: Implement and evaluate supervised/unsupervised learning models.
CO3: Design and train neural networks for classification, regression, and generative tasks.
CO4: Build generative AI systems (e.g., text, image synthesis) using modern frameworks.
CO5: Analyze ethical challenges (bias, privacy) in ML and generative AI.

Unit – I

Introduction to Machine Learning: Types of learning: Supervised, unsupervised, reinforcement. Applications

in healthcare, finance, robotics etc.

Math Essentials: Linear algebra (vectors, matrices), calculus (gradients), probability (Bayes' theorem). **Python for ML:** NumPy, pandas, Matplotlib.**Data preprocessing:** Normalization, feature engineering.

Unit – II

Supervised Learning: Regression: Linear Regression, Polynomial Regression. Classification: Decision Trees, Naïve Bayes, Support Vector Machines (SVM). **Ensemble Methods:** Random Forest, Gradient Boosting, XGBoost. Model Evaluation & Hyperparameter Tuning.

Unsupervised Learning: Clustering: K-Means, Hierarchical Clustering, DBSCAN. Dimensionality Reduction: PCA, t-SNE, LDA.**Association Rule Learning:** Apriori, FP-Growth

Unit – III

Neural Networks and Deep Learning: Basics of Neural Networks & Backpropagation, Activation Functions: Sigmoid, ReLU, Softmax, Feedforward Neural Networks (FNN), Convolutional Neural Networks (CNNs) – Image Classification, Recurrent Neural Networks (RNNs), LSTMs – Sequence Prediction.

Optimization Techniques: Gradient Descent, Adam, RMSprop. Frameworks: TensorFlow/Keras or PyTorch basics.

Unit – IV

Generative AI: Generative Adversarial Networks (GANs): Architecture (generator vs. discriminator), training challenges, DCGANs. **Variational Autoencoders (VAEs):** Latent space, reparameterization trick.

Transformers & LLMs: Attention mechanisms, BERT, GPT architectures. Fine-tuning pre-trained models. Applications:Text generation (ChatGPT etc), image synthesis (DALL-E), music generation. Ethics & Governance: Deepfakes, misinformation, AI regulation (GDPR, AI Act).

Suggested Readings:

1. C.M. Bishop: Pattern Recognition and Machine learning, Springer.
2. Hastie, Tibshirani, Friedman: Introduction to statistical machine learning with applications in R, Springer.
3. Tom Mitchell: Machine Learning, McGraw Hill.
4. Parag Kulkarni: Reinforcement and Systemic Machine learning for Decision Making, Wiley-IEEEPress.
5. Any other book(s) covering the contents of the paper in more depth.

Note: Latest and additional good books may be suggested and added from time to time.

List of Practical/Programs/Lab Assignments

1. Write a program to implement basic matrix operations (addition, subtraction, multiplication) using NumPy and also to calculate the determinant, inverse, and rank of a matrix using NumPy.
2. Write a program to compute the gradient of a simple function using NumPy.
3. Write a program to implement Bayes' theorem for a simple probability problem.
4. Write a program to visualize a dataset using Matplotlib (scatter plot, bar chart, histogram).
5. Write a program to load a dataset using pandas and display basic statistics (mean, median, mode, standard deviation).
6. Write a program to perform missing value imputation using mean, median, and mode in pandas.
7. Write a program to perform one-hot encoding and label encoding on categorical data.
8. Write a program to apply feature scaling (Normalization & Standardization) to a dataset.
9. Write a program to perform feature selection using correlation matrix and remove highly correlated features.
10. Write a program to implement Linear Regression from scratch using NumPy and Polynomial Regression using scikit-learn.
11. Write a program to implement Decision Tree classification on a dataset using scikit-learn.
12. Write a program to implement Naïve Bayes classification for spam detection.
13. Write a program to implement Support Vector Machine (SVM) classification on an image dataset.
14. Write a program to perform K-Means clustering and visualize clusters on a 2D dataset.
15. Write a program to implement Hierarchical Clustering and generate a dendrogram.

16. Write a program to perform DBSCAN clustering and visualize noise points.
17. Write a program to implement a simple Perceptron from scratch using NumPy.
18. Write a program to implement Backpropagation in a simple neural network using NumPy.
19. Write a program to implement an Artificial Neural Network (ANN) using TensorFlow/Keras.
20. Write a program to implement Convolutional Neural Network (CNN) for image classification using TensorFlow.
21. Write a program to build a Recurrent Neural Network (RNN) for text generation using TensorFlow.
22. Write a program to implement LSTM .
23. Write a program to implement a simple Generative Adversarial Network (GAN) using TensorFlow.

Any other Practical/Programs/Lab Assignments given by the teachers.