Syllabi and Scheme of Examinations
for
4-Year B.Sc. (Computer Science) – Single Major Program

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

With Effect from the session 2024-25

MAHARSHI DAYANAND UNIVERSITY
ROHTAK (HARYANA)

Credit Structure for Undergraduate Programmes (Single Major)

Semester	Discipline-Specific Courses (DSC) / Major Course	Minor(MIC)/ Vocational (VOC)/ Skill Enhancement Courses (SEC)/ Internship	Multidisciplinary courses (MDC)	Ability Enhancement courses (AEC)	Dissertation	Value-Added Courses (VAC)	Total Credits
I	DSC - A1 @ 4 credits DSC – A2 @ 4 credits	MIC1 @ 4 credits SEC1@ 3 credits	MDC1 @ 3 credits	AEC1 @ 2 credits	----------	VAC1 @ 2 credits	22
II	DSC – A3 @ 4 credits DSC – A4 @ 4 credits	MIC2 @ 4 credits SEC2@ 3 credits	MDC2 @ 3 credits	AEC2 @ 2 credits	----------	VAC2 @ 2 credits	22
Students exiting the programme after second semester and securing 48 credits including 4 credits of summer internship will be awarded UG Certificate in the relevant Discipline/Subject							
III	DSC – A5 @ 4 credits DSC – A6 @ 4 credits	MIC3 @ 4 credits SEC3@ 3 credits	MDC3 @ 3 credits	AEC3 @ 2 credits		VAC3 @ 2 credits	22
IV	DSC – A7 @ 4 credits DSC – A8 @ 4 credits DSC – A9 @ 4 credits DSC – A10 @ 4 credits	MIC4(VOC)@ 4 credits	----------	AEC4 @ 2 credits	----------	VAC4 @ 2 credits	24
Students exiting the programme after fourth semester and securing 94 credits including 4 credits of summer internship will be awarded UG Diploma in the relevant Discipline/Subject							
V	DSC – A11 @ 4 credits DSC – A12 @ 4 credits DSC – A13 @ 4 credits DSC – A14 @ 4 credits	MIC5(VOC)@ 4 credits Internship @ 4 credits#	----------	----------	----------	----------	24
VI	DSC – A15 @ 4 credits DSC – A16 @ 4 credits DSC – A17 @ 4 credits DSC – A18 @ 4 credits	MIC6(VOC)@ 4 credits SEC3@ 2 credits	----------	----------	----------	----------	22
Students will be awarded 3-year UG Degree in relevant major Discipline/Subject upon securing 136 credits.							
VII	DSC – H1 @ 4 credits DSC – H2 @ 4 credits DSC – H3 @ 4 credits DSC – H4 @ 4 credits DSC – H5 @ 4 credits	SEC4 @ 4 credits OR MIC7 (VOC) @ 4 credits OR Internship @ 4 credits	----------	----------	----------	----------	24
VIII (4yr UG Hon.)	DSC – H6 @ 4 credits DSC – H7 @ 4 credits DSC – H8 @ 4 credits DSC – H9 @ 4 credits DSC – H10 @ 4 credits	SEC5 @ 4 credits OR MIC8 (VOC) @ 4 credits OR Internship @ 4 credits	----------	----------	----------	----------	24
VIII (4yr UG Hon. with Research)	DSC – H6@ 4 credits DSC – H7@ 4 credits	SEC5 @ 4 credits OR MIC8 (VOC) @ 4 credits OR Internship @ 4 credits	----------	----------	Research project/ Dissertation @ 12 credits	---------- TOTAL CREDITS	24 184

Note: Four credits of internship earned by a student during summer internship after 2nd semester or 4th semester will be counted in 5th semester of a student who pursue 3 year UG Programmes without taking exit option.

Discipline Specific Courses/ Major Course			**Credits Distribution**			**Total Credits**	**Workload**			**Total Workload**	**Marks**			
	Nomenclature of Course	**Course Code**	**L**	**T**	**P**		**L**	**T**	**P**		**Theory**		**Practical**	**Total Marks**
											Internal	**External**		
Semester I (Session 2024-25)														
DSC - A1 @ 4 credits	Mathematical Foundations of Computing	24CSCS401DS01	4	0	0	4	4	0	0	4	30	70	0	100
DSC – A2 @ 4 credits	Computing Fundamentals and C Programming	24CSCS401DS02	3	0	1	4	3	0	2	5	25	50	25	100
Semester II (Session 2024-25)														
DSC – A3 @ 4 credits	Object Oriented Programming using C++	24CSCS402DS01	3	0	1	4	3	0	2	5	25	50	25	100
DSC – A4 @ 4 credits	Operating System Concepts	24CSCS402DS02	3	0	1	4	3	0	2	5	25	50	25	100
Semester III (Session 2025-26)														
DSC – A5 @ 4 credits	Data Structures and Algorithms	25CSCS403DS01	3	0	1	4	3	0	2	5	25	50	25	100
DSC – A6 @ 4 credits	Digital Logic Design	25CSCS403DS02	3	0	1	4	3	0	2	5	25	50	25	100
Semester IV (Session 2025-26)														
DSC – A7 @ 4 credits	Computer Networks	25CSCS404DS01	4	0	0	4	4	0	0	4	30	70	0	100
DSC – A8 @ 4 credits	Elements of Software Engineering	25CSCS404DS02	4	0	0	4	4	0	0	4	30	70	0	100
DSC – A9 @ 4 credits	DBMS and MySQL	25CSCS404DS03	3	0	1	4	3	0	2	5	25	50	25	100
DSC – A10 @ 4 credits	Computer Graphics and Animation	25CSCS404DS04	3	0	1	4	3	0	2	5	25	50	25	100
Semester V (Session 2026-27)														

DSC – A11 @ 4 credits	Artificial Intelligence	26CSCS405DS01	3	0	1	4	3	0	2	5	25	50	25	100
DSC – A12 @ 4 credits	Computational Theory	26CSCS405DS02	4	0	0	4	4	0	0	4	30	70	0	100
DSC – A13 @ 4 credits	Mobile Application Development	26CSCS405DS03	3	0	1	4	3	0	2	5	25	50	25	100
DSC – A14 @ 4 credits	Computer System Architecture	26CSCS405DS04	4	0	0	4	4	0	0	4	30	70	0	100
Semester VI (Session 2026-27)														
DSC – A15 @ 4 credits	Compiler Design	26CSCS406DS01	4	0	0	4	4	0	0	4	30	70	0	100
DSC – A16 @ 4 credits	Security in Computing	26CSCS406DS02	3	0	1	4	3	0	2	5	25	50	25	100
DSC – A17 @ 4 credits	Advance Database Systems	26CSCS406DS03	3	0	1	4	3	0	2	5	25	50	25	100
DSC – A18 @ 4 credits	Wireless Sensor Networks	26CSCS406DS04	3	0	1	4	3	0	2	5	25	50	25	100

L: Lecture; T: Tutorial; P: Practical

Note:

The Syllabi and Scheme of Examinations (SOE) for Discipline Specific Courses/Major Courses for UG Semester 7 and Semester 8 will be same as applicable for Syllabi and S.O.E. for Post Graduate semester 1 and semester 2 respectively.

Syllabi for 4-year Bachelor of Science (Computer Science) Program w.e.f. the session 2024-25

Academic Session: 2024-25
First Semester

Name of the Program	**B.Sc. Computer Science (Single Major)**	**Program Code**	**-----**
Name of the Course	**Mathematical Foundations of Computing**	**Course Code**	**24CSCS401DS01**
Hours/Week	**4 (4+0+0)**	**Credits(L:T:P)**	**4:0:0**
Max. Marks.	**Theory:100(70+30)**	**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. Students will be required to attempt five questions in all by selecting one question from each Unit and Question No.1, which is compulsory.

Course Objectives:
The objective of this course is to inculcate in students the fundamental mathematic al background in Computer Science and to provide reasoning, learning and understanding skills. The students get exposure of various concepts of mathematics such as sets, relations, functions, trigonometry, limit, continuity, derivatives, matrix and determinants.

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

CO1: Understand and solve the problems onset, relation and functions.
CO2: Understand the concepts of propositional logic, predicate logic and trigonometry.
CO3: Solve the problems on limit and continuity.
CO4: Understand the concepts and solutions of derivatives.
CO5: Understand the concept of Matrix and Determinants.

Unit–I

Sets: Sets, Subsets, Equal Sets Universal Sets, Finite and Infinite Sets, Operation on Sets, Union, Intersection and Complements of Sets, Cartesian Product, Cardinality of Set, Practical applications of set theory.

Relations And Functions: Properties of Relations, Equivalence Relation, Partial Order Relation. Function: Domain and Range, Onto, Into and One to One Functions, Composite and Inverse Functions.

Unit–II

P**ropositional Logic**: Proposition logic, basic logic, Logical Connectives, truth tables, tautologies, contradiction, Logical implication, Logical equivalence. Inference rules.
Predicate Calculus: Predicates and quantifiers.

Trigonometry: Introduction, measurement of angles, trigonometric functions, relation between trigonometric functions, signs of trigonometric functions, and trigonometric functions of standard angles. Basics of inverse trigonometry.

Unit–III

Limits and Continuity: Limit at a point, properties of limits, computation of limits of various types of functions, and continuity of a function at a point.

Differentiation: Derivative of a function, Derivatives of sum, differences, product & quotient of functions, Derivatives of polynomial, trigonometric, exponential, logarithmic, inverse trigonometric and implicit functions, Logarithmic Differentiation, Chain rule and differentiation by substitution.

Unit–IV

Matrices: Definition, types of matrices, addition, subtraction, scalar multiplication, and multiplication of matrices.

Determinants: Definition, minors, cofactors, properties of determinants, applications of determinants in finding the area of a triangle, adjoint of a matrix, inverse of a matrix, and solving a system of linear equations using the matrix method.

Suggested Readings:

1. C.L. Liu: Elements of Discrete Mathematics, McGraw Hill.
2. Lipschutz, Seymour: Discrete Mathematics, Schaum's Series.
3. Babu Ram: Discrete Mathematics, Vinayek Publishers, New Delhi.
4. Trembley, J.P. & R. Manohar: Discrete Mathematical Structure with Applications to Computer Science, TMH.
5. Kenneth H. Rosen: Discrete Mathematics and its Applications, TMH.
6. Doerr, Alan &Levasseur, Kenneth: Applied Discrete Structures for Computer Science, Galgotia Pub. Pvt. Ltd.
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.

Name of the Program	**B.Sc. Computer Science (Single Major)**	**Program Code**	**-----**
Name of the Course	**Computing Fundamentals and C Programming**	**Course Code**	**24CSCS401DS02**
Hours/Week	**5 (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. Students will be required to attempt five questions in all by selecting one question from each Unit and Question No.1, which is compulsory.

Course Objectives:
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 gain 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 fundamentals principles and history, Generations of Computers, Major components of Computer, 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, Types of Network, Network Topologies, Internet and its applications; Operating system and its functions.

Unit– II

Problem Solving: 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 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 and Branching: Decision making with IF statement, IF-ELSE statement, Nested IF statement, ELSE-IF ladder, switch statement, go to statement.

Decision Making and 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 and 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; Advanced control structures: switch, break, and continue statements.

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. YashwantKanetker: Let us C, BPB.
7. Rajaraman, V.: Computer Programming in C, PHI.
8. YashwantKanetker: 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 Programs/Programs/Lab Assignments

1. WAP to convert number from Decimal to Binary.
2. WAP to check whether number is Palindrome or not.
3. WAP to swap two numbers without using third variable.
4. WAP to check whether a number if Armstrong or not.
5. WAP to find factorial of a number.
6. WAP to check whether number is Perfect Square or not.
7. WAP to find SUM and AVERAGE of two numbers.
8. WAP to convert temperature from Fahrenheit to Celsius and Celsius to Fahrenheit.
9. WAP to read and print an employee's detail using structure.
10. WAP to check whether a number if Even or Odd.
11. WAP to print all leap years from 1 to N.
12. WAP to print tables of numbers from 1 to 20.
13. WAP to print star/pyramid series.
14. WAP to convert temperature from Celsius to Fahrenheit and vice versa.
15. WAP to convert number from Binary to Decimal.
16. WAP to print ASCII Table.
17. WAP to check whether a given number is palindrome or not using Bitwise Operator
18. WAP to count number of bits set to 1 in an Integer
19. WAP to check if all the bits of a given integer is one (1)
20. WAP to find the Highest Bit Set for any given Integer
21. WAP to Count the Number of Trailing Zeroes in an Integer
22. WAP to find the Biggest Number in an Array of Numbers using Recursion
23. WAP to accept Sorted Array and do Search using Binary Search
24. WAP to Cyclically Permute the Elements of an Array
25. WAP to print the sum and product of digits of an integer.
26. WAP to reverse a number.
27. WAP to compute the sum of the first n terms of the following series

S = 1+1/2+1/3+1/4+……

28. WAP to compute the sum of the first n terms of the following series

S =1-2+3-4+5…….

29. 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.
30. Write a function to find whether a given no. is prime or not. Use the same to generate the prime numbers less than 100.
31. WAP to compute the factors of a given number.
32. Write a macro that swaps two numbers. WAP to use it.
33. WAP to print a triangle of stars as follows (take number of lines from user):

```
*
***
*****
*******
*********
```

34. 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.

35. WAP that prints a table indicating the number of occurrences of each alphabet in the text entered as command line arguments.
36. Write a program that swaps two numbers using pointers.
37. Write a program in which a function is passed address of two variables and then alter its contents.
38. 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.
39. Given two ordered arrays of integers, write a program to merge the two-arrays to get an ordered array.
40. WAP to display Fibonacci series (i)using recursion, (ii) using iteration
41. WAP to calculate Factorial of a number (i)using recursion, (ii) using iteration

42. WAP to calculate GCD of two numbers (i) with recursion (ii) without recursion.
43. WAP to design calculator with basic operations using switch.
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

Any other Programs/Lab Assignments/Programsgiven by the teachers.

Academic Session: 2024-25
Second Semester

Name of the Program	**B.Sc. Computer Science (Single Major)**	**Program Code**	**-----**
Name of the Course	**Object Oriented Programming using C++**	**Course Code**	**24CSCS402DS01**
Hours/Week	**5 (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. Students will be required to attempt five questions in all by selecting one question from each Unit and Question No.1, which is compulsory.

Course Objectives:
This course provides in-depth coverage of object oriented programming principles and techniques using C++. This also provides the knowledge how to write class components that protect data integrity and produce classes that are re-usable and maintainable and to design class responsibilities and relationships with the aim of maximizing class cohesion and minimizing class dependencies and use the generic programming features of C++.

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

CO1: Understand concept of object oriented programming and its features.
CO2: Elaborate the C++ features and access specifiers.
CO3: Able to understand importance of polymorphism and inheritance.
CO4: Learn use Streams and Files.
CO5: Explore the applications of Generic programming with templates.

Unit – I

Introduction to OOP: Basic concepts of Object-Oriented Programming (OOP). Comparison with procedural programming, Characteristics of object-oriented programming, Advantages, disadvantages and applications of OOP.

Basics of C++: Data Types, Variables, Operators, Expressions, Structure of a C++ program. Creating the source files, compiling and linking programs. Creating class and objects, Arrays, String, Structure, Recursion, Control Statements.

Unit – II

Functions in C++: Passing arguments to and returning values from functions, Inline functions, Default arguments, Function overloading.

Arrays: Arrays within a class, Arrays of objects, Pointers to members, passing objects to functions, friend function.

Access Specifiers: Private, Public and Protected, Member functions of the class, Constructor and Destructor, types of Constructors and their advantages.

Unit – III

Concept of Inheritance: Reusability, types of inheritance, Derived class and their constructs, Class hierarchies, Public and private inheritance levels.

Polymorphism: Static class members, Static member function, Friend Function, Operator Overloading, Abstract class, virtual functions.

Memory management: new, delete, object Creation at Runtime. Exceptional handling.

Unit – IV
Input/ Output: Streams, Stream classes, Unformatted Input/ Output operations, Formatted console Input/ Output operations, Managing output with manipulators. **File-Operations:** Classes for file stream operations, Opening and Closing a file. File pointers and their manipulations, Random access. Error handling during file operations, Command-line arguments.
Suggested Readings: 1. K.R.Venugopal: Mastering C++ 2. BjarneStroustrup: The C++ Programming Language. 3. Balaguruswami: Object Oriented with C++, TMH. 4. K.R VenugopalRajkumar: Mastering C++, TMH. 5. Maria litvin, Gary litvin: Programming in C++, VPH. 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.

Name of the Program	**B.Sc. Computer Science (Single Major)**	**Program Code**	**-----**
Name of the Course	**Operating System Concepts**	**Course Code**	**24CSCS402DS02**
Hours/Week	**5 (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. Students will be required to attempt five questions in all by selecting one question from each Unit and Question No.1, which is compulsory.			
Course Objectives: This course, Operating Systems, is a fundamental component of any computer science education. The objective of this course is to learn the processes of operating systems such as process management, process synchronization, memory management, file system implementation, storage structures utilized in OS, and protection principles along with how efficiently the OS uses CPU resources with the aid of various techniques.			
Course Outcomes: By the end of the course the students will be able to: CO1: Regulate who has access to a computer and what files may be shared. CO2:Demonstrate an understanding of computer components and their functions in computing. CO3:Recognize and fix user problems in regular operating conditions. CO4: Understand the concept of Controlled access to systems through authentication. CO5:Learn how programming languages, operating systems, and architectures work and how to utilize them effectively.			
Unit – I			
Introduction: Introduction to Operating Systems operations and services, process management, memory management, storage management, protection and security, distributed systems. **Structures And Functions:** Operating system Functions, Characteristics, Structures - Simple Batch, Multi programmed, time shared, Personal Computer, Parallel, Distributed Systems, Real-Time Systems, System Calls, Virtual Machines.			

Unit – II

Process Management and Scheduling: Process concept, process scheduling, operations on processes, threads, inter-process communication, precedence graphs, critical section problem, semaphores, classical problems of synchronization, critical regions, monitors.

Deadlocks: Deadlock problem, Methods for Deadlock Handling, Deadlock prevention, avoidance and recovery.

Unit – III

Memory Management: Concepts of Memory Management, Logical and Physical address space, swapping, contiguous and non-contiguous memory allocation, paging, segmentation, and paging with segmentation.

Virtual Memory: Introduction to Virtual Memory, demand paging, concepts of page replacement and frames allocation, page replacement algorithms, thrashing, demand segmentation.

Unit – IV

File System Interface: Definition, Access methods, Directory Structure, Protection, File System Structure, Allocation methods, Free-space Management, Directory Management, Directory Implementation, Efficiency and Performance.

I/O Management: I/O software and its types, Disk Scheduling.

Shell Programming: Concept of shell, Types of shell, the command line, Using Editors for shell programming (e.g. vi), basics of Shell programming – shell building blocks. Writing Linux/Unix shell scripts.

Suggested Readings:

1. Silberschatz& Galvin: Operating System Concept, Wiley.
2. Milan Milenkovic: Operating Systems, Tata McGraw – Hill.
3. William Stallings: Operating Systems, PHI.
4. Yashawant Kanetkar: Unix Shell Programming, BPB.
5. H.M. Deitel -Operating Systems.
6. A.S. Tanenbaum: Modern Operating Systems, Pearson/PHI.
7. Dhamdhere: Operating Systems, Tata McGraw Hill.
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 Practicals /Programs/Lab Assignments

1. Shell script program to create and print the value of variables
2. Shell script program to add two numbers
3. Shell script program to swap two numbers
4. Shell script program to read two integer numbers and print the subtraction of both variables
5. Shell script program to multiply two numbers
6. Shell script program to print program name using command line argument
7. Shell script program to add two numbers using command line arguments
8. Shell script program to execute 'ls' command
9. Shell script program to demonstrate the '$#' variable
10. Shell script program to print the current process id

Any other Programs/Lab Assignments/Programs assigned by the teachers.

Academic Session: 2025-26
Third Semester

Name of the Program	**B.Sc. Computer Science (Single Major)**	**Program Code**	
Name of the Course	**Data Structures & Algorithms**	**Course Code**	**25CSCS403DS01**
Hours/Week	**5 (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.

Course Objectives:

The course aims at developing the ability to use basic data structures like arrays, stacks, queues, lists, trees to solve problems. C++ is chosen as the language to understand implementation of these data structures.

Course Outcomes:

By the end of the course the students will be able to:
CO1: Understand the basic concepts of data structures.
CO2: Understand the concepts of Linear and Non-linear data structures.
CO3: To know about implementation of various data structures.
CO4: Identify the appropriate data structure(s) for a given application and understand the trade-offs involved in terms of time and space complexity.
CO5: Search and sort the lists in an efficient way.

Unit – I

Data structure: Elementary data organization, Data Structure definition, Data type vs. data structure, Categories of data structures, Data structure operations, Applications of data structures.

Arrays: Introduction, Linear arrays, Representation of linear array in memory, address calculations, Traversal, Insertions, Deletion in an array, Multidimensional arrays, Parallel arrays, Sparse arrays.

Unit – II

Searching: Introduction, Sequential search, Binary search, Prerequisite for binary search, Comparison in terms of efficiency.
Sorting: Bubble sort, Selection sort, Insertion sort, Quick sort, Merge sort, Comparison in terms of their efficiency.
Stack: Introduction, Array and linked representation of stacks, Operations on stacks, Applications of stacks: Polish notation, Recursion.

Unit – III

Queues: Introduction, Array and linked representation of queues, Operations on queues, Deques, Priority Queues, Applications of queues.

Linked List: Introduction, Representation of linked lists in memory, Traversal, Insertion, Deletion, Searching in a linked list, Header linked list, Circular linked list, Two-way linked list, Threaded lists, Garbage collection, Applications of linked lists.

Unit – IV

Trees: Introduction, binary tree, binary tree representation in memory, tree traversal algorithms using stack, binary search trees and various operations.
Graphs: Introduction, various representation of graphs, graph traversal algorithms, spanning tree algorithms, applications of graph.

Suggested Readings:

1. Lipschutz, Seymour, and Marc Lars Lipson: "SCHAUM'S." Outline of Theory and Problems of Data Structures," McGraw: Hill Book Company.
2. Goodrich, M.T, Tamassia, R., & Mount, D.: Data Structures and Algorithms Analysis in C++, 2nd edition. Wiley.
3. Drozdek, A.: Data Structures and Algorithms in C++, 4th edition, Cengage Learning.
4. Langsam Y.: Augenstein, M. J., &Tanenbaum, A. M. Data Structures Using C and C++, Pearson.
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 Practicals/Programs/Lab Assignments

1. Perform matrix addition and multiplication.
2. Print various patterns.
3. Implement Factorial of a number using function.
4. Implement N^{th} Fibonacci number using function.
5. Implement Power function (X^Y) using function.
6. Implement singly linked lists.
7. Implement doubly linked lists.
8. Implement stack data structure and its operations using arrays.
9. Implement stack data structure and its operations using linked lists.
10. Convert Prefix expression to Infix and Postfix expressions, and evaluate.
11. Implement queue data structure and its operations using arrays.
12. Implement various searching operations (linear search and binary search)
13. Implement different sorting algorithms

Any other Programs/Lab Assignments/Programs assigned by the teachers.

Name of the Program	**B.Sc. (Computer Science) Single Major**	**Program Code**	**----------**
Name of the Course	**Digital Logic Design**	**Course Code**	**25CSCS403DS02**
Hours/Week	**5 (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.

Course Objectives:

The main objective is to introduce the fundamentals of Digital Logic Design in a simple and accessible manner to all undergraduate students, regardless of their specialization. It aims to provide a strong foundation for pursuing advanced technical programs, professional careers, and industry certifications. This course also prepares students to perform the analysis and design of various digital electronic circuits, design and analyze sequential and combinational logic circuits.

Course Outcomes:

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

CO1: Understand the fundamental concepts of Digital logic design, including Number systems, Boolean algebra, and Logic gates.

CO2: Analyze and design Combinational logic circuits such as Adders, Multiplexers and Decoders.
CO3: Understand and design Sequential circuits, including Flip-Flops, Counters, and Shift registers.
CO4: Gain knowledge about memory devices and their applications.
CO5: Develop problem-solving skills by implementing digital logic design concepts in practical applications and real-world scenarios.

Unit – I

Digital Systems: Digital Signals, Digital Waveforms, Digital Computers and Digital Integrated Circuits.

Number Systems: Binary Number Systems, Octal and Hexadecimal Number System. Number Base Conversions. Complements, Signed Binary Numbers and Binary Codes.

Decimal Codes: BCD, Excess-3, Gray Code, Binary Arithmetic, Error Detection and Correction codes.

Unit-II

Gates: AND, OR, NOT, NAND, NOR, Exclusive-OR operations, Exclusive-NOR operations, Boolean algebra, Basic Properties and theorems of Boolean algebra and functions.

Simplification of Boolean Functions: Standard representation for logic functions, K-map representation, and simplification of logic functions using K-map, minimization of logical functions, Don't care conditions.

Unit – III

Combinational Digital Circuits: Adders, Subtractors, Multiplexer, De-Multiplexer, Encoders, Decoders, BCD arithmetic, Carry look ahead adder, Parity checker/Generator, Code converters, Priority encoders.

Unit – IV

Sequential circuits and systems: A1-bit memory, Circuit properties of Bistable Latch, Clocked SR flip flop, J-K , T and D types flip flops, Applications of flip flops, Shift registers, Applications of shift registers, Counters design using flip flops: Asynchronous counters, Synchronous counters.

Suggested Readings:

1. M. Mano: Computer System Architecture, Pearson.
2. Gill, Nasib Singh and Dixit J.B.: Digital Design and Computer Organisation, University Science Press (Laxmi Publications), New Delhi.
3. Mano, M.M.: Digital Design, Prentice-Hall of India.
4. Carl Hamacher et al.: Computer Organization and Embedded Systems, McGraw-Hill.
5. R P Jain: Modern Digital Electronics, Tata McGraw Hill.
6. William Stallings: Computer Organization and Architecture- designing for Performance.
7. Anrew S. Tanenbaum: Structured Computer Organization, Pearson.
8. John P. Hayes: Computer Architecture and Organization, Tata McGraw Hill.
9. 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 Practicals/Programs/Lab Assignments

1. Load, store, move, and exchange data between registers and memory.
2. Write a program to display a pyramid of numerals or alphabets of a given string
3. Learn Basic syntax, instruction formats, and simple programs in Assembly Language of x86 Machines
4. Write a program using Macros.
5. Write a program to Load, store, move, and exchange data between registers and memory.
6. Write a program to perform addition, subtraction, multiplication, and division Using x86 Arithmetic Instructions
7. Write a program to AND, OR, XOR, NOT, shift, and rotate operations using x86 Logical and Bit Manipulation Instructions
8. Implementing Branching in x86 Assembly Language
9. Implementation of Loop Structures in x86 Assembly Language
10. Perform Array Processing in x86 Assembly Language: Accessing, iterating, and manipulating arrays.
11. Development of Procedures and Macros in x86 Assembly Language

12. Using x86 String Instruction to perform operations on strings using MOVS, LODS, STOS, SCAS, and CMPS instructions.
13. Implementing a Palindrome program for a given input.
14. Implementing a program for searching an element in a given array

Any other Programs/Lab Assignments/Programs assigned by the teachers

Academic Session: 2025-26
Fourth Semester

Name of the Program	**B.Sc. (Computer Science) Single Major**	**Program Code**	
Name of the Course	**Computer Networks**	**Course Code**	**25CSCS404DS01**
Hours/Week	**4 (4+0+0)**	**Credits (L:T:P)**	**4:0:0**
Max. Marks.	**Theory: 100 (70+30)**	**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:

The objective of this course is to inculcate knowledge in students about computer networks and data communication paradigms, about network models and standards, network protocols and their use, wireless technologies.

Course Outcomes:

By the end of the course the students will be able to:
CO1: Learn basics of computer networks.
CO2: Learn various network models used in networking.
CO3: Learn basics and categories of various transmission media.
CO4: Learn various routing techniques and congestion control techniques.
CO5: Understand the working of different applications.

Unit – I

Introduction: Data Transmission concepts, Transmission Media:Magnetic, twisted pair, coaxial cable, fiber optics, wireless transmission (radio, microwave, infrared), transmission impairments, switching, modulation, multiplexing.

Network: Basic concepts of network, Network topology & Protocols, types of network: LAN, MAN, WAN, Wireless networks, Inter-networks, network layer, interfaces and services.

Reference Models: OSI, TCP/IP and their comparison

Unit – II

Data Link Layer: Framing, Error control, sliding window protocols (one bit, Go back n, selective repeat). DLL Protocols–HDLC, PPP.

Medium Access Sub Layer: Channel Allocation, MAC protocols – ALOHA, CSMA protocols, Collision free protocols, Limited Contention Protocols, Wireless LAN protocols, IEEE 802.3, 802.4, 802.5 standards and their comparison.

Unit – III

Network Layer: Design issues, Routing algorithms (shortest path, flooding, flow based, distance vector, hierarchical, broadcast, multicast, for mobile hosts), Congestion control algorithms (Leaky bucket, Token bucket, Choke Packet, Load shedding), Internetworking, IP Protocol, ARP, RARP.

Unit – IV

Transport Layer: Addressing, establishing and releasing connection, flow control, buffering, Internet Transport Protocol (TCP and UDP).

Application Layer: Domain name system, E-mail, File transfer protocol, HTTP, HTTPS, TELNET

Suggested Readings:

1. Tanenbaum, Andrew S.: Computer Networks (4th Edition), PHI.
2. Forouzan, B. A.: Data Communications and Networking, Fourth Edition, Tata McGraw Hill.
3. Douglas E. Comer: Internet Working with TCP/IP (Vol.1, 4th Edition), CPE.
4. Gill, Nasib Singh: Handbook of Computer Fundamentals, Khanna Book Publishing Company (Pvt.) Limited, New Delhi.
5. Stallings, William: Data and Computer Communications (8th Edition), PHI.
6. Nance, Bary: Introduction to Networking, PHI, 4th Edition.
7. DoerrAlan & Levasseur Kenneth : Applied Discrete Structures for Computer Science, Galgotia Pub. Pvt .Ltd.
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.

Name of the Program	**B.Sc. (Computer Science) Single major**	**Program Code**	**----------**
Name of the Course	**Elements of Software Engineering**	**Course Code**	**25CSCS404DS02**
Hours/Week	**4 (4+0+0)**	**Credits (L:T:P)**	**4:0:0**
Max. Marks.	**Theory: 100 (70+30)**	**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 Objectives:

Students will gain experience on various processes used in Software industry for the development of a software product. They also learn about design, testing and maintenance of software products.

Course Outcomes:

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

CO1: Analyze and resolve software crisis issues using systematic and scientific approaches in the development of software systems.

CO2: Aiming to develop the software system with low cost, high quality and within the given time frame.

CO3: Use a variety of scripting tools and languages to automate routine tasks such as analysis, design, coding and testing tasks, security issues to the implementation of software systems.

CO4: Install, configure, troubleshoot, maintain, and upgrade software components.

CO5: Provide efficient and effective4 technical support to clients in a manner that promotes safe computing practices and reduces software risk.

Unit-I

Introduction to Software Engineering: The evolving role of Software, Characteristics of Software, Software crisis, Software engineering Approach and Challenges, Principles of software engineering.

Software Development Process Models: Waterfall, Prototype, Iterative enhancement, Spiral Models RAD Model and Automation through software environments. Quality Standards like ISO 9001, SEI-CMM.

Unit-II

Software Requirements Analysis & Specifications: Requirement engineering, requirement elicitation techniques, requirements analysis using DFD, Data dictionaries & ER Diagrams, Requirements documentation, Nature of SRS, Characteristics & organization of SRS.

Software Metrics and Measure: Need and benefits of Software Metrics, Size Metrics: Line of code, Token metrics, Function point metrics, Control Complexity Metrics, Software Project Estimation Models- COCOMO models.

Unit-III

Software Design Process: Basic Concept of Software Design, Architectural Design, Low Level Design: Modularization, Design Structure Charts, Flow Charts, Coupling and Cohesion Measures, Design Strategies: Function Oriented Design, Top-Down and Bottom- Up Design, Interface Design.

Coding: Goals of coding phase, Programming style, Structured programming: objectives of structured programming, Principles of structured programming, advantages and disadvantages of structured programming.
Computer Aided Software Engineering (CASE), CASE Tools..

Unit-IV

Software Testing: Testing Process, Design of Test Cases, Types of Testing, Functional Testing, Structural Testing, Test Activities, Unit Testing, Integration Testing and System Testing, Debugging Activities.

Software Maintenance: Maintenance as part of software evaluation, Reason for maintenance, Type of maintenance, Designing for maintainability, Techniques for maintenance, Software reliability, Software documentation.

Suggested Readings:

1. Gill, Nasib Singh: Software Engineering, Khanna Book Publishing Co.(P) Ltd, N. Delhi
2. Jalote, Pankaj: An Integrated Approach to Software Engineering, Narosa Publications.
3. Ghezzi, Carlo: Fundaments of Software Engineering, PHI.
4. Fairley, R.E.: Software Engineering Concepts, McGraw-Hill.
5. Lewis, T.G.: Software Engineering, McGraw-Hill.
6. Shere: Software Engineering & Management, Prentice Hall.
7. Deutsch, Willis: Software Quality Engineering: A Total Technical and Management Approach, Prentice Hall.
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.

Name of the Program	**B.Sc. (Computer Science) Single Major**	**Program Code**	**----------**
Name of the Course	**DBMS and MySQL**	**Course Code**	**25CSCS404DS03**
Hours/Week	**5(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:

The course introduces the foundations of database management systems by focusing on significance of a database, relational data model, schema creation and normalization, transaction processing. It also provides the practical exposures on various database operations.

Course Outcomes:

By the end of the course the students will be able to:
CO1: Identify major components of DBMS and their functions.
CO2: Design database schemas according to an application's data.
CO3: Write queries in relational algebra, SQL or MySQL.
CO4: Normalize a given database schema to avoid data anomalies and data redundancy.
CO5: Perform different operations on databases.

Unit – I

Introduction to Database Management Systems: Characteristics of database approach, data models, DBMS architecture and data independence. Entity Relationship and Enhanced ER Modeling: Entity types, relationships, Schema Definition, constraints, and object modelling

Unit – II

Relation data model: Relational model concepts, relational constraints, relational algebra
Database design: Mapping ER/EER model to relational database, functional dependencies, Lossless decomposition, Normal forms (1NF to 5NF).

Unit – III

Transaction and data storage: Introduction to transaction processing: ACID properties, concurrency control; Introduction to indexing structures for files, operations on files.

Introduction of MySQL: Features, Installation, Data Types, Variables.

Databases: Create Database, Select Database, Show Database, Drop Database, Table & Views, Create Table, Drop Table, Alter Table, Show Table, Truncate Table, Temporary Table, Copy Table, Views.

Unit – IV

Queries: Query Constraints, insert Query, Update Query, Delete Query, Select Query, MySQL Clauses: Where, AND, OR, LIKE, Limit, Order By, Group By, Having, Distinct, Union, Union All.

Joins: Inner Join, Outer Join, Self Join, Primary Key, Foreign Key, Unique Key, Aggregate Function: count(), sum(), avg(), max(), min(), first(), last()

Suggested Readings:

1. R. Elmasri, S.B. Navathe, Fundamentals of Database Systems 6th Edition, Pearson Education.
2. R. Ramakrishanan, J. Gehrke, Database Management Systems 3rd Edition, McGraw-Hill.
3. A. Silberschatz, H.F. Korth, S. Sudarshan, Database System Concepts 6th Edition, McGraw Hill.
4. R. Elmasri, S.B. Navathe Database Systems Models, Languages, Design and application Programming, 6th Edition, Pearson Education.
5. M. Neider, T. Davis, & M. Woo, OpenGL Programming Guide: Version 4.5, Addison-Wesley.
6. Timothy Boronczyk, Martin E. Psinas, "PHP and MYSQL (Create-Modify-Reuse)", Wiley India Private

Limited.
7. Robin Nixon, "Learning PHP, MySQL, JavaScript, CSS & HTML5", 3rd Edition Paperback, O'reilly.
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 Practicals/Programs/Lab Assignments

1. Write DDL Commands to
   (a) Create table
   (b) Alter table
   (c) Truncate the table
   (d) Drop table
2. Write DML Commands to
   (a) Select
   (b) Update
   (c) Delete
   (d) Insert
3. Perform condition specification using Boolean and comparison operators (and, or, not, =, <>, >, <,>=, <=) on a table.
4. Perform various arithmetic operators and aggregate functions (Count, sum, avg, Min, Max) on any table.
5. Perform multiple table queries like join on different or same tables.
6. Perform nested select statements on any table.
7. Use pattern matching, range searching and logical operators.
8. Perform various joins on the tables.
9. Creation of tables with different constraints.

Any other Programs/Lab Assignments/Programs assigned by the teachers

Name of the Program	**B.Sc. (Computer Science) Single Major**	**Program Code**	**----------**
Name of the Course	**Computer Graphics and Animation**	**Course Code**	**25CSCS404DS04**
Hours/Week	**5 (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:
This course introduces the students the fundamentals and mathematics of Computer Graphics. It provides the required concepts and principles of drawing various elementary mathematical objects such as point, line, circle and provide a base of 2D and 3D graphics programming and techniques so that learner can take on higher order skills like animation and computer-based design.

Course Outcomes:
By the end of the course the students will be able to:
CO1: Understand the fundamental concepts of computer graphics and its applications.
CO2: Implement 2D and 3D geometric transformations and understand their mathematical foundations.
CO3: Develop algorithms for graphics primitives like line drawing, polygon filling, and clipping.
CO4: Explore 3D graphics, viewing pipelines, and animation principles.
CO5: Apply computer graphics and animation concepts to build practical applications using OpenGL.

Unit –I

Introduction to Computer Graphics: Basics of Computer Graphics, Definition, Applications, Raster and

Vector graphics, **Graphics Hardware:** Display devices (CRT, LCD, LED, Plasma)**,**

Input devices: mouse, keyboard, graphic tablets.
Rasterization and Graphics Primitives: Line drawing algorithms (DDA, Bresenham's), Circle drawing algorithms (Midpoint, Bresenham's).

Unit – II

2D Geometric Transformations: Translation, Rotation, Scaling, Reflection, Shearing, Homogeneous coordinates and composite transformations. **Polygon filling algorithms:** scan-line, boundary-fill

3D Geometric Transformations: Translation, Rotation, Scaling, and **Projections**-types, matrices for Orthographic and perspective projections.

Unit – III

Clipping: 2D clipping algorithms (Cohen-Sutherland, Liang-Barsky), Basics of 3D clipping.

Viewing in 3D: Viewing pipeline, viewport transformations, Hidden surface removal (Z-buffer, Painter's algorithm).

Unit – IV

Illumination and Shading: Light models (ambient, diffuse, specular), Shading models (flat, Gouraud, Phong).

Basics of Animation: Principles of animation (timing, squash/stretch, anticipation), Frame-by-frame, keyframe, and procedural animation, Interpolation techniques.

Suggested Readings:

1. Hearn, D., & Baker, M. P., Computer Graphics: C Version, Pearson.
2. Foley, J. D., van Dam, A., Feiner, S. K., & Hughes, J. F., Computer Graphics: Principles and Practice, Addison-Wesley.
3. Hill, F. S., & Kelley, S., Computer Graphics Using OpenGL, Pearson.
4. Parent, R., Computer Animation: Algorithms and Techniques, Morgan Kaufmann.
5. Watt, A., 3D Computer Graphics, Addison-Wesley.
6. Angel, E., &Shreiner, D., Interactive Computer Graphics: A Top-Down Approach with OpenGL, Addison-Wesley.
7. M. Neider, T. Davis, & M. Woo, OpenGL Programming Guide: Version 4.5, Addison-Wesley.
8. Rick Parent, Animation Techniques and Applications, Morgan Kaufmann.
9. J. Vince, Essential Computer Animation: Algorithms and Techniques, Springer.
10. 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 Practicals /Lab Assignments/Programs

1. Implement a moving object of your choice.
2. Implement DDA line drawing algorithm
3. Implement Bresenham's line drawing algorithm.
4. Implement the circle drawing using polynomial approach.
5. Implement polar circle drawing algorithm.
6. Implement the midpoint algorithm for circle drawing
7. Develop a scan-line polygon filling program.
8. Perform basic 2D transformations (translation, scaling, rotation).
9. Implement composite transformations for complex 2D objects.
10. Implement a bouncing ball.
11. Demonstrate the Cohen-Sutherland clipping algorithm.
12. Visualize a 3D co-ordinate system and perform 3D translation transformation.
13. Visualize a 3D co-ordinate system and perform 3D rotation transformation.
14. Visualize a 3D co-ordinate system and perform 3D scaling transformation.
15. Draw smiley face as in chats.
16. Create a 3D object of your choice.
17. Build a basic animation using keyframe interpolation.
18. Develop a mini animation project integrating some transformations and rendering of your choice.

Any other Practicals/Lab Assignments/Programs assigned by the teachers.

Academic Session: 2026-27
Fifth Semester

Name of the Program	**B.Sc. (Computer Science) Single Major**	**Program Code**	**----------**
Name of the Course	**Artificial Intelligence**	**Course Code**	**26CSCS405DS01**
Hours/Week	**5 (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:

This course introduces students to fundamental concepts of Artificial Intelligence (AI), including problem-solving, knowledge representation, reasoning, and machine learning. The course aims to equip students with the ability to develop AI-based solutions and understand how AI techniques are applied in real-world scenarios.

Course Outcomes:

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

CO1: Grasp core AI concepts and problem-solving techniques.
CO2: Utilize knowledge representations methods in AI systems.
CO3: Comprehend the role and development of expert systems.
CO4: explore the concepts of perception, learning, Neural Network and Fuzzy Logic.
CO5: Grasp the knowledge of Prolog..

Unit – I

Introduction: Definition, Concepts and applications of Artificial Intelligence, Problem solving: Defining problem as State space search, Production systems, Problem characteristics, Search techniques: Brute force and Heuristic search and their different searching techniques.

Intelligent Agents: Agent types (simple reflex, model-based, goal-based, utility-based, learning agents). Agent environments (static/dynamic, observable/partially observable, deterministic/stochastic).

Unit – II

Knowledge representation: Types of knowledge, Inferencing, Inference rule, Knowledge Representation: Logic based Knowledge representation, Rule based knowledge representation;

Knowledge representation: Non-Monotonic reasoning, Knowledge representation based on probability and uncertainty; Knowledge representation schemes: Formal logic, Semantic net, Frame, Scripts.

Unit – III

Expert System: Definition, Role of Knowledge in expert system, Architecture of Expert system.

Expert system development life cycle: Problem selection, Prototype construction, Formalization, Implementation, Evaluation, Knowledge acquisition and different acquisition techniques, Knowledge engineering, Cognitive behaviour.

Perception: Sensing, Speech recognition, Vision, Action.

Unit – IV

Learning, Planning and Understanding: Learning and its different types, Planning, understanding.

Neural Networks: Introduction, Comparison of artificial neural networks with biological neural networks, Learning in neural networks, Perceptions, Back propagation networks, application of neural networks.

Fuzzy logic: Definition, Difference between Boolean and Fuzzy logic, fuzzy subset, fuzzy membership function, Approximate reasoning,Fuzzy Inference system. ***Implementation using Prolog***

Suggested Readings:

1. Russell S. and Norvig P.: Artificial Intelligence: A Modern Approach, Prentice-Hall.
2. Elaine Rich, Kevin Knight and Nair: Artificial Intelligence, TMH.
3. Luger G. F. and Stubblefield W. A.: Artificial Intelligence: Structures and strategies for ComplexProblem Solving, Addison Wesley.
4. Nilsson Nils J.: Artificial Intelligence: A New Synthesis, Morgan Kaufmann Publishers Inc.
5. Patrick Henry Winston: Artificial Intelligence, Addison-Wesley Publishing Company.
6. M. Mitchell: An Introduction to Genetic Algorithms, Prentice-Hall.
7. J.S.R. Jang, C.T. Sun and E. Mizutani: Neuro-Fuzzy and Soft Computing, PHI, Pearson Education.
8. Davis E. Goldberg: Genetic Algorithms: Search, Optimization and Machine Learning, Addison Wesley.
9. 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 Practicals/Lab Assignments/Programs

1. Write a program to classify AI as Narrow AI or General AI based on user input.
2. Implement a basic agent that interacts with an environment (e.g., a reflex vacuum cleaner agent).
3. Implement a PEAS framework for a simple task, such as a chatbot assistant.
4. Represent a simple problem like the "Missionaries and Cannibals" problem in state-space form.
5. Implement an agent that reaches a defined goal using rule-based decision-making.
6. Implement Breadth-First Search (BFS) and Depth-First Search (DFS) on a simple graph.
7. Implement the A* search algorithm to solve the 8-puzzle problem.
8. Implement the hill climbing algorithm to solve a mathematical optimization problem.
9. Implement the Minimax algorithm for an AI that plays Tic-Tac-Toe optimally.
10. Improve the Minimax algorithm with alpha-beta pruning to enhance efficiency.
11. Implement a Bayesian network for weather prediction.
12. Build a simple linear regression model to predict house prices.
13. Implement logistic regression for classifying email as spam or not spam.
14. Apply k-means clustering to segment customer data into different groups.
15. Implement a decision tree classifier to predict student grades based on study hours.
16. Image Classification using CNNs – Train a Convolutional Neural Network (CNN) on a dataset like MNIST.
17. Sentiment Analysis with NLP – Implement a sentiment analysis model using pre-trained NLP models.

Any other Practicals/Lab Assignments/Programs assigned by the teachers.

Name of the Program	**B.Sc. (Computer Science) Single Major**	**Program Code**	**----------**
Name of the Course	**Computational Theory**	**Course Code**	**26CSCS405DS02**
Hours/Week	**4 (4+0+0)**	**Credits (L:T:P)**	**4:0:0**
Max. Marks.	**Theory: 100 (70+30)**	**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 basic theory of computer science and formal methods of computation. It will enable learners analyze and design finite automata, regular language, context free grammar. It provides enough knowledge on the relationships among machines, languages and grammars.

Course Outcomes:

By the end of the course the students will be able to:
CO1: Analyze and design finite automata, formal languages, and grammars.
CO2: Explain the basic concepts of deterministic and non-deterministic finite automata, regular language, context-free language, Turing machines
CO3: Construct context free grammar for various languages and find ambiguous grammars, able to construct derivative trees
CO4: Describe the formal relationships among machines, languages and grammars

Unit – I

Fundamentals: Alphabets, Strings, Languages, Problems, Graphs, Trees, Finite State Systems, Definitions, Finite Automaton Model, Acceptance of Strings, And Languages.

Types of Finite Automata: Deterministic Finite Automaton and Nondeterministic Finite Automaton, Transition Diagrams, Transition Tables, Equivalence of DFA. Finite Automata with ε-Moves, Significance, Acceptance of Languages, ε-Closure, Conversion of Regular Expression to NFA, Minimization of Finite Automata, Finite Automata with Output– Moore and Mealy Machines, Conversion of Moore machine to Melay Machine & Vice-Versa.

Unit – II

Regular Language: Properties of Regular languages, Conversion of DFA to Regular Expression, Arden Theorem, Pumping Lemma, Properties and Limitations of Finite state machine, Decision properties of Regular Languages, Application of Finite Automata.

Unit – III

Context Free Grammar and Push Down Automata: Context Free Grammar, Derivation tree and Ambiguity, Application of Context free Grammars, Chomsky Normal form, Properties of context free grammar, Push down Stack Machine, Design of Deterministic and Non-deterministic Push-down stack.

Unit – IV

Turing Machine: Turing machine definition and design of Turing Machine, Variations of Turing Machines, combining Turing machine, Universal Turing Machine, Post Machine and Chomsky Hierarchy of grammars. Definitions Of P And NP Problems, NP Complete and NP Hard Problems.

Suggested Readings:

1. John C. Martin, "Introduction to Language and theory of Computation", McGraw Hill.
2. John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman, Introduction to Automata Theory Languages and Computation, Pearson Education
3. K. L. P Mishra, N. Chandrashekaran, Theory of Computer Science-Automata Languages and Computation, 2nd edition, Prentice Hall of India, India.
4. K.Krithivasan and R.Rama; Introduction to Formal Languages, Automata Theory and Computation; Pearson Education.
5. Harry R. Lewis and Christos H. Papadimitriou, Elements of the theory of Computation, Second Edition, Prentice-Hall of India Pvt. Ltd.
6. 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.

Name of Program	**B.Sc. (Computer Science) Single Major**	**Program Code**	**----------**
Name of Course	**Mobile Application Development**	**Course Code**	**26CSCS405DS03**
Hours/Week	**5 (3+0+2)**	**Credits (L:T:P)**	**3:0:1**
Max. Marks	**Theory: 75(50+25) Practical: 25**	**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 Objectives:

This course focuses on leveraging Java for developing mobile applications with Android. Students will learn to design user interfaces, handle data persistence, and integrate multimedia and device sensors into mobile applications. The course culminates in preparing applications for deployment to app stores.

Course Outcomes:

By the end of the course, the students will be able to:
CO1: Build mobile applications by understanding the Android development environment and implementing UI design principles using layouts and views.
CO2: Manage activities and navigation in Android applications using intents and fragments.
CO3: Utilize local and cloud data storage solutions like SQLite and Firebase for managing application data.
CO4: Incorporate multimedia, sensors, and location-based features into applications and prepare them for publishing on the Google Play Store.

Unit I

Introduction to Android Development: Overview of Mobile App Development, Introduction to Android Operating System, setting up the Android Studio IDE, Understanding Android Architecture and Components, Creating Your First Android Application.

User Interface Design: XML Basics for Layouts; Views and ViewGroups: LinearLayout, RelativeLayout, ConstraintLayout, etc.; Widgets: Buttons, TextViews, EditTexts, ImageViews, etc.; RecyclerView and ListView, Material Design Principles, Event Handling and Listeners.

Unit II

Activities and Intents: Activity Lifecycle and States, Passing Data Between Activities, Implicit and Explicit Intents, Navigating Between Screens.

Fragments and Navigation: Introduction to Fragments, Fragment Lifecycle, Creating Dynamic UIs with Fragments, Navigation Component and Navigation Graphs.

Unit III

Data Storage and Management: Shared Preferences, Internal and External Storage, SQLite Databases, Room Persistence Library, Introduction to Cloud Storage (Firebase Realtime Database).

Networking and APIs: Introduction to REST APIs and JSON, Making HTTP Requests with Retrofit/Volley, Parsing JSON Data, Implementing Authentication and Authorization, Handling Network Responses and Errors.

Unit IV

Multimedia and Sensors: Playing Audio and Video, Capturing Images and Videos Using Camera, Accessing Device Sensors (e.g., Accelerometer, GPS), Location-Based Services and Google Maps Integration.

Publishing Your App: Preparing the App for Release, Generating Signed APKs, Play Store Guidelines and Policies, and Publishing the App on Google Play Store.

Suggested Readings:

1. ZigurdMednieks, Laird Dornin, G, BlakeMeike and Masumi Nakamura: Programming Android, O'Reilly Publications.
2. Wei-Meng Lee: Beginning iPhone SDK Programming with Objective-C, Wiley India Ltd.
3. James C.S: Android Application development, CENGAGE Learning.
4. Gargenta M., Nakamura M.: Learning Android, O'Reilly Publications.
5. Reto Meier: Professional Android 2 Application Development, WROX Publication-Wiley-India.
6. James Edward: J2ME: The Complete Reference, James Edward – Publication.
7. Chris Haseman: Android Essentials, Apress Publication.
8. Mark L Murphy: Beginning Android - Wiley India Pvt Ltd.
9. Sayed Y Hashimi and SatyaKomatineni: Pro Android – Wiley India Pvt Ltd.
10. Lauren Darcey, Shane Conder: Android Wireless Application Development, 2nd edition –Pearson Education.
11. 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 Practicals/Lab Assignments/Programs

1. Set up Android Studio and develop a "Hello World" application.
2. Design an interactive user interface using multiple layout types.
3. Create a basic Android calculator app using event listeners.
4. Implement implicit and explicit intents for activity transitions.
5. Develop an app demonstrating the use of fragments in UI design.
6. Implement shared preferences to store user settings and retrieve them later.
7. Create an application that saves and retrieves user information using SQLite.
8. Develop an app that makes HTTP requests using Retrofit and displays JSON data.
9. Implement authentication using Firebase in an Android application.
10. Develop a simple notes app that allows data storage using Firebase Firestore.
11. Implement Google Maps API to display real-time locations.
12. Create a media player application that plays audio files.
13. Develop an app that captures and stores images using the device camera.
14. Design an Android To-Do list app with CRUD functionality.
15. Implement RecyclerView to display a list of users fetched from an API.
16. Create a QR Code scanner app using a third-party library.
17. Build a push notification system using Firebase Cloud Messaging (FCM).
18. Design and test an e-commerce app prototype using Android components.
19. Implement dark mode and app theme customization.
20. Publish an Android application to the Google Play Store.

Any other Practicals/Lab Assignments/Programs assigned by the teachers.

Name of the Program	**B.Sc. Computer Science (Single Major)**	**Program Code**	**----------**
Name of the Course	**Computer System Architecture**	**Course Code**	**26CSCS405DS04**
Hours/Week	**4 (4+0+0)**	**Credits (L:T:P)**	**4:0:0**
Max. Marks.	**Theory: 100 (70+30)**	**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 7 parts (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.

Course Objectives:

Upon completion of this course, students will have a better understanding of basic operations, principles, and components of Computer Architecture. Students will learn about different types of registers, instructions and interrupts used in Computer System. This course will empower the students to understand complex concepts related to computer architecture with ease, such as addressing modes, types of memory, etc.

Course Outcomes:

By the end of the course the students will be able to:
CO1: Understand the basic Computer concepts such as its different functional units and bus structure.
CO2: Interpret the functional architecture of Computer system in terms of its different components.
CO3: Analyze different types of Instructions and Interrupts used in Computer system.
CO4: Identify and compare different methods for computing Input/Output.
CO5: Explore Memory organization and different operations related to it.

Unit – I

Basic Computer Concepts: Computer Types: Analog and Digital, Digital Computer Organization and Architecture: Functional Units, Basic Organizational Concepts, Von-Neumann Architecture, Bus Structure. Data representation: Number Systems, Fixed and Floating-point representation.

Register Transfer and Microoperations: Basic concepts and types of Registers, Register Transfer Language, Data Transfer between Registers, Bus and Memory Transfer, Microoperations: Arithmetic, Logic and Shift Microoperations.

Unit – II

Basic Computer Design: Instruction codes, Common Bus System Architecture, Computer Instructions: Instruction Set and Instruction Cycle. Types of Instructions: Register Reference, Memory Reference and Input-Output Reference.

Interrupts: Classifications of Interrupts: Maskable and Non maskable, Hardware and Software. Interrupt Service Routine, Context Switching, Interrupt Identification: Daisy Chaining, Polling and Vectored Interrupt. Interrupt Cycle.

Unit – III

Central Processing Unit: Introduction, General Register Organization, Stack Organization: Register and /memory Stack, Reverse Polish Notation. Instruction Formats: Three Address, Two Address, One Address and Zero Address.

Data Transfer and Manipulation: Addressing Modes: Implied, Immediate, Register Direct and Indirect Mode and Direct Address Mode. Data Transfer and Data Manipulation Instructions, Program Control, Reduced Instruction Set Computer (RISC) and Complex Instruction Set Computer (CISC).

Unit – IV

Memory Organization: Memory Hierarchy, Main Memory Technologies, Auxiliary Memory, Associative Memory: Hardware requisites, Working Principle and Operations. Cache Memory: Characteristics, Types of Mapping, Writing into Cache, Cache Coherence.

Input-Output Organization: Peripheral Devices, Input-Output Interface, Asynchronous Data Transfer, Modes

of Transfer: Programmed I/O and Interrupt Initiated I/O. Concepts related to Priority Interrupt: Daisy Chaining Priority and Parallel Priority Interrupt. Direct Memory Access (DMA): DMA Controller and DMA Transfer.

Suggested Readings:

1. Mano, M.M.: Computer System Architecture, Prentice-Hall of India.
2. Gill Nasib Singh and Dixit J.B: Digital Design and Computer Organisation, University Science Press (Laxmi Publications), New Delhi.
3. Stallings, William: Computer Organisation & Architecture, Pearson Education.
4. Mano, M.M.: Digital Logic and Computer Design, Prentice-Hall of India.
5. Hwang Kai: Advanced Computer Architecture, McGraw Hill International.
6. Hayes J. P: Computer System Architecture & Organization, McGraw-Hill Education.
7. Hamacher V. Carl: Computer Organization, McGraw Hill.
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.

Academic Session: 2026-27
Sixth Semester

Name of the Program	**B.Sc. (Computer Science) Single Major**	**Program Code**	**----------**
Name of the Course	**Compiler Design**	**Course Code**	**26CSCS406DS01**
Hours/Week	**4 (4+0+0)**	**Credits (L:T:P)**	**4:0:0**
Max. Marks.	**Theory: 100 (70+30)**	**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 7 parts (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.

Course Objectives:

This course focuses on the fundamental concepts of compiler design and its phases. It familiarize students with the principles of parsing techniques, symbol tables, intermediate code generation and explore optimization techniques and code generation strategies.

Course Outcomes:

By the end of the course the students will be able to:
CO1: Explain the role of compilers in program execution and describe different phases of compilation.
CO2: Implement lexical analysis using finite automata and regular expressions.
CO3: Apply various parsing techniques to perform syntax analysis of a given source code.
CO4: Design and implement syntax-directed translations for semantic analysis.
CO5: Generate intermediate representations and optimize code for better performance.

Unit – I

Introduction to Compilers: Overview of the compilation process, structure of a compiler, phases of compilation, comparison of interpreters and compilers.

Lexical Analysis: Role of lexical analyzer, tokens, lexemes, and patterns, finite automata and regular expressions, designing a lexical analyzer, transition diagrams, handling reserved words and identifiers.

Lexical Analyzer Implementation: Introduction to Lex tool, tokenization, error detection and recovery in lexical analysis.

Unit – II

Syntax Analysis: Role of parsers, syntax-directed definitions, parse trees and derivations, ambiguous grammars.

Parsing Techniques: Top-down parsing, Recursive Descent Parsing, LL(1) parsers, predictive parsing.

Bottom-Up Parsing: Shift-reduce parsing, LR parsers (SLR, CLR, LALR), parser generator tools (e.g., Yacc).

Unit – III

Semantic Analysis: Type checking, type conversion, static and dynamic scoping, syntax-directed translation.

Symbol Table Management: Data structures for symbol tables, scope management.

Intermediate Code Generation: Three-address code (TAC), abstract syntax trees, postfix notation, quadruples and triples.

Run-Time Environments: Activation records, storage organization, stack-based memory allocation.

Unit – IV

Code Optimization Techniques: Basic blocks and flow graphs, optimization of basic blocks, loop optimization, peephole optimization.

Code Generation: Issues in the design of a code generator, target machine architecture, instruction selection, register allocation, and instruction scheduling.

Compiler Construction Tools: Overview of compiler generation tools such as LLVM, GCC, Lex, and Yacc.
Recent Trends in Compiler Design: Just-In-Time (JIT) compilation, dynamic compilation, and parallelizing compilers.

Suggested Readings:

1. Donovan: Systems Programming, Tata McGraw Hill.
2. Dhamdhere: System Software, Tata McGraw Hill.
3. Sunitha, K. V. N. Compiler construction. Pearson Education India.
4. Alfred V.Aho and Jeffrey D.Ullman: Principles of Compiler Design, Addison Wesley.
5. Aho, Alfred V., Sethi, Ravi, & Ullman, Jeffrey D. – Compilers: Principles, Techniques, and Tools (2nd Edition) – Pearson Education (Also known as the Dragon Book).
6. Dhamdhere, D. M. – Compiler Construction: Principles and Practice – McGraw Hill.
7. Kenneth C. Louden: Compiler Construction: Principles and Practice, Cengage Learning.
8. Holub, Allen I.: Compiler Design in C, Prentice Hall.
9. Andrew W. Appel: Modern Compiler Implementation in C/Java, Cambridge University Press.
10. Steven S. Muchnick: Advanced Compiler Design and Implementation, Morgan Kaufmann.
11. Tremblay, Jean-Paul, & Sorenson, Paul G.: The Theory and Practice of Compiler Writing, McGraw Hill.
12. 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.

Name of the Program	**B.Sc. (Computer Science) Single Major**	**Program Code**	**----------**
Name of the Course	**Security in Computing**	**Course Code**	**26CSCS406DS02**
Hours/Week	**5 (3+0+2)**	**Credits (L:T:P)**	**3:0:1**
Max. Marks.	**Theory: 75 (50+25)** **Practical: 25**	**Time of Examinations**	**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. Students 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:

To equip students with the knowledge of security principles, cryptographic techniques, and risk management strategies, enabling them to design and implement secure systems, mitigate vulnerabilities, safeguard networks and data, and stay informed about evolving cybersecurity threats and tools to ensure the reliability and confidentiality of computing environments.

Course Outcomes:

By the end of the course, the students will be able to:
CO1: Identify threats, vulnerabilities, and risks in computing systems.
CO2: Apply cryptographic techniques for secure data communication
CO3: Evaluate network and web security mechanisms.
CO4: Implement secure software development practices to mitigate risks.
CO5: Use security tools to configure and manage secure systems.

Unit – I

Fundamentals of Security: Introduction to Security in Computing: Concepts, Need, and Scope, Principles of Information Security: Confidentiality, Integrity, Availability (CIA Triad), Threats, Vulnerabilities, and Attacks: Types (Malware, Phishing, DoS etc.), Overview of Cryptography: Symmetric and Asymmetric Key Cryptography, The Data Encryption Standard, The AES Encryption Algorithm, Public Key Encryptions, Uses of Encryption, Security Policies, Risk Assessment, and Management, Case Studies: Real-world breaches and lessons learned.

Unit – II

System and Network Security: Network Security: Firewalls, VPNs, and Intrusion Detection Systems (IDS), Secure Network Protocols: HTTPS, SSL/TLS, IPsec, System Security: Hardening Techniques, Secure Configuration, and Patching, User Authentication and Access Control Mechanisms, Case Studies: Securing Enterprise Networks.

Unit – III

Application and Cloud Security: Secure Software Development Lifecycle (SDLC), Web Application Security: OWASP Top 10 Vulnerabilities, Cloud Security: Data Protection, Identity Management, and Security Tools, Security Testing Techniques: Penetration Testing, Vulnerability Scanning, Case Studies: Cloud Data Breaches.

Unit – IV

Advanced Topics and Legal Aspects: Advanced Cryptographic Techniques: Blockchain, Zero-Knowledge Proofs, Security in IoT and Emerging Technologies, Cyber Laws and Ethical Hacking, Incident Response and Forensics, Future of Security: AI in Cybersecurity, Quantum Computing Threats.

Suggested Readings:

1. Pfleeger, C. P., &Pfleeger, S. L.: Security in Computing, Pearson Education.
2. Stallings, W.: Cryptography and Network Security: Principles and Practice, Pearson Education.
3. Schneier, B.: Applied Cryptography: Protocols, Algorithms, and Source Code in C, Wiley.
4. Bishop, Matt (Matthew A.): Computer Security: Art and Science, Second edition, Addison-Wesley (Pearson Education).
5. Pramanik, S., Samanta, D., Vinay, M., &Guha, A. (Eds.), Cyber Security and Network Security, Wiley.
6. Gupta, B. B. (Ed.), Computer and Cyber Security, CRC Press.
7. Daimi, Kevin: Computer and Network Security Essentials, Springer International Publishing.
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 Practicals/Lab Assignments/Programs

1. Write a program to implement the Caesar cipher encryption algorithm.
   (Encrypt a given string by shifting characters by a fixed number and then decrypt it back.)
2. Write a program to calculate the MD5 hash of a given string.
   (Implement the MD5 hashing algorithm to compute a hash value for any user-entered string.)
3. Write a program to demonstrate symmetric key encryption using XOR cipher.
   (Encrypt and decrypt a string using a simple XOR encryption technique with a user-provided key.)
4. Write a program to implement basic Digital Signature generation and verification.
   (Using RSA or a simpler method, implement signing and verification of a message to demonstrate authenticity.)
5. Write a program to implement a basic firewall rule set.
   (Implement a program that allows, or blocks traffic based on certain conditions like source IP, destination IP, or port.)
6. Write a program to implement a password strength checker.
   (Develop a program that checks the strength of a password based on certain criteria (length, complexity, etc.).)
7. Write a program to demonstrate SSL/TLS handshake simulation.
   (Simulate a simple SSL/TLS handshake using public/private key pairs for secure communication.)
8. Write a program to implement an RSA encryption and decryption scheme.
   (Implement RSA encryption and decryption using user-generated prime numbers for key generation.)
9. Write a program to implement an Access Control List (ACL) for file permissions.
   (Implement a basic ACL system that checks whether a user has permission to read, write, or execute a file.)
10. Write a program to implement a simple Intrusion Detection System (IDS).
    (Implement a system that detects certain predefined patterns in network traffic or user activity.)

Any other Practicals/Lab Assignments/Programs assigned by the teachers.

Name of the Program	**B.Sc. (Computer Science) Single Major**	**Program Code**	**----------**
Name of the Course	**Advance Database Systems**	**Course Code**	**26CSCS406DS03**
Hours/Week	**5 (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 5 parts (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.

Course Objectives:

This course aims to provide students with an in-depth understanding of advanced database models, architectures, query optimization techniques, and emerging database technologies. It covers object-oriented, object-relational, parallel, and distributed databases, as well as modern applications like spatial, multimedia, and XML databases.

Course Outcomes:

By the end of the course the students will be able to:
CO1: Understand the concepts of E-R Model and EER models.
CO2: Explore the concepts related to object-oriented and object-relational database systems.
CO3: Understand the working of parallel, and distributed database systems.
CO4: Explore Client/Server & Server architectures and how to retrieve information.
CO5: Analyze modern database applications, including Active, Temporal, spatial, multimedia, and XML-based databases etc.

Unit – I

EER Model: Prerequisite of Entity-Relationship Model: Entity Types, Entity Sets, Attributes & Keys, Relationship Types & Instances, ER Diagrams, Enhanced Entity Relationship (EER) Model: Subclasses, Superclasses, Inheritance, Specialization and Generalization, Constraints & Characteristics of Specialization and Generalization, Union/Category.

Object-Oriented & Object-Relational Databases: Object-Oriented Concepts, Object Identity Object Structure, Type Constructors, Encapsulation of operations, Methods and Persistence, Type hierarchies and Inheritance, Complex objects, Schema design for OODBMS, OQL, OODBMS architectures and Storage issues, Transaction Management & Concurrency Control.

Unit – II

ObjectRelationalDatabaseand: DatabasedesignforanORDBMS– Nestedrelationsandcollections;storageandaccessmethods,QueryprocessingandOptimization, Advance Querying: User define data types, manipulating objects table, objectviews;

InformationRetrieval Information Retrieval & ways toretrieveinformation.

Client/Server Architecture: Client/Server Architectures, Server Architectures

Unit – III

Parallel Databases: Architectures for Parallel Databases, I/O Parallelism, Interquery Parallelism, Intraquery Parallelism.

Distributed Databases: ArchitecturesforDistributedDatabase,DataFragmentation,Replication, and Allocation Techniques for Distributed Database Design, Query processing inDistributedDatabases; ConcurrencyControl and Recoveryin DistributedDatabases.

Unit – IV

Enhanced Data Models for Advanced Applications: Active database-syntax and semantics (DB2,Oracle), applications, design principles for active rules, Temporal database concepts, Spatial databases, Deductive databases.

Emerging Database Technologies: Mobile databases, Multimedia Databases, Geographic Information systems (GIS); XML and Internet Databases: Structured, Semi-structured and Unstructured Data, Web databases.

List of Practicals /Lab Assignments/Programs

1. Write SQL commands to create tables (Students, Courses, Instructors) and insert sample data.
2. Retrieve specific data using SELECT, WHERE, ORDER BY, and GROUP BY.
3. Convert an ER diagram of a Library Management System into SQL tables.
4. Define relationships between tables (Students and Courses).
5. Write SQL queries with INNER JOIN, LEFT JOIN, and RIGHT JOIN.
6. Write SQL scripts to create tables (Employees, Employee_Log) and an AFTER INSERT trigger to log new employee entries into the log table.
7. Write a BEFORE UPDATE trigger on the Employees table to store old and new salary values in a Salary_Audit table whenever salary is updated.
8. Create a BEFORE DELETE trigger to archive student records from the Students table into a Students_Archive table before deletion.
9. Write a BEFORE INSERT trigger on the Products table to prevent insertion if the product price is less than zero.
10. Write an AFTER INSERT trigger to automatically update stock quantity in the Products table whenever a new order is added to the Orders table.

Any other Programs/Lab Assignments given by the teachers.

Any other Practicals /Lab Assignments/Programs assigned by the teachers.

Name of Program	**B.Sc. (Computer Science) Single Major**	**Program Code**	**----------**
Name of the Course	**Wireless Sensor Networks**	**Course Code**	**26CSCS406DS04**
Hours Per Week	**5 (3+0+2)**	**Credits**	**3:0:1**
Maximum Marks	**Theory: 75 (50+25)** **Practical: 25**	**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. 1consisting of short-answer type questions covering the entire syllabus. Students 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:

To provide a comprehensive understanding of wireless sensor networks, including architecture, protocols, and energy-efficient design, emphasizing their application in IoT, environmental monitoring, healthcare, and security. Students will learn to address challenges such as data aggregation, scalability, and security to design resilient and efficient wireless systems.

Course Outcomes:

By the end of the course, the students will be able to:
CO1: Understand the architecture and operational principles of wireless sensor networks.
CO2: Design efficient communication protocols for sensor networks.
CO3: Analyze security and energy optimization challenges in WSNs.
CO4: Develop applications integrating WSN technologies for IoT and other domains.
CO5: Utilize simulation tools to design and test WSN solutions.

Unit – I

Introduction and Basics: Overview of Wireless Sensor Networks (WSNs): Characteristics, Challenges, and Applications, Sensor Node Architecture: Hardware, Energy Sources, and Communication Modules, Wireless Communication Fundamentals: Frequency Spectrum, Modulation Techniques, Protocol Stack: Physical, MAC, and Network Layers, Case Studies: Environmental Monitoring and Smart Cities.

Unit – II

Protocols and Data Management: Medium Access Control (MAC) Protocols: TDMA, FDMA, CDMA, Routing in WSNs: Data-centric, Hierarchical, and Location-Based Routing, Data Aggregation, Clustering, and Compression Techniques, Energy-Efficient Protocols and QoS in WSNs, Security in WSNs: Threats and Countermeasures.

Unit – III

Deployment and Network Design: Deployment Strategies: Static vs. Dynamic Networks, Localization and Positioning Algorithms, Topology Control and Network Scalability, Synchronization and Scheduling Techniques, Simulation Tools for WSNs (e.g., NS2, TOSSIM).

Unit – IV

Advanced Applications and Emerging Trends: Internet of Things (IoT) and WSN Integration, Mobile Sensor Networks and Drones, Machine Learning Applications in WSNs, Case Studies: Healthcare and Industrial Automation, Future Trends in Wireless Sensor Technologies.

Suggested Readings:

1. Sohraby, K., Minoli, D., &Znati, T.: Wireless Sensor Networks: Technology, Protocols, and Applications, Wiley.
2. Karl, H., &Willig, A.: Protocols and Architectures for Wireless Sensor Networks, Wiley.
3. Kumar, D. P. A., & Dutta, S.: Wireless Sensor Networks: An Information Processing Approach, Morgan Kaufmann.
4. Forouzan, B. A.: Data Communications and Networking, Fourth Edition, Tata McGraw Hill.
5. Douglas E. Comer: Internet Working with TCP/IP (Vol.1, 4th Edition), CPE.
6. Stallings, William: Data and Computer Communications (8th Edition), PHI.
7. Tanenbaum, Andrew S.: Computer Networks (4th Edition), PHI.
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 Practicals/Lab Assignments/Programs

1. Write a program to simulate a basic sensor node that generates temperature, humidity, or light data at random intervals.
2. Write a program to create a system where multiple sensor nodes send data to a central node, which aggregates and processes the data.
3. Write a program to simulate communication between sensor nodes using a routing protocol like flooding or gossip.
4. Write a program to calculate the energy used by a sensor node to transmit data to a neighboring node.
5. Write a program to implement a program to simulate the transmission path of a message from a source node to a destination node.
6. Write a program to simulate the AODV routing algorithm to determine the best route in a WSN.
7. Write a program to calculate the ratio of successfully delivered packets over total sent packets in a network simulation.
8. Write a program to calculate the position of a node based on the distances to at least three reference nodes.
9. Write a program to implement a simple data transmission and reception process with basic error-checking mechanisms.
10. Write a program to implement a scenario where nodes lose packets and require retransmission using an error-checking protocol.

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