UNIVERSITY OF CALCUTTA

Notification No.CSR/20/2023

It is notified for information of all concerned that in terms of the provisions of Section 54 of the Calcutta University Act, 1979, (as amended), and, in exercise of her powers under 9(6) of the said Act, the Vice-Chancellor has, by an order dated 25.07.2023 approved the syllabus of the under mentioned subjects semester wise Four-year (Honours & Honours with Research) /Three-year (Multidisciplinary) programme of U.G. courses of studies, as applicable under CCF,2022, under this University, as laid down in the accompanying pamphlet.

1. Food & Nutrition
2. Defence Studies
3. Human Development
4. Sanskrit (Revised syllabus after incorporating some amendments, in the syllabus published in CSR/18/23,dt.24.7.23)
5. Persian
6. Computer Science
7. B.Mus.(Honours) and Music (Minor)
8. Marks distribution for semesters 1 & 2 and Amendments in the syllabus of English (4-year Honours/3-year MDC)
9. Amendments in SEC paper of Physiology (Honours & Honours with Research) Courses (as mentioned in CSR/13/23,11.07.2023)
10. Environmental Science (Revised syllabus after incorporating some amendments, in the syllabus published in CSR/18/23, DT.24.7.2023)

The above shall take effect from the academic session 2023-2024.

SENATE HOUSE

Kolkata-700073

The 28th July ,2023

28/7/2023

Prof.(Dr.) Debasis Das

Registrar

SEM	CORE SUBJECTS (DSCC)		SEC
	PAPER NAME	PRACTICAL DETAILS	
1	Computer Fundamentals & Digital Logic (4)	Logic Design using ICs (basic circuits, focus on combinational part)	Data Visualization using Spreadsheets (4)
2	Problem Solving Using C (4)	Problem solving using C Lab (using gcc compiler)	Web Development (4) (HTML, PHP)
3	Data Structures (4)	Upto BSTs (using C)	Mobile App Development (4) (using Android Studio)
	Computer Architecture & Organization (4)	Logic Design using ICs (building logic blocks, & sequential ckts)	
4	Computational Mathematics (4)	Numerical Methods (using C)	**N.A.**
	Microprocessor (4)	8085 MPU Programming	
	Operating System (4)	Shell Programming (including system calls)	
	Object Oriented Programming (4)	Java Lab	
5	Design & Analysis of Algorithms (4)	Graph algorithms (using C++)	**N.A.**
	Data Communication and Networking (4)	Tutorial	
	Theory of Computation (4)	Tutorial	
	Database Management System (DBMS) (4)	MySQL & JS	
6	Software Engineering (4)	Tutorial (System Analysis & Design Lab)	**N.A.**
	Programming in Python (4)	Python Lab	
	Linear Algebra & Statistical Methods (4)	Related to theory (using Python)	
7	Compiler Design (4)	Tutorial	**N.A.**
	Machine Learning (4)	Related to theory (using Python)	
	Computer Graphics (4)	Using Python	
	IoT & Embedded Systems (4)	IoT & Embedded Systems (Python and IoT, Embedded Hardware)	
	Big Data Analytics / Research Project (4)	Big Data Analytics (Hadoop, MongoDB, Java Spark)	
8	Digital Image Processing (4)	Python with OpenCV	**N.A.**
	Cryptography (4)		
	Data Warehousing (4)		
	Mobile & Wireless Computing/Research Project (4)	Mobile & Wireless Computing (Network Simulation)	
	Cloud Computing (4) / Project	Cloud Computing (Using Cloud Simulator Learning Virtualisation and Developing Cloud Services)	

University of Calcutta

B.Sc (Honours and Honours with Research) 4 - years degree program in Computer Science under credit framework.

(2023)

Semester – I & II

Semester - I				
Paper	**Paper type**	**Paper name**	**Credit**	**Contact hours**
DSC/CC-1	Theory	Computer fundamentals and Digital Logic	3	45
	Practical	Computer fundamentals and Digital Logic lab	1	30
SEC – 1	Theory	Data visualization using spreadsheet	3	45
	Practical	Data visualization using spreadsheet Lab	1	30

Semester - II				
Paper	**Paper type**	**Paper name**	**Credit**	**Contact hours**
DSC/CC-2	Theory	Problem Solving using C	3	45
	Practical	Problem Solving using C Lab	1	30
SEC – 2	Theory	Web Development	3	45
	Practical	Web Development Lab	1	30

Semester - I				
Paper	Paper type	Paper name	Credit	Contact hours
DSC/CC-1	Theory	Computer fundamentals and Digital Logic	3	45
	Practical	Computer fundamentals and Digital Logic lab	1	30
SEC – 1	Theory	Data visualization using spreadsheet	3	45
	Practical	Data visualization using spreadsheet Lab	1	30

CMSA- Theory: Computer Fundamentals and Digital Logic
Core Course, Theory, Semester – 1, Credits - 03, Contact hours - 45.

Course description:

The course introduces the fundamental principles and concepts of digital logic, which form the foundation of digital systems and computer architecture. Students will learn about Boolean algebra, logic gates, combinational and sequential circuits, and the design and analysis of digital systems.

Course Objectives:

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

1. Understanding of Computer fundamentals, generations, classification of computers and brief understanding of languages used.
2. Understand the principles and terminology of digital logic.
3. Analyze and simplify Boolean expressions using Boolean algebra.
4. Design and implement combinational logic circuits using logic gates.
5. Design and analyze sequential logic circuits, including flip-flops and registers.
6. Apply digital logic concepts to solve practical problems.
7. Utilizing discrete logic gates and integrated circuits on breadboards for the design of digital circuits to enhance hands-on experience and practical understanding.

Computer Fundamentals	
Central Processing Unit (CPU), Primary memory and Secondary Storage devices, I/O devices, generation and classification of Computers: Super, Mainframe, Mini and Personal Computer, System and Application Software, basic concepts on machine, assembly and high level language.	2 hours
Number Systems	
Weighted and Non - Weighted Codes, Positional, Binary, Octal, Hexadecimal, Binary Coded Decimal (BCD), Gray Codes, Alphanumeric codes, ASCII, EBCDIC, Conversion of bases, signed arithmetic, 1's, 2's complement representation, Parity bits. **Single bit error detection and correcting codes:** Hamming Code. **Fixed and floating point Arithmetic**.	3 hours
Boolean Algebra	
Fundamentals of Boolean Expression: Definition of Switching Algebra, Basic properties of Switching Algebra, Huntington's Postulates, Basic logic gates (AND, OR, NOT), De-Morgan's Theorem, Universal Logic gates (NAND & NOR), XOR and others, Minterm, Maxterm, Minimization of Boolean Functions using Karnaugh-Map up to four (4)	4 hours

variables, two level and multilevel implementation using logic gates, simplification of logic expressions.	
Combinational Circuits	
Adder & Subtractor: Half adders (2-bit), half Subtractor (2-bit), Full Adder (3-bit), Full Subtractor (3-bit) realization using logic gates, Carry Look Ahead adders, BCD adder, 1's and 2's complement adders/subtractor unit using 4-bit parallel adders.	5 hours
Data Selector/Multiplexer: Realization of multiplexers (4 to 1 and 8 to 1) using logical gates, expansion (Cascading), realization of AND, OR and NOT using multiplexers, realization of different Boolean expressions (SOP) using multiplexers.	5 hours
Data Distributor: De-multiplexer, Cascading, realization of various functions.	2 hours
Encoders: Realization of simple and priority encoders using basic and universal logic gates.	2 hours
Chip Selector/Minterm Generator: Realization of decoders using logic gates, function realization, BCD Decoders, Seven Segment display and decoders, cascading.	3 hours
Parity bit, Code Converters and magnitude comparators: Parity bit generator/checker, Gray to binary code, binary to Gray code and Gray to Excess-3 code converter, 2 & 3 bit magnitude comparators.	2 hours
Sequential Circuits	
Latch & Flip-Flops: Basic Set/Reset (SR) Latch using NAND and NOR gates, Gated S-R latches, Gated D Latch, Gated J-K Latch, race around condition, Master-Slave J-K flip flop, negative and positive clock edge detector circuits, edge triggered SR, D, JK, and T flip flop, flip-flop Conversions.	5 hours
Registers: Serial Input Serial Output (SISO), Serial Input Parallel Output (SIPO), Parallel input Serial Output (PISO), Parallel Input Parallel Output (PIPO), Universal Shift Registers.	3 hours
Counters: **Asynchronous Counter** UP/DOWN Counters, Mod - N Counters, BCD Counter (Counter Construction using J-K and T Flip Flops).	4 hours
Synchronous Counter: UP/DOWN Counters, Mod-N Counters, Ring & Johnson Counters.	3 hours
Integrated Circuits (Qualitative Study): DTL, **TTL:** Concepts of Fan in & out, TTL NOT, TTL NAND & NOR, NMOS, PMOS, CMOS, **IC fabrication (Concepts only):** SSI, MSI, LSI, VLSI, ULSI.	2 hours

Core Course/DSE, CMSA- Practical: Computer Fundamentals and Digital Logic Lab, Semester – 1, Credits - 01, Contact hours - 30.

Combinational Circuits

1. Study and prove De-Morgan's Theorem.
2. Realization of Universal functions using NAND and NOR gates.
3. Implementation different functions (SOP, POS) using digital logic gates.
4. Implementation of half (2-bit) and full adder (3-bit) using basic (AND, OR and NOT) and Universal logic gates (NAND & NOR).
5. Design 4 to 1 multiplexer using basic or Universal logic gates and implement half and full adder/subtractor.
6. Design and implement half and full adder/subtractor and other functions using multiplexers 74151/74153 and other necessary logic gates.
7. Cascading of Multiplexers.
8. Design 2 to 4 decoder using basic or universal logic gates, study 74138 or 74139 and implement half and full Adder/Subtractor and other functions.
9. Design a display unit using Common anode or cathode seven segment display and decoders (7446/7447/7448)
10. Design and implement 4-input 3-output (one output as valid input indicator) priority encoder using basic (AND, OR & NOT) logic gates.
11. Design a parity generator and checker using basic logic gates.

Sequential Circuits

1. Realization of SR, D, JK Clocked/Gated, Level Triggered flip-flop using logic gates.
2. Master Slave flip-flop using discrete digital logic gates.
3. Conversion of flip-flops: D to JK, JK to D, JK to T, SR to JK, SR to D Flip-flop.
4. Design asynchronous counters MOD-n (upto 4 bits) UP/ DOWN.
5. Construction Synchronous UP/Down Counter (maximum 4 bits).

Note: The assignments listed below are illustrative examples and not an exhaustive list. They serve as a starting point to cover various aspects of the course.

Recommended Books

1. Digital Fundamentals, 11th Edition by Pearson Eleventh Edition, Thomas L. Floyd.
2. Digital Logic and Computer Design, M Morris Mano, Pearson.
3. Digital Electronics, Principles, Devices and Applications, Anil K. Maini, John Wiley & sons.
4. Digital Principles and Applications, Leach, Malvino, Saha, Tata McGraw Hill Education.
5. Digital Systems, Principal and Applications, Widmer, Moss and Tocci, Pearson.

CMSA- Theory: Data visualization using spreadsheet
SEC-1, Theory, Semester – 1, Credits - 03, Contact hours - 45.

Course Description

This Skill Enhancement Course (SEC) provides a comprehensive introduction to essential concepts and practical skills required for proficient utilization of spreadsheets. Students will gain proficiency in data management, visualization, analysis, and presentation using a widely-used open source spreadsheet software application such as Open Office, Libre Office, or Google Spreadsheets. Through this course, students will acquire the ability to proficiently create, format, manipulate, and analyze data within spreadsheets to meet a diverse range of needs.

Course Objectives

1. The purpose and potential applications of spreadsheets.
2. Create, format, and modify spreadsheets.
3. Use of formulas, functions, and calculations to perform data visualization.
4. Understanding and utilization of advanced spreadsheet features such as data validation, conditional formatting, and pivot tables.
5. Design visually appealing charts and graphs to represent data.
6. Collaborate and share spreadsheets with others.
7. Apply spreadsheet skills to real-world scenarios and problem-solving.
8. Role of spreadsheets in data analysis.
9. Import, clean, and transform data for analysis.
10. Applicability of statistical and mathematical functions for data visualization.
11. Advanced features and tools for data visualization.
12. Perform exploratory data analysis and identify patterns and trends.
13. Create informative reports and summaries based on data analysis.
14. Apply data analysis techniques to real-world problems.

Description	Teaching hours
Introduction to Spreadsheets Spreadsheets and their applications, overview of spreadsheet software (e.g., Open office, Google Sheets, Excel), creating workbooks, modifying workbook, modifying workbook, zooming in on a worksheet, arranging multiple workbook windows, adding buttons to the quick access toolbar, customizing the ribbon, maximizing usable space in the program window navigating the spreadsheet interface, entering and editing data in cells saving, opening, and closing spreadsheet files.	2 hours
Working with Data and Tables Entering and revising data, moving data within a workbook, finding and replacing data, correcting and expanding upon worksheet data, defining tables.	2 hours
Performing Calculations on Data Naming groups of data, creating formulas to calculate values (e.g., SUM, AVERAGE, COUNT), summarizing data that meets specific conditions (e.g., AVERAGEIF, COUNTA, COUNTBLANK, COUNTIFS, SUMIF, IFERROR etc), finding and correcting errors in calculations.	2 hours

Changing Workbook Appearance Formatting Cells, defining styles, workbook themes and table styles, making numbers easier to read, changing the appearance of data based on its value, adding images to worksheets.	2 hours
Data Analysis and Manipulation Limiting data appearance on screen, working with text functions for data cleaning, Splitting and combining data, Data normalization and standardization, working with ranges and named ranges, conditional formatting, data validation and error checking, using logical functions (e.g., IF, AND, OR), sorting and filtering data.	4 hours
Advanced Spreadsheet Features Creating and managing tables, creating and modifying pivot tables, using lookup functions (e.g., VLOOKUP, HLOOKUP), working with charts and graphs, importing and exporting data.	4 hours
Statistical Functions and Analysis Descriptive statistics (mean, median, mode, variance, etc.), Calculating measures of central tendency and dispersion, Correlation and regression analysis, Hypothesis testing and confidence intervals, Analysis of variance (ANOVA).	5 hours
Pivot Tables and Data Aggregation Creating pivot tables for data summarization, grouping and aggregating data by categories, Applying filters and slicers to pivot tables, calculating calculated fields and items.	4 hours
Advanced Data Visualization Creating charts and graphs for data representation, Customizing chart elements (titles, axes, legends), Using sparklines and data bars for visual analysis, Creating interactive dashboards, Incorporating trendlines and forecasting in charts.	5 hours
Exploratory Data Analysis Identifying patterns and outliers in data, Creating histograms and box plots, Using conditional formatting for data visualization, Data segmentation and drill-down analysis, Applying data validation rules for data integrity.	4 hours
Advanced Analysis Techniques Using goal seek and solver for optimization problems, Performing "what-if" analysis with data tables, Simulating data using random number functions, Monte Carlo simulation for risk analysis, creating scenario analysis models.	4 hours
Reporting and Presentation of Results Designing informative reports and summaries, creating interactive dashboards for data presentation, data visualization best practices, documenting data analysis processes presenting findings to stakeholders.	3 hours
Collaboration and Sharing Protecting worksheets and workbooks, sharing spreadsheets with others, tracking changes and commenting, collaborating in real-time, using version history and revision control.	4 hours

CMSA- Practical - Data visualization using spreadsheet
SEC, Laboratory, Semester – 1, Credits - 01, Contact hours - 30.

1. Create a personal budget spreadsheet that tracks income, expenses, and savings over a specified period. Use formulas and functions to calculate totals, percentages, and remaining balances.

2. A dataset containing sales data for a company to be provided. A spreadsheet to be created that calculates monthly sales totals, identifies top-selling products, and visualizes sales trends using line charts or bar graphs. Use conditional formatting to highlight exceptional sales performances.

3. Design a grade book spreadsheet that calculates students' final grades based on assignments, exams, and participation. Incorporate weighted grading systems, formulas for calculating averages, and conditional formatting to indicate performance levels. Generate reports to track individual student progress.

4. Create a spreadsheet that tracks inventory for a hypothetical business. Include columns for item names, quantities, prices, and total values. Use formulas to automatically update inventory totals, generate alerts for low stock, and create visualizations to represent inventory levels over time.

5. Loan parameters, such as principal amount, interest rate, and loan term to be provided. Create a spreadsheet that calculates monthly loan payments, remaining balances, and interest paid over time using appropriate formulas. Create a chart to visualize the loan's repayment schedule.

6. Dataset to be provided which will allow various data analysis tasks using spreadsheets. Calculation of summary statistics, sorting and filtering data, creating pivot tables for deeper insights, and generation of charts or graphs to visualize patterns or trends within the data.

7. A dataset to be selected (e.g., stock prices, weather data, population growth, etc) and create line charts or area charts to visualize trends over time. Students should choose appropriate chart types, label axes, and add titles and legends to make the visualization clear and informative.

8. A dataset containing information about different products or variables (e.g., sales data, customer satisfaction ratings) to be provided and following to be done; create bar charts or column charts to compare the performance or rankings of the items. Use color, data labels, and chart elements to enhance the visual comparison.

9. A dataset containing time-series data for multiple variables (e.g., monthly sales data for different products) to be provided and the following task to be performed; to create a combo chart with lines and columns to compare the trends of the variables and identify any relationships or patterns.

10. To create a unique visualization using advanced spreadsheet features and tools. For example, an experiment with sparklines, radar charts, or treemaps to represent specific types of data or explore innovative ways to visualize information.

Note: The assignments listed below are illustrative examples and not an exhaustive list. They serve as a starting point to cover various aspects of the course.

Recommended Text books

1. Data Analysis and Decision Making with Microsoft Excel" by S. Christian Albright.
2. Microsoft Excel 2019 Data Analysis and Business Modeling, Sixth Edition, Wayne L. Winston, Pearson education.
3. Excel 2019 Bible, Michael Alexander, 11th edition, Wiley.
4. Microsoft Office 2019 for Dummies, Wallace Wang, Wiley.

Recommended Application Software

1. Google Spreadsheets
2. Libre/Open Office
3. Excel sptreadsheets

Semester - II				
Paper	**Paper type**	**Paper name**	**Credit**	**Contact hours**
DSC/CC-2	Theory	Problem Solving using C	3	45
	Practical	Problem Solving using C Lab	1	30
SEC – 2	Theory	Web Development	3	45
	Practical	Web Development Lab	1	30

CMSA- Theory: Problem Solving using C
DSC/CC-2, Theory, Semester – 2, Credits - 03, Contact hours - 45.

Objective of the Course

The objectives of this course are to make the student understand programming language, programming, concepts of Loops, reading a set of Data, stepwise refinement, Functions, Control structure, Arrays. After completion of this course the student is expected to analyze the real life problem and write a program in 'C' language to solve the problem. The main emphasis of the course will be on problem solving aspect i.e. developing proper algorithms.

After completion of the course the student will be able to;

1. Develop efficient algorithms for solving a problem.
2. Use the various constructs of a programming language viz. conditional, iteration and recursion.
3. Implement the algorithms in "C" language.
4. Use simple data structures like arrays, stacks and linked list in solving problems.
5. Handling File in "C".

Outline of Course

S. No.	Topic	Minimum number of hours
1	Introduction to Programming	03
2	Algorithm/ Flowchart for Problem Solving	06
3	Introduction to 'C' Language	02
4	Conditional Statements and Loops	05
5	Arrays	05
6	Functions	04
7	Storage Classes	02
8	Structures and Unions	05
9	Pointers	05
10	Self-Referential Structures and Linked Lists	04
11	File Processing	02
12	Organizing C Projects	02
Lectures = 45 Practical/tutorials = 30, Total = 75		

Detailed Syllabus

Description	Teaching hours
Introduction to Programming The Basic Model of Computation, Algorithms, Flow-charts, Programming Languages, Compiler, Interpreter, Assembler, Linker and Loader, Testing and Debugging, Documentation.	03 hours
Algorithms/ Flowchart for Problem Solving Exchanging values of two variables, summation of a set of numbers, decimal base to binary base conversion, reversing digits of an integer, GCD (Greatest Common Division) of two numbers, test whether a number is prime, organize numbers in ascending order using bubble sort, find integer square root of a number, factorial computation, Fibonacci sequence, evaluate 'sin x' as sum of a series, reverse order of elements of an array, find largest number in an array, print elements of upper triangular matrix, multiplication of two matrices, evaluate a Polynomial.	06 hours
Introduction to 'C' Language Character set, variables, identifiers and their nomenclature, built-in data types, variable declaration, arithmetic operators and expressions, constants and literals, simple assignment statement, basic input/output statement, simple 'C' programs.	02 hours
Conditional Statements and Loops Decision making within a program, conditions, relational operators, logical connectives, if statement, if-else statement, **Loops:** while loop, do while, for loop, nested structure, infinite loops, switch-case, break, continue statement, structured programming.	05 hours
Arrays One dimensional arrays: Array manipulation; Searching, Insertion, deletion of an element from an array; finding the largest/smallest element in an array; two dimensional arrays, addition/multiplication of two matrices, Transpose of a square matrix; null terminated strings as array of characters, standard library string functions.	05 hours
Functions Top-down approach of problem solving, modular programming and functions, standard library of C functions, Prototype of a function: Formal parameter list, return type, function call, block structure, passing arguments to a function: call by reference, call by value, Recursive functions, arrays as function arguments.	04 hours
Storage Classes Scope and extent, Storage Classes in a single source file: auto, extern and static, register, Storage Classes in a multiple source files: extern and static	02 hours
Structures and Unions Structure variables, initialization, structure assignment, nested structure, structures and functions, structures and arrays: arrays of structures, structures containing arrays, unions	05 hours
Pointers Address operators, pointer type declaration, pointer assignment, pointer initialization, pointer arithmetic, functions and pointers, Array of Pointers, pointer to an array, pointers and structures, dynamic memory allocation.	05 hours

Self-Referential Structures and Linked Lists Creation of a singly connected linked list, Traversing a linked list, Insertion into a linked list, Deletion from a linked list	04 hours
File Processing Concept of Files, File opening in various modes and closing of a file, Reading from a file, Writing onto a file, Appending to a file.	02 hours
Organizing C projects, working with multiple source directories, makefiles.	02 hours

Recommended books main reading

1. Byron S Gottfried "Programming with C" Second edition, Tata McGraw Hill, 2007 (Paperback)
2. R.G. Dromey, "How to solve it by Computer", Pearson Education, 2008.
3. Kanetkar Y, "Let us C", BPB Publications, 2007.
4. Hanly J R & Koffman E.B, "Problem Solving and Program design in C", Pearson Education, 2009.
5. Kashi Nath Dey and Samir Bandyopadhayay "C Programming Essentials" Pearson India Education, 2010.

Supplementary reading.

1. E. Balagurusamy, "Programming with ANSI-C", Fourth Edition,2008, Tata McGraw Hill.
2. Venugopal K. R and Prasad S. R, "Mastering 'C'", Third Edition, 2008, Tata McGraw Hill.
3. B.W. Kernighan & D. M. Ritchie, "The C Programming Language", Second Edition, 2001, Pearson education.
4. ISRD Group, "Programming and Problem-Solving Using C", Tata McGraw Hill,2008.
5. Pradip Dey, Manas Ghosh, "Programming in C", Oxford University Press, 2007.

CMSA- Practical: Problem Solving using C
DSC/CC-2, Practical, Semester – 2, Credits - 01, Contact hours - 30.

Algorithms / Flowchart (Sample and simple assignments)

1. Design a flowchart/ Algorithm for a basic calculator that accepts two numbers and an operator (+, -, *, /) as input from the user and performs the corresponding operations, and displaying/print the result.
2. Create a flowchart/Algorithm that converts a temperature from Celsius to Fahrenheit or vice versa based on user input.
3. Design a flowchart/Algorithm that calculates the factorial of a given positive integer provided by the user.
4. Create a flowchart/Algorithm that finds and displays the largest number among three input numbers given by the user.
5. Design a flowchart/Algorithm to implement the linear search algorithm to find a specific element in an array of integers. The array and the element to search for should be taken as user input.
6. Create a flowchart/Algorithm that calculates the area and perimeter/circumference of different shapes (e.g., circle, rectangle, triangle) based on user input for dimensions.
7. Design a flowchart/Algorithm that checks whether a given input string is a palindrome or not.

Introduction to 'C' Language (Assignments/examples related to simple C program.)

8. Write a program in C to read two numbers and produce the sum and product of those numbers and show the result separately.
9. Write a program in C to read two numbers and print the greater number, if both the numbers are same them print "EQUAL".
10. Write a program in C multiple numbers say n and print the greatest and the third greatest.
11. Write a program in C to read n numbers and print the even/odd numbers up to n.
12. Write a program in C to read a number and print the sum of n natural numbers.
13. Write a program in C to read a number n and print factor of n.
14. Write a program in C to read a number n and print first 10 multiples of n.
15. Write a program in C to read a number n and print if n is "PRIME" or "COMPOSITE".
16. Write a program in C to calculate the average of a set of N numbers.
17. Write a program in C convert the temperature given in Celsius to Fahrenheit or vice-versa.
18. Write a program in C to determine and print the sum of the following harmonic series for a given value of n: 1+1/2+1/3+……..1/n.
19. Write a program in C that reads a floating-point number and then displays the right most digits of integral part of the number.
20. Write a program in C to accept the length and breadth in meters and calculate the area and perimeter and also determine if it is a rectangle or a square based on the inputs given.
21. Write a program in C to accept an input and determine if the input entered is a number or alphabet or a special character.
22. Write a program in C to accept a word and then print the reverse case that is lower to upper or upper to lower case.
23. Write an interactive program in C which will demonstrate the process of division/multiplication, the user should be asked to enter two-digit numbers.

Conditional Statements and Loops (simple examples)

24. Write a program in C to read a number n and print n terms of the Fibonacci series.
25. Write a program in C to read a number n and print a single digit answer showing sum of the digits of n. (example – input 8626, expected output – 4, explanation 8+6+2+6 = 22, 2+2 = 4).
26. Write a program in C to read a number n and print all the prime numbers up to n.
27. Write a program in C to read a number n and print the following pattern (input = 5, expected output
    1
    12
    123
    1234
    12345).
28. Write a program in C to check if the given number is the Armstrong number or not (e.g 153 = 13+53+33).
29. Write a program in C to check the type of the given triangle whether it is equilateral, isosceles or scalene.

Arrays (examples of few simple programs)

30. Write a program in C to read a string and store it into a character array. Check whether the string is a palindrome or not and display accordingly.

31. Write a program in C to read a list of numbers stored in an integer array and while saving them arrange in ascending order.
32. Write a program in C to read two matrices and perform addition.
33. Write a program in C to read two matrix and check their compatibility for multiplication, if compatible then find product and print it.
34. Write a program in C to read a string and print the triangular pattern using the string.

Functions

35. Write a program in C to print all the Armstrong number from 1 to 500.
36. Write a function ***convert ()*** that returns a weight in Kg after being given a weight in pounds.
37. Write a function to find all perfect numbers from 1 to 100 (perfect numbers are positive integers where the sum of perfect divisor is the number itself, e.g., 6 = 1+2+3).
38. Write a function power () to find base raise to power [$\mathbf{base}^{\mathbf{power}}$].
39. Write a program in C to find solution of a quadratic equation [$x = \frac{-b \pm \sqrt{b^2-4a}}{2a}$] where values a, b and c to be accepted from the user as input.
40. Accept inputs from the user and echo it on to the screen in normal as well as in reverse using void recursive function.
41. Accept any number from the user and calculate the factorial of the number using recursion
42. Accept numbers n and print the odd/even numbers up to n using recursive function.
43. Write a program in C in compute the cubes of all numbers from 10 to 20.
44. Write a program in C to find the GCD of a number.
45. Write a program in C to generate all combinations of 1, 2, 3, 4 using recursion, e.g.,1234, 2341….. etc.

Storage Classes

46. Write a program in C to accept a number and find the factorial of the number demonstrating use of automatic variables.
47. Write a program in C to accept two numbers and find the sum of the number demonstrating use of external variables.
48. Write a program in C to accept two numbers and find the sum of the number demonstrating use of global variables.
49. Write a program in C to illustrate the use of static variables.
50. Write a program in C to accept numbers till a negative number is entered and calculate the sum of a list of numbers read using static variable.
51. Write a program in C to sum integers and use static variable to store the cumulative sum.

Pointers

52. Write a program in C to swap two numbers of n length.
53. Write a program in C for swapping numbers using functions.
54. Write a program in C to illustrate the Call by Value and Call by reference a rule in C programming.
55. Write a program in C to use a double dimensional array and print each cells value and address.
56. Write a program in C to show the use of Array, declared at compilation time (static manner) to read 10 numbers and display them.
57. Write a program in C to show the use of Array, declared dynamically to read 10 numbers and display them.
58. Write a program in C to read a string in a dynamic array and determine whether it is palindrome or not.

Structures and Unions

59. Write a program in C to read the data of a student, store it in a structure and display it.
60. Write a program in C to read the data of many students, store it in a structure and display the student's data and average percentage of the class.
61. Write a program in C to accept two dates from the user, validate both of them and check if they are different dates.
62. Write a program in C to accept students' data from the user. Check if the student stream is science, commerce or arts. If the stream is arts, then print the class of students. If the stream is science, then print the grade and if the stream is commerce, then print the percentage.

Files

63. Write a program in C showing the technique of opening and closing a file say **result.dat** and writing a list of numbers and its square into the file.
64. Write some texts into a file, reopen the file in read mode and reproduce the text on the monitor (use of putc() and fputc()).
65. Write a few numbers in the file created earlier. Reopen it in Read mode, write odd numbers in one file and even number in another file (use the **getw** and **putw** functions).
66. Write programs to demonstrate the use of getc(), fgetc() and ungetc().
67. Write programs to demonstrate the use of String I/O, Formatted I/O and End of file eof() and feof().

Recommended assignment content/structure

- Objective
- Algorithm/Flowchart
- Code
- Result
- Conclusion

Platform/Compiler

- GCC

Note: The assignments listed below are illustrative examples and not an exhaustive list. They serve as a starting point to cover various aspects of the course.

CMSA- Theory: Web development
SEC, Theory, Semester – 2, Credits - 03, Contact hours - 45.

Course Description

This course provides an introduction to web development using HTML (Hypertext Markup Language) and CSS (Cascading Style Sheets). Students will learn the core concepts and practical skills needed to create and style web pages. The course covers the fundamentals of HTML structure, CSS styling properties, and responsive web design principles.

Course Objectives

1. Understanding the basics of web development and the role of HTML and CSS.
2. Create well-structured HTML documents using proper tags and elements.

3. Apply CSS to style web pages, including layout, typography, colors, and images.
4. Implement responsive design techniques to ensure optimal display on different devices.
5. Incorporate multimedia elements, such as images, videos, and audio, into web pages.
6. Understand best practices for organizing and maintaining code in web development projects.
7. Develop and deploy a basic website using HTML and CSS.

Description	Teaching hours
Introduction to Web development Overview of web technologies and the role of HTML and CSS, understanding the structure of a web page, introduction to web browsers and developer tools.	3 hours
HTML Fundamentals Introduction to HTML tags and elements, creating headings, paragraphs, lists, and links, working with images and multimedia content, creating forms for user input.	3 hours
CSS basics Introduction to CSS and its role in web page styling, selectors, properties, and values, applying inline, internal, and external style sheets, formatting text, backgrounds, and borders.	3 hours
CSS Layout and box model Understanding the box model and its impact on layout, working with margins, padding, and borders, positioning elements using floats, positioning properties, and flexbox, creating responsive layouts with media queries.	3 hours
Typography and colors Styling text with fonts, sizes, weights, and styles, formatting text using CSS properties, understanding color models and applying colors to elements.	4 hours
Images and multimedia Working with images: sizing, aligning, and optimizing, incorporating videos and audio into web pages, implementing responsive images and media.	4 hours
CSS Selectors and specificity Understanding CSS selectors and specificity, applying styles to specific elements and classes, using pseudo-classes and pseudo-elements.	5 hours
Responsive Web design Introduction to responsive design principles, creating fluid layouts using CSS media queries, adapting web pages for different screen sizes and devices.	4 hours
CSS Frameworks and libraries Overview of popular CSS frameworks (e.g., Bootstrap, Foundation), using pre-built CSS components and grids, customizing and integrating CSS frameworks into web projects.	5 hours
Web development best practices Organizing and structuring code files and directories, validating HTML and CSS code, optimizing web pages for performance, introduction to version control with Git.	3 hours
Building and deploying a website Planning and designing a basic website structure, Implementing HTML and CSS to create the website, testing and debugging the website across different browsers, deploying the website to a local host/web server	6 hours

CMSA- Web development
SEC, Laboratory, Semester – 2, Credits - 01, Contact hours - 30.

1. Creating a personal portfolio website using HTML and CSS. There should be sections for an about me, projects, skills, and contact information's. Using CSS to style the layout, typography, and colors to create a visually appealing and professional-looking portfolio.

2. To design a responsive website that adapts to different screen sizes. They should create a layout that adjusts fluidly using CSS media queries and responsive design techniques.

3. To create a product landing page for a fictional product or an existing one. HTML to be used to structure the page and CSS to style the layout, typography, buttons, and images. Main focus to be on creating an engaging page that effectively showcases the chosen product.

4. To incorporate CSS animation effects into a web page. Use CSS transitions, transforms, and keyframe animations to add interactive and engaging elements to the website. Create animations for hover effects, scrolling effects, image sliders, or menu transitions.

5. Redesign an existing website using HTML and CSS. Analyze the original design and propose improvements to the layout, typography, color scheme, and overall user experience.

6. Create a webpage layout using CSS Flexbox or CSS Grid. Design a responsive layout that organizes content in a visually appealing way. Experiment can be performed with different grid or flexbox properties to create flexible and responsive designs.

7. To design and style an interactive form using HTML and CSS. They should incorporate various form elements such as text inputs, checkboxes, radio buttons, and select dropdowns. Apply CSS styling to improve the form's visual appearance and user experience.

Note: The assignments listed below are illustrative examples and not an exhaustive list. They serve as a starting point to cover various aspects of the course.

Suggested Readings.

1. Mastering HTML, CSS & Java Script Web Publishing, Laura Lemay, Rafe Colburn, Jennifer Kyrnin, BPB Publication.
2. Web designing and development, Satish Jain, BPB Publications.
3. HTML & CSS: The complete reference, Thomas Powell, McGraw Hill education.
4. Web programming with HTML5, CSS and JavaScript, John Dean, Joneas and Bartlet learning.
5. Sams Teach Yourself HTML, CSS, and JavaScript All in One, Julie C Meloni, Pearson Education.
6. Learning Web App development, Semmy Purewal, O'Reilly.

INTERDISCIPLINARY COURSE

Fundamentals of Computer Science and its applications 45 hrs

Course Outcome:

- Demonstrate the basic concepts of Computer science, such as Computer Architecture, Data representation, Algorithms, and Data structures.
- Write basic programs in a high-level programming language, such as Python.
- Explain how computers communicate with each other over a network.
- Explain how artificial intelligence is used in real-world applications.
- Use ICT tools to create documents, spreadsheets, and presentations.

Detailed Syllabus

- Introduction to computers and computing 08 hr.

History of computing and the different types of computers that are available today, Generations of computers, Basic Building blocks (CPU, Memory, I/O Devices), types of computer (Mainframe, Desktop, Laptop, System on Chip). Classification of Software – System and Application Software, Basic Security Anti-Virus.

- Data representation and number systems 04 hr

Concept of binary code, ASCII and how it is used to represent data in computers, How different number systems work

- Algorithms and data structures 06 hr

Basic concepts of algorithms and data structures: Common algorithms and data structures, such as sorting algorithms and linked lists.

- Office suite 08 hr

  Word processors, Spreadsheets, and Presentation

- Programming languages 08 hr

Basic concepts of programming languages: types of programming languages , machine language, assembly language, high level language, Introduction to writing basic programs in Python (Finding prime numbers, finding GCD of two numbers etc,)

- Networking 05 hr

Basic concept of networking and how computers communicate with each other, LAN, WAN, Introduction to the concept of the internet and how it works. Mobile communication

- Artificial intelligence 05 hr

Basic concept of artificial intelligence and how it is used in computers. Introduction to Machine Learning, Preliminary concept of Big Data, Recommendation System, Conversation Agents like ChatGPT, Prompt Engineering

- Information and Communications (ICT) Tools 01 hr

Importance of ICT tools, different types of ICT tools and their uses

Recommended Books:

1. Computer Science: An Interdisciplinary Approach, Robert Sedgewick (Author), Kevin Wayne (Author)
2. Introduction To Computer Science, Anita Goel Pearson India

Structure of Core Courses in Computer Science for Three-year MDC

Semester	Course / Paper Code	Course Name
1	CMS- MD- CC1- 1- Th/ P	Computer Fundamentals and Digital Logic
2	CMS- MD- CC2- 2- Th / P	Problem Solving using C

Structure of Minor Courses in Computer Science for MDC

Semester	Course / Paper Code	Course Name
3	CMS- MD- MC1- 3- Th / P	Computer Fundamentals and Digital Logic
4	CMS- MD- MC2- 4- Th / P	Problem Solving using C

Structure of Skill Enhancement Courses in Computer Science for MDC

Semester	Course / Paper Code	Course Name
Semester 1/2/3	CMS- MD- SEC2- 2- Th / P	Web Development (HTML / PHP)

Structure of Skill Interdisciplinary Courses in Computer Science for MDC

Semester	Course / Paper Code	Course Name
Semester 1/2/3	CMS- MD- IDC2- 2- Th / P	Fundamentals of Computer Science and their Applications