UNIVERSITY OF KALYANI

UG 4 YEAR Computer Science (HONOURS/ HONOURS WITH RESEARCH) SYLLABUS (Upto 4th Semester)

(Under NEP 2020)

W.E.F. the Academic Session 2023-24

COURSE STRUCTURE Computer Science (NEP-2020)

SEMESTER I

Course Code	Course title	Nature of Course	Credit of Course	Class hour/week	Evaluation		Total
					Internal	Semester End	
CS-MJ-T-1	Computer Fundamentals and Programming using C	Major	4	4	15	60	75
CS-MJ-P-1	Programming using C Lab	Major	2	3	Problem: 60, Viva: 10, Lab Notebook: 5		75
CS-MI-T-1	Computer Fundamentals and Programming using C	Minor	3	3	10	25	35
CS-MI-P-1	Programming using C Lab	Minor	1	2	Problem+ Viva:15		15
CS-MU-T-1	Computer Science for Beginners	Multidisciplinary Course	3	3	10	35	45
CS-SEC-P-1	Office Automation Lab	Skill Enhancement Course	3	3	Problem: 35, Viva: 5, Lab Notebook: 5		45
		Value Added Course	4	4	10	40	50
Total			**20**	**22**			**340**

SEMESTER II

Course Code	Course title	Nature of Course	Credit of Course	Class hour/week	Evaluation		Total
					Internal	Semester End	
CS-MJ-T-2	Digital System Design	Major	4	4	15	60	75
CS-MJ-P-2	Digital System Design Lab	Major	2	3	Problem: 60, Viva: 10, Lab Notebook: 5		75
CS-MI-T-2	Computer Fundamentals and Programming using C	Minor	3	3	10	25	35
CS-MI-P-2	Programming using C Lab	Minor	1	2	Problem + Viva =15		15
CS-MU-P-2	Office Automation	Multidisciplinary Course	3	3	10	35	45
AECC-1		Ability Enhancement Course	4	4	10	40	50
CS-SEC-P- 2	Web Development and Applications Lab	Skill Enhancement Course	3	3	Problem: 35, Viva: 5, Lab Notebook: 5		45
Total			**20**				**340**

SEMESTER III

Course Code	Course title	Nature of Course	Credit of Course	Class hour/week	Evaluation		Total
					Internal	Semester End	
CS-MJ-T-3	Computer Organization & Architecture	Major	6	6	15	60	75
CS-MI-T-3	Database Management Systems	Minor- (theory)	3	3	10	25	35
CS-MI-P-3	Database Management Systems(Lab)	Minor-(Practical)	1	2	Problem + Viva: 15		15
CS-MU-T-3	AI for Everyone	Multidisciplinary Course	3	3	10	35	45
		Ability Enhancement Course					
CS-SEC-P-3	Data Analysis through Python/R(Lab)	Skill Enhancement Course	3	3	Problem: 35, Viva: 5, Lab Notebook: 5		45
		Value Added Course	4	4	10	40	50
			20	**19**			**265**

SEMESTER IV

Course Code	Course title	Nature of Course	Credit of Course	Class hour/week	Evaluation		Total
					Internal	Semester End	
CS-MJ-T-4	Data Structure	Major	4	4	15	60	75
CS-MJ-P-4	Data Structure Lab	Major	2	3	Problem: 60, Viva: 10, Lab Notebook: 5		75
CS-MJ-T-5	Discrete Mathematics	Major- (theory)	6	6	15	60	75
CS-MI-T-4	Database Management Systems	Minor	3	3	10	25	35
CS-MI-P-4	Database Management Systems(Lab)	Minor-(Practical)	1	2	Problem + Viva: 15		15
		Ability Enhancement Course					
CS-SI-1	Summer Internship	Additional for Certificate Diploma	4	4			50
			20	**19**			**325**

Detail Syllabus of Semester-I

CS-MJ-T-1- Theory: Computer Fundamentals and Programming using C
Major Course, Theory, Semester – I, Credits - 04, Contact hours - 40

Course description:

The course introduces the fundamental principles and concepts of digital logic, Number systems and codes along with C programming language. Students will learn about number system, and codes, various tasks to be carried out by C language.

Course Outcomes (COs):

After completing this course satisfactorily, a student will be able to:

• Confidently operate computers to carry out computational tasks
• Understand working of Hardware and Software and the importance of operating Systems
• Understand programming languages, number systems, peripheral devices, networking, multimedia and internet concepts
• Read, understand and trace the execution of programs written in C language
• Write the C code for a given problem
• Perform input and output operations using programs in C
• Write programs that perform operations on arrays, strings , structures, unions and functions

Introduction to Computer and Problem Solving	
Information and Data. Hardware: CPU, Primary and Secondary storage, I/O devices Software: Systems and Application.	
Super, Mainframe, Mini and Personal Computer. Introduction to Programming Languages: Machine Language, Assembly Language, High Level Language. Problem Solving: Flow Charts, Decision Tables and Pseudo codes.	
Number representation: Weighted Codes, Non-weighted codes, Positional, Binary, Octal, Hexadecimal, Binary Coded Decimal (BCD), Conversion of bases. Complement notions. Binary Arithmetic, Binary Codes: Gray, Alphanumeric, ASCII, EBCDIC; Parity Bits. Single Error-Detecting and Correcting Codes, Hamming Codes, Fixed and Floating Point Arithmetic:	
Addition, Subtraction, Multiplication and Division.	

Boolean Algebra	
Fundamentals of Boolean Algebra, Switches and Inverters, Functionally Complete Gates (AND, OR, NOT), NAND, NOR. Switching function and Boolean function. De Morgan's theorem, Minterm and Maxterm, Truth table and minimization of switching function up to four variables, Algebraic and K-map method of logic circuit synthesis: Two level and Multi level.	
C character set, Identifiers and keywords, Data types, Declarations, Expressions, statements and symbolic constants.	
Input-Output: getchar, putchar, scanf, printf, gets, puts, functions. **Pre-processor commands**: #include, #define, #ifdef	
Operators and expressions: Arithmetic, unary, logical, bit-wise, assignment and conditional operators **Storage types**: Automatic, external, register and static variables.	
Functions: Defining and accessing, passing arguments, Function prototypes, Recursion, Library functions, Static functions	
Arrays: Defining and processing, Passing arrays to a function, Multi dimensional arrays. **Strings**: Defining and operations on strings.	
Pointers: Declarations, Passing pointers to a function, Operations on pointers, Pointer Arithmetic, Pointers and arrays, Arrays of pointers function pointers.	
Structures: Defining and processing, Passing to a function, Unions, typedef, array of structure, and pointer to structure	
File structures: Definitions, concept of record, file operations: Storing, creating, retrieving, updating Sequential, relative, indexed and random access mode, Files with binary mode(Low level), performance of Sequential Files, Direct mapping techniques: Absolute, relative and indexed sequential files (ISAM) concept of index, levels of index, overflow of handling. **File Handling**: File operation: creation, copy, delete, update, text file, binary file.	

Recommended Books:

1. P. K. Sinha & Priti Sinha , ―Computer Fundamentals‖, BPB Publications, 2007.
2. Dr. Anita Goel, Computer Fundamentals, Pearson Education, 2010.
3. Kernighan, Brian W., and Dennis M. Ritchie.The C programming language. 2006.

4. Schildt, Herbert, and C. Turbo. "C: the complete reference, Osborne." (2000).
5. Balagurusamy, E. programming in ANSI C. Tata McGraw-Hill Education, 2002.
6. Kanetkar, Yashavant P. Let us C. BPB publications, 2016.

CS-MJ-P-1- Lab: Programming using C Lab
Major Course, Practical, Semester – I, Credits - 02, Contact hours - 40

The following activities be carried out/ discussed in the lab during the initial period of the semester.
1. Basic Computer Proficiency
a. Familiarization of Computer Hardware Parts
b. Basic Computer Operations and Maintenance.
c. Do's and Don'ts, Safety Guidelines in Computer Lab
2. Familiarization of Basic Software, Internet Browsers, Integrated Development Environment (IDE) with Examples.
3. Type Program Code, Debug and Compile basic programs covering C Programming fundamentals discussed during theory classes.

Programming Lab

1. Write a C Program to read radius of a circle and to find area and circumference
2. Write a C Program to read three numbers and find the biggest of three
3. Write a C Program to demonstrate library functions in *math.h*
4. Write a C Program to check for prime
5. Write a C Program to generate n primes
6. Write a C Program to read a number, find the sum of the digits, reverse the number and check it for palindrome
7. Write a C Program to read numbers from keyboard continuously till the user presses 999 and to find the sum of only positive numbers
8. Write a C Program to read percentage of marks and to display appropriate message (Demonstration of else-if ladder)
9. Write a C Program to find the roots of quadratic equation (demonstration of switch-case statement)
10. Write a C program to read marks scored by n students and find the average of marks (Demonstration of single dimensional array)
11. Write a C Program to remove Duplicate Element in a single dimensional Array
12. Write a C Program to demonstrate string functions.
13. Write a C Program to demonstrate pointers in C
14. Write a C Program to check a number for prime by defining *isprime()* function
15. Write a C Program to read, display and to find the trace of a square matrix
16. Write a C Program to read, display and add two m x n matrices using functions
17. Write a C Program to read, display and multiply two matrices using functions
18. Write a C Program to read a string and to find the number of alphabets, digits, vowels, consonants, spaces and special characters.
19. Write a C Program to Reverse a String using Pointer

20. Write a C Program to Swap Two Numbers using Pointers
21. Write a C Program to demonstrate student structure to read & display records of n students.
22. Write a C Program to demonstrate the difference between structure & union.
23. File related programs.

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.

CS-MI-T-1- Theory: Computer Fundamentals and Programming using C
Minor Course, Theory, Semester – I, Credits - 03, Contact hours - 40

Course description:

The course introduces the fundamental principles and concepts of digital logic, Number systems and codes along with C programming language. Students will learn about number system, and codes, various tasks to be carried out by C language.

Course Outcomes (COs):

After completing this course satisfactorily, a student will be able to:

- Confidently operate computers to carry out computational tasks
- Understand working of Hardware and Software and the importance of operating systems
- Understand programming languages, number systems, peripheral devices, networking, multimedia and internet concepts
- Read, understand and trace the execution of programs written in C language
- Write the C code for a given problem
- Perform input and output operations using programs in C
- Write programs that perform operations on arrays, strings , structures, unions and functions

Introduction to Computer and Problem Solving	
Information and Data. Hardware: CPU, Primary and Secondary storage, I/O devices Software: Systems and Application.	3 L
Number Systems	
Super, Mainframe, Mini and Personal Computer. Introduction to Programming Languages: Machine Language, Assembly Language, High Level Language. Problem Solving: Flow Charts, Decision Tables and Pseudo codes.	3 L
Number Systems and Codes	
Number representation: Weighted Codes, Non-weighted codes, Positional, Binary, Octal, Hexadecimal, Binary Coded Decimal (BCD), Conversion of bases.	4 L
Boolean Algebra	

Fundamentals of Boolean Algebra, Switches and Inverters, Functionally Complete Gates (AND, OR, NOT), NAND, NOR. Switching function and Boolean function. De Morgan's theorem, Minterm and Maxterm, Truth table and minimization of switching function up to four variables, Algebraic and K-map method of logic circuit synthesis: Two level and Multi level.	3 L
C Language preliminaries	
C character set, Identifiers and keywords, Data types, Declarations, Expressions, statements and symbolic constants.	3 L
Input-Output: getchar, putchar, scanf, printf, gets, puts, functions. **Pre-processor commands**: #include, #define, #ifdef	2 L
Operators and expressions: Arithmetic, unary, logical, bit-wise, assignment and conditional operators **Storage types**: Automatic, external, register and static variables.	3 L
Functions: Defining and accessing, passing arguments, Function prototypes, Recursion, Library functions, Static functions	3 L
Arrays: Defining and processing, Passing arrays to a function, Multi dimensional arrays. **Strings**: Defining and operations on strings.	3 L
Pointers: Declarations, Passing pointers to a function, Operations on pointers, Pointer Arithmetic, Pointers and arrays, Arrays of pointers function pointers.	3 L
Structures: Defining and processing, Passing to a function, Unions, typedef, array of structure, and pointer to structure	4 L
File structures: Definitions, concept of record, file operations: Storing, creating, retrieving, updating Sequential, relative, indexed and random access mode, Files with binary mode(Low level), performance of Sequential Files, Direct mapping techniques: Absolute, relative and indexed sequential files (ISAM) concept of index, levels of index, overflow of handling. **File Handling**: File operation: creation, copy, delete, update, text file, binary file.	6 L

Recommended Books:

1. P. K. Sinha & Priti Sinha , ―Computer Fundamentals‖, BPB Publications, 2007.
2. Dr. Anita Goel, Computer Fundamentals, Pearson Education, 2010.
3. Kernighan, Brian W., and Dennis M. Ritchie.The C programming language. 2006.
4. Schildt, Herbert, and C. Turbo. "C: the complete reference, Osborne." (2000).

5. Balagurusamy, E. programming in ANSI C. Tata McGraw-Hill Education, 2002.
6. Kanetkar, Yashavant P. Let us C. BPB publications, 2016.

CS-MIP-1- Lab: Practical, Programming using C Lab
Minor Course, Practical, Semester – I, Credits – 01. Contact Hours: 30.

Programs to be written on the following topics:

C Programming elements: Character sets, Keywords, Constants, Variables, Data Types, Operators- Arithmetic, Relational, Logical and Assignment; Increment and Decrement and Conditional, Operator Precedence and Associations; Expressions, type casting. Comments, Functions, Storage Classes, Bit manipulation, Input and output.

C Preprocessor: File inclusion, Macro substitution.

Statements: Assignment, Control statements- if, ifelse, switch, break, continue, goto, Loops- while, do-while, for.

Functions: argument passing, return statement, return values and their types, recursion

Arrays: String handling with arrays, String handling functions.

Pointers: Definition and initialization, Pointer arithmetic, Pointers and arrays, String functions and manipulation, Dynamic storage allocation.

User defined Data types: Enumerated data types, Structures. Structure arrays, Pointers to Functions and Structures, Unions.

File Access: Opening, Closing, I/O operations.

CS-MU-T-1- Theory: Computer Science for Beginners
Multidisciplinary Course, Theory, Semester – 1, Credits - 03, Contact hours - 40.

Objective: The "Computer Science for Beginners" course aims to provide an introductory understanding of the field. It covers the fundamentals of computer hardware, software, and programming languages. Students will learn about number systems, Boolean algebra, and logic gates used in digital circuits. Additionally, the course introduces concepts of database management systems, the history of the internet, and relevant information technology laws, fostering a well-rounded understanding of computing basics and their real-world applications.

Course Outcomes:

(i) Understand the historical development and evolution of computers, including the main features and advancements of each generation.
(ii) Describe the components of modern computers, including the CPU, primary and secondary storage, and various I/O devices.
(iii) Gain knowledge of different number systems and fundamentals of Boolean Algebra and circuit design.
(iv) Understand problem-solving techniques using flowcharts, decision tables, and pseudo codes.
(v) Comprehend the history of programming languages, including Machine Language,

Assembly Language, and High-Level Language.

(vi) Gain an overview of different types of DBMS architectures and their applications.

(vii) Understand the history of the internet, its role in daily life, and different internet service providers.

(viii) Learn about Information Technology laws related to electronic commerce, electronic signatures, data protection, cybersecurity, penalties, offenses under the IT Act, and dispute resolution.

Generation of Computers	
A brief history of generation of computers, Super, Mainframe, Mini and Personal Computer.	2 L
Introduction to Computer Hardware and Softwares	
Components of modern computers: CPU, Primary and Secondary storage, I/O devices Software: Systems and Application	4 L
Number Systems	
A brief history of different number systems. Number representations and conversion rules in different number systems such as binary, octal, hexadecimal and decimal.	5 L
Boolean Algebra	
Fundamentals of Boolean Algebra, Switches and Inverters, Functionally Complete Gates (AND, OR, NOT), NAND, NOR. Switching function and Boolean function. De Morgan's theorem, Minterm and Maxterm, Truth table and minimization of switching function up to four variables, Algebraic and K-map method of logic circuit synthesis: Two level and Multi level.	8 L
Problem Solving	
Flow Charts, Decision Tables and Pseudo codes.	5 L
Programming Languages	
A brief history of Programming languages: Machine Language, Assembly Language, High Leve l Language.	4 L
Introduction to Database Management Systems	
What is DBMS? Difference between DBMS and File structure, Architectures of DBMS. Different Types of DBMS.	4 L
Internet	
History of the internet and different internet enabled services used in our daily life. Different internet service providers. Cloud services and service providers.	3 L
Information Technology Laws	
Information Technology laws provided to electronic commerce – electronic signatures, data protection, cyber security; penalties & offences under the IT Act, dispute resolution, and other contemporary issues.	5 L

Recommended Books and resources:

1. P. K. Sinha & Priti Sinha , ―Computer Fundamentals‖, BPB Publications, 2007.
2. Dr. Anita Goel, Computer Fundamentals, Pearson Education, 2010.
3. V. Rajaraman, Neeharika Adabala,- Fundamentals Of Computers, PHI
4. IT Act 2000(INDIA).

CS-SEC-P-1- Practical: Office Automation
Skill Enhancement Course, Practical, Semester – I, Credits - 03, Contact hours - 40

Course Objectives:

The course objective of this course is to teach students the fundamental concepts and principles of using technology and software to streamline office processes. It aims to enhance their proficiency in essential office automation tools such as word processing, spreadsheets and presentations. The course focuses on increasing productivity, improving communication, and promoting collaboration among team members using automation features.

Course Outcomes (COs):

After completing this course satisfactorily, a student will be able to:

- Compare and contrast various types of operating systems
- Explain the purpose of office automation
- Describe how information is stored and retried in/from computer memory
- Know about various types of office automation software and their applications
- Create document using word processing software
- Design presentation using presentation software
- Create worksheets and analyse data using spreadsheet software

Computer software and Introduction to Operating System and Installation
Computer software : Introduction, Software definition, Software categories, Installing and uninstalling software, Software piracy, Software terminologies. Introduction to Operating System with GUI, CUI and installation of different OS with required software. **Office package** : Introduction, Office user interface, Different office package software.
Word Processing
Introduction, Starting Word, working with word documents, working with text, working with tables checking spelling and grammar, adding graphs to the document, mail merge, header and footers, page numbers, protect the document, working with formatting tools.
Presentation
Starting Presentation, Working with Presentation, Creating, Saving and Printing a presentation, Working with Animation, adding a slide to presentation, Navigating through a presentation, Slide-sorter, Slide-show, Editing slides, Working with Graphics and Multimedia (Inserting Photo, Video & Sound)
Spreadsheet
Introduction, starting Spreadsheet, Spreadsheet environment, Working with Spreadsheet workbook, Working with worksheet – Entering data, formatting tips and Techniques, Generating graphs, Formulas and Functions, Inserting charts, sorting, Pivot Tables, data extraction, adding clip art, add an image from a file, Printing in Spreadsheet. **Formulas and Functions** Understanding formulas and cell references, basic mathematical operations, using common functions (e.g., SUM, AVERAGE, COUNT), applying absolute and relative cell references, nesting functions **Data Analysis and Manipulation** 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.
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.
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.
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).
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.
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.
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.
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
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.

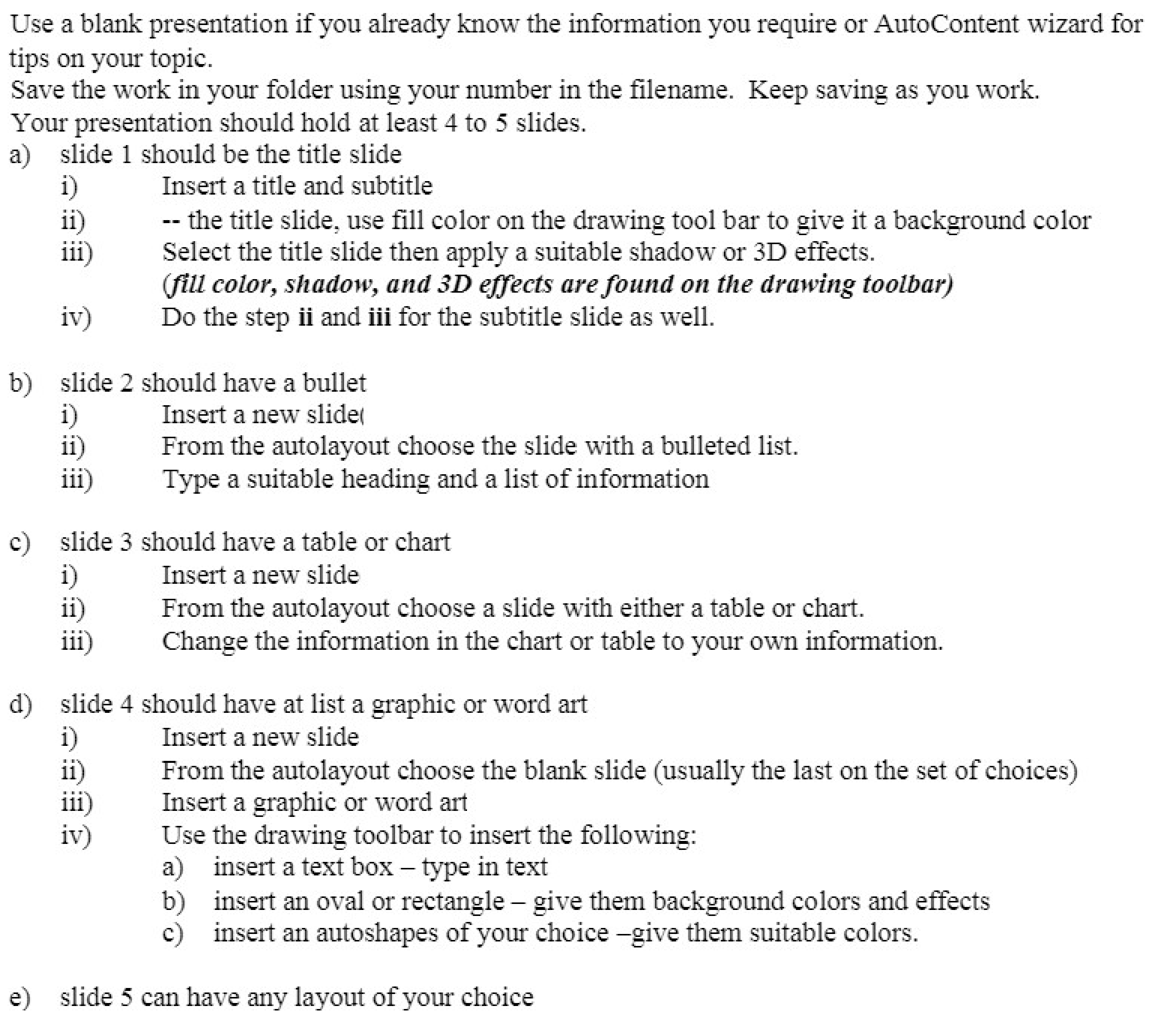
1. Open an existing word document or create a new one.
   Insert a table with a specific number of rows and columns.
   Enter data into the table and format it (e.g., adjusting cell width, changing cell background color).
   Add a new row and column to the existing table.
   Create a simple line graph based on the table data.
   Edit the graph to add axis labels, a title, and customize the appearance. Add a header with your name and student ID.
   Insert a footer with the document title and page count. Customize the header and footer layout and formatting.

Adjust the page numbering format (e.g., Roman numerals, starting from a specific page).

2. Prepare a sample file in Spreadsheet or CSV format, including fields like "Name," "Address," and "Email."

   Create a template for a standard letter.
   Perform a mail merge to personalize the letter with information from the database.
   Review and edit merged documents to ensure correct personalization. Print
   the personalized letters or save them as separate documents.

3. Choose a topic of your own interest and prepare a presentation according to the following instructions:



Use a blank presentation if you already know the information you require or AutoContent wizard for tips on your topic.
Save the work in your folder using your number in the filename. Keep saving as you work.
Your presentation should hold at least 4 to 5 slides.

a) slide 1 should be the title slide
   i) Insert a title and subtitle
   ii) -- the title slide, use fill color on the drawing tool bar to give it a background color
   iii) Select the title slide then apply a suitable shadow or 3D effects.
   (fill color, shadow, and 3D effects are found on the drawing toolbar)
   iv) Do the step **ii** and **iii** for the subtitle slide as well.

b) slide 2 should have a bullet
   i) Insert a new slide(
   ii) From the autolayout choose the slide with a bulleted list.
   iii) Type a suitable heading and a list of information

c) slide 3 should have a table or chart
   i) Insert a new slide
   ii) From the autolayout choose a slide with either a table or chart.
   iii) Change the information in the chart or table to your own information.

d) slide 4 should have at list a graphic or word art
   i) Insert a new slide
   ii) From the autolayout choose the blank slide (usually the last on the set of choices)
   iii) Insert a graphic or word art
   iv) Use the drawing toolbar to insert the following:
      a) insert a text box – type in text
      b) insert an oval or rectangle – give them background colors and effects
      c) insert an autoshapes of your choice –give them suitable colors.

e) slide 5 can have any layout of your choice

4. incorporate multimedia elements and graphics into a presentation.

   Tasks:

   Insert a relevant photo/image onto a specific slide in the presentation. Embed a video clip from an external source (e.g., YouTube) into another slide. Add background audio or sound effect to the presentation.

Ensure proper alignment and resizing of multimedia elements.
Test the multimedia elements to ensure they play correctly during a slideshow.

5. 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.

6. A dataset containing sales data for a company be provided. Create a spreadsheet 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.
7. 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.

8. 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.

9. 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.

10. 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.

11. 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.

12. 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.

13. Design an interactive dashboard using a spreadsheet. Combine various chart

types, slicers, and drop-down menus to allow users to explore and interact with the data dynamically. Create an intuitive and user-friendly interface.

14. 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.

15. To create a unique visualization using advanced spreadsheet features and tools. For example, an experiment with sparklines, radar charts, or tree maps 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.

Detail Syllabus of Semester-II

CS-MJ-T-2: Digital System Design

Major Course, Theory, Semester – II, Credits - 04, Contact hours - 40

Course Objectives: This course will teach the basic understanding of Digital Logic Design and its application for digital computer. The course focuses on understanding the Introduction to organization of digital computer, Concept of Memory, Boolean Algebra & its Simplification of Boolean Functions, and Combinational Logic and Sequential Logic.

Course Outcomes: On successful completion of the course, students will be able to
(i) explain the concept of organization of digital computer its different hardware components such as Input Unit, Output Unit, Storage Unit, CPU. Control Unit, Arithmetic Logic Unit
(ii) solve problems on different number systems, binary arithmetic operation, floating point number and signed magnitude number representation, overflow, under flow, and different computer error detection and correction codes
(iii) have thorough idea on memory Hierarchy, and different types of memory, hit and miss
(iv) solve problems on Boolean algebra and simplification of boolean Functions
(v) Design different digital *Combinational and Sequential Logic circuitry.*

Combinational Circuits	
Realization of AND and OR Gates using diodes and NOT Gate using transistors, Standard Gate Assemblies, IC chips packaging nomenclature, Half and Full Adder(3 bits), Multi-bit adders – Ripple carry and Carry Look Ahead Adder, Adder/subtractor, BCD-Adder, Data selectors/multiplexers – expansions, reductions, function realization, universal function realization, multi-function realization, Decoders/Demultiplexers : function realization, De-multiplexer and function realization, Encoder, Priority Encoder, Parity bit Generator/checker, Gray Code Generator, Code Converters, I/O features of BCD to 7- segment LED decoder/driver(7447/7448), Seven segment display unit, Comparators.	
Sequential Circuits	
Model of Sequential computing, Difference between Combinational and Sequential circuit, RS-Latch: using NAND and NOR Gates, RS Latch as a Static RAM Cell, Problems of Basic Latch circuits, Digital Clock – Duty Cycle, Rising time, Falling time, Clocked Flip Flops - SR, JK, D, T, Level Trigger and Edge Trigger, Excitation Functions of each flip-flops, Flip-flops with Preset and Clear, Application of Flip-flops: Asynchronous Counter (UP/DOWN) up to 4 bit counter, Mod – n Counter, Synchronous Counters – different mod counters, Ring counter, Registers: Registers with serial and parallel load, Shift Registers.	
Data Converter	
D/A Conversion principle using basic circuit, R-2R Ladder circuit, Counter based A/D converter, Successive approximation method for A/D conversion. DTL and TTL NAND gate circuits and its operations, Fan in & Fan out, Noise margin, SSI, MSI, LSI, and VLSI classifications.	

Recommended Books:

(1) Digital Circuits, Combinational Circuit, Vol. 1 by D. Roy Choudhuri, Platinam Publication.
(2) Digital Circuits, Sequential Circuit, Vol. 2 by D. Roy Choudhuri, Platinam Publication.
(3) Digital Logic and Computer Design by M.Morris Mano, PHI
(4) Digital Principle and Applications by Malvino & Leach, TMH
(5) Digital Systems Principles and Applications by Ronal J. Tocci and Neal S. Widmer, PHI
(6) Digital Fundamentals by Floyd, Pearson Education

CS-MJ-P-2 : Digital System Design Lab
Major Course, Practical, Semester – II, Credits - 02, Contact hours - 40

Combinational Circuits

1. Implement Half Adder/Half Subtractor / Full Adder / Full Subtractor using Logic Gates. Realize a logic function using basic/universal gates in SOP and POS form. Study the functionalities of 7483 and design a BCD adder using 7483 or equivalent.
2. Design a 4 bit 2‘s complement adder – subtractor unit using 7483 or equivalent and XOR gates.
3. Design a circuit to convert BCD numbers to corresponding gray codes.
4. Design a 4:1 MUX using NAND gates. Study of 74153 and 74151. Design Full Adder / Subtractor using MUX.
5. Design a 2:4 decoder using NAND gates. Study of 74155 and 74138. Design Full Adder / Subtractor using decoders.
6. Design a parity generator/checker using basic gates.
7. Design magnitude comparator using basic/universal gates. Study of 7485.
8. Design a seven-segment display unit.

Sequential Circuits

1. Realize S-R, D, J-K and T flip-flop using basic gates. (Study the undefined state in S-R flip-flop).
2. Study the functional characteristic of IC 74194 with emphasis on timing diagram.
3. Design Asynchronous and Synchronous counters. (Mod-8, Mod-10 up counter).
4. Study the functional characteristics of RAM IC chip. Study of open collector and tri- state output. Horizontal expansion of RAM chips by cascading. (Use 74189, 7489, or any available chip).

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

CS-MU-P-2: Practical, Office Automation
Multidisciplinary Course, Practical, Semester – II, Credits - 03, Contact hours – 40

Course Objectives:
The course objective of this course is to teach students the fundamental concepts and principles of using technology and software to streamline office processes. It aims to enhance their proficiency in essential office automation tools such as word processing, spreadsheets and presentations. The course focuses on increasing productivity, improving communication, and promoting collaboration among team members using automation features.

Course Outcomes (COs):
After completing this course satisfactorily, a student will be able to:
- Compare and contrast various types of operating systems
- Explain the purpose of office automation
- Know about various types of office automation software and their applications
- Create document using word processing software
- Design presentation using presentation software
- Create worksheets and analyse data using spreadsheet software

Computer software and Introduction to Operating System
Computer software: Introduction, Software definition, Software categories, Installing and uninstalling different operating systems and software, Software piracy, Software terminologies .   **Introduction to Operating System**, operating with GUI and CUI, use of help features, starting an application, essential accessories, creating shortcuts, control panel, finding

folders and files, System utilities. **Office package**: Introduction, Office user interface, Different office packaged software **.**
Word Processing
Introduction, Starting Word, working with word documents, working with text, working with tables checking spelling and grammar, adding graphs to the document, mail merge, header and footers, page numbers, protect the document, working with formatting tools.
Spreadsheet
Introduction, starting Spreadsheet, Spreadsheet environment, Working with Spreadsheet workbook, working with worksheet – Entering data, formatting tips and Techniques, Generating graphs, Formulas and Functions, Inserting charts, Sorting, Pivot Tables, data extraction, adding clip art, add an image from a file, Printing in Spreadsheet.
Presentation
Starting presentation, Working with Presentation, Creating, Saving and Printing a presentation, Working with Animation, adding a slide to presentation, navigating through a presentation, Slide-sorter, Slide-show, Editing slides, Working with Graphics and Multimedia (Inserting Photo, Video & Sound).

List of Suggested Practical / Laboratory Experiments to be conducted but not limited to the following:

1. Open an existing word document or create a new one.

   Insert a table with a specific number of rows and columns.

   Enter data into the table and format it (e.g., adjusting cell width, changing cell background color).

   Add a new row and column to the existing table.

   Create a simple line graph based on the table data.

   Edit the graph to add axis labels, a title, and customize the appearance. Add a header with your name and student ID.

   Insert a footer with the document title and page count. Customize the header and footer layout and formatting.

   Adjust the page numbering format (e.g., Roman numerals, starting from a specific page).
2. Prepare a sample file in Spreadsheet or CSV format, including fields like "Name," "Address," and "Email."

   Create a template for a standard letter.

   Perform a mail merge to personalize the letter with information from the file.

   Review and edit merged documents to ensure correct personalization. Print the personalized letters or save them as separate documents.
3. Choose a topic of your own interest and prepare a presentation according to the following instructions:

Use a blank presentation if you already know the information you require or AutoContent wizard for tips on your topic.
Save the work in your folder using your number in the filename. Keep saving as you work.
Your presentation should hold at least 4 to 5 slides.

a) slide 1 should be the title slide
   i) Insert a title and subtitle
   ii) -- the title slide, use fill color on the drawing tool bar to give it a background color
   iii) Select the title slide then apply a suitable shadow or 3D effects.
   (fill color, shadow, and 3D effects are found on the drawing toolbar)
   iv) Do the step **ii** and **iii** for the subtitle slide as well.

b) slide 2 should have a bullet
   i) Insert a new slide(
   ii) From the autolayout choose the slide with a bulleted list.
   iii) Type a suitable heading and a list of information

c) slide 3 should have a table or chart
   i) Insert a new slide
   ii) From the autolayout choose a slide with either a table or chart.
   iii) Change the information in the chart or table to your own information.

d) slide 4 should have at list a graphic or word art
   i) Insert a new slide
   ii) From the autolayout choose the blank slide (usually the last on the set of choices)
   iii) Insert a graphic or word art
   iv) Use the drawing toolbar to insert the following:
      a) insert a text box – type in text
      b) insert an oval or rectangle – give them background colors and effects
      c) insert an autoshapes of your choice –give them suitable colors.

e) slide 5 can have any layout of your choice

4. Incorporate multimedia elements and graphics into a presentation.

   Tasks:

   Insert a relevant photo/image onto a specific slide in the presentation. Embed a video clip from an external source (e.g., YouTube) into another slide. Add background audio or sound effect to the presentation.
   Ensure proper alignment and resizing of multimedia elements.
   Test the multimedia elements to ensure they play correctly during a slideshow.
5. 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.
6. A dataset containing sales data for a company be provided. Create a spreadsheet 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.
7. 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.

8. 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.
9. 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.
10. 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.

CS-SEC-P-2: Practical, Web Development and Applications
Skill Enhancement Course, Practical, Semester – II, Credits - 03, Contact hours – 40

Course Objectives: This course is intended to give the students an introduction to the Hypertext Markup Language and its various components. Use of different HTML components like links, images, tables, headers, forms, CSS, etc., to design an ergonomic and efficient webpage will be taught in this course.

Course Outcomes: On successful completion of the course, students will be able to:
(i) Understand basics of HTML.
(ii) Use of CSS in web design.
(iii)Understand the basics of Javascript
(iv)They will be able to build basic static web pages.

Hypertext Markup Language (HTML)
The Basics: The Head, the Body, Colors, Attributes, Lists (ordered and unordered). • **Links:** Introduction, Relative Links, Absolute Links, Link Attributes, Using the ID Attribute to Link Within a document • **Images:** Putting an Image on a Page, Using Images as Links, Putting an Image in the Background. • **Table:** Creating a Table, Table Headers, Captions, Spanning Multiple columns, Styling Table • **Form:** Basic Input and Attributes, Other Kinds of Inputs,
Cascading Style Sheets (CSS):
• Introduction • CSS Basics • Anatomy of a CSS ruleset • Different types of selectors • Fonts and text
JavaScript Fundamentals

• Data types and variables, • Functions, methods and events, • Controlling program flow, • JavaScript object model, built-in objects • Operators
Document Object Model (DOM)
• DOM Tree • Accessing DOM elements by ID , Class and Tag

List of Suggested Practical / Laboratory Experiments to be conducted but not limited to the following:

1. Create HTML document with following formatting – Bold, Italics, Underline, Colors, Headings, Title, Font and Font Width, Background, Paragraph, Line Brakes, Horizontal Line, Blinking text as well as marquee text.

2. Create HTML document with Ordered and Unordered lists, Inserting Images, Internal and External linking.

3. Create HTML document with Table:

			Some Image Here	

4. Create Form with Input Type, Select and Text Area in HTML.

5. Create an HTML containing Roll No., student's name and Grades in a tabular form.

6. Create an HTML document (having two frames) which will appear as follows:

About	This frame would show the contents according to the link clicked by the user on the left frame.
Department 1	
Department 2	
Department 3	

7. Create an HTML document containing horizontal frames as follows:

Department Names (could be along with Logos)
Contents according to the Link clicked

8. Create a website of 6 – 7 pages with different effects as mentioned in above problems.
9. Create HTML documents (having multiple frames) in the following three formats:

Frame 1
Frame 2

Frame 1	
Frame 2	Frame 3

List of Practical Assignments to be done using Javascript :

Create event driven program for following:

1. Print a table of numbers from 5 to 15 and their squares and cubes using alert.

2. Print the largest of three number

3. Find the factorial of a number n.

4. Enter a list of positive numbers terminated by Zero. Find the sum and average of these numbers.

5. A person deposits Rs 1000 in a fixed account yielding 5% interest. Compute the amount in the account at the end of each year for n years.

6. Read n numbers. Count the number of negative numbers, positive numbers and zeros in the list.

Suggested Readings:
Text Books:
1. Learning Web Design: A Beginner`s Guide To HTML, CSS, JavaScript, And Web Graphics
2. D.R. Brooks, An Introduction to HTML and Javascript for Scientists and Engineers, Springer
W. Willard,2009

Reference Books/Links:
1. HTML A Beginner's Guide, Tata McGraw-Hill Education, 2009
2. J. A. Ramalho, Learn Advanced HTML 4.0 with DHTML, BPB Publications, 2007
3. https://developer.mozilla.org/en-US/docs/Web/CSS4.
4. https://developer.mozilla.org/en-US/docs/Web/HTML
5. https://developer.mozilla.org/en-US/docs/Web/JavaScript

Detail Syllabus of Semester-III

CS-MJ-T-3- Theory: Computer Organization & Architecture

Major Course, Theory, Semester – III, Credits - 06, Contact hours - 60

Course description:

This course provides a foundational understanding of digital logic, data representation, computer organization, and the architecture of modern computer systems. It begins with basic concepts like logic gates, Boolean algebra, and sequential circuits. Students will then explore number systems, computer arithmetic, and essential components of computer organization, such as the central processing unit (CPU) and memory. The course also covers input/output organization, memory hierarchy, and key architectural principles like RISC, CISC, and pipelining. By the end of the course, students will gain practical insights into how computers process, store, and retrieve data, enabling them to understand the inner workings of modern computer systems.

Course Outcomes (COs):

1. After completing this course satisfactorily, a student will be able to:
Demonstrate proficiency in number systems, fixed and floating-point representation, and ALU operations.
2. Understand the structure and operation of essential computer components.
3. Understand CPU design and operation
4. Understand memory hierarchy and organization
5. Describe input-output organization

1. **Introduction** 9L
Logic gates, Boolean algebra, combinational circuits, circuit simplification, flip-flops and sequential circuits, decoders, multiplexers, registers, counters and memory units.

2. **Data Representation and Basic Computer Arithmetic** 12L
Number systems, complements, fixed and floating point representation, character representation, addition, subtraction, magnitude comparison, multiplication and division algorithms for integers.

3. **Basic Computer Organization and Design** 10L
Computer registers, bus system, instruction set, timing and control, instruction cycle, memory reference, input-output and interrupt, Interconnection Structures, Bus Interconnection design of basic computer.

4. **Central Processing Unit** 9L
Register organization, arithmetic and logical micro-operations, stack organization, micro programmed control. Instruction formats, addressing modes, instruction codes, machine language, assembly language, input output programming, RISC, CISC architectures, pipelining and parallel architecture.

5. **Memory Organization** 10L
Cache memory, Associative memory, mapping.

6. **Input-Output Organization** 10L
Input / Output: External Devices, I/O Modules, Programmed I/O, Interrupt-Driven I/O, Direct

Memory Access, I/O Channels.

Recommended Books:

1. M. Mano, Computer System Architecture, Pearson Education 1992
2. A. J. Dos Reis, Assembly Language and Computer Architecture using C++ and JAVA, Course Technology, 2004
3. W. Stallings, Computer Organization and Architecture Designing for Performance, 8th Edition, Prentice Hall of India,2009
4. M.M. Mano , Digital Design, Pearson Education Asia,2013
5. Carl Hamacher, Computer Organization, Fifth edition, McGrawHill, 2012.

CS-MI-T-3: Database Management Systems

Minor Course, Theory, Semester – III, Credits - 01, Contact hours – 30

Course Objective: The importance of a database, relational data model, schema design and normalization, transaction processing, indexing, and the related data structures (files and B+- trees) are covered in this course.

Course Outcome: On successful completion of the course, students will be able to:

(i) Gain knowledge of database systems and database management systems software.
(ii) Ability to model data in applications using conceptual modelling tools such as ER Diagrams and design data base schemas based on the model.
(iii) Formulate, using SQL, solutions to a broad range of query and data update problems.
(iv) Demonstrate an understanding of normalization theory and apply such knowledge to the normalization of a database.
(v) Be acquainted with the basics of transaction processing and concurrency control. Become familiar with database storage structures and access techniques. Compare, contrast and analyse the various emerging technologies for database systems such as NoSQL.
(vi) Analyse strengths and weaknesses of the applications of database technologies to various subject areas.

Course Details:

Introduction

Concept & Overview of DBMS, Data Models, Database Languages, Database Administrator, Database Users, Three Schema architecture of DBMS. Characteristics of database approach, data models, database system architecture and data independence. 3L

Entity Relationship(ER) Modeling

Basic concepts, Design Issues, Mapping Constraints, Keys, Entity-Relationship Diagram, Weak Entity Sets, Extended E-R features. 5L

Relation data model

Structure of relational Databases, Relational Algebra, Relational Calculus, Extended Relational Algebra Operations, Views, Modifications Of the Database. 5L

SQL and Integrity Constraints

Concept of DDL, DML, DCL. Basic Structure, Set operations, Aggregate Functions, Null Values, Domain Constraints, Referential Integrity Constraints, assertions, views, Nested Subqueries, Database security application development using SQL, Stored procedures and triggers 7L

Relational Database Design

Functional Dependency, Different anomalies in designing a Database., Normalization using functional dependencies, Decomposition, Boyce-Codd Normal Form, 3NF, Normalization using multi-valued dependencies, 4NF, 5NF 7L

Internals of RDBMS	
Physical data structures, Query optimization: join algorithm, statistics and cost bas optimization. Transaction processing	3L
Transaction Processing	
ACID properties, Concurrency control and Recovery Management : transaction model properties, state serializability, lock base protocols, two phase locking	5L
File Structure and Indexing	
Operations on files, File of Unordered and ordered records, overview of File organizations, Indexing structures for files (Primary index, secondary index, clustering index), Multilevel indexing using B and B+ trees.	5L

Recommended Books:

1. Henry F. Korth and Silberschatz Abraham, ―Database System Concepts‖, Mc.Graw Hill.
2. Elmasri Ramez and Novathe Shamkant, ―Fundamentals of Database Systems‖, Benjamin Cummings Publishing. Company.
3. Ramakrishnan: Database Management System, McGraw-Hill
5. Date C. J., ―Introduction to Database Management‖, Vol. I, II, III, Addison Wesley.
1. Ullman JD., ―Principles of Database Systems‖, Galgottia Publication.

CS-MI-P-3: Database Management Systems Lab

Minor Course, Practical, Semester – III, Credits - 01, Contact hours – 30

Course Objectives:

The course objectives are to equip students with practical skills in creating and managing databases. They will learn to design efficient database schemas, write complex SQL queries to retrieve and manipulate data, and administer databases effectively. The course also aims to provide real-world application scenarios for students to apply their SQL knowledge and problem-solving skills. By the end of the course, students should be proficient in handling relational databases and SQL queries.

Course Outcomes:

(i) Proficiency in SQL
(ii) Database Design Skills
(iii) Data Manipulation and Administration

List of Suggested Practical / Laboratory Experiments to be conducted on the following topics:

Structured Query Language (SQL)

1. **Creating Database:**
   Creating a Database Creating a Table
   Specifying Relational Data Types Specifying Constraints Creating
   Indexes
2. **Table and Record Handling:**
   INSERT statement
   Using SELECT and INSERT together DELETE, UPDATE, TRUNCATE statements DROP, ALTER statements
3. **Retrieving Data from a Database:**
   The SELECT statement Using the WHERE clause Using
   Logical Operators in the WHERE clause

Using IN, BETWEEN, LIKE , ORDER BY, GROUP BY and HAVING clasue
Using Aggregate Functions Combining Tables Using JOINS Subqueries

4. Condition specification using Boolean and comparison operators (and, or, not,=,<>,>,<,>=,<=)
5. Arithmetic operators and aggregate functions(Count, sum, avg, Min, Max)
6. Multiple table queries (join on different and same tables)
7. Nested select statements
8. Set manipulation using (any, in, contains, all, not in, not contains, exists, not exists, union, intersect, minus, etc.)
9. Create a database and tables for a simple student management system.
10. Insert, update, delete, and retrieve data using SQL commands.
11. Use SELECT with WHERE, ORDER BY, GROUP BY, and HAVING clauses.
12. Create tables with PRIMARY KEY, FOREIGN KEY, UNIQUE, NOT NULL, and CHECK constraints.
13. Demonstrate ALTER TABLE to modify schema and constraints.
14. Retrieve specific records using SELECT with DISTINCT, LIKE, and BETWEEN operators.
15. Use aggregate functions (COUNT, SUM, AVG, MIN, MAX) with GROUP BY.
16. Implement UNION, INTERSECT, and EXCEPT operations.
17. Perform JOIN operations: INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL OUTER JOIN, and SELF JOIN.
18. Write queries using IN, EXISTS, ANY, ALL, and CORRELATED SUBQUERIES.
19. Retrieve data from multiple tables using nested queries.
20. Create and manipulate VIEWS to simplify complex queries.
21. Implement transactions using COMMIT, ROLLBACK, and SAVEPOINT.
22. Convert unnormalized tables into 1NF, 2NF, and 3NF.
23. Design a relational database schema from an ER model.

1. Develop a simple database project (e.g., Library Management System, Employee Records, Online Store) using SQL.

CS-MU-T-3- Theory: AI for Everyone

Multidisciplinary Course, Theory, Semester – III, Credits - 03, Contact hours - 40.

Course Objectives:

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

1. Understand the foundational concepts and evolution of Artificial Intelligence (AI).
2. Explore key AI subfields like machine learning, deep learning, and natural language processing through real-world applications.
3. Identify and assess the impact of AI in sectors such as healthcare, finance, transportation, and education.
4. Analyze the ethical, social, and economic implications of AI, including concerns around bias, privacy, and job displacement.
5. Discuss emerging trends and future possibilities of AI, including its role in creativity, innovation, and responsible use.

Introduction to Artificial Intelligence: Definition and scope of AI; historical overview and key milestones; differentiating AI from human intelligence.

AI Subfields and Technologies: Introduction and basic concepts of machine learning, including supervised, unsupervised, and reinforcement learning, deep learning and neural networks (without technical details); basic concepts of natural language processing (NLP) and computer vision.

Applications of AI: AI in healthcare (diagnosis, treatment, medical imaging); AI in finance (fraud detection, algorithmic trading, risk assessment); AI in transportation (autonomous vehicles, traffic optimization); AI in education (personalized learning, intelligent tutoring systems).

Ethical and Social Implications of AI: Bias and fairness in AI systems; privacy and data protection concerns; impact of AI on employment and the workforce; AI and social inequality.

Emerging Issues and Future Trends: Ethical guidelines and responsible AI practices; AI and innovation; emerging trends and future directions in AI; AI and creativity (generative models, artistic applications).

Books:

1. S. Goswami, A. K. Das, A. Chakrabarti, "AI for Everyone: A Beginner's Handbook for Artificial Intelligence (AI)", Pearson, 2024.

2. P. Verdegem, "AI for Everyone?: Critical Perspectives", University of Westminster Press, 2021.

3. Shawn Schuster, "AI For All: How Everyday People Can Benefit from Artificial Intelligence", UMLAUT Publishing, 2021.

CS-SEC-P-3- Practical: Data Analysis through Python/R(Lab)

Skill Enhancement Course, Practical, Semester – III, Credits - 03, Contact hours - 40.

Objective:

The objective of the "Data Analysis through Python/R (Lab)" course is to equip students with the skills and knowledge required to analyze, manipulate, and visualize data using Python or R programming. The course focuses on fundamental data analysis techniques, data preprocessing, exploratory data analysis (EDA), statistical modeling, and machine learning. Students will learn how to work with real-world datasets, develop data-driven insights, and apply various algorithms and models to extract meaningful information. The course also emphasizes effective communication of findings through visualizations and reports.

Course Outcomes:

Upon successful completion of this course, students will be able to:

1. Understand and apply fundamental data structures in Python/R for data manipulation.
2. Import, clean, and preprocess datasets from various sources such as CSV, Excel, and databases.
3. Perform exploratory data analysis (EDA) using statistical measures and visualizations to identify patterns and trends.
4. Implement and apply various data transformation techniques such as normalization, standardization, and feature engineering.
5. Conduct statistical tests and understand concepts like correlation and hypothesis testing.
6. Build and evaluate regression and classification models using Python/R for predictive analytics.
7. Apply clustering techniques for unsupervised learning and pattern discovery.
8. Analyze and forecast time series data using appropriate methods such as ARIMA or exponential smoothing.
9. Perform dimensionality reduction using techniques like Principal Component Analysis (PCA).
10. Communicate findings and insights effectively through visualizations and comprehensive reports. Develop and evaluate end-to-end data analysis projects, including data acquisition,

analysis, model building, and result presentation.

Sample Programs:

1. Write a Python/R script to create, manipulate, and perform basic operations on lists (Python) or vectors (R). Perform operations such as slicing, indexing, and appending elements.
2. Write a Python/R program to load and manipulate data using dictionaries (Python) or data frames (R).
3. Write a Python/R script to load a CSV file into a Pandas DataFrame (Python) or a data frame (R). Perform basic operations like viewing the first few rows, summary statistics, and exporting the modified dataset to a new CSV file. Import data from an Excel file and perform similar operations.

4. Write a Python/R script to identify and handle missing data (drop or impute missing values) in a dataset. Detect and treat outliers using statistical techniques. Perform data normalization or standardization on numerical columns.
5. Perform univariate and multivariate analysis using Python (Pandas, Matplotlib, Seaborn) or R (ggplot2, dplyr).
   Generate summary statistics (mean, median, mode, standard deviation, etc.) for numerical columns.
   Create visualizations such as histograms, boxplots, scatter plots, and pair plots to explore relationships between variables.
6. (a) Write a Python/R script to apply log transformations, binning, or scaling to numerical data.
   (b) Create new features using existing ones (e.g., adding a column for a calculated field).
   (c) Perform one-hot encoding for categorical variables and label encoding for target variables.
7. Write a Python/R script to compute and visualize the correlation matrix for a dataset. Perform hypothesis testing using t-tests or ANOVA to compare means between groups. Apply chi-square tests for independence on categorical data.
8. Create line plots, bar plots, pie charts, and heatmaps using Python (Matplotlib/Seaborn) or R (ggplot2).
   Create advanced visualizations like violin plots, KDE plots, and facet grids to represent multi-dimensional data.
   Customize plots by adding titles, labels, legends, and changing color schemes.

9. Implement a simple linear regression model in Python (Scikit-learn) or R to predict a target variable. Evaluate the model using metrics like R-squared, mean squared error, and visualize the regression line. Extend to multiple linear regression and evaluate its performance.

10. Implement a logistic regression model in Python/R to classify binary data. Evaluate model performance using confusion matrix, accuracy, precision, recall, and F1-score. Explore other classification algorithms like Decision Trees and K-Nearest Neighbors (KNN) for comparison.
11. Implement the K-means clustering algorithm in Python/R to group data based on similarity. Visualize clusters and calculate the silhouette score to evaluate cluster quality. Apply hierarchical clustering and compare the results with K-means.
12. Load and visualize time series data using Python (Pandas) or R. Perform decomposition of time series into trend, seasonality, and residuals. Implement ARIMA or exponential smoothing to forecast future data points.
13. Write a Python/R script to perform PCA on a dataset and reduce its dimensions. Visualize the explained variance and transformed data. Use the reduced dataset to perform further analysis or machine learning.
14. Implement cross-validation techniques to evaluate model performance. Use techniques like GridSearchCV (Python) or tune. grid (R) to find the best hyperparameters for models. Compare multiple models and choose the best one based on performance metrics.

Detail Syllabus of Semester-IV

CS-MJ-T-4- Theory: Data Structure

Major Course, Theory, Semester – IV, Credits - 04, Contact hours - 40

Objective: This course aims to provide a comprehensive understanding of data structures, covering both linear and non-linear structures, their representations, and applications. Students will learn about arrays, stacks, queues, linked lists, recursion, trees, and graphs, along with efficient searching, sorting, and hashing techniques. The course emphasizes the implementation of data structures, including stack and queue operations, binary trees, AVL trees, B-trees, and graph traversal methods like BFS and DFS. Additionally, students will explore various sorting algorithms, collision resolution in hashing, and file structures such as indexed and hashed files. By the end of the course, students will gain a strong foundation in data organization, algorithmic efficiency, and problem-solving techniques essential for software development.

Outcome:

1. Explain basic definitions, classifications, and representations of data structures, including Abstract Data Types (ADT).
2. Utilize single and multi-dimensional arrays, represent sparse matrices, and apply row-major and column-major ordering.
3. Implement stack operations, convert between prefix, infix, and postfix expressions, evaluate postfix expressions, and explore stack-based applications.
4. Develop and manipulate singly, doubly, and circular linked lists, including self-organizing and skip lists.
5. Understand queue operations, including Deque and Priority Queues, using both array and linked representations.
6. Develop recursive solutions, understand recursion's advantages and limitations, and optimize recursion using tail recursion or removal techniques.
7. Implement and traverse binary trees (BSTs, AVL trees, B-trees, B+ trees), and apply BFS and DFS algorithms for graph traversal.
8. Analyze and implement various searching (linear, binary) and sorting algorithms (Bubble, Selection, Merge, Heap, Radix, etc.), comparing their efficiency.
9. Implement hashing, resolve collisions using different techniques (Open Addressing, Separate Chaining, Perfect Hashing), and optimize hash functions.
10. Understand sequential, direct access, indexed (B+ tree), multi-indexed, inverted, and hashed file organization for efficient data management.

Course description:

1. Basic definitions; classifications; ADT; Linear Data Structures - Sequential representations; Non- linear data structures – representations. 2L
2. Arrays
   Single and Multi-dimensional Arrays; Sparse Matrices (Array and Linked Representation); Row- major and column-major order; different applications 4L
3. Stacks
   Implementing single / multiple stack/s in an Array; Prefix, Infix and Postfix expressions, Utility and conversion of these expressions from one to another; Evaluation of postfix expression using stack; Applications of stack; Limitations of Array representation of stack 6L

4. Linked Lists
   Singly, Doubly and Circular Lists (Array and Linked representation); Normal and Circular representation of Stack in Lists; Self Organizing Lists; Skip Lists 10L
5. Queues
   Array and Linked representation of Queue, De-queue, Priority Queues
6. 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; When not to use recursion; Removal of recursion. 4L
7. Trees and Graphs
   Introduction to Tree as a data structure; Binary Trees (Insertion, Deletion , Recursive and Iterative Traversals on Binary Search Trees); Different properties of Binary trees; Threaded Binary Trees (Insertion, Deletion, Traversals); Height-Balanced Trees (Various operations on AVL Trees); B- tree, B+ tree; Graphs - Representations, Breadth-first and Depth-first Search. 10L
8. Searching and Sorting
   Linear Search, Binary Search, Comparison of Linear and Binary Search; Bubble Sort, Selection Sort, Insertion Sort, Heap Sort, Merge Sort, Radix Sort, Bucket Sort, Shell Sort; Comparison of Sorting Techniques 10L
9. Hashing
   Introduction to Hashing, Deleting from Hash Table, Efficiency of Rehash Methods, Hash Table Reordering, Resolving collision by Open Addressing, Coalesced Hashing, Separate Chaining, Dynamic and Extendible Hashing, Choosing a Hash Function, Perfect Hashing Function 4L
10. File Structures
   Sequential and Direct Access. Relative Files, Indexed Files - B+ tree as index. Multi-indexed Files, Inverted Files, Hashed Files. 4L

Reference Books:

1. Adam Drozdek, "Data Structures and algorithm in C++", Third Edition, Cengage Learning, 2012.
2. SartajSahni, Data Structures, "Algorithms and applications in C++", Second Edition, Universities Press, 2011.
3. Aaron M. Tenenbaum, Moshe J. Augenstein, YedidyahLangsam, "Data Structures Using C and C++:, Second edition, PHI, 2009.
4. Robert L. Kruse, "Data Structures and Program Design in C++", Pearson,1999.
5. D.S Malik, Data Structure using C++,Second edition, Cengage Learning, 2010.
6. Mark Allen Weiss, "Data Structures and Algorithms Analysis in Java", Pearson Education, 3rd edition, 2011
7. Aaron M. Tenenbaum, Moshe J. Augenstein, YedidyahLangsam, "Data Structures Using Java, 2003.
8. Robert Lafore, "Data Structures and Algorithms in Java, 2/E", Pearson/ Macmillan Computer Pub,2003
9. John Hubbard, "Data Structures with JAVA", McGraw Hill Education (India) Private Limited; 2 edition, 2009
10. Herbert Schildt, "Java The Complete Reference (English) 9th Edition Paperback", Tata McGraw Hill, 2014.
11. D. S. Malik, P.S. Nair, "Data Structures Using Java", Course Technology, 2003.

CS-MJ-T-4-: Data Structure Lab

Major Course, Practical, Semester – IV, Credits - 02, Contact hours - 40

Objective:

The objective of the Data Structure Lab is to provide hands-on experience in implementing and analyzing various data structures such as arrays, linked lists, stacks, queues, trees, and graphs. Students will develop proficiency in writing programs for operations like insertion, deletion, traversal, searching, and sorting while understanding their time and space complexities. The lab emphasizes the implementation of stack-based applications, expression evaluation, and recursion. It also covers tree and graph traversal techniques such as BFS and DFS, hashing for efficient data retrieval, and collision resolution methods. Additionally, students will work with file structures, including indexed and hashed files, to understand data storage and retrieval mechanisms. Through practical exercises, the lab enhances problem-solving and algorithmic thinking skills essential for real-world applications.

Outcome:

1. Implement and analyze various data structures, including arrays, linked lists, stacks, queues, trees, and graphs.
2. Perform insertion, deletion, traversal, searching, and sorting operations while evaluating their time and space complexities.
3. Apply stack-based applications such as expression evaluation and conversion between infix, prefix, and postfix notations.
4. Develop and implement recursion, understand its internal workings, and optimize it when necessary.
5. Utilize tree and graph traversal techniques, including Breadth-First Search (BFS) and Depth-First Search (DFS).
6. Apply hashing techniques for efficient data retrieval and implement collision resolution strategies.
7. Work with indexed and hashed file structures for effective data storage and retrieval.
8. Gain hands-on experience in implementing data structures to solve real-world problems.
9. Enhance algorithmic thinking and problem-solving skills through practical coding exercises.

Sample Programs:

1. Write a program to search an element from a list. Give user the option to perform Linear or Binary search. Use Template functions.
2. WAP using templates 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 using templates. Include functions for insertion, deletion and search of a number, reverse the list and concatenate two linked lists.
4. Implement Doubly Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list.
5. Implement Circular Linked List using templates. 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. Use Templates.
8. Perform Queues operations using Circular Array implementation. Use Templates.
9. Create and perform different operations on Double-ended Queues using Linked List implementation.
10. WAP to scan a polynomial using linked list and add two polynomial.
11. WAP to calculate factorial and to compute the factors of a given no. (i)using recursion, (ii) using

iteration

12. (ii) WAP to display fibonacci series (i)using recursion, (ii) using iteration
13. WAP to calculate GCD of 2 number (i) with recursion (ii) without recursion
14. WAP to create a Binary Search Tree and include following operations in tree: (a) Insertion (Recursive and Iterative Implementation) (b) Deletion by copying (c) Deletion by Merging (d) Search a no. in BST (e) Display its preorder, postorder and inorder traversals Recursively (f) Display its preorder, postorder and inorder traversals Iteratively (g) Display its level-by-level traversals (h) Count the non-leaf nodes and leaf nodes (i) Display height of tree (j) Create a mirror image of tree (k) Check whether two BSTs are equal or not
15. WAP to convert the Sparse Matrix into non-zero form and vice-versa.
16. WAP to reverse the order of the elements in the stack using additional stack.
17. WAP to reverse the order of the elements in the stack using additional Queue.
18. WAP to implement Diagonal Matrix using one-dimensional array.
19. WAP to implement Lower Triangular Matrix using one-dimensional array.
20. WAP to implement Upper Triangular Matrix using one-dimensional array.
21. WAP to implement Symmetric Matrix using one-dimensional array.
22. WAP to create a Threaded Binary Tree as per inorder traversal, and implement operations like finding the successor / predecessor of an element, insert an element, inorder traversal.
23. WAP to implement various operations on AVL Tree.

CS-MJ-T-5- Theory: Discrete Mathematics

Major Course, Theory, Semester – IV, Credits - 06, Contact hours - 60

Objective:

This course aims to provide a fundamental understanding of discrete mathematics concepts essential for computer science. It covers sets, functions, relations, binary relations, and their properties, along with combinatorial principles such as the Pigeonhole Principle, Permutations, Combinations, Mathematical Induction, and the Principle of Inclusion and Exclusion. Students will learn about the growth of functions, asymptotic notations, summation techniques, and integral approximations. The course also introduces recurrence relations, generating functions, and various solution techniques, including the Substitution Method, Recurrence Trees, and the Master Theorem. In graph theory, students will explore basic terminologies, graph representations, connectivity, Euler and Hamiltonian paths, planar graphs, graph coloring, and spanning trees. Additionally, the course covers propositional logic, logical connectives, well-formed formulas, tautologies, equivalences, and inference theory, providing a strong foundation for logical reasoning and computational problem-solving.

Outcome:

1. Understand and apply the concepts of sets, functions, relations, and binary relations along with their properties.
2. Utilize combinatorial principles such as the Pigeonhole Principle, Permutations, Combinations, Mathematical Induction, and the Principle of Inclusion and Exclusion.
3. Analyze the growth of functions using asymptotic notations, summation techniques, and integral approximations.
4. Solve recurrence relations using generating functions, Substitution Method, Recurrence Trees, and the Master Theorem.
5. Understand fundamental graph theory concepts, including graph models, representations, and isomorphism.
6. Analyze graph connectivity, Euler and Hamiltonian paths and circuits, planar graphs,

and graph coloring.
7. Learn tree structures, their properties, and applications, including spanning trees.
8. Apply propositional logic, logical connectives, and well-formed formulas to solve logical reasoning problems.
9. Use tautologies, logical equivalences, and inference theory to derive valid conclusions.
10. Develop problem-solving and analytical skills essential for theoretical and applied computer science.

Course description:

1. Introduction:
   Sets - finite and Infinite sets, uncountably Infinite Sets; functions, relations, Properties of Binary Relations, Closure, Partial Ordering Relations; counting - Pigeonhole Principle, Permutation and Combination; Mathematical Induction, Principle of Inclusion and Exclusion.

2. Growth of Functions:
   Asymptotic Notations, Summation formulas and properties, Bounding Summations, approximation by Integrals

3. Recurrences:
   Recurrence Relations, generating functions, Linear Recurrence Relations with constant coefficients and their solution, Substitution Method, Recurrence Trees, Master Theorem

4. Graph Theory
   Basic Terminology, Models and Types, multigraphs and weighted graphs, Graph Representation, Graph Isomorphism, Connectivity, Euler and Hamiltonian Paths and Circuits, Planar Graphs, Graph Coloring, Trees, Basic Terminology and properties of Trees, Introduction to Spanning Trees

5. Propositional Logic
   Logical Connectives, Well-formed Formulas, Tautologies, Equivalences, Inference Theory

Recommended Books:

1. C.L. Liu , D.P. Mahopatra, Elements of Discrete mathematics, 2nd Edition , Tata McGraw Hill, 1985,
2. Kenneth Rosen, Discrete Mathematics and Its Applications, Sixth Edition ,McGraw Hill 2006
3. T.H. Coremen, C.E. Leiserson, R. L. Rivest, Introduction to algorithms, 3rd edition Prentice Hall on India, 2009
4. M. O. Albertson and J. P. Hutchinson, Discrete Mathematics with Algorithms , John wiley Publication, 1988
5. J. L. Hein, Discrete Structures, Logic, and Computability, 3rd Edition, Jones and Bartlett Publishers, 2009
6. D.J. Hunter, Essentials of Discrete Mathematics, Jones and Bartlett Publishers, 2008