UNIVERSITY OF CALCUTTA

Notification No. CSR/82/2024

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 the exercise of her powers under 9(6) of the said Act, the Vice-Chancellor has, by an order dated 20.092024 approved the new revised syllabus (Semester-1 to 4) of Computer Science (4-year Honours & Honours with Research and Three-year MDC & Minor), under this University, as laid down in the accompanying pamphlet.

The new CSR shall be applicable for semester-1 to 4 and take effect from the Odd semester Examinations, 2024 and onwards.

26/9/2024

SENATE HOUSE

Kolkata-700073

26.09.2024

Prof.(Dr.) Debasis Das

Registrar

University of Calcutta

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

(2024)

Semester – I, II, III & IV

Curriculum Structure for a 4-Year Bachelor of Science or Bachelor of Arts (Honours or Honours with Research) Degree under CCF as per CSR-05/2023

COURSE STRUCTURE-CCF, 2022

	DSCC/ Core (Major)	Minor (m1 & m2)	IDC	AEC	SEC	CVAC	Summer Internship	Dissertation/ Research work	Total Credit
Semester	22x4= 88	8x4= 32	3x3= 9	4x2= 8	3x4= 12	4x2= 8	1x3=3	(1x4= 4)+ (1x8= 8) = 12	172
1	1x4= 4 3TH+1P/TU	1x4= 4 (m1) 3TH+1P/TU	1x3= 3 2TH +1P/TU	1x2= 2 2TH +0P/TU	1x4= 4	2x2= 4			21
2	1x4= 4 3TH+1P/TU	1x4= 4 (m1) 3TH+1P/TU	1x3= 3 2TH +1P/TU	1x2= 2 2TH +0P/TU	1x4= 4	2x2= 4			21
3	2x4= 8 2x(3TH+ 1P/TU)	1x4= 4 (m2) 3TH+1P/TU	1x3= 3 2TH +1P/TU	1x2= 2 2TH +0P/TU	1x4= 4				21
4	4x4= 16 4x(3TH+ 1P/TU)	1x4= 4 (m2) 3TH+1P/TU		1x2= 2 2TH +0P/TU					22
5	4x4= 16 4x(3TH+ 1P/TU)	m1+m2 2x4= 8 2x(3TH+ 1P/TU)							24
6	3x4= 12 3x(3TH+ 1P/TU)	2x4= 8 m1+m2 2x(3TH+ 1P/TU)							20
7	4x4= 16 4x(3TH+1P/ TU)							1x4*	20
8	3x4= 12 3x(3TH+1P/ TU)							1x8 *	20
Credits	**22x4= 88**	**8x4= 32**	**3x3= 9**	**4x2= 8**	**3x4= 12**	**4x2= 8**		**(1x4)+(1X8)= 12**	**169+3= 172**
Marks	**22x100= 2200**	**8x100=800**	**3x75= 225**	**4x50= 200**	**3x100= 300**	**4x50 = 200**		**1x100+1x200= 300**	**Total Marks =4300**

Marks= 25 marks per credit.

*Students who will not pursue Dissertation/ Research work then the candidate will have to study additional 1 DSC/Core paper o f 4 credits in the 7th Semester & 2 DSC/ Core Papers of 4 Credits each in the 8th Semester.

Minor courses will come from two subjects of same broad discipline as Major (m1, m2).

Total credit=169+3 (for summer internship) = 172

Reference: CSR-05/2023

Curriculum Structure for a 4-Year Bachelor of Science (Honours or Honours with Research) Degree in Computer Science under CCF (Major discipline).

Semester	Core Subjects - (DSCC) Computer Science as Major Discipline		SEC
	Theory Paper Credits – 03 (75 marks)	**Practical paper Credits – 01 (25 marks)**	**Theory – 02 Credits Practical – 02 credits**
1	Computer fundamentals & Digital Logic	Digital Logic Circuit Lab	Data Visualization using spreadsheets
2	Problem Solving Using C	Problem Solving using C Lab	Web Development
3	Data Structures	Data Structures using C	Mobile App Development
	Computer Architecture & Organization.	Digital Computer Design Lab	
4	Computational Mathematics	Numerical Methods lab	Not applicable
	Microprocessor and Its Applications	Programming Microprocessor 8085	
	Operating System	Shell Programming	
	Object Oriented Programming	Programming in Java	
5	Design & Analysis of Algorithms	Graph algorithms Lab using C++	Not applicable
	Data Communication and Networking	Networking Lab	
	Theory of Computation	Tutorial	
	Database Management System (DBMS)	RDBMS Lab	
6	Software Engineering	Tutorial	Not applicable
	Programming in Python	Programming in Python	
	Linear Algebra & Statistical Methods	Linear Algebra & Statistical Methods using Python	
7	Compiler Design	Tutorial	Not applicable
	Machine Learning	Machine learning Lab using Python	
	Computer Graphics	Computer Graphics	
	Embedded Systems	Embedded Lab	
	Big Data Analytics / Research Project	Big Data Analytics Lab	
8	Digital Image Processing	Digital Image Processing using Python and Open CV.	Not applicable
	Cryptography	Cryptography using Python	
	Data Warehousing	Data Warehousing Lab	
	Mobile & Wireless Computing/ResearchProject	Mobile & Wireless Computing lab	
	Cloud Computing / ResearchProject	Cloud Computing Lab	

Note: For **Semesters 7 and 8**, the courses highlighted in yellow will be taken by students who either choose not to pursue Honours with Research or do not meet the eligibility criteria for Honours with Research. Please refer to notification number **CSR-05/2023** for more details.

Curriculum Structure for Computer Science as a Minor subject/discipline of study for students opting for a different major discipline in a 4-Year Bachelor of Science (Honours or Honours with Research) Degree.

Semester	Computer Science as Minor (M1) study/discipline		Computer Science as Minor (M2) study/discipline	
	Theory Paper Credits – 03 (75 marks)	**Practical paper Credits – 01 (25 marks)**	**Theory Paper Credits – 03 (75 marks)**	**Practical paper Credits – 01 (25 marks)**
1	Computer fundamentals & Digital Logic	Digital Logic Circuit Lab	Not applicable	Not applicable
2	Problem Solving Using C	Problem Solving using C Lab		
3	Not applicable	Not applicable	Computer fundamentals & Digital Logic	Digital Logic Circuit Lab
4	Not applicable	Not applicable	Problem Solving Using C	Problem Solving using C Lab
5	Data Structures	Data Structures using C	Data Structures	Data Structures using C
6	Operating System	Shell Programming	Operating System	Shell Programming
Credits	**4 x 4 = 16**		**4 x 4 = 16**	
Full marks	**Marks 4 x 100 = 400 (Minor – M1)**		**Marks 4 x 100 = 400 (Minor – M2)**	

Curriculum Structure for a 3-Years Bachelor of Science in Multidisciplinary Course (MDC) in Computer Science

COURSE STRUCTURE-MDC

	CC1	CC2	Minor	IDC	AEC	SEC	CVAC	Summer Internship	Total Credit
Semester	**8x4= 32**	**8x4= 32**	**6x4= 24**	**3x3=9**	**4x2= 8**	**3x4=12**	**4x2=8**	**1x3= 3**	**128**
1	1x4= 4 3TH+ 1P/TU	1x4= 4 3TH+ 1P/TU		1x3=3 2TH +1P/TU	1x2= 2 2TH +0P/TU	1x4= 4	2x2=4		21
2	1x4= 4 3TH+ 1P/TU	1x4= 4 3TH+ 1P/TU		1x3=3 2TH +1P/TU	1x2= 2 2TH +0P/TU	1x4= 4	2x2=4		21
3	1x4= 4 (3TH+ 1P/TU)	1x4= 4 3TH+ 1P/TU	1x4= 4 3TH+1P/ TU	1x3=3 2TH +1P/TU	1x2= 2 2TH +0P/TU	1x4= 4			21
4	2x4=8 4x(3TH+ 1P/TU)	2x4= 8 2x(3TH+ 1P/TU	1x4= 4 (3TH+1P/ TU)		1x2= 2 2TH +0P/TU				22
5	2x4= 8 2x(3TH+ 1P/TU)	1x4= 4 3TH+ 1P/TU	2x4= 8 2x(3TH+ 1P/TU						20
6	1x4= 4 (3TH+ 1P/TU)	2x4= 8 2x(3TH+ 1P/TU)	2x4= 8 2x(3TH+ 1P/TU)						20
Credits	**8x4= 32**	**8x4= 32**	**6x4= 24**	**3x3= 9**	**4x2= 8**	**3x4= 12**	**4x2= 8**	**1x3= 3**	**128**
Marks	**8x100= 800**	**8x100= 800**	**6x100= 600**	**3x75= 225**	**4x50= 200**	**3x100= 300**	**4x50= 200**	**3x25= 75**	**Total Marks =3200**

Marks= 25 marks per credit.

Total credit=125+3 (for summer internship) = 128
Summer Internship: As mentioned in clause no. 8 (G)

Curriculum Structure for a 3-Years Bachelor of Science in Multidisciplinary Course (MDC) and Computer Science as one of the subjects.

<table>
<tr><th>Semester</th><th>If taken as CC1</th><th>If taken as CC2</th><th>If taken as Minor</th><th>SEC</th><th>IDC</th></tr>
<tr><td>1</td><td>Computer fundamentals & Digital Logic, (Th-3/P-1)*</td><td>Computer fundamentals & Digital Logic, (Th-3/P-1)</td><td>Not Applicable</td><td rowspan="3">Web Development (Th-2/P-2)</td><td rowspan="3">Fundamentals of Computer Science and their Applications. (Th-2/P-1)</td></tr>
<tr><td>2</td><td>Problem Solving Using C (Th-3/P-1)</td><td>Problem Solving Using C (Th-3/P-1)</td><td>Not Applicable</td></tr>
<tr><td>3</td><td>Data Structures (Th-3/P-1)</td><td>Data Structures (Th-3/P-1)</td><td>Computer fundamentals & Digital Logic, (Th-3/P-1)</td></tr>
<tr><td rowspan="2">4</td><td>Operating System (Th-3/P-1)</td><td>Operating System (Th-3/P-1)</td><td rowspan="2">Problem Solving Using C (Th-3/P-1)</td><td rowspan="4">Not Applicable</td><td rowspan="4">Not Applicable</td></tr>
<tr><td>Object Oriented Programming (Th-3/P-1)</td><td>Object Oriented Programming (Th-3/P-1)</td></tr>
<tr><td rowspan="2">5</td><td>Database Management System (DBMS) (Th-3/P-1)</td><td rowspan="2">Database Management System (Th-3/P-1)</td><td>Data Structures (Th-3/P-1)</td></tr>
<tr><td>Data Communication and Networking (Th-3/P-1)</td><td>Operating System (Th-3/P-1)</td></tr>
<tr><td rowspan="2">6</td><td rowspan="2">Programming in Python (Th-3/P-1)</td><td>Data Communication and Networking (Th-3/P-1)</td><td>Object Oriented Programming (Th-3/P-1)</td></tr>
<tr><td>Programming in Python (Th-3/P-1)</td><td>Database Management System (DBMS) (Th-3/P-1)</td></tr>
<tr><td>Credit</td><td>Theory + Practical = 4 x 8</td><td>Theory + Practical = 4 x 8 = 32</td><td>Theory + Practical = 4 x 6 = 24</td><td>Theory + Practical = 4 x 3 = 12</td><td>Theory + Practical = 3 x 3 = 9</td></tr>
<tr><td>Marks</td><td>8 x 100 = 800</td><td>8 x 100 = 800</td><td>6 x 100 = 600</td><td>3 x 100 = 300</td><td>3 x 75 = 225</td></tr>
</table>

- **(Th – Theory, P – Practical), Theory -01 credit = 1 contact hour/week, Practical -01 credit = 2 contact hour/week**
- **1 Credit = 25 marks.**
- **Syllabus remains the same as prescribed semester-wise in 4 Year B.Sc (Honours or Honours with Research) in Computer Science.**

Computer and other hardware recommended for laboratory (Upgrade/New installation)

1. Minimum System requirement

Computer Hardware upgradation recommended

- **Processor:** Ryzen-3 (3200) series or Ryzen-5 (4600G/5600G) series with compatible motherboard.
  Or
- **Processor:** Intel i-3 10^{th} generation and above, i5 12^{th} generation and above with compatible mother board with integrated graphics.
- **Memory:** DDR-4/5 (3200), 8 GB (minimum recommended) or more
- **Operating System:**Window-10/11 (64 - bit), or Linux (Ubuntu latest version).
- **Open Office**
- Upgrade hard disk to SSD.

2. Hardware laboratory

Digital Circuit and Microprocessor 8085 lab

- +5V dc Regulated power supply
- Digital multimeter
- Integrated Circuits – 7400, 7402, 7404, 7408, 7410, 7411, 7420, 7432, 7442, 7447, 7446, 7474, 7476, 7483/74283, 7486, 7489/74189, 7490, 74112, 74138, 74147, 74151, 74153, 74157, 74194, 74244, 74373.
- LED, Jumper wires, Cutters.
- Resistors: 100 Ω, 220Ω, 330Ω, 470Ω, 560Ω, 1KΩ, 1.5KΩ, 2.2KΩ, 4.7KΩ, 10KΩ, 15KΩ, 22kΩ, 100KΩ.
- Semiconductor devices: 1N4007.
- Microprocessor 8085 Trainer Kit (Dynalog/NVIS/ALS).

Syllabus for 4-Year Bachelor of Science (Honours or Honours with Research) degree in Computer Science under CCF (Major discipline)/Minor and 3-year MDC course (wherever applicable) for Semesters I, II, III and IV.

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	2	30
	Practical	Data visualization using spreadsheet Lab	2	45
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	2	30
	Practical	Web Development Lab	2	45
Semester - III				
Paper	**Paper type**	**Paper name**	**Credit**	**Contact hours**
DSC/CC-3	Theory	Data Structure	3	45
	Practical	Data Structure Lab	1	30
DSC/CC-4	Theory	**Computer Architecture & Organization**	3	45
	Practical	**Computer Architecture & Organization** lab	1	30
SEC – 3	Theory	Mobile App Development	2	30
	Practical	Mobile App Development lab	2	45
Semester - IV				
Paper	**Paper type**	**Paper name**	**Credit**	**Contact hours**
DSC/CC-5	Theory	Computational Mathematics	3	45
	Practical	Computational Mathematics lab using C Lab	1	30
DSC/CC-6	Theory	Microprocessor and its Applications	3	45
	Practical	Microprocessor – 8085 Lab	1	30
DSC/CC-7	Theory	Operating System	3	45
	Practical	Operating System Lab	1	30
DSC/CC-8	Theory	Object Oriented Programming	3	45
	Practical	Object Oriented Programming 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	2	30
	Practical	Data visualization using spreadsheet Lab	2	45

CMSM- Theory: Computer Fundamentals and Digital Logic
DSC/CC-1/ -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) variables, two level and multilevel implementation using logic gates, simplification of logic expressions.	4 hours
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, CMSM**DSCC-1- 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.
6. Digital Circuits, Volume I & II, Ray Chaudhuri, Platinum Publishers.
7. Digital Circuits and Design, Salivahanan and Arivazhagan, 5th Edition, Oxford Higher Education publishers.

CMSM- Theory: Data visualization using spreadsheet
SEC-1, Theory, Semester – 1, Credits - 02, Contact hours - 30.

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 OpenOffice, LibreOffice, 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 windownavigating the spreadsheet interface, entering and editing data in cells saving, opening, and closing spreadsheet files.	1 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, IFERROretc), 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.	3 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.	3 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).	3 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.	2 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.	2 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.	3 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.	2 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.	2 hours

CMSM- Practical - Data visualization using spreadsheet
SEC-1, Laboratory, Semester – 1, Credits - 02, Contact hours - 45.

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 2019Data Analysis andBusiness 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	2	30
	Practical	Web Development Lab	2	45

CMSM- 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	06
7	Storage Classes	02
8	Structures and Unions	05
9	Pointers	06
10	File Processing	03
11	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) oftwo 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 array: 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.	06 hours
Storage Classes Scope and extent, Storage Classes in a single source file: auto, extern and static, register, Storage Classes in 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.	06 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.	03 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.

CMSM- 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 [$x = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$] 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.

CMSM- **Theory: Web development**

SEC-2, Theory, Semester – 2, Credits - 02, Contact hours - 30.

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	2 hours

structure of a web page, introduction to web browsers and developer tools.	
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.	2 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.	2 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.	2 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.	3 hours
Images and multimedia Working with images: sizing, aligning, and optimizing, incorporating videos and audio into web pages, implementing responsive images and media.	3 hours
CSS Selectors and specificity Understanding CSS selectors and specificity, applying styles to specific elements and classes, using pseudo-classes and pseudo-elements.	3 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.	3 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.	2 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.	2 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

CMSM- Web development

SEC-2, Laboratory, Semester – 2, Credits - 02, Contact hours - 45.

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: Thecompletereference, Thomas Powell, McGraw Hill education.
4. Web programming with HTML5, CSS and JavaScript, John Dean, Joneas and Bartlet learning.
5. SamsTeachYourself HTML, CSS, and JavaScript AllinOne, Julie C Meloni, Pearson Education.
6. Learning Web App development, Semmy Purewal, O'Reilly.

Semester - III				
Paper	**Paper type**	**Paper name**	**Credit**	**Contact hours**
DSC/CC-3	Theory	Data Structure	3	45
	Practical	Data Structure Lab	1	30
DSC/CC-4	Theory	**Computer Architecture & Organization**	3	45
	Practical	**Computer Architecture & Organization** lab	1	30
SEC – 3	Theory	Mobile App Development	2	30
	Practical	Mobile App Development lab	2	45

CMSM - Theory: Data Structure

DSC/Core Course-3, Theory, Semester – 3, Credits - 03, Contact hours - 45.

Introduction to Data Structure Abstract Data Type.	01 hour
Arrays Multi-dimensional Arrays, Sparse Matrices. Polynomial representation	02 hours
Linked Lists **Introduction to Linked Lists:**Definition and importance, Comparison with arrays, applications of linked lists. **Types of Linked Lists:** Singly, Circular and Doubly Lists, Polynomial representation, **Basic Operations:** Creation of a singly connected linked list, traversing a linked list, Insertion into a linked list, deletion from a linked list.	08 hours
Stacks Array and linked representation of stack, Prefix, Infix and Postfix expressions, utility and conversion of these expressions from one to another, evaluation of postfix and prefix expression using stack, applications of stack, limitations of Array representation of stack.	04 hours
Queues Array and Linked representation of Queue, Circular Queue, De-queue, Priority Queues.	04 hours
Recursion Developing Recursive Definition of Simple Problems and their implementation; Advantages and Limitations of Recursion; Understanding what goes behind Recursion (Internal Stack Implementation), Tail recursion.	04 hours
Trees Introduction to Tree as a data structure: Binary Trees (Recursive and Iterative Traversals), Binary Search Tree (Traversal, Insertion, Deletion and Searching), Threaded Binary Trees (Traversal and advantages).	12 hours
Searching and Sorting Linear Search, Binary Search, Comparison of Linear and Binary Search with respect to time complexity, Selection Sort, Bubble sort, Insertion Sort, Merge Sort, Quick sort, Heap sort, Shell Sort, Radix sort, Comparison of Sorting Techniques with respect to time complexity.	06 hours
Hashing Introduction to Hashing, Different hashing Techniques, Collision and resolving collision by Open Addressing, Closed Hashing, Separate Chaining, Choosing a Hash Function.	04 hours

CMSM - Practical: Data Structure Lab
DSC/Core Course-3, Practical, Semester – 3, Credits - 01, Contact hours - 30.
Platform/Compiler: GCC

Lab based on Data Structure theory except Threaded Binary Tree, Shell Sort, Radix Sort and hashing.

1. Write a program to search an element from a list. Give user the option to perform Linear or Binary search.
2. Write a program to sort a list of elements. Give user the option to perform sorting using Insertion sort, Bubble sort or Selection sort.
3. Implement Linked List. Include functions for insertion, deletion and search of a number, reverse the list and concatenate two linked lists.

4. Implement Doubly Linked List. Include functions for insertion, deletion and search of a number, reverse the list.
5. Implement Circular Linked List. Include functions for insertion, deletion and search of a number, reverse the list.
6. Perform Stack operations using Linked List implementation.
7. Perform Stack operations using Array implementation.
8. Perform Queue operations using Array and linked list implementation.
9. Create and perform different operations on Double-ended Queues using Linked List implementation.
10. Write a program to scan a polynomial using linked list and add two polynomials.
11. Write a program to create a Binary Search Tree and include following operations in tree:
    (a) Insertion (Recursive and Iterative Implementation).
    (b) Deletion.
    (c) Search a node in BST.
    (d) Display its preorder, postorder and inorder traversals recursively.
    (e) Display its preorder, postorder and inorder traversals Iteratively.
    (f) Display its level-by-level traversals.
    (g) Count the non-leaf nodes and leaf nodes.
    (h) Display height of tree.
    (i) Create a mirror image of tree.
12. Write a program to reverse the order of the elements in the stack using additional stack.
13. Write a program to reverse the order of the elements in the stack using additional Queue.

Note: These are only sample programs; more can be included related to the theory.

Text/ Reference Books

1. Fundamentals of Data Structures in C, Horowitz, Sahni, Anderson-Freed, University Press.
2. Data Structures: A Pseudocode approach with C, Gilberg and Forouzan, Cengage Learning.
3. Data Structure using C, E Balagurusamy, McGraw Hill.
4. Data Structures Using C and C++, Aaron M. Tanenbaum, M J. Augenstein, Langsam, PHI.
5. Classic Data Structures, Debasis Samanta, Second Edition, EEE, PHI.
6. Data Structures, Seymour Lipschutz, Schaum's Outlines, Tata McGraw Hill.
7. Data Structures Through C (A practical approach), G.S Baluja, Dhanpat Rai & Co.

CMSM - Theory: Computer Architecture & Organization

DSC/Core Course-4, Theory, Semester – 3, Credits - 03, Contact hours - 45.

Basic Structure of Computers (Qualitative discussion) Basic functional units, basic operational concept, bus structure, software, performance, multiprocessor and multicomputer, IAS Computer, Historical Perspectives.	2 hours
Register Transfer and Micro-operation Register transfer language, register transfer, bus and memory transfers, three state bus buffers, memory transfer, arithmetic and logical micro-operations, shift and arithmetic shifts.	4 hours
Basic Computer Organization and Design Stored program organization, computer registers, common bus system, timing and control, instruction cycle, fetch decode, Computer Instructions, register reference instructions, memory reference instruction, input-output and Interrupt, design of basic	5 hours

computer, design of accumulator logic.	
CPU Organization Arithmetic and Logic Unit (ALU) - Combinational ALU, 2'S complement subtraction unit, Booth's algorithm for multiplication, restoration division algorithm and hardware. General register organization, control word, accumulator based, register based, Stack type CPU organization.	6 hours
Control Unit Hardwired Control Unit (basic concept), Micro-programmed Control Unit: Control memory, address sequencing, conditional branching, mapping of instructions, subroutine.	6 hours
CPU Registers Program Counter, Stack Pointer Register, Memory Address Register, Instruction Register, Memory Buffer Register, Flag registers, Temporary Registers.	4 hours
Instructions. Operational code, operands, zero, one, two and three address instruction, instruction types, addressing modes, data transfer and manipulation instructions, Program control instructions.	5 hours
CISC and RISC processors Introduction, relative merits and De-merits.	1 hour
Computer Peripherals VDU, Keyboard, Mouse, Printer, Scanner (Qualitative approach).	3 hours
Input / Output Organization: Polling, Interrupts, subroutines, memory mapped I/O, I/O mapped IO, DMA, I/O bus and protocol, SCSI, PCI, USB, bus arbitration.	4 hours
Memory Primary memory: ROM, PROM, EPROM, EEPROM, Flash memory, SRAM, DRAM, **Cache Memory:** mapping functions, replacement algorithms, interleaving, hit and rate penalty, virtual memories, address translation, memory management requirements, **Secondary Storage:** Solid State drives (SSD), Magnetic hard disks, Optical disks, magnetic tape systems.	5 hours

Computer Organization Lab.

CMSM -DSC/Core Course-3, Practical, Credits: 01, Contact hours:30.

(1). Construct an Arithmetic Unit capable of performing 4-bit subtraction and Addition using 2's complement method. Use Parallel Adders and other necessary logic gates.
(2). Construct a 2-bit logical unit using logic gates capable of performing 2-bit, Bitwise ORing, ANDing, XORing and inversion
(3). Construct a 4-bit ALU unit which can perform the following operation;

Selection		Function
S_1	S_0	
0	0	Addition
0	1	Subtraction
1	0	XOR-ing
1	1	Complement

(4). Construct a 2-bit Carry Look Ahead (CLA) Adder using logic gates.
(5). Study and construct a 1-digit BCD/Decimal adder using parallel adders and other necessary logic gates.
(6). Construct a Binary Multiplier using basic logic gates.
(7). Subtraction with 1's complement method using parallel adders and logic gates(necessary).

(8). Construction of BCD Subtractor with 9'S complement method using parallel adders.
(9). Construction of BCD Subtractor with 10'S complement method using parallel adders.
(10). Binary magnitude comparators (up to 4 bits) using parallel adder and logic gates.
(11). Cascading of 4-bit parallel adder (7483/74283) to construct an 8-bit adder circuit.
(12). Construct a Serial in Serial out 2/4-bit register.
(13). Construct a 2-bit Universal Shift register.
(14). Construct a 2/4-bit ring counter using edge triggered D Flip-Flops.
(15). Construct a 4 - bit Johnson Counter.
(16). Horizontal and Vertical Cascading of Memory modules (7489/74189).
(17). Code converters using memory modules.

Text/Reference Books

1. Computer System Architecture, Morries Mano, Pearson.
2. Computer Organization & Architecture, Williams Stallings, Pearson.
3. Computer Organization, Hamacher, Vranesic and Zaky, McGraw Hill.
4. Computer Architecture and Organization, Govindrajalu, Tata McGraw Hill.
5. Computer Architecture and Organization, J P Hayes, Tata McGraw Hill.
6. Structured Computer Organization, Andrew S. Tanenbaum, Austin, Pearson.

Note: Laboratory work must be conducted using integrated circuits on a breadboard, along with other necessary devices and equipment.

CMSM - Theory: Mobile Application Development
SEC-3 Course, Theory, Semester – 3, Credits - 02, Contact hours - 30.

Mobile App development using Flutter and Dart

Introduction to Flutter Introducing Flutter, defining widgets and elements, understanding Widget lifecycle events, understanding the Widget tree and the element tree, Installing the Flutter SDK, **Android Setup:** Install Android Studio, Setup the Android Emulator.	02 hours
Creating Your First Flutter App Setting Up the Project, using hot reload, using themes to style your App, understanding Stateless and Stateful Widgets, using external packages.	02 hours
Learning Dart basic Purpose of DART and its use, Commenting code, Running the **main() entry Point**, referencing variables, declaring variables, using Operators, using flow statements, using functions, Import packages, using classes, Implementing Asynchronous Programming.	04 hours
Creating Starter Project Template Creating and Organizing Folders and Files, Structuring Widgets.	02 hours
Widget Tree Introduction to Widgets, Building the Full Widget Tree, Building a Shallow Widget Tree	02 hours

Using Common Widgets Using basic widgets, using Images and Icons, using decorators, using the Form Widget to validate text fields, checking orientation.	02 hours
User Interface (UI) Development **Animation:** Animated container, Crossfade, Opacity, controller. **Navigation:** Using navigator, Hero Animation, Bottom navigation bar, Bottom app bar, Tabbar and view. **Scrolling:** Card, list view, list tile, Gridview, using Stack. **Layout:** High level view of layout, Creating layout. **Interactivity:** Set up Gesture detector, Draggable and Dragtarget Widgets, Moving and Scaling, Dismissible Widget.	08 hours
Finalizing App development Understanding the JSON Format, Using Database Classes to Write, Read, and Serialize JSON, Formatting Dates, sorting a list of dates, Retrieving Data with the FutureBuilder, Building the Journal App, Adding the Journal Database Classes, Adding the Journal Entry Page, Finishing the Journal Home Page. **Adding Firebase and Firestore Backend** Introduction to Firebase and Cloud Firestore, Structuring and Data Modelling Cloud Firestore, Viewing Firebase Authentication Capabilities, Viewing Cloud Firestore Security Rules, Configuring the Firebase Project, adding a Cloud Firestore Database and Implementing Security, Building the Client Journal App, Adding Authentication and Cloud Firestore Packages to the Client App, Adding Basic Layout to the Client App, Adding Classes to the Client App.	08 hours

CMSA- Practical: Mobile App Development
SEC-3, Practical, Semester – 3, Credits - 02, Contact hours - 45.

Here are some practical assignments for mobile app development using Flutter:

1. Basic Flutter Mobile App
   - Create a simple Flutter app with a single screen that displays "Hello, Flutter!".
   - Add a button that changes the text to "Hello, World!" when pressed.

2. Personal Profile MobileApp
   - Develop an app that displays your personal profile, including your name, photo, and a brief bio.
   - Use different Flutter widgets such as `Container`, `Row`, `Column`, and `Text`.

3. Weather MobileApp
   - Develop a weather app that fetches and displays weather information for a given location.
   - Use an API like OpenWeatherMap or any other suitable and display the data using Flutter widgets.

4. Quiz MobileApp
   - Create a quiz app with multiple-choice questions.

- Show the user's score at the end of the quiz.
- Use `ListView` for displaying questions and options.

5. Photo Gallery Mobile App
   - Develop a photo gallery app that displays images from a user's device.
   - Implement features like viewing images in full screen, deleting, and sharing images.

Optional (App development for practice)

1. Simple Calculator
   - Build a basic calculator app that can perform addition, subtraction, multiplication, and division.
   - Use `TextField` for input and `RaisedButton` for operations.

2. Todo List App
   - Create a todo list app where users can add, edit, and delete tasks.
   - Use a `ListView` to display the list of tasks.

3. Recipe App
   - Create a recipe app that displays a list of recipes.
   - Each recipe should have a detail page with ingredients and instructions.
   - Implement navigation between the list and detail pages.

4. Notes App
   - Build a notes app where users can create, view, edit, and delete notes.
   - Use local storage (e.g., `SharedPreferences` or `sqflite`) to save the notes.

5. Expense Tracker
   - Develop an expense tracker app that allows users to log their expenses.
   - Display a summary of expenses by category and date.
   - Use charts to visualize spending patterns.

6. E-commerce App
   - Develop a simple e-commerce app with a product list and product detail pages.
   - Implement a shopping cart where users can add products and proceed to checkout.

7. Chat App
   - Build a basic chat application with a login screen and a chat screen.
   - Use a backend service like Firebase for real-time messaging.

8. Fitness Tracker
   - Create a fitness tracker app that logs workout activities.
   - Display statistics like total time spent, calories burned, and workout history.

9. Music Player
   - Create a music player app that can play audio files from the device.
   - Implement basic controls like play, pause, next, and previous.

10. Travel Guide App
   - Build a travel guide app that provides information about different travel destinations.
   - Include features like a map view, destination details, and user reviews.

Reference books and other resources

1. Flutter Complete Reference by Alberto Miola
2. Beginning Flutter: A Hands-On Guide to App Development by Marco L. Napoli
3. Flutter in Action by Eric Windmill
4. Flutter for Beginners by Thomas Bailey and Alessandro Biessek
5. Pragmatic Flutter by Priyanka Tyagi
6. Online documentations by Google on Flutter: https://docs.flutter.dev/

Semester - IV				
Paper	**Paper type**	**Paper name**	**Credit**	**Contact hours**
DSC/CC-5	Theory	Computational Mathematics	3	45
	Practical	Computational Mathematics lab using C Lab	1	30
DSC/CC-6	Theory	Microprocessor and its Applications	3	45
	Practical	Microprocessor – 8085 Lab	1	30
DSC/CC-7	Theory	Operating System	3	45
	Practical	Operating System Lab	1	30
DSC/CC-8	Theory	Object Oriented Programming	3	45
	Practical	Object Oriented Programming lab	1	30

Semester - 4

CMSM- Theory: Computational Mathematics
DSC/Core Course-5, Theory, Semester – 4, Credits - 03, Contact hours - 45.

Introduction Set Theory: Finite and Infinite Sets, Uncountable Infinite Sets, Relations: Properties of Binary Relations, Closure, Partial Ordering Relations, Equivalence, Functions: definition, one-to-one, onto and invertible, Mathematical Functions: Exponential and Logarithmic, Counting: Mathematical Induction, Pigeonhole Principle, Permutation and Combination, Binomial Theorem, Principle of Inclusion and Exclusion.	08 hours
Introduction to Probability Elementary events, Sample space, Classical and Axiomatic definition of Probability, Theorems on Total Probability, Conditional Probability, Bernoulli Trials and Binomial Distribution, Bayes' Theorem, Random Variables, Expectation, Variance, Standard Deviation.	08 hours
Growth of Functions Asymptotic Notations, Standard notations and common functions with simple	04 hours

examples.	
Recurrences Relations, Generating Functions, Linear Recurrence Relations with Constant Coefficients and their solution, Substitution Method, Recurrence Trees.	05 hours
Numerical Methods (Algorithmic Approach) **Errors:** Approximate and Rounding of Numbers, Significant digits, Errors and their types, Propagation of errors. **Interpolation:** Newton Forward and Backward interpolation, Lagrange interpolation. Solving a Set of Linear Equations: Gaussian Elimination, Gauss–Jordan, Iteration methods and their convergence conditions, Gauss-Seidel, Gauss-Jacobi Iterative Methods. **Solving Non-linear equations:** Bisection, Regula-falsi, Secant and Newton-Raphson, their order of convergence. **Solving Differential Equations:** Euler, Runge-Kutta second and fourth order methods. **Numerical Integration:** Trapezoidal and Simpson's 1/3rd rules. **Curve fitting:** Least square approximation, Linear regression, Polynomial regression, Fitting Exponential and Trigonometric functions	12 hours
Graph Theory Basic Terminology, Models and Types, Multi graphs and Weighted graphs, Graph Representation, Graph Isomorphism, Connectivity, Euler and Hamiltonian Paths and circuits, Planar Graphs, Trees and their basic terminologies and properties.	08 hours

CMSM - Practical: Computational Mathematics
DSC/Core Course-5, Practical, Semester – 4, Credits - 01, Contact hours - 30.

Laboratory assignments based on Numerical Methods using C.

1. **Gauss-Seidel Method**: Implement the Gauss-Seidel iterative method to solve a system of linear equations.
2. **Bisection Method**: Write a program to find the root of a nonlinear equation using the bisection method.
3. **Newton-Raphson Method**: Implement the Newton-Raphson method to find the root of a given function.
4. **Matrix Inversion using Gauss-Jordan Method**: Develop a program to find the inverse of a matrix using the Gauss-Jordan elimination method.
5. **Interpolation**: Implement Lagrange interpolation to approximate a function from given data points.
6. **Numerical Integration**: Write programs for the trapezoidal rule and Simpson's rules (1/3) to approximate definite integrals.
7. **Solution of Ordinary Differential Equations (ODEs)**: Implement Euler's method and Runge-Kutta methods (second-order and fourth-order) to solve ODEs.
8. **Curve Fitting**: Fit linear, exponential, and polynomial curves to discrete data using the least squares method.
9. **System of Linear Equations**: Solve a system of linear equations using Gauss elimination or Gauss-Jordan methods.

Note: These are only sample programs; more can be included related to the theory.

Text/ Reference Books

1. Elements of Discrete mathematics, C.L. Liu & Mahapatra, Tata McGraw Hill.
2. Discrete Mathematics and Its Applications, Rosen, McGraw Hill.
3. Introduction to algorithms, T.H. Cormen, C.E. Leiserson, R. L. Rivest, Prentice Hall.
4. Discrete Mathematics with Algorithms, Albertson and Hutchinson, John Wiley Publication.
5. Discrete Structures, Logic, and Computability, J. L. Hein, Jones and Bartlett Publishers.
6. Essentials of Discrete Mathematics, D.J. Hunter, Jones and Bartlett Publishers.
7. Numerical Analysis and Computational Procedures by Mollah, New Central Book.
8. Computer Oriented Numerical Methods, 3rd Edition, V Rajaraman, PHI
9. Graph Theory with Applications to Engineering and Computer Science by Narsingh Deo, PHI.
10. Graph Theory by J.A. Bondy and U.S.R. Murty, Springer.
11. Introduction to Graph Theory by D B West, 2nd edition, Pearson Education

CMSM - Theory: Microprocessors
DSC/Core Course-6, Theory, Semester – 4, Credits - 03, Contact hours - 45.

Introduction to Microcomputer based system Evolution of Microprocessor and Microcontrollers and their advantages and disadvantages.	02 hours
Microprocessor Architectureand memory interfacing Microprocessor 8085 pin-out and signals, basic architecture of microprocessor 8085 and explanation of each block, flag registers, general and special purpose registers, Timing and control unit, need for multiplexing of lower order address and data bus. **Memory Interfacing** Basic concepts in memory interfacing, address decoding (absolute and partial), memory map, foldback memory, and interfacing of memory segment 8155.	06 hours
Control signal generation and de-multiplexing Generation of control signals (IOR, IOW, MEMR, MEMW) using basic and universal logic gates, ALE and use of it for de-multiplexing of lower order address-data bus (**AD_0 to AD_7**).	03 hours
Instruction Set addressing modes, instruction formats, instruction cycle, instruction timing diagram, **Instruction set:** data transfer group, arithmetic and logic group, branch group, machine control group.	04 hours
Interfacing I/O Devices Basic interfacing concepts, Peripheral I/O instructions (I/O mapped I/O), IN & OUT instruction, device selection and data transfer, absolute and partial decoding, realization of input and output ports, memory mapped I/O techniques, Comparison of memory mapped I/O and Peripheral mapped I/O.	10 hours

Programming 8085 Different programming techniques, Stack and Subroutines, counter and time delays, code conversion, BCD Arithmetic and 16-bit data operation.	05 hours
Interfacing Peripheral devices and its applications Concept of Interrupts driven data transfer in microprocessor-8085, software interrupts using RST instructions, hardware interrupt, vectored interrupts (maskable and non-maskable), multiple interrupts and priorities.	05 hours
Interfacing of data acquisition devices, PPI, Keyboard and DMA Digital to Analog Converters (DAC), Analog to Digital converters (ADC), keyboard interfacing, Programable Peripheral Interface (PPI) 8255 (Mode - 0, BSR), keyboard/display interface 8279, and DMA controller 8237.	07 hours
Microprocessor 8086 The 8086 microprocessor- Architecture, Instruction set, Addressing modes, Interrupts, Memory interfacing with 8086.	03 hours

CMSM - Practical: Programming Microprocessor 8085
DSC/Core Course-6, Practical, Semester – 4, Credits - 01, Contact hours - 30.

1. Assembly Language Programming for Arithmetic Operations like Addition, Subtraction,
2. Multiplication and Division on 8, 16-bit data.
3. Assembly Language Programming for different logical operations.
4. Assembly Language Programming for code conversions.
5. Assembly Language Programming for different sorting techniques.
6. Assembly Language Programming for memory block transfer.
7. Assembly Language Programming for AP series and Fibonacci series.
8. Assembly Language Programming for Searching.
9. Assembly Language Programming to determine the frequency distribution of elements in an Array.
10. Assembly Language Programming for block replacement and data transfer of array elements to another memory block.

Note: The above-mentioned are only samples many more programs related to Microprocessor 8085 programming techniques can be included.

CMSM - Theory: Operating System.
DSC/Core Course-7, Theory, Semester – 4, Credits - 03, Contact hours - 45.

Introduction Basic OS functions, types of operating systems- batch processing, multiprogramming, time sharing, multiprocessing, distributed and real time systems.	02 hours
Operating System Organization Processor and user modes, kernels, system calls and system programs.	05 hours

Process System view of the process and resources, process control block, I/O and CPU bound process, process hierarchy, concept of threads Process Scheduling: Pre-emptive and non-pre-emptive scheduling, long term scheduling, short term/CPU scheduling (FCFS, SJF, SRJF, RR and priority) and medium-term scheduling **Process Synchronization:** Concurrent processes, critical section, semaphores and application, methods for inter-process communication;	12 hours
Deadlock Definition, Prevention, Avoidance, Detection, Recovery.	07 hours
Memory Management Physical and logical address space; memory allocation strategies – fixed and variable partitions, paging, segmentation, virtual memory.	12 hours
File and I/O Management Directory structure, file operations, file allocation methods, disk management.	05 hours
Protection and Security Policy mechanism, Authentication	02 hours

CMSM - Practical: Operating System
DSC/Core Course-7, Practical, Semester – 4, Credits - 01, Contact hours - 30.

1. Write a shell script to convert the content of a file from lower case to upper case.
2. Write a shell script to count the words, lines and characters of a given file. File name should be provided at run time.
3. Write a shell script that take a word from user and find out the frequency of the word in a given file.
4. Write a shell script that gets executed at the moment of user login and it displays Good Morning, Good afternoon, Good Evening, Good Night, depending upon the time at which the user logs on.
5. Write a shell script to print Pascal diamond.
6. Write a shell script to find a number using sequential search method.
7. Write a shell script to find a number using binary search technique.
8. Write a shell script to sort a set of integer numbers using bubble sort.
9. Write a shell script to find out the factorial of a given number.
10. Write a shell script to reverse a string and check whether it is a palindrome.
11. Write a shell script to find the roots of a quadratic equation $ax^2 + bx + c = 0$, considering all possible cases.
12. Write a shell script for menu-based system to insert records for employees with employee ID, name, designation, salary in a data file, also display records when necessary. Display salary for the employee asked.

Note: These are just examples; additional ones can be added based on the syllabus.

Text/ Reference Books

1. Operating Systems Concepts, A Silberschatz, P.B. Galvin, G. Gagne, WileyPublications.
2. Modern Operating Systems, A.S. Tanenbaum, 3rd Edition, Pearson Education.

3. Operating Systems: A Modern Perspective, G. Nutt, Pearson Education.
4. Operating Systems, Internals & Design Principles, Stallings, PHI.
5. Operating Systems- Concepts and design, M. Milenkovic, Tata McGraw Hill.
6. Sumitabha Das, UNIX Concepts and Applications, Tata McGraw-Hill.
7. Understanding the Linux Kernel, D. P. Bovet and M. Cesati, O'Reilly.

CMSM - Theory: Object Oriented Programming (OOP's)
DSC/Core Course-8, Theory, Semester – 4, Credits - 03, Contact hours - 45.

Concept of OOPs Difference with procedure-oriented programming, Data abstraction and information hiding: Objects, Classes, methods.	02 hours
Introduction to Java Java Architecture and features, understanding the semantic and syntax differences between C++ and Java, Compiling and executing a Java Program, variables, constants, keywords data types, Operators (Arithmetic, Logical and Bitwise) and expressions, comments, doing basic program output, decision making constructs (conditional statements and loops) and nesting, Java methods (defining, scope, passing and returning arguments, type conversion and type and checking, built-in Java class methods).	04 hours
Arrays, Strings and I/O Creating & using arrays (One dimension and multi-dimensional), referencing arrays dynamically, Java Strings: The Java String class, creating & using string objects, manipulating strings, string immutability & equality, passing strings to & from methods, string buffer classes. Simple I/O using System.out and the scanner class, byte and character streams, Reading/Writing from console and files.	06 hours
Object-Oriented Programming Overview Principles of Object-Oriented Programming, defining & using classes, controlling access to class members, class constructors, method overloading, Class variables & methods, Objects as parameters, final classes, Object class, garbage collection.	04 hours
Inheritance, Interfaces, Packages, Enumerations, Autoboxing and Metadata. Single Level and Multilevel, Method Overriding, Dynamic Method Dispatch, Abstract Classes, Interfaces and Packages, extending interfaces and packages, Package and Class Visibility, Using Standard Java Packages (util, lang, io, net), Wrapper Classes, Autoboxing/Unboxing, Enumerations and Metadata.	10 hours
Exception Handling, Threading, Networking and Database Connectivity Exception types, uncaught exceptions, throw, built-in exceptions, creating your own exceptions; Multi-threading: The Thread class and Runnable interface, creating single and multiple threads, Thread prioritization, synchronization and communication, suspending/resuming threads. Using java.net package, Overview of TCP/IP and Datagram programming. Accessing and manipulating databases using JDBC.	09 hours
Applets Java Applets: Introduction to Applets, Writing Java Applets, Working with Graphics, Incorporating Images & Sounds. Event Handling Mechanisms, Listener Interfaces, Adapter and Inner Classes. The design and Implementation of GUIs using the AWT controls, Swing components of Java Foundation Classes such as labels, buttons, textfields, layout managers, menus, events and listeners; Graphic objects for drawing	10 hours

figures such as lines, rectangles, ovals, using different fonts. Overview of servlets.	

CMSM - Practical: Object Oriented Programming
DSC/Core Course-7, Practical, Semester – 4, Credits - 01, Contact hours - 30.

OOPs Lab using JAVA

Text/Reference Books

1. Java: The Complete Reference, Herbert Schildt, McGraw-Hill Education.
2. The Java Language Specification, Java SE by James Gosling, Bill Joy, Guy L Steele Jr, Gilad Bracha, Alex Buckley, Published by Addison Wesley.
3. Effective Java by Joshua Bloch, Publisher: Addison-Wesley.
4. Core Java 2 by Cay S. Horstmann, Gary Cornell, Volume 1, Prentice Hall.
5. Programming with Java by E. Balagurusamy, McGraw Hill.
6. Java: How to Program by Paul Deitel, Harvey Deitel, Prentice Hall.
7. Programming with JAVA by John R. Hubbard, Schaum's Series.

UNIVERSITY OF CALCUTTA

Notification No. CSR/67/2025

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 the exercise of her powers under 9(6) of the said Act, the Vice-Chancellor has, by an order dated 22.09.2025, approved the Course Structure & Syllabus for Major Courses of semester-5 & 6 of 4-year Honours and Honours with Research Courses of studies and revised course structure for Minor Courses of 4-year Honours and Honours with Research Courses of studies in Computer Science under CCF,2022.

Detail Syllabus of Minor Courses was Published under CSR/82/2024, dt. 26.09.2024.

The above shall take effect from the Odd semester examinations, 2025 and onwards.

13/10/2025

SENATE HOUSE

Kolkata-700073

13.10.2025

Prof.(Dr.) Debasis Das

Registrar

University of Calcutta

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

(2024)

Semester – V & VI

Semester - 5

Theory Paper	Credit	Contact hours	Practical/Tutorial Paper	Credit	Contact hours
Design & Analysis of Algorithms	03	45	Graph algorithms Lab using C++	1	30
Data Communication and Networking	03	45	Networking Lab	1	30
Theory of Computation	03	45	Tutorial	1	30
Database Management System (DBMS)	03	45	RDBMS Lab	1	30

Theory: Design & Analysis of Algorithms
Credits - 03, Contact hours - 45.

Introduction to Algorithms: Definition, Characteristics, Recursive and Non-recursive algorithms	3 hours
Asymptotic Complexity Analysis of Algorithms: Space and Time Complexity, Efficiency of an algorithm, Growth of Functions, Polynomial and Exponential Complexity, Asymptotic Notations: Big O Notation and Small o notation, Big Ω and Small ω, Big Θ and Small φ Notations, Properties: Best case/worst case/average case analysis of well-known algorithms.	6 hours
Algorithm Design Techniques: Concepts and simple case studies of Greedy algorithms. Divide and conquer: Basic concepts, Case study of selected searching and sorting problems using divide and conquer techniques: Dynamic programming: General issues in Dynamic Programming.	12 hours
Graph Representation and Algorithm: Graph traversal algorithms: BFS, DFS, Minimal spanning trees: Prim's Algorithm, Kruskal's Algorithm, Shortest path algorithms: Floyd's Algorithm, Floyd-Warshall Algorithm, Dijkstra's Algorithm, Graph Colouring Algorithms.	22 hours
Classification of Problems: Concept of P, NP.	2 hours

Text/References Books

1. Introduction to Algorithms, Cormen, Leiserson, Rivest and Stein, TMH.
2. The Design and Analysis of Algorithms, Aho, Hopcroft and Ullman, Pearson Education.
3. The Art of Computer Programming, D.E. Knuth, Pearson Education.
4. Algorithm Design, Jon Kleiberg and Eva Tardos,Pearson Education.
5. Data Structures and Algorithms - K.Mehlhorn.
6. Computer Algorithms, S.Baase, Pearson Education.
7. Fundamentals of Computer Algorithms, E. Horowitz and Sahani, Galgotia
8. Combinational Algorithms- Theory and Practice, E.M. Reingold, J. Nievergelt and N. Deo, PHI.

Practical: Graph algorithms Lab using C++
Credits - 01, Contact hours - 30.

Pre-requisite;

1. Programming Foundations in C++

- Syntax, data types, loops, conditionals, and functions
- STL basics — especially vector, queue, stack, and map
- Dynamic memory and pointers (essential for building graph structures)

2. Graph Theory Fundamentals

- Terminology: vertices, edges, adjacency, degree, cycles
- Graph types: directed vs undirected, weighted vs unweighted
- Representations: adjacency matrix vs adjacency list

3. Math & Logic Readiness

- Basic combinatorics and logic reasoning
- Set theory (for understanding disjoint sets and connectivity)
- Matrices (helpful for adjacency matrix representations)

4. Compilers

- C++ compiler GCC
  or
- MSVC (Microsoft Visual C++), Visual Studio Community Edition.

Laboratory exercises to be based on Graph Theory using C++ and based on the following;

Implementation of Graph algorithms: Single Spanning Tree Generation using - BFS, DFS, Minimal Spanning Tree Generation using - Prim's Algorithm, Kruskal's Algorithm, Shortest Path finding using Floyd's Algorithm, Floyd-Warshall Algorithm, Dijkstra's Algorithm, Graph Partitioning Algorithm.

Sample questions

1. Write a C++ program to generate a single spanning tree using Breadth-First Search (BFS). Trace and display visited nodes.
2. Develop a Depth-First Search (DFS)-based C++ algorithm to construct a spanning tree from a connected undirected graph. Compare traversal order with BFS.
3. Design and implement Prim's Algorithm using adjacency list and priority queue in C++. Display the edges included in the MST and their weights.
4. Construct Kruskal's Algorithm using disjoint set union in C++. Identify cycle prevention strategy and display final edge set of the MST.
5. Write a C++ program to compute the shortest paths from a single source using Dijkstra's Algorithm. Provide output trace with distances and paths.
6. Implement Floyd's Algorithm to find all-pairs shortest paths in a weighted graph using adjacency matrix representation. Analyze time complexity.
7. Demonstrate Floyd-Warshall Algorithm with intermediate node storage. Display final distance matrix and reconstructed paths.
8. Apply a basic graph partitioning strategy using BFS clustering in C++. Divide the graph into subgraphs and count nodes in each.
9. Implement a heuristic-based Graph Partitioning Algorithm in C++ and evaluate inter-partition edge cuts. Visualize the result if possible.

Note: ***The questions provided are illustrative samples. Students are encouraged to practice additional exercises aligned with the prescribed syllabus topics for comprehensive understanding.***

Theory: Data Communication and Networking
Credits - 03, Contact hours - 45.

Overview of Data Communication and Networking **Introduction:** Data communications Components, data representation, direction of data flow (simplex, half duplex, full duplex). **Network Hardware:** Physical structure (type of connection, topology), categories of network (LAN, MAN, WAN). **Internet:** Brief history, Protocols and standards, **Reference models:** OSI reference model, properties of all the layers, TCP/IP reference model, their comparative study.	03 hours
Physical Layer **Data & Signals:** Analog & Digital Data and Signals, periodic and non-periodic signals, composite signals, bandwidth, bit rate, transmission of digital signals. **Transmission Impairments:** Attenuation, Distortion and Noise. **Data Rate Limits:** Noiseless Channel: Nyquist Data rate, Noisy Channel: Shannon's Capacity, calculation of data rate using both limits. **Digital Transmission** **Digital to Digital Conversion:** Line coding, schemes (RZ, NRZ, Manchester, Differential Manchester), block coding. **Analog to Digital Conversion:** Sampling, Nyquist rate of sampling, Pulse code modulation (PCM), Delta Modulation (DM), Adaptive Delta Modulation (ADM), parallel and serial transmission. **Analog Transmission** **Digital to Analog conversion:** Amplitude shift keying (ASK), Frequency Shift Keying (FSK), Phase Shift Keying (PSK), Quadrature Amplitude Modulation (QAM). **Analog to Analog Conversion (qualitative):** Amplitude Modulation (AM), Frequency Modulation (FM), Phase Modulation.	11 hours
Bandwidth Utilization Techniques Multiplexing: FDM, Synchronous & Statistical TDM, WDM.	03 hours
Transmission Media **Guided media:** Twisted pair, Coaxial, Fiber optics. **Unguided:** Radio waves, microwaves, Infrared, Antenna, Communication satellites (qualitative study only).	04 hours
Switching and Telephone network Circuit switched networks, Packet Switched networks, Virtual Circuit switch. Major components of telephone network, Dial up modem, DSL and ADSL modems, Cable TV for data transfer (qualitative study only)	03 hours
Data link Layer Types of errors, framing (character and bit stuffing), error detection & correction methods, Linear and cyclic codes, checksum. **Protocols:** Stop & wait ARQ, Go-Back- N ARQ, Selective repeat ARQ, HDLC (qualitative study only). **Physical addressing:** MAC address and its format	03 hours
Medium Access sub layer Point to Point Protocol, **Token Ring:** Reservation, Polling. **Multiple access protocols:** Pure & Slotted ALOHA, CSMA, CSMA/CD, CSMA/CA. **Channelization:** FDMA, TDMA, CDMA (Qualitative study only). **Wired and Wireless LAN:** Standards, fast Ethernet, Protocol 802.11, Bluetooth.	05 hours

Network layer **Internetworking & devices:** Repeaters, Hubs, Bridges, Switches, Router, Gateway, **Addressing:** IP addressing, Subnetting, Routing techniques: static vs. dynamic routing, **Protocols:** RARP, ARP, IP, ICMP.	07 hours
Transport layer **Process to Process delivery:** UDP, TCP	02 hours
Application Layer Introduction to DNS, Remote logging, FTP, Electronic mail, WWW & HTTP.	04 hours

Text/ Reference Books

1. Data Communication and Networking, B.A. Forouzan, Tata McGraw Hill.
2. Computer Networks, A.S. Tanenbaum, Pearson Education.
3. Data and Computer Communication, W. Stallings, Pearson Education.
4. Data & Computer Communication, Black, PHI.
5. Internet & World Wide Web: How to program, Harvey M. Deitel& Paul J. Deitel.

Practical: Networking Lab
Credit: 01, Contact hour: 30.

1. Simulate Simplex, Half Duplex, and Full Duplex Communication Use serial and Ethernet ports to show directional data flow between two nodes.
2. Create LAN, MAN, and WAN Topologies Use appropriate devices (switches, routers, cloud) to build various category-wise networks.
3. Design Physical Topologies Implement star, bus, ring, and mesh topologies using appropriate devices.
4. Model Protocol Stack using TCP/IP Reference Model Layers Simulate communication between clients and servers, annotating each layer's activity.
5. Compare OSI vs TCP/IP Reference Models Create two scenarios showing layered encapsulation in both stacks and interpret packet flows.
6. Visualize Bandwidth and Bit Rate Concepts Use different cable types and configure link speeds to demonstrate throughput variations.
7. Simulate Signal Impairments Using Distorted Packets Introduce corruption manually and trace signal behavior across a noisy channel.
8. Compare Guided Media: Coaxial vs Fiber vs Twisted Pair Deploy multiple link types and measure latency and throughput across them.
9. Implement Multiplexing Models (FDM & TDM) Use time-sequenced communication across nodes to simulate multiplexed channel behavior.
10. Simulate Packet Switching Using Routers Design a scenario where data packets are rerouted based on availability.
11. Create a Virtual Circuit Setup Between Devices Use Label Switching or logical pathways to model a fixed route data exchange.
12. Demonstrate MAC Addressing and Frame Inspection Use simulation mode to trace MAC-level transmission and address resolution.
13. Simulate Error Detection Using Checksums and CRC Introduce errors and verify how they're flagged and corrected.
14. Frame Construction Using Bit and Character Stuffing Manual framing via scripts or event annotation showcasing stuffing techniques.
15. Compare CSMA/CD vs CSMA/CA Protocols Use wired Ethernet vs wireless 802.11 environments to simulate contention handling.

16. Create WLAN Using 802.11 Standard Devices Configure a wireless network with access points, laptops, and mobile nodes.
17. Simulate Bluetooth and PAN Communication Establish small-scale device-to-device networks to reflect low-energy protocol behavior.
18. Configure Static and Dynamic Routing Using RIP/EIGRP/OSPF Route between multiple routers with table inspection and simulation.
19. Implement IP Addressing and Subnetting Assign subnets to devices and verify connectivity across routed domains.
20. Simulate TCP and UDP Communication Use PCs and servers to demonstrate reliable vs unreliable transport with packet inspection.

Simulator: CISCO Packet Tracer

Note: ***The questions provided are illustrative samples. Students are encouraged to practice additional exercises aligned with the prescribed syllabus topics for comprehensive understanding.***

Theory: Theory of Computation
Credits - 03, Contact hours - 45.

Finite Automata Definition of a Finite Automaton, Model, Representation, Classification – with respect to output function Mealy and Moore Machines, with respect to State Transition – Deterministic and Non-Deterministic Machine, Examples, conversion algorithms Mealy to Moore and Moore to Mealy, Finite and Infinite state machines, Finite Automaton, Deterministic and Non-Deterministic Finite automaton, non-deterministic to equivalent Deterministic Automaton-Optimized and Non-optimized technique ideas and algorithms, Acceptability of String by a Finite Automaton.	12 hours
Formal Languages and Grammar Introduction to Formal Grammar and Language, Chomsky's Classification of Grammar – Type-0, Type-1 or Context Sensitive, Type-2 or Context Free and Type-3 or Regular Grammar, Illustration of each of these classes with example, Sentential form, Sentences – Languages or strings, Derivations, Ambiguous Grammar and Language, Designing of Grammar for a language, Find the Language for given Grammar, Definition and basic idea about Push Down Automaton.	11 hours
Regular Expression Basic Idea and Definition, Regular Expression basic Identities, Arden's Theorem – Statement (without Proof) and application for reduction of equivalent regular expressions, Regular expression to Finite Automata conversion, State Transition System to Regular Expression conversion algorithm by Arden's Algebraic Method, FA to Regular Grammar and Regular Grammar to FA conversion algorithms and applications.	12 hours
Turing Machine Concepts of Turing Machine, Formal Definitions, Classifications – Deterministic and Non-Deterministic Turing Machines, Simple Design of Turing Machines: Odd / even count and concepts of Universal Turing Machines, Difference and Similarities between Turing Machine and a General-Purpose Computer, Definition and significant of Halting Problem in Turing Machine.	10 hours

Text/ Reference Books

1. Introduction to Automata Theory, Languages, and Computation by John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman, 3rd Edition, Pearson.
2. Theory of Computer Science (Automata, Languages & Computation) by K L P Misra & N Chandrasekharan, 3rd Edition, PHI.
3. Introduction to Theory of Computation by MichealSipser, 3rd Edition, Cengage Learning.
4. Switching and Finite Automata Theory by ZviKohavi, Niraj.K.Jha, 3rd Edition, TMH.
5. Formal Language and Automata, P. Linz, Narosa.

Practical: Tutorial
Credits - 01, Contact hours - 30.

Theory: Database Management System (DBMS)
Credits - 03, Contact hours - 45.

Introduction Drawbacks of Legacy System; Advantages of DBMS; Layered Architecture of Database, Data Independence; Data Models; Schemas and Instances; Database Languages; Database Users, DBA; Data Dictionary.	03 hours
Entity Relationship (ER) Modelling Entity, Attributes and Relationship, Structural Constraints, Keys, ER Diagram of Some Example Database, Weak and strong Entity Set, Specialization and Generalization, Constraints of Specialization and Generalization, Aggregation.	04 hours
Relational Model Basic Concepts of Relational Model; Relational Algebra; Tuple Relational Calculus; Domain Relational Calculus.	08 hours
Integrity Constraints Domain Constraints, Referential Integrity, View.	04 hours
Relational Database Design Problems of Un-Normalized Database; Functional Dependencies (FD), Derivation Rules, Closure of FD Set, Canonical Cover; Normalization: Decomposition to 1NF, 2NF, 3NF or BCNF Using FD; Lossless Join Decomposition Algorithm; Dependency preservation.	10 hours
SQL Basic Structure, Data Definition, Constraints and Schema Changes; Basic SQL Queries (Selection, Insertion, Deletion, Update); Order by Clause; Complex Queries, Aggregate Function and Group by Clause; Nested Sub Queries; Views, Joined Relations; Set Comparisons (All, Some); Derived Relations.	11 hours
Record Storage and File Organization (Concepts only) Fixed Length and Variable Length Records; Spanned and Un-Spanned Organization of Records; Primary File Organizations and Access Structures Concepts; Unordered, Sequential, Hashed; Concepts of Primary and Secondary Index; Dense and Sparse Index; Index Sequential Files; Multilevel Indices.	05 hours

Text/ Reference Books

1. Fundamentals of Database Systems 6th Edition, R. Elmasri, S.B. Navathe, Pearson Education.
2. Database Management Systems, R. Ramakrishanan, J. Gehrke,3rd Edition, McGraw-Hill.
3. Database System Concepts 6th Edition, A. Silberschatz, H.F. Korth, S. Sudarshan, McGraw Hill.
4. Database Systems Models, Languages, Design and application Programming, R. Elmasri, S.B. Navathe, Pearson Education.
5. SQL and Relational Theory: How to Write Accurate SQL Code, Christopher J. Date, O'Reilly Media.
6. Database Systems: A Practical Approach to Design, Implementation and Management, Thomas M. Connolly and Carolyn E. Begg, Pearson.

Practical: RDBMS Lab
Credits - 01, Contact hours - 30.

1. Design a student-course enrollment schema using Excel (legacy format), then recreate the same in MySQL using normalized tables to highlight redundancy reduction and relational integrity.
2. Create a login system using MySQL that demonstrates layered architecture. Separate user credentials (user layer), schema definitions (schema layer), and storage engine settings (storage layer).
3. Alter a table's structure (e.g., add/remove columns) and demonstrate how MySQL preserves application-level logic, illustrating data independence.
4. Draw an ER diagram for a college admission system. Then, create MySQL tables for Student, Course, and Department reflecting primary/foreign key relationships.
5. Model weak and strong entities by implementing Dependent and Employee tables in MySQL. Use partial key constraints and foreign key links.
6. Demonstrate specialization/generalization by creating an entity Vehicle, and specialized entities Car and Truck in MySQL using table inheritance logic.
7. Convert your ER diagram from Exercise 4 into relational schemas and implement the tables in MySQL.
8. Use MySQL to perform relational algebra operations:
   Selection: Retrieve students from a particular department
   Projection: List unique course names
   Union: Combine student lists from two departments
   Join: Get student names with course titles they enrolled in*
9. Write queries using tuple and domain relational calculus concepts translated into SQL. Example: "Find all students who have enrolled in Math but not Physics."
10. Create a student-course database in MySQL. Apply domain constraints (e.g., NOT NULL, CHECK), and referential constraints using FOREIGN KEYs.
11. Define a MySQL view that shows the average marks of students per department. Write SQL to query from that view.
12. Design a student performance table with repeated subject columns. Show data redundancy and anomalies during updates.
13. Normalize the above table into 1NF, 2NF, 3NF, and BCNF. Create each normal form version using MySQL CREATE TABLE statements.
14. Demonstrate lossless join and dependency preservation using SQL joins. Verify consistency of decomposed relations.
15. *Create a Sales database in MySQL. Write SQL queries to:
    SELECT all products
    INSERT a new order
    UPDATE product price
    DELETE an obsolete order*

16. Write complex queries using GROUP BY and aggregate functions to calculate total sales per region and highest priced product per category.
17. Use nested subqueries and SET comparison operators (ALL, ANY, SOME) to find customers with purchases exceeding others.
18. Join customer and order tables. Create a view for total spending per customer. Query the view to find top spenders.
19. Write derived table queries (subqueries inside FROM) to filter high-value orders exceeding ₹20,000.
20. Apply schema constraints using PRIMARY KEY, FOREIGN KEY, and CHECK. Perform ALTER TABLE operations to modify schema.

RDBMS software tool: MYSQL

Note: ***The questions provided are illustrative samples. Students are encouraged to practice additional exercises aligned with the prescribed syllabus topics for comprehensive understanding.***

Semester - 6

Theory Paper	Credit	Contact hours	Practical/Tutorial Paper	Credit	Contact hours
Software Engineering	03	45	Tutorial	1	30
Programming in Python	03	45	Programming in Python	1	30
Linear Algebra & Statistical Methods	03	45	Linear Algebra & Statistical Methods using Python	1	30

Theory: Software Engineering
Credits - 03, Contact hours - 45.

Introduction Defining system, open and closed system, modelling of system through computer hardware, communication systems, external agents and software systems; Importance of Engineering Methodology towards computerization of a system.	3 hours
Software Life Cycle Classical and Iterative Waterfall Model; Spiral Model; Prototype Model; Evolutionary model and its importance towards application for different system representations, Comparative Studies.	6 hours
Software Requirement and Specification Analysis Requirements Principles and its analysis principles; Specification Principles and its Representations Software Design Analysis – Different level of DFD Design, Physical and Logical DFD, Use and Conversions between them, Decision Tables and Trees, Structured analysis, Coupling and Cohesion of different modules Software Cost Estimation Modelling –COCOMO.	15 hours
Software Testing Software Verification and Validation; Testing objectives, Testing Principles, Testability; Error and Faults; Unit Testing, White Box and Blank Box Testing, Test Case Design: Test Vector, Test Stub.	15 hours

Software Quality Assurances Concepts of Quality, Quality Control, Quality Assurance, IEEE Standard for Statistical Software Quality Assurances (SSQA) criterions.	06 hours

Text/References Books

1. Software Engineering: A Practitioner's Approach by R.S. Pressman, McGraw-Hill.
2. An Integrated Approach to Software Engineering by P. Jalote, Narosa Publishing House.
3. Software Engineering by K.K. Aggarwal and Y. Singh, New Age International Publishers.
4. Software Engineering by I. Sommerville, Addison Wesle.
5. Software Engineering for Students by D. Bell, Addison-Wesley.
6. Fundamentals of Software Engineering by R. Mall, PHI.

Practical: Tutorial
Credits - 01, Contact hours - 30.

Theory: Programming In Python
Credits - 03, Contact hours - 45.

Introduction Introducing Python, features of Python, the paradigms, chronology and uses, setting up Python on Windows and other Operating systems, introducing IDLE, Installation of Anaconda and miniconda.	02 hours
Parts of Python Programming Language Identifiers, Keywords, Statements and Expressions, Variables, Operators, Precedence and Associativity, Data Types, Indentation, Comments, Reading Inputs, print output, Type Conversions, The type() function and Is operator, Dynamic and strongly typed language.	04 hours
Control Flow Statements If, if.....else, if.....elif...else decision control statement, nested if statement, While loop, Continue and break, catching exceptions using try and except.	04 hours
Functions Built in functions, commonly used modules, Function Definition and Calling the Function, return and void function, Scope and life time of variables, default parameters, keyword arguments, *args and **kwargs, command line arguments.	04 hours
String Creating and storing strings, basic string operation, Accessing Characters in string by index, String slicing and joining, string methods, formatting strings.	05 hours
Lists Creating lists, basic list operations, indexing and slicing in lists, built-in functions used on lists, list methods, del statements,	05 hours
Dictionaries Creating Dictionary, accessing and modifying key value, built-in functions used on dictionaries, dictionary methods.	03 hours

Tuples and Sets Creating tuples, basic tuple operations, Indexing and slicing in Tuples, built-in functions on tuples, relation between tuples and lists, relation between tuples and dictionaries, Tuple methods, Zip() function, Sets, Sets methods, frozen set.	04 hours
Files Types of files, creating and reading text data, file method to read and write data, reading and writing binary, the pickle module, reading and writing CSV files, python os and os.path modules,	03 hours
Regular expression Using special characters, regular expression methods, named groups in python regular, regular expression with glob module.	03 hours
Object Oriented Programming Classes and objects, creating classes in python, creating objects in python, constructor method, classes with multiple objects, class versus data attributes, encapsulation, inheritance, polymorphism.	04 hours
Introduction to Data Science Functional programming, JSON and XML in Python, Numpy with Python, Pandas, altair	04 hours

Text/ Reference Books

1. Introduction to Computation and Programming Using Python: With Application to Understanding Data, Guttag, John V. MIT Press.
2. Learn Python 3 the Hard Way: A Very Simple Introduction to the Terrifyingly Beautiful World of Computers and Code, Shaw, Zed A, Addison-Wesley Professional.
3. Think Python 2e. Green Tea Books, Downey, Allen B.
4. Practical Programming: An Introduction to Computer Science Using Python 3.6. Pragmatic Bookshelf, Gries, Paul, Jennifer Campbell, and Jason Montojo.
5. Introduction to Python Programming, Gowrishankar S, Veena A, Taylor and Francis, CRC Press.
6. Python Cookbook, David Beazley and Brian K. Jones, O'Reilly.

Practical: Programming in Python
Credits - 01, Contact hours - 30.

1. Develop a Python program that reads marks from three different subjects and calculates their average.
2. Create a Python script to convert weight from kilograms to pounds.
3. Write a program that takes the lengths of the three sides of a prism as input (integers) and computes its surface area using the formula: 2ab + 2bc + 2ca.
4. Design a Python program to calculate the speed of a plane, given a distance of 395,000 meters and a travel time of 9,000 seconds.
5. Build a program that determines how long it will take to empty a swimming pool of dimensions 12m × 7m × 2m using a pump with a flow rate of 17 cubic meters per hour.
6. Write a Python script to convert temperature from Celsius to Fahrenheit (input as a float).
7. Create a program to compute the total number of seconds in one full day.
8. Develop a Python program to calculate the acceleration of a car that starts from rest and reaches a velocity of 10 m/s in 20 seconds, using the formula: $(v_{final} - v_{initial})$/time.

9. Write a program to display Pascal's triangle.
10. Write a program to display the following pattern using nested loops.
    1 1
    22 21
    333 321
    4444 4321
    55555 54321
11. Write a program that uses a while loop to add up all the even numbers between 100 and 200.
12. Write a program to print the sum of the following series
    a. **1 + ½ + 1/3 +. …. + 1/n**
    b**. 1/1 + 2^2/2 + 3^3/3 + ……. + n^n/n**
13. Write a program to find the depreciation value of an asset (property) by reading the purchase value of the asset (amt), year of the service (year) and the value of depreciation.
14. Write a program that accepts a string from the user and display the same string after removing vowels from it.
15. Write a function to insert a string in the middle of the string.
16. Write a program to sort a string lexicographically.
17. Write a program to replace a string with another string without using built-in methods.
18. Write a program to concatenate two strings into another string without using the + operator.
19. Write a program to strip a set of characters from a string.
20. Write a program to extract the first n characters of a string.
21. Write a program that creates a list of 10 random integers. Then create two lists by name **odd_list** and **even_list** that have all odd and even values of the list respectively.
22. Write a program to sort the elements in ascending order using insertion sort.
23. Write a Python program to use binary search to find the key element in the list.
24. Make a list of the first eight letters of the alphabet, then using the slice operation do the following operations:
    a. Print the first three letters of the alphabet.
    b. Print any three letters from the middle.
    c. Print the letters from any particular index to the end of the list.
25. Write a program to sort the elements in ascending order using selection sort.
26. Write a program that prints the maximum value of the second half of the list.
27. Write a program that creates a list of numbers 1–100 that are either divisible by 5 or 6.
28. Write a function that prompts the user to enter five numbers, then invoke a function to find the GCD of these numbers.
29. Write a function named addfruit, which is passed with a set of fruit names and their prices and returns a dictionary containing the entered information and raises a **ValueError** exception if the fruit is already present.
30. Write a function to add the air quality index as the value and the date as the key; create the dictionary for the entered information.
31. Create a dictionary that contains usernames as the key and passwords as the associated values. Make up the data for five dictionary entries and demonstrate the use of clear and **fromkeys()** methods.
32. Write Pythonic code to create a dictionary that accepts a country name as a key and its capital city as the value. Display the details in sorted order.
33. Write a program that has the dictionary of your friends' names as keys and phone numbers as its values. Print the dictionary in a sorted order. Prompt the user to enter the name and check if it is present in the dictionary. If the name is not present, then enter the details in the dictionary.
34. Write a program to create a dictionary containing the author name as the keys and ISBN number as the value. Make up the data for five dictionary entries and demonstrate the use of **clear()** and **fromkeys()** methods.
35. Create a list containing three elements, and then create a tuple from that list.

36. Write a program to unpack a tuple to several variables.
37. Write a program to check whether an item exists within a tuple.
38. Write a program to unzip a list of tuples into individual lists.
39. Write a program to create an intersection, union, set difference, and symmetric
40. difference of sets.
41. Write a program to demonstrate the use of **issubset()** and **issuperset()** methods.
42. Write a program that takes a range and creates a list of tuples within that range with the first element as the number and the second element as the square of the number.
43. Write a program to clear a set.
44. Write a program to find the length of the set.
45. Write a program to store the latitude and longitude of your house as a tuple and display it.
46. Write a program that prompts the user to enter a text file, reads words from the file, and displays all the non-duplicate words in ascending order.
47. Write a program to get the file size of a plain text file.
48. Write a program that prompts the user to enter a text filename and displays the number of vowels and consonants in the file.
49. Write a program to read the first n lines of a file. Prompt the user to enter the value for n.
50. Write a program that reads the contents of the file and counts the occurrences of each letter. Prompt the user to enter the filename.
51. Write a program to read the last n lines of a file. Prompt the user to enter the value for n.
52. Write a program to combine each line from the first file with the corresponding line in the second file.
53. Write a program to remove newline characters from a file.
54. Write a program to read the random line from a file.
55. Write a program to read and write the contents from one csv file to another.
56. Write a Python program that matches a word containing 'z'.
57. Write a Python program to remove all leading zeros' from an IP address
58. Write a Python program to search the numbers (0–9) of length between 1 to 3 in a given string.
59. Write a Python program to find the substrings within a string
60. Write a Python program to extract year, month and date from an url.
61. Write a Python program to read a file and to convert a date of yyyy-mm-dd format to dd-mm-yyyy format.
62. Write a Python program to abbreviate 'Street' as 'St.' in a given string.
63. Write a Python program to find all five characters long word in a string.
64. Create a class named quadratic, where a, b, c are data attributes and the methods are
    a. __init__() to initialize the data attributes
    b. roots() to compute the quadratic equation
65. Define a class called student. Display the marks details of top five students using inheritance.
66. Create a class called library with data attributes like acc_number, publisher, title and author. The methods of the class should include
    a. read() – acc_number, title, author.
    b. compute() - to accept the number of days late, calculate and display the fine charged at the rate of $1.50 per day.
    c. display the data.
67. Create two base classes named clock and calendar. Based on these two classes define a class calendarclock, which inherits from both the classes which displays month details, date and time.
68. Write a program to add two polynomials using classes.
69. Define JSON. Construct a simple JSON document and write Pythonic code to parse JSON document.
70. Elaborate on the differences between XML and JSON.
71. Define XML. Construct a simple XML document and write Python code to loop through XML nodes in the document.

72. Explain NumPy array creation functions with examples.
73. Explain NumPy integer indexing, array indexing, Boolean array indexing and slicing with examples.
74. Write Python program to create and display a one-dimensional array-like object containing an array of data using pandas library.
75. Write Python program to add, subtract, multiply and divide two Pandas Series.
76. Write Python program to create and display a Data Frame from a dictionary data which has the index labels.

Note: ***The questions provided are illustrative samples. Students are encouraged to practice additional exercises aligned with the prescribed syllabus topics for comprehensive understanding.***

Theory: Linear Algebra & Statistical Methods
Credits - 03, Contact hours - 45.

Linear Algebra	
Vectors Introduction, Vector in Rn, vector addition and scalar multiplication, Dot (Inner) product, orthogonality, Cauchy-Schwarz inequality, spatial vectors(R3). Definition of matrices, elementary operations on matrices (addition, scalar multiplication, matrix multiplication, transpose), different types of matrices, determinants and their properties, matrix inverse. Definition of linear transformations (Linear mappings), properties, kernel and image, matrix representation of linear transformations. Representation of systems of linear equations in matrix form, consistent and inconsistent systems, Gaussian elimination, Gauss-Jordan method, echelon matrices, row canonical form, row equivalence. Definition of vector spaces, examples, spanning sets, linear span, linear dependence and independence, basis and dimension, rank, change of basis	**15 hours**
Eigenvalues and Eigenvectors Definition, characteristic equation, computing eigenvalues and eigenvectors, diagonalizing matrices. Applications in computer graphics and machine learning: Principal Component Analysis (PCA), Singular Value Decomposition (SVD) for dimensionality reduction and data analysis.	**5 hours**
Statistical methods	
Descriptive Statistics & Probability basics Data representation and summarization: Types of data (quantitative, qualitative), measurement scales (nominal, ordinal, interval, ratio), frequency distributions, graphical representation (histograms, frequency polygons), stem and leaf plots. Measures of central tendency: mean, median, mode. Measures of dispersion: range, variance, standard deviation. Moments, skewness, and kurtosis: Characteristics of data distribution. Moment-generating functions of random variables.	**5 hours**

Probability distributions 10 hours Common distributions: Binomial, Poisson, Normal, Uniform, Exponential. Sampling distributions of mean and variance.	**10 hours**
Statistical inference Point and interval estimation, estimating population parameters from sample data. Hypothesis testing, formulating hypotheses, Type I and Type II errors, significance levels, various tests (t-tests, chi-square tests, ANOVA.	**5 hours**
Regression and Correlation Definition of correlation, Karl Pearson coefficient of correlation, Spearman's rank correlation coefficient. Linear regression, least squares method, simple linear regression, fitting of polynomials and exponential curves.	**5 hours**

References:

1. Introduction to Linear Algebra, Gilbert Strang, 6th ed.
2. Linear Algebra and Its Applications, 5th Edition - Pearson by David C. Lay.
3. Linear Algebra: Step By Step, by Kuldeep Singh, Oxford University Press.
4. Linear Algebra, G. Hadley
5. Probability and Statistical Inference, by Robert V. Hogg, Elliot Tanis, Dale Zimmerman, 10th ed.
6. Fundamental Of Mathematical Statistics, S C Gupta & V K Kapoor, Sultan Chand and sons.
7. Introduction to Probability and Statistics for Engineers and Scientists, by Sheldon M. Ross, Academic Pr; 5th edition.
8. Statistical Methods, N G Das, Tata McGraw Hill.
9. Statistical Methods An Introductory Text, by Jyotiprasad Medhi, New Age International (P) Limited, Wiley.

Practical: Linear Algebra and Statistical methods Lab.
Credits - 01, Contact hours - 30.

1. Implementing machine learning algorithms, understanding their underlying mechanisms by using linear algebra concepts like vectors, matrices, eigenvalues, and eigenvectors

2. Solving systems of linear equations: Gaussian elimination, Gauss-Jordan elimination, and LU decomposition

3. Statistical methods like analysing datasets by utilizing descriptive statistics by mean, median, standard deviation

4. Data visualization techniques (histograms, boxplots) to analyse and interpret datasets containing information like heights and weights of individuals, identify outliers, and assess normality

5. Designing experiments and analysing results by applying principles of experimental design, like one-way and two-way ANOVA and for tasks such as testing the efficacy of different pain management methods

6. Predicting trends and classifying data by building regression models to predict outcomes based on given data and employing classification techniques to categorize data into different classes

7. Understanding and evaluating machine learning algorithms like hypothesis testing and confidence intervals which are vital in evaluating the performance and reliability of machine learning models

(These are suggestive applications of the theory portion. The software mentioned above can be used in the applications).

Tools/Simulator/Programming Language – Python Programming language with **libraries like NumPy, Sage Math, or Scilab.**

Minor Papers for Computer Science
(4-Year Honours & Honours with Research Courses of Studies)

Semester	Computer Science as Minor (M1) study/discipline		Computer Science as Minor (M2) study/discipline	
	Theory Paper Credits – 03 (75 marks)	**Practical paper Credits – 01 (25-marks)**	**Theory Paper Credits – 03 (75 marks)**	**Practical paper Credits – 01 (25 marks)**
I	Computer fundamentals & Digital Logic.	Digital Logic Circuit Lab	Not Applicable	Not Applicable
II	Problem Solving Using C	Problem Solving using C Lab	Not Applicable	Not Applicable
III	Not Applicable	Not Applicable	Computer fundamentals & Digital Logic.	Digital Logic Circuit Lab
IV	Not Applicable	Not Applicable	Problem Solving Using C	Problem Solving using C Lab
V	Data Structures	Data Structures using C	Data Structures	Data Structures using C
	Operating System	Shell Programming	Operating System	Shell Programming
For candidates who elect to pursue M1 or M2 in Computer Science during Semester-VI				
VI	Data Structures	Data Structures using C	Data Structures	Data Structures using C
	Operating System	Shell Programming	Operating System	Shell Programming