VIDYASAGAR UNIVERSITY

Midnapore, West Bengal

PROPOSED CURRICULUM&SYLLABUS (DRAFT) OF

BACHELOR OF COMPUTER APPLICATION- BCA (HONOURS)

4-YEAR UNDERGRADUATE PROGRAMME

(w.e.f. Academic Year 2023-2024)

Based on

Curriculum & Credit Framework for Undergraduate Programmes (CCFUP), 2023& NEP, 2020

VIDYASAGAR UNIVERSITY
BACHELOR OF COMPUTER APPLICATION- BCA (HONOURS)
(under CCFUP, 2023)

Level	YR.	SEM	Course Type	Course Code	Course Title	Credit	L-T-P	Marks		
								CA	ESE	TOTAL
BSW. (Hons.)	2nd	III	SEMESTER-III							
			Major-3	BCAHMJ03	T:Computer Architecture, P: Practical	4	3-0-1	15	60	75
			Major-4	BCAHMJ04	T:Discrete Mathematics	4	3-1-0	15	60	75
			SEC	BCASEC03	P: PYTHON (Practical)	3	0-0-3	10	40	50
			AEC	AEC03	Communicative English -2 (***common for all programmes***)	2	2-0-0	10	40	50
			MDC	MDC03	Multidisciplinary Course -3 (***to be chosen from the list***)	3	3-0-0	10	40	50
			Minor-3	BCAMIN03	T:Data Structure and Algorithm, P:Practical	4	3-0-1	15	60	75
					Semester-III Total	20				375
		IV	SEMESTER-IV							
			Major-5	BCAHMJ05	T:Database Management Systems; P: Practical	4	3-0-1	15	60	75
			Major-6	BCAHMJ06	T:Operating Systems; P: Practical	4	3-0-1	15	60	75
			Major-7	BCAHMJ07	T: Computer Network P: Practical	4	3-0-1	15	60	75
			AEC	AEC04	MIL-2 (***common for all programmes***)	2	2-0-0	10	40	50
			Minor-4	BCAMIN04	T:Artificial Intelligence; P: Practical	4	3-0-1	15	60	75
			Summer Intern.	INT	**Internship/ Apprenticeship**	4	0-0-4	-	-	50
					Semester-IV Total	22				400
					TOTAL of YEAR-2	42				775

MJ = Major, MI = Minor Course, SEC = Skill Enhancement Course, AEC = Ability Enhancement Course, MDC = Multidisciplinary Course, CA= Continuous Assessment, ESE= End Semester Examination, T = Theory, P= Practical, L-T-P = Lecture-Tutorial-Practical, MIL = Modern Indian Language

MAJOR (MJ)

MJ-3: Computer Architecture **Credits 04 (Full Marks: 75)**

OBJECTIVE OF THE COURSE

This course offers a deep understanding of computer system design, covering processor architecture, memory hierarchies, input/output systems, and instruction set architectures (ISA). It explores data paths, control units, pipelining, parallel processing, and multi-core architectures, highlighting advancements in computing. Students gain practical experience through labs and programming in assembly language and hardware simulation tools. The course emphasizes the impact of architectural choices on software performance, preparing students to analyze, optimize, and pursue advanced studies in computer system design and engineering.

MJ-3T: Computer Architecture **Credits 03**

Module I: Fundamentals of Computer Architecture **(08 Lectures.)**

Boolean algebra, Logic gates, combinational circuits; circuit simplification, decoders, multiplexers, sequential circuits; flip-flops and, registers, counters and memory units.

Module II: Data Representation and Arithmetic **(07 Lectures)**

Number systems, complements, fixed and floating point representation, character representation, addition, subtraction, magnitude comparison, multiplication and division algorithms for integers

Module III: Basic Computer Organization and Design **(05 Lectures)**

Computer registers, bus system, instruction set, timing and control, instruction cycle, memory reference, Organization of a basic single-bus computer system.

Module IV: Central Processing Unit **(10 Lectures)**

Register organization, arithmetic and logical operations, Instruction formats, addressing modes, instruction codes, machine language, assembly language, RISC, CISC architectures, Hardwired and microprogrammed control unit design.

Module V: Memory Organization **(10Lectures)**

Memory interfacing and addressing, cache memory organization.

Module VI: Input-Output Organization **(05Lectures)**

Input / Output: External Devices, I/O Modules, Programmed I/O, Interrupt-Driven I/O, Direct Memory Access, I/O Channels.

Suggested Readings:

1. 2. W. Stallings, Computer Organization and Architecture Designing for Performance, 8 Edition, Prentice Hall of India,2009
2. M. Mano, Computer System Architecture, Pearson Education 1992
3. Carl Hamacher, Computer Organization, Fifth edition, McGrawHill, 2012.
4. M.M. Mano , Digital Design, Pearson Education Asia,2013

MJ-3P: Computer Architecture Lab **Credits 01**

All laboratory assignments focus on simulation or implementation with hardware kits.

1. Arithmetic Operations: Design and simulate 8-bit addition, multiplication, and division circuits.
2. 8-bit ALU Design: Simulate an Arithmetic Logic Unit capable of performing basic operations.
3. 8-bit CPU Design: Implement a simple CPU with basic instruction set architecture.
4. 8-bit Register Design: Create a register and perform read/write operations.
5. Memory Unit Design: Design a memory unit and demonstrate basic memory operations.
6. CPU and Memory Interfacing: Simulate and test the interaction between a CPU and a memory unit.

9. Create the micro operations and associate with instructions as given in the chapter (except interrupts). Design the register set, memory and the instruction set. Use this machine for the assignments of this section.

10. Design a fetch routine for the instruction cycle.

11. Simulate the machine for the memory-reference instructions referred in above question with I= 1 and address part = 082. The instruction to be stored at address 026 in RAM. Initialize the memory word at address 082 with the value 298. Initialize the memory word at address 298 with operand B8F2 and AC with A937. Determine the contents of AC, DR, PC, AR and IR in hexadecimal after the execution.

12. Simulate the machine to determine the hexadecimal contents of the AC, E, PC, AR, and IR registers after executing each of the following register reference instructions:

a. CMA	e. CIR	i. SNA
b. CLE	f. CIL	j. SZA
c. CLA	g. INC	k. SZE
d. CME	h. SPA	l. HLT

Initialize the contents of AC to $(A937)_{16}$, that of PC to $(022)_{16}$ and E to 1.

13. Modify the machine created in Practical 1 according to the following instruction format:

Instruction format

0 2 3 4 15

Opcode I Address

a. The instruction format contains a 3-bit opcode, a 1-bit addressing mode and a 12-bit address. There are only two addressing modes, I = 0 (direct addressing) and I = 1 (indirect addressing).

b. Create a new register I of 1 bit.

c. Create two new microinstructions as follows:

i. Check the opcode of instruction to determine type of instruction (Memory

Reference/Register Reference/Input-Output) and then jump accordingly.

ii. Check the I bit to determine the addressing mode and then jump accordingly.

14. Simulate the machine for the following memory-reference instructions with I= 0 and address part = 082. The instruction to be stored at address 022 in RAM. Initialize the memory word at address 082 with the operand B8F2 and AC with A937. Determine the contents of AC, DR, PC, AR and IR in hexadecimal after the execution.

a. STA f. BSA

b. ISZ g. AND

c. LDA d. ADD

e. BUN

MJ-4: Discrete Mathematics **Credits 04(Full Marks: 75)**

OBJECTIVE OF THE COURSE

The objective of the Discrete Mathematics course is to introduce students to essential mathematical concepts and techniques used in computer science, engineering, and mathematics, focusing on logical reasoning, problem-solving, and the study of discrete structures like sets, functions, relations, combinatorics, and graph theory. The course aims to enhance students' ability to apply mathematical principles to real-world problems, develop skills in mathematical proof techniques, and establish a strong foundation for advanced topics in computer science, such as algorithms, cryptography, and network theory. By the end of the course, students will be equipped to approach complex problems with rigorous mathematical tools and techniques.

MJ-4T: Discrete Mathematics **Credits04**

Module 1: Introduction **(12 Lectures)**

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.

Module 2: Growth of Functions **(8 Lectures)**

Asymptotic Notations, Summation formulas and properties, Bounding Summations, approximation by Integrals

Module 3: Recurrences **(8 Lectures)**

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

Module 4: Graph Theory **(10 Lectures)**

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

Module 5: Prepositional Logic **(12 Lectures)**

Logical Connectives, Well-formed Formulas, Tautologies, Equivalences, Inference Theory

Module 6: Matrix Algebra **(10 Lectures)**

Types of matrices, algebra of matrices–addition, subtraction, and multiplication of matrices, determinant of a matrix, symmetric and skew-symmetric matrices, orthogonal matrix, rank of a matrix,

inverse of a matrix, applications of matrices to solve system of linear equations, Eigen values and Eigen vectors, Caley-Hamilton theorem.

Suggested Readings:

1. Discrete Mathematics and Graph Theory, Satyanarayana& Prasad
2. Discrete Mathematics with Graph Theory, 3rd ed., Goodaire&Parmenter
3. Discrete Mathematics and Graph Theory, 2nd ed., Biswal
4. Discrete Mathematics, Rajendra Akerkar and Rupali Akerkar
5. Discrete Mathematics, Babu Ram

MJ-5: Database Management Systems **Credits 04(Full Marks: 75)**

OBJECTIVE OF THE COURSE

- Understanding core concepts of DBMS.
- Gaining proficiency in database design and SQL.
- Applying advanced database techniques.

OUTCOME OF THE COURSE:

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

- Understand database concepts and design principles.
- Apply SQL for data retrieval and manipulation.
- Implement transactions and concurrency control.
- Explore NoSQL databases and big data technologies.
- Develop secure and efficient database solutions.

MJ-5T: Database Management Systems **Credits 03**

Course Contents

Introduction **(12 hours)**

Definition of Data, Database, and DBMS, Overview of Database Applications, Advantages, and Disadvantages of DBMS, Roles of Database Users and Administrators Data Models: Introduction to Data Models, Types of Data Models: Hierarchical, Network, Relational, Object-Oriented, Importance of Data Models in DBMS, Database Design: Keys: Primary Key, Candidate Key, Super Key, Foreign Key, Composite Key, Alternate Key, Unique Key, Surrogate Key, Constraints: Primary Key, Foreign Key, Unique Key, NOT NULL, CHECK Entity-Relationship (ER) Model: Entities and Entity Sets, Attributes and Relationships, ER Diagrams, Key Constraints, Weak Entity Sets, Extended ER Features: Introduction to Relational Model and Relational Schema.

Relational Algebra, SQL, and Normalization **(16 hours)**

Relational Algebra and Calculus: Introduction to Relational Algebra, Operations: Selection, Projection, Set Operations, Join Operations, Division, Tuple and Domain Relational Calculus
Structured Query Language (SQL): SQL Basics: DDL and DML, Aggregate Functions (Min(), Max(), Sum(), Avg(), Count()), Logical Operators (AND, OR, NOT), Predicates (LIKE, BETWEEN, DISTINCT), Clauses: GROUP BY, HAVING, ORDER BY, Joins: Inner Join, Natural Join, Outer Joins (Left, Right, Full), Equi Join, Advanced SQL: Analytical Queries, Hierarchical Queries, Recursive Queries, Views, Cursors, Stored Procedures and Functions, Triggers, Dynamic SQL, Normalization and Database Design: Functional Dependencies: Armstrong's Axioms, Reflexivity, Augmentation, Transitivity, Types of Functional Dependencies: Trivial, Non-Trivial, Partial, Full, Normal Forms: 1NF, 2NF, 3NF, BCNF, Denormalization

Transaction Management and Indexing **(8 hours)**

Transaction Management: ACID Properties, Transactions and Schedules, Concurrent Execution of Transactions, Lock-Based Concurrency Control, Performance of Locking, Transaction Support in SQL, Crash Recovery, Two-Phase Locking (2PL), Serializability, Recoverability, Lock Management

and Deadlock Resolution, Database Storage and Indexing: Data on External Storage, File Organizations and Indexing, Index Data Structures and Comparison of File Organizations, Indexes and Performance Tuning, Guidelines for Index Selection.

Advanced Databases and Security (9 hours)

NoSQL Databases and Big Data: Introduction to NoSQL Databases, Data Models: Document, Key-Value, Column Family, Graph, Features of NoSQL Databases, CAP Theorem, BASE vs. ACID, CRUD Operations in MongoDB, MongoDB Operators, Big Data Technologies: Overview of Hadoop, MongoDB, Cassandra, Database Security and Advanced Topics: Introduction to Database Security, Access Control, Discretionary Access Control, Introduction to Data Warehousing, OLAP, and Data Mining.

Suggested Readings:

Text Books

1. Raghu Ramakrishnan, Johannes Gehrke, "Database Management Systems," Third Edition, McGraw–Hill, 2018.
2. Benjamin Rosenzweig, Elena Rakhimov, "Oracle PL/SQL by Example," Fifth Edition, Prentice Hall, 2015.
3. Brad Dayley, "NoSQL with MongoDB in 24 Hours," First Edition, Sams Publishing, 2024.

Reference Books

1. Korth, Silbertz, Sudarshan, "Database System Concepts," Seventh Edition, McGraw-Hill, 2019.
2. R.P. Mahapatra, GovindVerma, "Database Management Systems," Khanna Publishing House, 2025.

Web Resources

1. Oracle Base Articles : https://oracle-base.com/articles/articles
2. Oracle Forums: https://forums.oracle.com/ords/r/apexds/community/home
3. Ask Tom Oracle: https://asktom.oracle.com/ords/f?p=100:1:0

MJ-5P: Database Management Systems- Lab Credits 01

List of practical exercises:

1. **Design an Entity-Relationship (ER) Diagram**: Create an ER diagram for a system of your choice—be it a hospital, university, or management system. Clearly define entities, attributes, and relationships to model the system effectively.
2. **Convert ER Diagram to Relational Tables**: Transform the ER diagram from Task 1 into a set of relational tables. Ensure proper normalization and define primary keys, foreign keys, and relationships between tables.
3. **Introduction to MySQL**: Get started with MySQL by learning how to create databases and tables. Understand the basic syntax and structure of SQL commands for database and table creation.

4. **Data Manipulation in MySQL**: Practice inserting, updating, modifying, and deleting data in MySQL tables. Learn how to retrieve data using basic SQL queries and understand the structure of SQL statements.
5. **Enforcing Integrity Constraints**: Implement various integrity constraints in your database, such as domain constraints, key constraints (primary and foreign keys), NOT NULL, UNIQUE, DEFAULT, and CHECK constraints.
6. **Working with Views**: Create and update views in MySQL. Use views to simplify complex queries and retrieve data efficiently.
7. **Aggregate Functions**: Explore the use of aggregate functions like AVG, COUNT, MIN, MAX, and SUM to perform calculations on datasets and extract meaningful insights.
8. **Join Operations**: Master the use of JOIN operators, including natural joins and outer joins (left, right, and full), to combine data from multiple tables.
9. **Query Optimization with Nested Queries**: Optimize queries using nested queries. Utilize logical connectives, set comparison operators (UNION, INTERSECT, EXCEPT), and EXISTS clauses to refine your queries.
10. **Grouping and Filtering Data**: Use the GROUP BY and HAVING clauses to group and filter data based on specific conditions. Learn how to create triggers to automate database operations.
11. **Index Creation**: Improve database performance by creating indexes using SQL. Understand how indexes optimize data retrieval.
12. **PL/SQL Programming**: Write basic PL/SQL programs to perform calculations and manipulate data. Get familiar with PL/SQL syntax and structure.
13. **Cursors for Data Retrieval**: Use cursors in PL/SQL to retrieve and process data row by row. Learn how to aggregate data using cursors.
14. **Triggers, Procedures, and Functions**: Implement triggers, stored procedures, and functions to automate and streamline database operations. Understand their roles in maintaining database integrity and efficiency.
15. **MongoDB Queries**: Explore MongoDB by writing queries to interact with NoSQL databases. Learn how to perform CRUD operations and retrieve data from MongoDB collections.

Suggested Readings:

1. The complete reference-By Coach and loney
2. A Beginners guide- By Abbey and corney
3. Database System-Elmasri and Navathe
4. Database system concepts- Silberschatz Abraham Korth Henry F. Jt. auth. Sudarshan S. Jt. Auth.
5. Database management system oracle SQL and PL/SQL- Das Gupta Pranab Kumar

MJ-6: Operating Systems **Credits 04(Full Marks: 75)**

OBJECTIVE OF THE COURSE

- To provide knowledge about the services rendered by operating systems
- To provide a detailed discussion of the various memory management techniques
- To discuss the various file-system design and implementation issues
- To discuss how the protection domains help to achieve security in a system

OUTCOME OF THE COURSE:

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

- Ability to comprehend the techniques used to implement the process manager
- Ability to comprehend virtual memory abstractions in operating systems
- Ability to design and develop file system interfaces, etc.
- Technical knowhow of the working principle of various types of operating systems

MJ-6T: Operating Systems **Credits 03**

Course Contents

Unit – I **(08 Lectures)**

Operating Systems –Definition – Types- Functions -Abstract view of OS- System Structures– operating systems generations – System Calls- Virtual Machines –Process Concepts –Threads – Multithreading- protection and security-distributed systems.

Unit – II **(12Lectures)**

Process concepts –process state, – process control block –Process Scheduling- Process Co-ordination –Synchronization –Semaphores –Monitors Hardware Synchronization –Deadlocks –Methods for Handling Deadlocks

Unit – III **(12 Lectures)**

Memory Management Strategies –Contiguous and Non-Contiguous allocation –Virtual memory Management – Swapping –contiguous memory allocation –paging –structure of the pagetable – Demand Paging- Page Placement and Replacement Policies - thrashing –case study - UNIX.

Unit – IV **(08 Lectures)**

File System –Basic concepts - File System design and Implementation - file system mounting - file sharing - protection - Mass Storage Structure –free-space management - Disk Scheduling –Disk Management – System Protection and Security- Case Study: Linux File Systems.

Unit – V **(05 Lectures)**

I/O Management - Principles of I/O Hardware - Disk structure - Disk scheduling algorithms.

Suggested Books:

1. Silberschatz, Galvin, Gagne, "Operating System Concepts", John Wiley and Sons.
2. William Stallings, “Operating Systems –Internals and Design Principles”, 8/E, Pearson Publications.
3. Andrew S. Tanenbaum, “Modern Operating Systems”, 4/E, Pearson Publications.

MJ-6P: Operating Systems Lab **Credits 01**

List of practical exercises:

1. **Greatest of Three Numbers**: Write a script that takes three numbers as command-line arguments and determines the greatest among them.
2. **Even or Odd Checker**: Create a script to check whether a given number is even or odd.
3. **Average of N Numbers**: Develop a script to calculate the average of a list of *n* numbers provided by the user.
4. **Prime Number Checker**: Write a script to determine whether a given number is prime or not.
5. **Number or String Checker**: Create a script to check if a given input is a number or a string.
6. **File Line Analyzer**: Write a script to compute the number of characters and words in each line of a given file.
7. **Fibonacci Series Generator**: Develop a script to print the Fibonacci series up to *n* terms.
8. **Factorial Calculator**: Write a script to calculate the factorial of a given number.
9. **Sum of Digits**: Create a script to calculate the sum of the digits of a given number.
10. **Palindrome Checker**: Write a script to check whether a given string is a palindrome.
11. **CPU Scheduling Algorithms**: Simulate the following CPU scheduling algorithms:
    a) First-Come-First-Serve (FCFS)
    b) Shortest Job First (SJF)
    c) Round Robin
    d) Priority Scheduling
12. **Banker’s Algorithm**: Simulate the Banker’s Algorithm for:
    a) Deadlock Avoidance
    b) Deadlock Prevention
13. **Dining Philosophers Problem**: Write a C program to simulate the classic Dining Philosophers problem, illustrating synchronization and potential deadlock scenarios.
14. **Producer-Consumer Problem**: Implement a C program to simulate the Producer-Consumer problem using semaphores for synchronization.
15. **Process Control and Communication**:
    a) Write a C program to implement the kill(), raise(), and sleep() functions.
    b) Write a C program to implement the alarm(), pause(), and abort() functions.
    c) Write a program to illustrate communication between two processes using unnamed pipes.

d) Write a program to illustrate communication between two processes using named pipes (FIFO).

e) Write a C program that receives a message from a message queue and displays it.

16. **Shared Memory Communication**: Write a C program to demonstrate communication between two processes using shared memory.
17. **Page Replacement Algorithms**: Simulate the following page replacement algorithms:
    a) First-In-First-Out (FIFO)
    b) Least Recently Used (LRU)
    c) Optimal
18. **Disk Scheduling Algorithms**: Write a C program to simulate the following disk scheduling algorithms:
    a) First-Come-First-Serve (FCFS)
    b) SCAN
    c) C-SCAN

Suggested Readings:

1. Operating Systems – Internals and Design Principles, William Stallings, Fifth Edition–2005, Pearson Education/PHI
1. Operating System - A Design Approach-Crowley, TMH.
2. Modern Operating Systems, Andrew S Tanenbaum, 2nd edition, Pearson/PHI
3. UNIX Programming Environment, Kernighan and Pike, PHI/Pearson Education
4. UNIX Internals: The New Frontiers, U. Vahalia, Pearson Education

MJ-7: Computer Networking **Credits 04(Full Marks: 75)**

The objective of the **Computer Networks** course is to provide students with a solid understanding of the principles, protocols, and technologies used in modern computer networks. The course aims to teach students how data is transmitted across different network types, including local area networks (LANs), wide area networks (WANs), and the internet. Topics such as network architecture, communication protocols, routing, switching, network security, and wireless networks will be covered. Students will learn to design, configure, and troubleshoot network systems while understanding key concepts like IP addressing, TCP/IP, and network-layer protocols. By the end of the course, students will be equipped with the knowledge and skills necessary to analyze, implement, and manage network systems effectively in both academic and real-world settings.

MJ-7T: Computer Networking **Credits 03**

Unit I : Physical Layer: Introduction to Computer Communication and Network : Network Topologies, Typesof Network, OSI Model, Protocol Stack, Network Protocols. Analog Signals & Digital Signals. DataTransmission: Sampling, Transmission Mode. Analog Transmission: Modulation (Analog & DigitalSignals). Multiplexing: FDM, WDM & TDM. Transmission Media: Guided Media, Unguided Media(Wireless). Circuit Switching. **(15 Lectures)**

Unit II : Data Link Layer: Error detection and correction: - Type of Errors, Detection, Error Correction,Framing. Data Link Control and Protocols: - Flow and Error control, CRC, REC, FEC, Hamming Code,Stop-and Wait ARQ, Go-Back, N ARQ, Selective Repeat ARQ, HDLC. ALOHA, CSMA, CSMA/CD.Multiple Access: Random Access, Controlled Access, Area Network: Ethernet, Wireless LANS: IEEE802-11, Frame Relay, ATM **(15 Lectures)**

Unit-III: Network Layer: Host to Host Delivery: IP Addressing and Routing, Gateway, N/W Layer Protocols:ARP, IPV4, ICMP, IPV6, Transport Layer: Process-to-Process Delivery: UDP, TCP Congestion Control& Quality of Service. **(10 Lectures)**

Unit-IV: Application Layer: Client Server Model, Domain Name System (DNS), E-mail (SMTP), File Transfer(FTP) HTTP, WWW. **(05 Lectures)**

Reference Books:

1. Data Communication & Networking – Behuouz A. Forouzan, TMH
2. Computer Network – A.S Tanenbaum, Pearson Education
3. Computer Newtworks- kundu – PHI
4. Computer Network – Rajesh – Vikash

MJ-7P: Computer Networking Lab **Credits 01**

List of practical exercises:

1. **Swap Using Pointers**: Write a C program to swap the contents of two variables using pointers.
2. **Extract Bytes from a Number**: Write a C program to extract each byte from a given number, store them in separate character variables, and print the content of those variables.
3. **Endianness Check and Conversion**: Write a C program to check whether the host machine uses Little Endian or Big Endian. Enter a number, print the content of each byte location, and convert the Endianness (Little to Big or vice versa).
4. **Client to Server Message**: Write a C program in UDP to send a message from the client to the server.
5. **Server to Client Message**: Write a C program in UDP to send a message from the server to the client.
6. **String Case Conversion**: Write a program to pass a string from the client to the server. Display the lowercase message on the server side. Convert the message to uppercase and send it back to the client for display.
7. **String Reversal**: Write a program to pass a string from the client to the server. Display the string on the server side, reverse it, and send the reversed string back to the client for display.
8. **Integer Array Transfer**: Write a program to pass an integer array from the client to the server and print the array on the server side.
9. **Greatest Element in Array**: Write a program to pass an integer array from the client to the server. Determine the greatest element on the server side and send it back to the client for display.
10. **Sorted Array**: For practice, pass an integer array from the client to the server, sort it on the server side, and return the sorted array to the client for display.
11. **Basic Message Exchange**: Write a C program in TCP to send the message "Hello KIIT" from the client to the server. The server will display the message and respond with "Thank you," which the client will display.
12. **Count Vowels in a String**: Write a C program in TCP to send the string "Welcome to KIIT" to the server. The server will count and display the number of vowels in the string.
13. **Palindrome Check**: Write a C program in TCP to send a string from the client to the server. The server will display the string and check if it is a palindrome.
14. **Sum of Array Elements**: Write a C program in TCP to send an integer array from the client to the server. The server will display the array, calculate the sum of its elements, and send the sum back to the client for display.
15. **Second Largest Element**: Write a C program in TCP to send an integer array from the client to the server. The server will display the array and determine the second-largest element.
16. **Array Transformation**: Write a program to send an integer array from the client to the server. On the server side, add two consecutive elements and store the result in the next location without using a third variable or extra array. Display the resultant array.
    Example:

Input: 10 20 30 40 50 60
Output: 10 30 50 70 90 110

17. **Count Consonants**: Write a program to send a string from the client to the server using UDP. The server will count the number of consonants in the string and display the result.
18. **I/O Multiplexing with SELECT()**: Write a C program in TCP to demonstrate the use of the select() system call for I/O multiplexing.

MINOR (MI)

Minor (MI) – 3: Data Structure and Algorithm **Credits 04(Full Marks: 75)**

OBJECTIVE OF THE COURSE

The objective of the **Data Structures and Algorithms** course is to provide students with a comprehensive understanding of the fundamental data structures and algorithms that form the backbone of computer programming and software development. The course aims to equip students with the skills to design, analyze, and implement efficient algorithms, using appropriate data structures to solve complex computational problems. Students will learn how to evaluate algorithm performance through time and space complexity analysis and gain hands-on experience with techniques such as searching, sorting, recursion, dynamic programming, and graph algorithms. By the end of the course, students will be capable of selecting and applying the right data structures and algorithms to optimize performance and solve real-world challenges in computer science and software engineering.

MI – 3T: Data Structure and Algorithm **Credits 03**

Course contents:

1. **Analysis of Algorithms:** **(05 Lectures)**
   Characteristics of an algorithm, Steps in Designing of Algorithms, Growth of function, Recurrence, Substitution Method, Iteration Method, Asymptotic Notations, Concept of efficiency of analysis of an algorithm, Comparative efficiencies of algorithms: Linear, Quadratic, Polynomial and Exponential, Calculation of Storage Complexity, Calculation of Run Time Complexity.
2. **Arrays:** **(04 Lectures)**
   Single and Multi-dimensional Arrays, Sparse Matrices, Row major and column major representation.
3. **Linked Lists:** **(03 Lectures)**
   Abstract Data Type-List,Singly, Doubly and Circular Lists (Array and Linked representation); Normal and Circular representation of Stack in Lists;
4. **Stacks:** **(03 Lectures)**
   Abstract Data Type-Stack, Implementation of stack using array and linked list, Prefix, Infix and Postfix expressions, Algorithmic Implementation of Multiple Stacks, Applications of stack.
5. **Queues:** **(03 Lectures)**
   Abstract Data Type-Queue, Array and Linked representation of Queue and circular Queue, De-queue, Priority Queues.
6. **Trees:** **(08 Lectures)**
   Abstract Data Type-Tree, Implementation of Tree, Tree Traversals, Binary Trees , Implementation of Binary Tree , Binary Tree Traversals(Recursive Implementation and Non Recursive Implementations),Binary Search Trees, AVL tree (Various operations on AVL Trees), Heap Tree.
7. **Searching and Sorting:** **(05 Lectures)**
   Linear Search, Selection Sort, Insertion Sort, Bubble Sort, Divide and Conquer approach(Binary Search, Quick Sort and Merge sort), Comparison of Sorting Algorithm Techniques.

8. **Hashing:** **(05 Lectures)**
   Hashing technique, Different types of hash function ,Collision resolution, chaining.
9. **Graph:** **(09 Lectures)**
   Definitions, Shortest Path Algorithms ,Dijkstra's Algorithm , Minimum cost Spanning Trees , Kruskal's Algorithm , Prims's Algorithm , Breadth First Search , Depth First Search.

Suggested Readings:

1. Data Structure using C – Rajni Jindal – Umesh Publication
2. Data Structure using C – B. BalujaDhanpatrai Publication
4. Classic Data Structures, 2nd ed., Samanta
5. Data Structures Using C and C++, 2nd ed., Langsam, Augenstein&Tenenbaum

MI – 3P: Data Structure and Algorithm Lab **Credits 01**

List of practical exercises:

1. **Linear Search**: Write a program to implement the Linear Search algorithm.
2. **Binary Search:**: Implement the Binary Search algorithm.. Use an iterative (non-recursive) function.
3. **Bubble Sort (Ascending Order)**: Write a program to arrange a given list of numbers in ascending order using the Bubble Sort algorithm.
4. **Bubble Sort (Descending Order)**: Write a program to arrange all the alphabets in the string "CSIPLEARNING" in descending order using Bubble Sort.
5. **Selection Sort**: Write a program to sort a given dataset using the Selection Sort algorithm.
6. **Selection Sort (Specific Data)**: Sort the following list using Selection Sort:
   [14, 21, 27, 41, 43, 45, 46, 57, 70]
7. **Merge Sort**: Implement the Merge Sort algorithm.
8. **Quick Sort**:
   a. Write a program to implement the Quick Sort algorithm.
   b. Use the following list to demonstrate Quick Sort:
   [50, 23, 9, 18, 61, 32]
9. **Stack Operations**:
   a. Implement the following stack operations:
      1. Insertion (Push)
      2. Deletion (Pop)
      3. Display
      b. Write a program to reverse a string using a stack.
10. **Queue Operations**: Implement the following queue operations.:
      1. Insertion (Enqueue)
      2. Deletion (Dequeue)
      3. Display
15. **Singly Linked List**: Write a program to implement the following singly linked list operations:
    a. Create a singly linked list.

b. Add elements to the singly linked list.
c. Access elements from the singly linked list.
d. Remove elements from the singly linked list.

16. **Doubly Linked List**: Write a program to implement the following doubly linked list operations:
    a. Create a doubly linked list.
    b. Add elements to the doubly linked list.
    c. Access elements from the doubly linked list.
    d. Remove elements from the doubly linked list.
17. **Stack Using List**: Write a program to implement a stack using a list.
18. **Queue Using List**: Write a program to implement a queue using a list.
19. **Binary Search Tree (BST) Operations**:
    a. Write a program to implement any one operation on a Binary Search Tree (e.g., insertion, deletion, or searching).
20. **Binary Tree Traversal**: Write a program to implement the following binary tree traversal methods:
    - Preorder
    - Inorder
    - Postorder

Minor (MI) – 4: Artificial Intelligence **Credits 04 (Full Marks: 75)**

OBJECTIVE OF THE COURSE

The objective of the **Artificial Intelligence** course is to introduce students to the fundamental concepts, techniques, and applications of AI in solving complex problems. The course aims to provide an understanding of key AI topics such as machine learning, search algorithms, natural language processing, knowledge representation, and robotics. Students will learn how to design and implement intelligent systems that can perform tasks typically requiring human intelligence, such as reasoning, decision-making, and pattern recognition. By the end of the course, students will be able to apply AI algorithms to real-world problems, evaluate their performance, and explore emerging trends in AI technologies, preparing them for advanced studies and careers in the field.

Minor (MI) T– 4: Artificial Intelligence **Credits 03 (45 Hrs.)**

Unit 1: (05 Lectures)
Overview: Foundations, Scope, Problems, and Approaches of AI
Intelligent Agents: Structure of Intelligent Agents, Environments

Unit 2: (10 Lectures)
Solving Problems by Searching: Graph Search, Uninformed Search, Informed/Heuristic Search, Constraint Satisfaction Search, Stochastic Search (like Hill climbing, Simulated Annealing), Population Based/Evolutionary Search (like Genetic Algorithm, Swarm Optimization), Adversarial Search – Game Playing, Planning as Search, Sample Applications

Unit 3: (8 Lectures)
Knowledge and Reasoning: Ontology, Foundations of Knowledge Representation and Reasoning, Propositional Logic, First Order Logic, Inference Rules, Unification, Soundness, Completeness, Control Strategies, Resolution-Refutation Systems, Sample Applications

Unit 4: (12 Lectures)
Reasoning under Uncertainty: Non Monotonic Reasoning Systems, Assumption based Truth Maintenance System, Modal and Temporal Logic, Probabilistic Reasoning (Bayes' Rule, Bayesian Network), Fuzzy reasoning

Unit 5: (5 Lectures)
Learning based AI: Learning (Supervised, Unsupervised, Reinforcement), Decision Trees and Random Forests, Neural Learning

Unit 6: (5 Lectures)
Some Applications of AI
Ethical and Legal Considerations in AI
Future of AI

Minor (MI) P– 4: Artificial Intelligence **Credits 01 (30 Hrs.)**

Laboratory Assignments:

1. Write a Prolog program to calculate the GCD of two numbers.
2. Write a Prolog program to calculate the factorial of a number.
3. Write a Prolog program to find the sum of N natural numbers.
4. Write a Prolog program to calculate the product of two numbers.
5. Write a prolog program to find the maximum and minimum number in a list.
6. Write a prolog program to find the number of elements in a list and find their sum.
7. Write a prolog program to find the reverse of a list and check whether it is palindrome or not.
8. Write a program to find the number of zero in a list and delete them.
9. Write a program to find the ascending order of the list using bubble sort and insertion sort..
10. Write a program to find a number is prime or not and the factorial of the number.
11. Write a program to add three elements at the end of a list.
12. Write a Prolog program to sort a list in descending order using selection sort.

Suggested Readings:

1. AI for All : Tools and Techniques – Biswapati Jana, DebkumarBera, Sharmistha Jana – Amitrakshar™ Publishers Kolkata
2. AI for Everyone- S Goswami, AK Das, AChakraborty- Pearson
3.**সকলের জন্য** AI -Biswapati Jana, DebkumarBera, Sharmistha Jana – Amitrakshar™ Publishers Kolkata

SKILL ENHANCEMENT COURSE (SEC)

SEC 3: PYTHON **Credits 03 (Full Marks: 50)**

OBJECTIVE OF THE COURSE -

The objectives of this course are to make the student understand programming language, programming, concepts of Loops, readinga 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 'Python' 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
- Develop efficient algorithms for solving a problem.
- Use the various constructs of a programming language viz. conditional, iteration and recursion.
- Implement the algorithms in "Python" language.
- Usesimpledatastructureslikearrays,stacks andlistinsolvingproblems.

SEC3P: PYTHON **Credits 03**

Course Outline:

Planning the Computer Program: Concept of problem solving, Problem definition, Problem design, Debugging, Types of Errors in programing, Documentation

Techniques of Problem Solving: Flowcharting, decision table, algorithms, Structured programing concepts, Programing methodologies viz. top-down and bottom-up

Overview to Python Programming: Structure of Python Program, Elements of Python

Introduction to Python: Python Interpreter, Python shell, Indentation, Atoms, Identifiers and keywords, literals, Strings, Operator (Arithmetic Operator, Relational Operator, Logical or Boolean Operator, Assignment Operator, Ternary operator, Bitwise Operator)

Creating Python Programs: Input and Output Statements, Control Statements (Branching, Looping, Conditional Statement, Exit, Function, Difference, between break, continue and pass). Defining Functions, Default arguments and Exception handling

Iterations and Recursions: Conditional execution, Alternative execution, Nested conditionals, Return statements, Recursion, Stack diagrams for recursive functions, Multiple assignment, While statement, For statement.

String and List: String asa compound data type, Length, Traversal and the for loop, String slices, String Comparison, A find function, Looping and counting, List values, Accessing elements, List length, List membership, List and for loops, List operations, List deletion, Cloning lists, Nested Lists

Object Oriented Programming: Introduction to Classes, Objects and Methods, Standard Libraries

Suggested Readings:

1. JhonV.Guttag,"IntroductiontoComputationandProgrammingUsingPython",MIT Press
2. AllenDowney,"Think Python: How to Think a Computer Scientist", O'Reilly
3. Mark Lutz, "Learning Python, 5th Edition", O'Reilly

INTERNSHIP/APPRENTICESHIP (INT)

Credit-04 Marks: 50 **(120 hours, 8 weeks)**

Guideline for internship/apprenticeship:

The internship program will commence at the beginning of the third semester and will be evaluated upon its completion at the end of the fourth semester.

1. A student may visit an industry for industry-related issues or a research institution, laboratory, or academic institute to engage in internship under the guidance of an industry official, scientist, or academician.
2. A student may work at a company's outlet or similar type of office, Professional bodies, etc. to develop programming / process etc. under the supervision of the respective official or a faculty of his/her own college teacher or a teacher from another college/university/industry person.
3. Interns may engage in advanced learning in topics beyond their course curriculum, under the guidance of their respective mentor.
4. Interns may be assigned a problem to solve using any programming language.
5. Interns may be assigned to design a webpage for college /department/ Entrepreneur/ start-up under the mentor's guidance.
6. Interns may be allowed to work as quantitative researchers in advance topics in Computer Science and applications from the reputed institute/ organization.

General instructions:

a) Each intern must maintain a daily logbook of activities.
b) At the end of the internship, a completion certificate must be obtained from the mentor, supervisor, or concerned authority.
c) Interns are expected to strictly adhere to the assigned tasks and deadlines.