B.N.M. Institute of Technology

An Autonomous Institution under VTU, Approved by AICTE

Department of Mathematics

Syllabus

Semester: III			
Course: Fourier Transform, Fundamentals of logic and Advanced Linear Algebra **Course Code: 25MAC131 (Common to CSE, ISE, AIML)**			
L:T:P:J	**2:2:0:0**	**CIA**	**: 50**
Credits:	**03**	**SEA**	**: 50**
Hours:	**40**	**SEA Duration**	**: 03 Hours**

Course Learning Objectives: The students will be able to

1 Have an insight into Fourier series, Fourier transforms.

2 Develop knowledge of Fundamentals of logic and Relations, Vector Spaces, Linear Transformation & Inner product spaces arising in engineering

Module-1: Fourier Series & Fourier Transforms	**No. of hours**	**Blooms cognitive Levels**
Examples from Engineering that require Fourier series and Fourier Transforms. **Fourier series**: Periodic functions, Introduction to Fourier Series, Dirichlet's condition. Problems on Fourier series over $(-l, l)$. **Fourier Transforms:** Introduction to infinite Fourier transform, Fourier sine and cosine transform and properties, problems on infinite Fourier transform, Discrete & Fast Fourier transform. *Experiential Learning component: Finding the Fourier series and Fourier Transform of a function*	**L : 04** **T : 04**	**L1** **L2** **L3**
Module-2: Fundamentals of logic and Boolean Algebra		
Examples from Engineering that require Fundamentals of logic and Relations. **Mathematical logic:** Basic connectives and truth tables, logic equivalence - the laws of logic, logical implication- rules of inference **Boolean Algebra:** Boolean functions, Representation of Boolean functions, Logic gates, minimization of circuits. *Experiential Learning component: Construction of combinational and sequential circuit.*	**L : 04** **T : 04**	**L1** **L2** **L3**
Module-3: Vector Spaces		
Examples from Engineering that require vector spaces Recap of system of linear homogenous and non-homogeneous equation and solution sets. Vector spaces, subspaces, linearly independent and dependent, Linear span of a set, Basis and dimension, coordinate vectors. *Experiential Learning component: Problems on linearly independent and dependent vectors, basis and dimension of a vector space.*	**L : 04** **T : 04**	**L1** **L2** **L3**
Module-4:Linear Transformation		
Examples from Engineering that require linear transformation. Linear transformations, algebra of linear transformations, representation of transformations by matrices, Non-singular linear transformation, Inverse of a linear transformation, Range space, Null space and problems on Rank-nullity theorem. *Experiential Learning component: Problems on Inverse of a linear transformation and Rank-nullity theorem*	**L : 04** **T : 04**	**L1** **L2** **L3**
Module-5: Inner Product Spaces		
Examples from Engineering that require Inner product spaces. Inner products Inner product spaces, Orthogonal set, orthogonal projections, orthonormal bases, Gram-Schmidt process, QR-factorization, Recap of Eigen values and Eigen vectors, problems on Singular value decomposition. *Experiential Learning component: Problems on QR-factorization and singular value decomposition*	**L : 04** **T : 04**	**L1** **L2** **L3**

Course Outcomes: After completing the course, the students will be able to	
CO 1:	Apply Fourier series & transform concepts in data visualization and cryptography.
CO 2:	Convert Boolean expressions to logic gates and vice-versa.
CO 3:	Apply the knowledge of vector spaces for solving problems in arising in engineering field
CO 4:	Apply the knowledge of linear transform for solving problems in arising in image processing
CO 5:	Compute orthogonal and orthonormal bases vectors and decomposition of a symmetric matrix using standard technique.

CO - PO Mapping:

COs	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO 1	3	2			2							
CO 2	3	2			2							
CO 3	3	2			2							
CO 4	3	2			2							
CO 5	3	2			2							

Reference Books:

1. E. Kreyszig: "Advanced Engineering Mathematics", John Wiley & Sons,10th Edition(Reprint), 2016.
2. B. S. Grewal: "Higher Engineering Mathematics", Khanna Publishers, 44th Ed., 2017.
3. C. Ray Wylie, Louis C.Barrett : "Advanced Engineering Mathematics", 6" Edition, 2. McGraw-Hill Book Co., New York, 1995.
4. James Stewart : "Calculus —Early Transcendentals", Cengage Learning India Private Ltd., 2017.
5. Srimanta Pal & Subodh C Bhunia: "Engineering Mathematics", Oxford University Press, 3"Reprint, 2016.
6. David C. Lay, Steven R. Lay and J. J. McDonald "Linear Algebra and its applications", 3rd Edition, Pearson Education Ltd., 2017.
7. Kenneth H Rosen, "Discrete Mathematics and its Applications, Special Indian Edition 2021, McGraw Hill publication (India).
8. Ralph P. Grimaldi, " Discrete and Combinatorial Mathematics, 5th Edition, Pearson Education 2004.

Web links and Video Lectures:

1. https://nptel.ac.in/courses/111106111
2. https://youtu.be/OynpZwylau8
3. https://archive.nptel.ac.in/courses/111/106/111106051/
4. https://www.youtube.com/watch?v=zvRdbPMEMUI
5. https://www.youtube.com/watch?v=PiG2BMkK3s4
6. https://www.youtube.com/watch?v=ATqV_I8DCh0

B.N.M. Institute of Technology

An Autonomous Institution under VTU, Approved by AICTE

Department of Artificial Intelligence and Machine Learning

SEMESTER – III

Computer Organization and Architecture (PCC)

Credit: 3

Course Code	25AML132	CIA Marks	50
Teaching Hours/Week (L: T: P: J)	3:0:0:0	SEA Marks	50
Total Number of Lecture Hours	40	Exam Hours	03

Course Learning Objectives:

This course will enable students to

- Explain the basic organization of a computer system and the concept of programs as sequences of machine instructions.
- Describe arithmetic and logical operations with integer operands.
- Describe memory hierarchy and the hardware requirements for cache memory.
- Demonstrate different ways of communicating with I/O devices and standard I/O interfaces.
- Illustrate organization of a simple processor, pipelined processor and other computing systems

Pre-Requisites:

- Computer Basics
- Digital Logic
- Mathematics
- Assembly Language

	Number of Hours	**Bloom's Level**
Module 1: CENTRAL PROCESSING UNIT		
Basic Structure of Computers: Basic Operational Concepts, Bus Structures, Performance – Processor Clock, Basic Performance Equation, Clock Rate, Instruction Set: CISC and RISC, Performance Measurement. **Machine Instructions and Programs:** Memory Location and Addresses, Instructions and Instruction Sequencing, Addressing Modes. Self study : Memory Operations	**8**	**(CO1) Apply**
Module 2 : COMPUTER ARITHMETIC		
Numbers, Arithmetic Operations and Characters, Addition and Subtraction of Signed Numbers, Multiplication of Positive Numbers, Signed Operand Multiplication, Fast Multiplication, Integer Division. Self Study: Design of Fast Adders	**8**	**(CO2) Apply**
Module 3 : MEMORY ORGANIZATION		

Basic Concepts, Semiconductor RAM Memories: Internal organization of memory chips, static memories, Asynchronous and synchronous DRAMs, Cache Memories – Mapping Functions. Self Study: Virtual memories	**8**	**(CO3) Apply**
Module 4 : INPUT - OUTPUT ORGANIZATION		
Accessing I/O Devices, Interrupts – Interrupt Hardware, Direct Memory Access, Buses, Interface Circuits Self Study: Exploring Hardware and Software Interrupts	**8**	**(CO4) Apply**
Module 5 : BASIC PROCESSING UNIT		
Basic Processing Unit: Some Fundamental Concepts, Execution of a Complete Instruction,Multiple Bus Organization, Hard-wired Control **Pipelining:** Basic concepts of pipelining, parallel processing Self Study: Microprogrammed control	**8**	**(CO5) Apply**

Course outcomes:

The students will be able to

1. Identify the basic organization of a computer system and the concept of programs as sequences of machine instructions. (Apply)
2. Solve the various arithmetic operation in computer hardware (Apply)
3. Identify the memory hierarchy and the hardware requirements for cache memory. (Apply)
4. Identify different ways of communicating with I/O devices and standard I/O interfaces. (Apply)
5. Estimate the processor time and CPU usage. (Apply)
6. Design and analyze Memory devices (Analyze- for Assignment)

Reference Books:

1. Carl Hamacher, ZvonkoVranesic, SafwatZaky, Computer Organization, 6 th Edition, TataMcGraw Hill, 2012.
2. William Stallings: Computer Organization & Architecture, 11th Edition, Pearson, 2022.
3. M. Morris Mano, “Computer system Architecture”, 3rd Edition, Prentice-Hall Publishers, 2017.

B.N.M. Institute of Technology

An Autonomous Institution under VTU, Approved by AICTE

Department of Artificial Intelligence and Machine Learning

SEMESTER – III

Foundation of Artificial Intelligence (PCI)

Credit: 4

Course Code	25AML133	CIA Marks	50
Teaching Hours/Week (L: T: P: J)	3:0:2:0	SEA Marks	50
Total Number of Lecture Hours	50	Exam Hours	03

Course Learning Objectives:

This course will enable students to

- Become familiar with basic principles of AI toward problem solving, inference, perception, knowledge representation, and learning.
- Investigate applications of AI techniques in intelligent agents, expert systems, artificial neural networks, and other machine learning models.
- Learn the methods of solving problems using Artificial Intelligence.
- Learn the knowledge representation techniques, reasoning techniques and planning

Prerequisites:

Problem solving skill and Python programming language

	Number of Hours	Bloom's Level
Module-1: Introduction		
Introduction to AI: history, Intelligent systems, foundation and sub-area of AI, applications. Introduction to Artificial General Intelligence (AGI). **Problem solving:** Production System, water jug problem, Missionaries and Cannibals Problem, 8-Puzzle problem, State space search, Control Strategies, Characteristics of Problem. **Practical:** 1. Write a Program to Implement Tic-Tac-Toe game using Python. 2. Write a Program to Implement Water Jug using Python. **Self-study:** current trend and development of AI, Implementation of Missionaries and Cannibals Problem.	**6+4**	**(CO1) Apply**
Module-2: Problem solving		
Uninformed Search Strategies: Hill climbing Algorithm, Breadth-First search, Uniform- Cost Search, Depth-first search, Depth-limited search, Iterative deepening depth-first search, comparing uninformed search strategies. **Informed (Heuristic) Search strategies:** Best-first search, A* algorithm, AO* algorithm **Constraint Satisfaction Problems:** Crypt-arithmetic problem **Practical:** 1. Write a program to analyze the output of various Hill-climbing algorithms. 2. Implement AO* Search algorithm. 3. Write a Program to Implement Breadth First Search using Python. **Self-study:** Bidirectional search, Memory bounded RBFS algorithm and SMA* algorithm	**6+4**	**(CO2) Apply**
Module-3: Game Playing		

Adversarial Search: Nim Game problem, minimax procedure, alpha-beta pruning. **Advanced problem-solving paradigm:** Planning: types of planning system, block world problem, logic-based planning, Linear planning using a goal stack, Sussman anomaly problem in goal stack. **Practical:** 1. Write a program to implement the Minimax Algorithm. 2. Write a Program to Implement Tower of Hanoi **Self-study:** Means End Analysis	**6+4**	**(CO3) Apply**
Module-4 Logical Reasoning		
Logical reasoning: propositional calculus, propositional logic, Natural Deduction system, Axiomatic system, Semantic Tableau system in propositional logic, resolution refutation in propositional logic, predicate logic, forward and backward chaining. **Practical:** 1. Implementation of problem-solving strategies: either using Forward Chaining or Backward Chaining. 2. Write a program using simple predicate logic for any given examples. [Sample examples: If a person is human, then the person is mortal. All students can study] **Self-study: L**ogic programming (LISP), Horn Clause	**6+4**	**(CO4) Apply**
Module-5: Knowledge Representation & Expert Systems		
Knowledge Representation: Approaches to knowledge representation, knowledge representation using semantic network, extended semantic network, Knowledge representation using Frames. **Expert Systems:** Architecture of expert systems, Roles of Expert systems, Typical expert systems - MYCIN. **Application of AI:** AI in Healthcare, AI in Finance **Practical:** 1. Implement MYCIN expert system 2. Write a python program to represent the following knowledge: Birds can fly. Sparrow is a Bird. Display the property inherited by Sparrow. 3. Create a frame for an employee. Update the salary and display the modified frame. 4. Create a semantic network of fruits and their colors. Ask the user for a fruit name and display its color. **Self-study:** DART, XCON, Edge AI and Distributed AI	**6+4**	**(CO5) Apply**

Course outcomes:

The students will be able to

1. Understand the concepts of AI, characteristics of problems, and apply various techniques for problem solving.
2. Apply appropriate search techniques to solve AI problems.
3. Apply algorithms that can learn to play games and make decisions
4. Develop knowledge base sentences using propositional logic and first order logic for logical reasoning.
5. Apply AI techniques for knowledge representation and implement various expert systems on real world problems.

Reference Books:

1. Saroj Kaushik, Artificial Intelligence, 2nd edition, Cengage learning India, 2023
2. Staurt Russel, Peter Norvig, Artificial Intelligence: A Modern Approach, Pearson Education, 4th Edition, 2022.

3. Elaine Rich, Kevin Knight, Artificial Intelligence, Tata McGraw Hill, 4th Revised edition

Online Resources:

1. https://www.ibm.com/think/topics/edge-ai#:~:text=Decreased%20bandwidth,simultaneous%20data%20transmission%20and%20reception.

B.N.M. Institute of Technology

An Autonomous Institution under VTU, Approved by AICTE

Department of Artificial Intelligence and Machine Learning

SEMESTER – III

Data Structures using C (PCI)

Credit: 4

Course Code	25AML134	CIA Marks	50
Teaching Hours/Week (L: T: P: J)	3:0:2:0	SEA Marks	50
Total Number of Lecture Hours	50	Exam Hours	03

Course Learning Objectives:

This course will enable students to:

- Develop the skills needed to efficiently manage and manipulate these data structures in various applications.
- Distinguish the conceptual and applicative differences in binary trees, binary search trees for effective data retrieval and manipulation.
- Apply heap trees, hash tables, and AVL trees to manage data efficiently.
- Apply graph theory concepts and algorithms to effectively solve graphical problems.

Prerequisite: C Concepts and Programming

	Number of Hours	**Bloom's Level**
Module 1: Stacks and Queues		
Concept Learning: Introduction to structure and unions, Data Structures: Classification (Primitive & Non-primitive), Operations, Pattern Matching Algorithms (Brute force, KMP). Stacks: Definition, Operations, Implementation using arrays, Applications of Stacks – Infix to Postfix Conversion and Postfix Expression Evaluation. Queues: Definition, Operations, Implementation, Applications, Circular Queue (Message queue using Circular queue), Doubly Ended Queue, Priority Queue. **Practical:** 1. Design, develop and Implement a Program in C for converting an Infix Expression to Postfix Expression. Program should support both parenthesized and free parenthesized expressions with the operators: +, -, *, /, % (Remainder), ^ (Power) and alpha numeric operands. 2. Design, Develop and Implement a Program in C to Evaluation of Suffix expression with single digit operands and operators: +, -, *, /, %, ^ using Stack. 3. Design, Develop and Implement a menu driven Program in C for the following operations on Circular QUEUE of Characters (Array	**6+4**	**CO1 Apply**

Implementation of Queue with maximum size MAX) • Insert an Element on to Circular QUEUE • Delete an Element from Circular QUEUE • Demonstrate Overflow and Underflow situations on Circular QUEUE • Display the status of Circular QUEUE Support the program with appropriate functions for each of the above operations. Self-Study: Pattern Matching Algorithm: Rabin-Karp, and Boyer-Moore		
Module 2: Linked List		
Linked Lists: Definition, Create, Insert, Delete, Update, Traverse, and Position-based Operations, Concatenate, Merge, and Reverse Lists, Doubly Linked List Implementation and Operations, Circular Linked List Implementation and Operations, Applications of Lists (Polynomial addition). Implementation of stacks and queues using Linked List **Practical:** 1. Design, Develop and Implement a menu driven Program in C for the following operations on Singly Linked List (SLL) of Student Data with the fields: USN, Name, Branch, Sem, PhNo. • Create a SLL of N Students Data by using front insertion. • Display the status of SLL and count the number of nodes in it • Perform Insertion / Deletion at End of SLL • Perform Insertion / Deletion at Front of SLL 2. Design, Develop and Implement a menu driven Program in C for the following operations on Doubly Linked List (DLL). • Create a DLL of N Data by using end insertion. • Display the status of DLL and count the number of nodes in it • Perform Insertion and Deletion at End of DLL • Perform Insertion and Deletion at Front of DLL Self-Study: Josephus Problem using Circular Linked List	**6+4**	**CO2 Apply**
Module 3: Trees		
Concept Learning: Trees: General Tree Representation, Traversals, Applications. Binary Trees: Definition, Properties, Traversals, Applications. Binary Search Tree: Definition, Implementation, Search, Insert, Delete operations. Building and Evaluating Binary Expression Tree. **Practical:** 1. Write a C program to find maximum depth or height of a complete binary tree. 2. Write a C program to print all the path from root to leaf path for given binary tree. 3. Write a C program to insert a new node as a left child in a threaded binary tree. 4. Design, Develop and Implement a menu driven Program in C for the following operations on Binary Search Tree (BST) of Integers.	**6+4**	**CO3 Apply**

• Create a BST of N Integers: 6, 9, 5, 2, 8, 15, 24, 14, 7, 8, 5, 2 • Traverse the BST in Inorder, Preorder and Post Order • Search the BST for a given element (KEY) and report the appropriate message Self-study: Minimax Trees (Game Trees)		
Module 4: Advanced Trees & Hashing		
Concept Learning: Heap Tree: Definition, Implementation, Insert, Delete, Peek operations. Hashing: Hash Table, Hash Functions, Collision Handling by Open Addressing, Chaining, AVL tree. **Practical:** 1. Write a C program to construct MAX-Heap. 2. Write a C program to implement hashing technique and collision resolution technique. Self-study: Red-Black Trees: Concept and Properties	**6+4**	**(CO4) Apply**
Module 5: Graphs		
Concept Learning: Graphs: Disjoint sets, Representation of Graphs - Adjacency/ Cost Matrix, Adjacency Lists, and Traversal of Graphs (BFS and DFS) **Practical:** 1. Design, Develop and Implement a Program in C for the following operations on Graph(G) of Cities, create a Graph of N cities using adjacency Matrix and print all the nodes reachable from a given starting node in a digraph using DFS/BFS. 2. Write a C Program to detect Cycle in a Directed Graph 3. Write a C Program to find if there is a path between two vertices in a directed graph. Self-study: Hamiltonian Path	**6+4**	**(CO5) Apply**
Course outcomes: The students will be able to 1. Apply and implement fundamental data structures like stacks, queues, and pattern matching algorithms. 2. Apply and manipulate various types of linked lists, including practical applications such as polynomial addition. 3. Demonstrate the ability to use tree structures for different traversal methods and implement binary search trees for data retrieval and manipulation. 4. Apply heap trees, hash tables, and AVL trees for efficient data management. 5. Apply graph theory concepts and algorithms to solve graphical problems effectively.		
Reference Books: 1. Data Structures Using C, Reema Thareja, Third Edition, 2023, Oxford Higher Education, ISBN-10: 978-93-5497-717-7 2. "Data Structures and Program Design in C", Robert Kruse, C L Tondo, Bruce Leung and Shashi Mogalla, PHI, 2nd Edition, 2015. 3. Y. Langasm, M. J. Augenstein, A. M. Tenenbaum (2001) Data Structures Using C and C++, Prentice Hall India, New Delhi, India. 4. T. H. Cormen, C. E. Leiserson and R. L. Rivest (1990) Introduction to Algorithms, Third		

Edition, MIT Press, MA.
5. Fundamentals of Data Structures in C -- by Horowitz, Sahni and Anderson-Freed (Silicon Press 2007).
6. Data Structures and Algorithm Analysis in C++, Mark Allen Weiss, 4th Revised edition; 2013, Addison-Wesley, ISBN-13: 978-8131714744.

B.N.M. Institute of Technology

An Autonomous Institution under VTU, Approved by AICTE

Department of Artificial Intelligence and Machine Learning

SEMESTER – III

MICROCONTROLLER AND EMBEDDED SYSTEMS (PCI)
Credit: 3

Course Code	25AML135	CIA Marks	50
Teaching Hours/Week (L: T: P: J)	1:2:2:0	SEA Marks	50
Total Number of Lecture Hours	50	Exam Hours	03

Course Learning Objectives:
This course will enable students to

- Illustrate the logic design concepts and combinational logic circuits
- Provide the student with the basic understanding of microcontroller and embedded systems design.
- Learn the addressing modes, instructions, and assembler directives and develop the ALP to solve problems.
- Develop embedded C programs for microcontrollers and run on the simulator, target board and various interfaced hardware devices.
- Integrate Hardware and Software to Implement the required embedded smart systems

	Number of Hours	Bloom's Level
Module-1		
Logic Design & Applications: Review of Basic Logic gates. **Combinational Logic Circuits:** Sum-of-Products Methods, Karnaugh Map simplifications, Don't – care Conditions, Product-of-Sums Simplifications, Simplification by Quine-McClusky Method, Map Entered Variable Method, Multiplexers.	**6**	**Apply (CO1)**
Laboratory Component: Realize the following digital circuits using Digital Trainer kit. 1. Realization of Boolean expression: $Y= \bar{A}\,\bar{B}\,\bar{C}\,D + \bar{A}\,B\bar{C}\,D + AB\bar{C}\,D + \bar{A}\,\bar{B}\,C\,D + \bar{A}\,B\,C\,D + A\,\bar{B}\,C\,D + A\,B\,C\,\bar{D} + A\,B\,C\,D$ 2. Realize Half Adder, Full Adder, Half Subtractor and Full Subtractor using Logic Gates. 3. Realize Binary to Gray & Gray to Binary Code Converters using Logic Gates 4. Implement the following Boolean expression using 8:1 MUX. $F(A, B, C)=\Sigma(1,2,4,7)$	**4**	**Apply (CO1, CO2)**
Module-2		
ARM-32 bit Microcontroller: Thumb-2 technology and applications of ARM, Architecture of ARM Cortex M3, Various Units in the architecture, debugging support, General Purpose Registers, Special Registers, exceptions, interrupts, stack operation, reset sequence.	**6**	**Apply (CO2)**
Laboratory Component:	**4**	**Apply (CO2)**

Conduct the following experiments on an ARM CORTEX M3 evaluation board to learn ALP and using evaluation version of Embedded 'C' & Keil uVision-4 tool/compiler. 1. Write a program to multiply two 16 bit binary numbers. 2. Write a program to find the sum of first 10 integer numbers. 3. Write a program to find factorial of a number. 4. Implement an assembly program to search for a key element in a given array.		
Module-3		
ARM Cortex M3 Instruction Sets and Programming: Assembly basics, Instruction list and description, Special instructions, Useful instructions, Assembly and C language Programming	**6**	**Understand (CO3)**
Laboratory Component: Conduct the following experiments on an ARM CORTEX M3 evaluation board to learn ALP and using evaluation version of Embedded 'C' & Keil uVision-4 tool/compiler. 1. Write a program to find the square of a number (1 to 10) using look-up table. 2. Write a program to find the largest/smallest number in an array of 32 numbers. 3. Write a program to arrange a series of 32-bit numbers in ascending/descending order. 4. Write a program to count the number of ones and zeros in two consecutive memory locations.	**4**	**Apply (CO3, CO4)**
Module-4		
Embedded System Components: Embedded Vs General computing system, History of embedded systems, Classification of Embedded systems, Major applications areas of embedded systems, purpose of embedded systems Core of an Embedded System, Memory, Sensors, Actuators, LED, 7 segment LED display, stepper motor, Keyboard, Communication Interfaces (12C, SPI, IrDA, Bluetooth, Wi-Fi, Zigbee only).	**6**	**Apply (CO4)**
Laboratory Component: Conduct the following experiments on an ARM CORTEX M3 evaluation board to learn ALP and using evaluation version of Embedded 'C' & Keil uVision-4 tool/compiler. 1. Display “Hello World” message using Internal UART. 2. Interface and Control the speed of a DC Motor. 3. Interface a Stepper motor and rotate it in clockwise and anti-clockwise direction. 4. Display the Hex digits 0 to F on a 7-segment LED interface, with an appropriate delay in between.	**4**	**Apply (CO4)**
Module-5		
Introduction to Raspberry Pi: Different Models of Raspberry Pi, Peripherals of Raspberry Pi. Applications of Raspberry Pi, The Voltage hazard Information, General information on other pins and their functionality.	**6**	**Apply (CO5)**
Laboratory Component: Conduct the following experiments by writing program using Raspberry Pi board. 1. The LED Interfacing, 2. The First Button Interface with Raspberry Pi,	**4**	**Apply (CO5)**

3. UART example.		

Course outcomes:

The students will be able to

- Apply Karnaugh Map, Quine-McClusky Method and Map Entered Variable Method to simplify digital circuits. (Apply)
- Describe the architectural features and instructions of 32-bit microcontroller ARM CortexM3. (Understand)
- Apply the knowledge gained for Programming ARM Cortex M3 for different applications. (Apply)
- Understand the basic hardware components and their selection method based on the characteristics and attributes of an embedded system. (Understand)
- Interact with Arduino using Arduino sketch to program the devices. (Apply)

Reference Books:

1. M. Morris Mano, Digital Design, 4th Edition, Pearson Prentice Hall 2008
2. Joseph Yiu, "The Definitive Guide to the ARM Cortex-M3", 211d Edition, Newnes, (Elsevier), 2010.
3. Shibu K V, "Introduction to Embedded Systems", Tata McGraw Hill Education Private Limited, 2nd Edition.
4. Exploring Arduino: Tools and Techniques for Engineering, Wizardry 1st Edition WILEY, ISBN-10: 1118549368, ISBN-13: 978-1118549360.

B.N.M. Institute of Technology

An Autonomous Institution under VTU, Approved by AICTE

Department of Artificial Intelligence and Machine Learning

SEMESTER – III

Object Oriented Programming Using JAVA (PBL)
Credit: 2

Course Code	25AML136	CIA Marks	50
Teaching Hours/Week (L:T: P: J)	0:0:2:2	SEA Marks	50
Total Number of Lecture Hours	30	Exam Hours	03

Course Learning Objectives:

This course will enable students to

- Understand and apply the basic concepts of object-oriented programming.
- Implement java programs for establish interfaces and to develop reusable software components.
- Build software development skills using java programming for real-world applications.

Pre-requisite:

Basic Programming Concepts: Data Types, Variables, Control Flow, Functions/Methods.

	Number of Hours	Bloom's Level
Task 1		
An Overview of Java, Data Types, Variables, and Arrays, Operators, Control Statements, Classes and Methods 1. Write a JAVA program to display message "Welcome to BNMIT"and "I am first batch of Autonomous" in two different lines. 2. Write a JAVA program to display at-least five student information by considering student USN, name, branch and semester. 3. Write a java program that prints all real solutions to the quadratic equation $ax^2+bx+c=0$. Read in a, b, c and use the quadratic formula. 4. Write a java program to create objects of class Students with student USN, name, branch and semester and display information. Self study: Write a JAVA program to display at-least five student information by considering student USN, name, branch and semester.	**2**	**(CO1) Apply**
Task 2		
Method overloading, Inheritance, polymorphism, encapsulation 1. Calculate area of Rectangle, Triangle and Circle by using method overloading 2. Write a java program to create a class Car that inherits from base class Vehicle using private strings and getter/setter methods to achieve encapsulation. 3. Write a java program to develop a suitable hierarchy, classes for Point, Shape, Rectangle, Square, Circle, Ellipse, Triangle, Polygon, etc. Design a simple test application to demonstrate dynamic polymorphism. 4. Write a java program to create a class named Employee. Extend Faculty class from Employee class. Extend Professor Class from Faculty class.	**2**	**(CO1) Apply**

Access members of super class using super keyword. Create an instance to sub class called Professor and access members of both Faculty and Professor using instance. 5. Write a java program to create an abstract class named Shape that contains two integers and an empty method named print Area(). Self study: Write a java program to develop a suitable hierarchy, classes for Point, Shape, Rectangle, Square, Circle, Ellipse, Triangle, Polygon, etc. Design a simple test application to demonstrate dynamic polymorphism.		
Task 3		
Enumerations, Strings 1. Write a java program to create an enum of restaurants that can be used to pick user choice restaurant. 2. Given an input string, you are expected to extract either all vowels, or all non-vowels from the string and return the result as all lowercase or uppercase, based on the options specified. • input1 represents the input string. • input2 represents the extraction option. 0 for extraction of all non-vowels. 1 for extraction of all vowels. • input3 represents the output case option. 0 for all lowercase letters. 1 for all UPPERCASE letters. 3. Write a java program to find the duplicate words and their number of occurrences in a string. 4. Write a Java program to replace each substring of a given string that matches the given regular expression with the given replacement. Self-study: Write a Java program to replace each substring of a given string that matches the given regular expression with the given replacement.	**2**	**(CO1) Apply**
Task 4		
Exception Handling 1. Write a program in java if number is less than 10 and greater than 50 it generates the exception out of range. Else it displays the square of number. 2. Write a program in java to enter the number through command line argument. If first and second number is not entered then it will generate the exception. Also divide the first number with second number and generate the arithmetic exception. Self Study: Write a program in java to enter the number through command line argument. If first and second number is not entered then it will generate the exception. Also divide the first number with second number and generate the arithmetic exception.	**2**	**(CO2) Apply**
Task 5		

Multithreaded Programming 1. Write a java program for multithread in which user thread and thread started from main method invoked at a time each thread sleep for 1 sec. 2. Write a java program that implements a multi-thread application thathas three threads. First thread generates random integer every 1second and if the value is even, second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of cube of the number. 3. Write an application that executes two threads. One thread displays "An" every 1000 milliseconds and other displays "B" every 3000 milliseconds. Create the threads by extending the Thread class. Self Study: Write a java program that implements a multi-thread application that has three threads. First thread generates random integer every 1 second and if the value is even, second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of cube of the number.	**2**	**(CO2) Apply**
Task 6		
Collections 1. Write a Java program to create a new array list, add some colors (string) and print out the collection. 2. Write a Java program to iterate through all elements in a linked list starting at the specified position. 3. Write a Java program to append the specified element to the end of a hash set. 4. Write a Java program to create a new tree set, add some colors (string) and print out the tree set. 5. Write a Java program to create a new priority queue, add some colors (string) and print out the elements of the priority queue. Self Study: Write a Java program to iterate through all elements in a linked list starting at the specified position.	**2**	**(CO1) Apply**
Task 7		
Collections 1. Write a Java program to associate the specified value with the specified key in a Tree Map. 2. Write a Java program for the following: i) Create a doubly linked list of elements. ii) Delete a given element from the above list. iii) Display the contents of the list after deletion. 3. Write a Java program to store content in Hash table and use enumeration to display contents of Hash Table. 4. Write a Java program to create a vector of n elements and perform the following operations: Adding elements, Removing elements and Display elements. 5. Write a program to add elements to the HashMap given the key and value data type is string, get size of HashMap, and check if HashMap is empty.	**2**	**(CO1) Apply**

Self Study: Write a Java program to store content in Hash table and use enumeration to display contents of Hash Table.		
Task 8		
Event Handling 1. Write a Java program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num 2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a Number Format Exception. If Num2 were Zero, the program would throw an Arithmetic Exception. Display the exception in a message dialog box.	**2**	**(CO3) Apply**
Task 9		
,Event Handling 1. Write a Java program that handles all mouse events and shows the event name at the center of the window when a mouse event is fired. Self Study: Write a Java program that handle keyboard events in Java	**2**	**(CO3) Apply**
Task 10		
Event Handling 1. Write a java program that simulates a traffic light. The program lets the user select one of three lights: Red, Yellow or Green with radio buttons. On selecting a button an appropriate message with "STOP" or "READY" or" GO" should appear above the buttons in selected color. Initially, there is no message shown. Self Study:Write a java program that works as a simple calculator. Use a grid layout to arrange buttons for the digits and for the +,-,*, % operations. Add a text field to display the result. Handle any possible exception like divided by zero.	**2**	**(CO3) Apply**
Task 11		
Event Handling 1. Write a java program to create a frame that contains two buttons and one text field. 2. Write a java program to handle the button click events by implementing ActionListener Interface. 3. Write a java program to create two textfields to display single line text string and one TextArea that is used to display multiple-line text string. Both should be editable in nature. Self Study: Write a java program to create a drop-down menu of choices. When a user selects a particular item from the drop-down then it is shown on the top of the menu.	**2**	**(CO3) Apply**
Task 12		

Java Script 1. Write a program to Swap Two Variables. 2. Write a program to Generate a Random Number. 3. Write a program to Check the Number of Occurrences of aCharacter in the String. 4. Write a program to Count the Number of Vowels in a String. 5. Write a java script program to pass a 'javascript function' as parameter. Self Study: Write a program to Count the Number of Vowels in a String.	**2**	**(CO1) Apply**
Task 13		
File handling 1. Write a java program that reads a file name from the user, and then displays information about whether the file exists, whether the fileis readable, whether the file is writable, the type of file and the length of the file in bytes. Self Study: Write a java program that displays the number of characters, lines and words in a text file.	**2**	**(CO2) Apply**
Task 14		
File handling 1. Write a java program that reads a file and displays the file on the screen with line number before each line. 2. Write a java program in which data is read from one file and shouldbe written in another file. Name of both file is given throughcommand line arguments. Self Study: Write a java program in which data is read from one file and should be written in another file line by line.	**2**	**(CO2) Apply**
Task 15		
Mini-Project • Develop real world application using graphical user interface and object-orient concept for selected problem statement. • The problem statement can be selected from the following title but not limitedto the same. 1. Electricity bill generation 2. Currency converter / Distance converter / Time converter 3. Pay slip generation 4. Online book store 5. Airline reservation system 6. Designing of simple calculator		**(CO4) Create**

Course Outcomes:

The students will be able to

1. Develop Java application programs to implement basic Object Oriented concepts. (Apply)
2. Apply the concepts of Multithreading, Exception handling, event and file handling to develop efficient and error free codes. (Apply)
3. Design event driven GUI and web related applications which mimic the real word scenarios. (Apply)
4. Design, implement, test, and debug graphical user interfaces to solve real time applications.(Create- for Mini project).

Reference Books:

1. Herbert Schildt, "Java, The complete Reference", Tata McGraw-Hill, 12th Edition,2022.
2. P. J. Deitel, H. M. Deitel, "Java for Programmers: with Generative AI", Pearson Education, PHI, 5th Edition,2025.
3. P. Radha Krishna, "Object Oriented Programming through Java", Universities Press, 2ndEdition, 2007
4. Bruce Eckel, "Thinking in Java", Pearson Education, 4th Edition, 2006.
5. Sachin Malhotra, Saurabh Chaudhary, "Programming in Java", Oxford University Press, 5thEdition, 2010.

B.N.M. Institute of Technology

An Autonomous Institution under VTU, Approved by AICTE

Department of Mathematics

Syllabus

Semester: IV

Course: Statistics, Probability and Graph theory

Course Code: 25MAC141 (Common to CSE, ISE, AIML)

L:T:P:J	**2:2:0:0**	**CIA: 50**
Credits:	**03**	**SEA: 50**
Hours:	**40**	**SEA Duration: 03 Hours**

Course Learning Objectives: The students will be able to

1	Provide an insight into applications of Graph Theory, Curve fitting & Statistical methods.
2	Develop the knowledge of probability, joint probability distribution and Queuing theory occurring in digital signal processing, design engineering and micro wave engineering.

Module-1: Curve fitting & Statistical methods	No. of hours	Blooms cognitive Levels
Examples from Engineering that require curve fitting and statistical methods. **Curve Fitting:** Curve fitting by the method of least squares-fitting the curves of the form: $y = ax+b$, $y = ax^b$ *and* $y = ax^2 + bx + c$. **Statistical methods:** Introduction to Moments, Skewness, Kurtosis and problems. Karl Pearson's coefficient of correlation and lines of regression. *Experiential Learning component: Problems on curve fitting and statistical methods*	**L: 04** **T: 04**	**L1** **L2** **L3**
Module-2: Probability distributions & Joint probability distribution		
Examples from Engineering that require Probability and Joint probability distribution **Probability distributions:** Review of basic probability theory. Discrete and continuous Random variables, probability mass/density functions (definitions only). Binomial, Poisson, exponential and normal distributions (without proof). **Joint probability distribution:** Joint Probability distribution for two discrete random variables, expectation, covariance and correlation. *Experiential Learning component: Problems on Binomial, Poisson, Exponential and Normal distributions*	**L: 04** **T: 04**	**L1** **L2** **L3**
Module-3:Markov chain & Sampling theory		
Examples from Engineering that require Markov Chain and Sampling Theory **Markov chain**: Introduction to Stochastic process, Probability vectors, Stochastic matrices, Regular stochastic matrices, Markov Chains, Higher transition probabilities, Stationary distribution of Regular Markov chains and absorbing states, Markovian processes. **Sampling theory**: Introduction to sampling theory, testing of hypothesis, level of significance, confidence limits, test of significance of mean and difference of means for large samples-z-test, test of significance of small Samples-Student's t- distribution. *Experiential Learning component: Problems on Markovian processes and, Sampling Theory*	**L: 04** **T: 04**	**L1** **L2** **L3**
Module-4: Queuing theory		
Examples from Engineering that require queueing theory Introduction, birth and death process, Kendall's Notation, Symbolic representation of a queuing model, single server Poisson queuing model with infinite capacity (M/M/1: ∞/FCFS), when $\lambda_n = \lambda$ and $\mu_n = \mu(\lambda < \mu)$, Multiple server Poisson queuing model with infinite capacity (M/M/S: ∞/ FCFS), when $\lambda_n = \lambda$ for all $n, (\lambda < S\mu)$, *Experiential Learning component: Problems on* (M/M/1: ∞/FCFS) and (M/M/S: ∞/ FCFS) queuing models	**L: 04** **T: 04**	**L1** **L2** **L3**
Module-5: Graph theory		
Examples from Engineering that require graph theory Basic concepts, types of graphs, order and size of a graph, in-degree and out-degree, bipartite-graphs, connected and disconnected graphs, Eulerian graph, Hamiltonian graphs, sub-graphs, isomorphic graphs. Matrix representation of graphs, adjacency matrix, incidence matrix. Planar graphs: definition, characterization of planar graphs, Kuratowski's theorem, Euler's formula and consequences. *Experiential Learning component: Problems on detection of planar and non-planar graphs*	**L: 04** **T: 04**	**L1** **L2** **L3**

Course Outcomes: After completing the course, the students will be able to

CO 1:	Make use of correlation and regression analysis to fit a suitable mathematical model for the statistical data.
CO 2:	Apply discrete and continuous probability and joint probability distributions in analyzing the probability models arising in engineering field.
CO 3:	Use Markov chain in prediction of future events and demonstrate the validity of testing the hypothesis.
CO 4:	Acquire skills in analyzing queuing models.
CO 5:	Apply the knowledge of Graph Theory in Network modeling, electrical network and computational algorithms.

CO - PO Mapping:

COs	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO 1	3	2			2							
CO 2	3	2			2							
CO 3	3	2			2							
CO 4	3	2			2							
CO 5	3	2			2							

Reference Books:

1. E. Kreyszig: "Advanced Engineering Mathematics", John Wiley & Sons, 10th Edition(Reprint), 2016.
2. B. S. Grewal: "Higher Engineering Mathematics", Khanna Publishers, 44th Edition, 2017.
3. S. D. Sharma : "Operations Research", Kedar Nath Ram Nath & Co. Meerut, 2014.
4. T. Veerarajan : Probability, Statistics and Random processes, McGraw Hill Education(India) Private Limited, Third edition, Nineteenth reprint 2017.
5. C. Ray Wylie, Louis C. Barrett : "Advanced Engineering Mathematics", 6th Edition, McGraw-Hill Book Co., New York, 1995.
6. B. V. Ramana: "Higher Engineering Mathematics" 11th Edition, Tata McGraw-Hill, 2010.
7. Srimanta Pal & Subodh C. Bhunia: "Engineering Mathematics", Oxford University Press, 3rd Reprint, 2016.
8. Narsingh Deo, "Graph Theory with Applications to Engineering and Computer Science", Prentice Hall of India, 2000.

Web links and Video Lectures:

1. https://nptel.ac.in/courses/111104098
2. https://www.youtube.com/watch?v=1YkfeR05YXY
3. https://archive.nptel.ac.in/courses/111/104/111104079/
4. https://www.youtube.com/watch?v=xGkpXk-AnWU
5. https://archive.nptel.ac.in/courses/106/104/106104170/

B.N.M. Institute of Technology

An Autonomous Institution under VTU, Approved by AICTE

Department of Artificial Intelligence and Machine Learning

Operating Systems (PCC)

SEMESTER–IV

Subject Code: **25AML142**	L:T:P: J: 3:0:0:0	CIA Marks:	50
Credits:	3	SEA Marks:	50
Total Number of Lecture Hours	40	SEA Duration:	3 Hours

Course Objectives:

- Introduce basic concepts of OS, Process management and CPU scheduling to improve CPU utilization
- Illustrate process synchronization and Deadlocks in managing concurrent processes.
- Illustrate the concepts of main memory, virtual memory and secondary memory such as SSD for effective memory management.
- Introduction to Unix File System and file related commands
- Illustrate file attributes and shell programming to write simple shell scripts

Prerequisites: Basic theoretical concepts of C programming

Module 1: Introduction to Operating System & ProcessManagement	**Teaching Hours**	**Blooms cognitive Levels**
Fundamental Concepts of Operating System: Introduction to Operating Systems, Operating system functions and services, historical evolution of operating systems, System boot. **Process Management:** Process abstraction, process address space, process management, system calls, threads. **CPU Scheduling:** Levels of scheduling, comparative study of scheduling algorithms. Multilevel Queue Scheduling, Multilevel Feedback Queue Scheduling **Practical:** Usage of system calls using C (fork, vfork, exec, read, write, seek, etc) **Self-Study Component:** Multi-processor scheduling.	**8**	**(CO1) Apply**
Module 2: Process Synchronization and Deadlocks		
Concurrent Processes: Critical section problem, semaphores, Classical problems of synchronization, monitors, inter-process communication, **Deadlocks:** Characterization, prevention and avoidance, deadlock detection and recovery. **Practical:** IPC using pipes, fifo, shared memory **Self-Study Component: M**essage passing mechanisms. Applications of deadlocks	**8**	**(CO2) Apply**
Module 3: Memory Management		
Memory Management: Background; Swapping; Contiguous memory allocation; Paging; Structure of page table; Segmentation, virtual memory concept. **Solid State Drives**-SSD Architecture, Flash Controller, Garbage Collection, Bad Block Management. **D**emand paging, page replacement algorithms, Disk Scheduling. **Practical:** Implementation of page replacement algorithms, disk scheduling algorithms **Self-Study Component:** Thrashing	**8**	**(CO3) Apply**
Module 4: UNIX file system		

Unix files: UNIX Architecture, Naming files. Basic file types/categories. Organization of files. Hidden files. Standard directories. Parent child relationship. The home directory and the HOME variable. Reaching required files- the PATH variable, manipulating the PATH, Relative and absolute pathnames. Directory commands – pwd, cd, mkdir, rmdir commands. The dot (.) and double dots (..) notations to represent present and parent directories and their usage in relative path names. File related commands – cat, mv, rm, cp, wc and od commands. **Practical:** Shell commands **Self-Study Component:** Execution of UNIX Shell Commands	**8**	**(CO4) Apply**
Module5: File attributes and Shell programming		
File attributes and permissions: The ls command with options. Changing file permissions: the relative and absolute permissions changing methods. Recursively changing file permissions. Directory permissions. The shells interpretive cycle: Wild cards. Removing the special meanings of wild cards. Three standard files and redirection. **Connecting commands**: Pipe, Basic and extended regular expressions, grep,egrep. **Shell programming**: Ordinary and environment variables. Read and read-only commands. Command line arguments. exit and exit status of a command. Logical operators for conditional execution. The test command and its shortcut. The if, while, for and case control statements. The set and shift commands and handling positional parameters. The here (<<) document. Simple shell program examples. **Practical:** Shell scripting **Self-Study Component:** Execution of Wildcards & UNIX Shell Programs	8	**(CO5) Analyze**

Course Outcomes: After completing the course, the students will be able to

1. Apply the concepts of process management and process scheduling techniques to enhance CPU efficiency and throughput.
2. Apply process synchronization mechanisms, Identify the need of policies and protection required in managing deadlock
3. Apply memory management concepts to optimize memory usage and performance.
4. Apply the concepts of Unix system and file commands to perform various tasks in files and system.
5. Analyze the concepts of file attributes and manage files through shell programming to automate tasks

Reference Books:

1. Sumitabha Das., Unix Concepts and Applications., 4th Edition., Tata McGraw Hill.
2. Abraham Silberschatz, Peter Baer Galvin, Greg Gagne, Operating System Principles 10th edition,Wiley-India, 2018.
3. William Stallings, Operating System: Internals and Design Principles, Prentice Hall, 9th Edition, 2018.
4. W. Richard Stevens: Advanced Programming in the UNIX Environment, 2nd Edition, PearsonEducation, 2005
5. Unix System Programming Using C++ - Terrence Chan, PHI, 1999.

Web links and Video Lectures:

1. https://academicearth.org/
2. https://archive.nptel.ac.in/courses/106/105/106105214/

BNM Institute of Technology

Dept. of Artificial Intelligence & Machine Learning
Choice Based Credit System (CBCS) and Outcome Based Education (OBE)

Semester: IV

Course Name: Database Management System (PCI)
Course Code: 25AML143

L: T: P: J	2: 0: :2 :0	**CIA Marks:** 50
Credits:	3	**SEA Marks:** 50
Total Number of Lecture Hours:	4	**SEA Duration:** 03 Hours

Course Learning Objectives: The students will be able to

1	Understand fundamental concepts, terminology and application of databases, SQL and NoSQL
2	Design concepts and creation of relational databases using relation algebra.
3	Practice SQL programming through a variety of database problems.
4	Demonstrate the use of Normalization, concurrency and transactions in database.

Module-1: Database System Concepts, Data Modeling	**No. of Hours**	**Bloom's Levels**
Databases and Databases Users: Characteristics of database Approach, Advantages of using the DBMS Approach. Database System Concepts and Architecture: Data Models-Schemas and Instances, Three-Schema Architecture and Data Independence, Database Languages and Interfaces. Data Modeling Using the Entity-Relationship (ER) Model: Entity Types-Entity sets- Attributes and Keys, Relationship types – Relationship Sets – Roles and structural Constraints, Weak Entity Types. **Self-Study Components:** Actors on the scene and behind the scenes, DBMS component modules interaction **Practical component:** Draw ER Diagram for the following Databases using GitMind software. Order Database Library Database Bank Database	8+2	Understand CO1
Module-2: Relational Data Model and Relational Algebra		
Concept of relations, schema-instance distinction, keys, referential integrity and foreign keys, converting the database specification in E/R notation to the relational schema. relational algebra operators: selection, projection, set operators, cross product, various types of joins, division, Basic SQL: SQL Data Definition and Data Types, Specifying Constraints in SQL, Basic Retrieval Queries in SQL. **Self-Study Components:** Example queries **Practical component:** Create Schema, insert at least 5 records in each table and add appropriate constraints for the following Library Database using ORACLE or MySQL DBMS under LINUX/Windows environment BOOK (Book_id, Title, Publisher_Name, Pub_Year) BOOK_AUTHORS (Book_id, Author_Name)	8+2	Apply CO2

PUBLISHER (Name, Address, Phone) BOOK_COPIES (Book_id, Branch_id, No-of_Copies) BOOK_LENDING (Book_id, Br_id, Card_No, Date_Out, Due_Date) LIBRARY_BRANCH (Branch_id, Branch_Name, Address) Write SQL queries to 1. Retrieve details of all books in the library – id, title, name of publisher,authors, number of copies in each branch, etc. 2. Get the particulars of borrowers who have borrowed more than 3 books,but from Jan 2020 to Jun 2022. 3. Delete a book in BOOK table. Update the contents of other tables toreflect this data manipulation operation.		
Module-3: SQL		
INSERT – DELETE and UPDATE Statements in SQL, Additional features in SQL **More SQL: Complex Queries, Triggers, Views and Schema Modification:** Complex SQL Retrieval Queries, Specifying Constraints as Assertions and actions as Triggers, Views (Virtual Tables) in SQL. **Self-study components**: Stored Procedures **Practical component:** Create Schema, insert at least 5 records for each table and add appropriate constraints for the following Order Database using ORACLE or MySQL DBMS under LINUX/Windows environment. SALESMAN (Salesman_id, Name, City, Commission) CUSTOMER (C_id, Cust_Name, City, Grade, Salesman_id) ORDERS (Ord_No, Purchase_Amt, Ord_Date, C_id, S_id) Write SQL queries to 1. Count the customers with grades above Bangalore's average. 2. Find the name and numbers of all salesman who had more than onecustomer. 3. List all the salesman and indicate those who have and don't have customers in their cities (Use UNION operation.) 4. Create a view that finds the salesman who has the customer with the highest order.	8+2	Apply CO3
Module-4: Functional Dependencies and Normalization		
Basics of Functional Dependencies and Normalization for Relational Database: Informal design guidelines for relation schema, Functional Dependencies, Armstrong's axioms for FD's, closure of a set of FDs, Equivalence of set FD's, minimal covers, Normal forms Based on Primary Keys, General Definitions of Second and Third Normal Forms, Boyce- Codd Normal Forms [BCNF], Multivalued Dependency and Fourth Normal Form, Join Dependencies and Fifth Normal Form. **Self-study Components:** Loss less Join Property and Dependency preservation property **Practical component:** Create Schema, insert at least 5 records for each table and add appropriate constraints for the following Company Database using ORACLE or MySQL DBMS under LINUX/Windows environment. EMPLOYEE (SSN, Name, Address, Sex, Salary, Super SSN, D No) DEPARTMENT (D No, D Name, Mgr. SSN, Mgr. Start Date)	8+2	Analyze CO4

DLOCATION(D No,D Loc) PROJECT (P No, P Name, P Location, D_No) WORKS_ON(SSN, P No, Hours) Write SQL queries to 1. Make a list of all project numbers for projects that involve an employee whose last name is 'Scott', either as a worker or as a manager of the department that controls the project. 2. Show the resulting salaries if every employee working on the 'IoT' project is given a 10 percent raise. **3.** Find the sum of the salaries of all employees of the 'Accounts' department, as well as the maximum salary and the average salary in this department.		
Module-5: Transaction Processing, Concurrency Control, NoSQL		
Introduction to Transaction Processing –Introduction to Transaction Processing, Desirable Properties on Transactions (ACID) **Concurrency Control Techniques:** Transactions and Schedules, Serializability, Precedence Graphs, Concurrency, Lock Based Protocols: 2PL, Strict 2PL Protocols, Deadlocks - Detection and Prevention **NoSQL:** SQL v/s NoSQL, The Emergence of NoSQL, BASE Properties, Schema less Database, Vector Database, Distributed Database. Self-study components: **Graph Databases**	**8+2**	Analyze CO5
Project: Build a NoSQL-based web application using MongoDB, few applications are listed here but not restricted to: 1. Student Course Management System 2. Library Management System 3. To-Do App with User Authentication 4. Personal Finance Tracker **5.** Online Polling System		

Course Outcomes: After completing the course, the students will be able to	
CO1	Understand the Database System Concepts along with Data Modeling Using the Entity-Relationship (ER) Model
CO2	Apply the concepts of relations on RDBMS, constraints, joints using relational algebra operators.
CO3	Apply Structured Query Language for database manipulation.
CO4	Analyze functional dependencies to normalize relations of relational database
CO5	Analyze transactions processing, schedules protocols, serializability issues, deadlocks in DBMS and concepts of NoSQL with its advantages

Textbooks

1. Ramez Elmasari, Shamkant B Navathe "Fundamentals of Database Systems", Pearson, Seventh Edition 2017.
2. "Database System Concepts", Silberschatz, H Korth, S Sudarshan, 6th Edition, McGraw-Hill, 2010
3. Pramod J Sadalage, Martin Fowler, "NOSQL Distilled", Pearson, 2013

B.N.M. Institute of Technology

An Autonomous Institution under VTU, Approved by AICTE

Department of Artificial Intelligence and Machine Learning

SEMESTER – IV

Design and Analysis of Algorithms (PCI)
Credit: 4

Course Code	25AML144	CIA Marks	50
Teaching Hours/Week (L: T: P: J)	3:0:2:0	SEA Marks	50
Total Number of Lecture Hours	50	Exam Hours	03

Course Learning Objectives:
This course will enable students to

- Understand and apply mathematical relations to analyze the asymptotic runtime complexity of algorithms.
- Utilize brute force, divide and conquer, and decrease and conquer methodologies to solve various computational problems.
- Synthesize efficient algorithms in common engineering design situations.
- Solve problems using backtracking and branch and bound methods and gain insights into the theoretical limits of algorithmic problem-solving and computational complexity.

Prerequisite: Data Structure Concepts and Java Programming

	Number of Hours	**Bloom's Level**
Module-1: Introduction		
Introduction: Notion of Algorithm, Fundamentals of Algorithmic Problem Solving, Fundamentals of the Analysis of Algorithmic Efficiency: Analysis framework, Asymptotic Notations and Basic Efficiency Classes, Mathematical Analysis of Non-Recursive and Recursive Algorithms. Self-Study: Space complexity **Practical:** 1. Apply brute force technique to recursively implement the following a. Linear Search b. To find the maximum and minimum from a given list of n elements. 2. There are 5 books in the shelf, find the number of ways to select 3 books from 5 books on the shelf using the NCR with recursion. 3. Find the next three terms of the sequence 15, 23, 38, 61, … Fibonacci series of the given number using recursion.	**6+4**	**(CO1) Analyze**
Module-2: Brute Force, Divide and conquer		
Brute Force: Selection Sort **Divide and Conquer:** General method: Binary search, Recurrence equation for divide and conquer, Finding the Maximum and Minimum, Merge sort, Quick sort, Strassen's matrix multiplication. Decrease and Conquer Approach: Topological Sort.		

Self-Study: Space-Time Tradeoffs: Sorting by Counting, Input Enhancement in String Matching.		

Practical: 1. Implement the Selection sort algorithm. 2. Write a program to search for a key in a given set of elements using the Binary search method and find the time complexity required to find the key. 3. Sort a given set of elements using the Merge Sort method and determine the time required to sort the elements. Plot a graph of number of elements versus time taken. Specify the time efficiency class of this algorithm. 4. Sort a given set of elements using Quick Sort method and determine the time required to sort the elements. Plot a graph of number of elements versus time taken. Specify the time efficiency class of this algorithm. **5.** Implement Topological sort using the source removal method to find the time required to sort the elements.	**6+4**	**(CO2) Apply**
Module-3: Greedy Method		
Greedy Method: General method, Coin Change Problem, Knapsack Problem, Job sequencing with deadlines. Minimum cost spanning trees: Prim's Algorithm, Kruskal's Algorithm. Single source shortest paths: Dijkstra's Algorithm. Optimal Tree problem: Huffman Trees and Codes. Self-Study: Minimize Number of Platforms (Train Station Problem) **Practical:** 1. Write a program to find maximum profit using the Knapsack technique. 2. Implement Prim's algorithm and Find Minimum Cost Spanning Tree of a given connected undirected graph. 3. Implement Kruskal's algorithm and Find Minimum Cost Spanning Tree of a given connected undirected graph. 4. Implement Dijkstra's algorithm finds shortest paths to other vertices from a given vertex in a weighted connected graph.	**6+4**	**(CO3) Apply**
Module-4: Dynamic Programming		
The General Method with examples, Multistage Graphs. Transitive Closure: Warshall's Algorithm, Floyd's Algorithm for the All-Pairs Shortest Paths Problem, Single-Source Shortest Paths: General Weights, 0/1 Knapsack, The Traveling Salesperson problem. Self-Study: Longest Common Subsequence **Practical:** 1. Implement all-pairs shortest paths problem using Floyd's algorithm. 2. Implement transitive closure problem using Warshal's algorithm. 3. Implement 0/1 Knapsack using Dynamic Programming. 4. Implementation of Bellman Ford Algorithm using a directed graph. 5. Implementation of Travelling Salesman Problem using dynamic programming.	**6+4**	**(CO4) Apply**

Module-5: Backtracking		
Backtracking: General method (T2:7.1), Backtracking: n - Queens problem, Hamiltonian Circuit Problem, Subset – Sum Problem, Graph coloring, **Branch-and-Bound:** Assignment Problem, Knapsack Problem, Traveling Salesperson Problem. Basic concepts of P, NP, NP-Complete and NP-Hard Problems, Challenges of Numerical Algorithms. Self -Study: Approximation algorithms **Practical:** **1.** Implementation of N Queen Problem using Backtracking technique. **2.** Implementation of SUM-SUBSET Problem. **3.** Design and implement the Hamiltonian Cycle in an undirected Graph G of n vertices. **4.** Implementation of 0/1 Knapsack Problem using Branch and Bound.	**6+4**	**(CO5) Apply**

Course outcomes:

The students will be able to

1. Analyze the asymptotic runtime complexity of algorithms by using mathematical relations that help to identify them in specific instances.
2. Apply and solve problems using brute force, divide and conquer techniques, decrease and conquer.
3. Apply problem-solving methodologies such as greedy methods to solve a given problem.
4. Apply dynamic programming to estimate the computational complexity of different algorithms.
5. Apply and solve problems using backtracking, branch and bound methods and describe P, NP and NP Complete problems.

Reference Books:

1. Anany Levitin, Introduction to the Design and Analysis of Algorithms, Pearson, 2nd Edition, 2009]
2. Thomas H. Cormen, Charles E. Leiserson, Ronal L. Rivest, Clifford Stein, Introduction to Algorithms, PHI, 3rd Edition
3. Ellis Horowitz, Satraj Sahni and Rajasekaran, Computer Algorithms/C++, Universities Press, 2nd Edition, 2014.

B.N.M. Institute of Technology

An Autonomous Institution under VTU, Approved by AICTE

Department of Artificial Intelligence and Machine Learning

SEMESTER – IV

Foundation of Machine Learning (PCI)

Credit: 3

Course Code	25AML145	CIA Marks	50
Teaching Hours/Week (L: T: P: J)	1:2:2:0	SEA Marks	50
Total Number of Lecture Hours	50	Exam Hours	03

Course Learning Objectives:

This course will enable students to

- Understand the basic concepts of learning, machine learning and its landscape,
- Understand the concept of linear models for classification and regression,
- Understand the concepts of decision trees and support vector machines,
- Understand the concept of ensemble learning, and
- Understand model-free learning over instance-based learning.

Prerequisites: Python, Linear Algebra

	Number of Hours	Bloom's Level
Module 1: Introduction to Machine Learning		
Machine Learning Landscape: Types of Machine Learning (ML), Main Challenges of ML, Testing and Validation **Typical End-to-End Machine Learning Project:** Working with Real Data, The Big Picture, Getting Data, Discovering and Visualizing the Data, Data Preparation- Feature engineering, Selecting and Training Models **Practical:** 1. Build a simple linear model to predict the life satisfaction index of a country given its GDP considering an appropriate dataset. 2. Perform data preprocessing and transformation considering an appropriate dataset. Self-Study: Model Optimization-Fine Tuning Model.	**6+4**	**(CO1) Apply**
Module 2: Linear Models		
Linear Models for Classification: Binary Classifier, Measuring Performance, Multiclass Classification, Multilabel Binary and Multilabel Multiclass Classification (Multioutput Classification), Error Analysis, Logistic Regression **Linear Models for Regression:** Linear Regression, Gradient Descent, Polynomial Regression, Learning Curves, Regularized Linear Models (Ridge Regression, Lasso Regression, Elastic Net Regression)	**6+4**	**(CO2) Apply**

Practical: 1. Build a machine learning model to classify a handwritten digit using a linear model. 2. Build a machine learning model to predict California house prices using different linear models. Self-Study: Early Stopping		
Module 3: Non-linear Models		
Support Vector Machines: Linear and Non-linear Support Vector Machine (SVM) Classification, SVM Regression, Training Linear SVM Classifiers **Decision Trees**: Training and Visualizing Decision Tree, Making Predictions, Estimating Class Probabilities, ID3 and CART Algorithm, Computational Complexity, Impurity Measurement – Gini and Entropy, Regularization, Regression **Practical:** 1. Build a machine learning model using both linear and kernelized SVM algorithms to predict wine class considering Wine recognition dataset. 2. Build a decision tree classifier to classify Iris flower species. Self-Study: Issues in Decision Trees – Sensitivity to Axis Orientation and High Variance	**6+4**	**(CO3) Apply**
Module 4: Ensemble Learning		
Voting Classifiers, Bootstrap Aggregating (Bagging), Pasting, Out-of-bag Evaluation, Random Patches, Random Subspaces, Random Forests, Boosting – Adaptive and Gradient Boosting **Practical:** 1. Implement a voting classifier to classify handwritten digits from MNIST dataset. 2. Implement a stacked ensemble for classification on an appropriate dataset and check if it can outperform a voting ensemble. Self-Study: Stacking	**6+4**	**(CO4) Apply**
Module 5: Instance-based Learning		
k-Nearest Neighbor Learning **-** Distance-Weighted Nearest Neighbor Algorithm and remarks; Locally Weighted Regression **-** Locally Weighted Linear Regression and remarks; Radial Basis Function; **Practical:**	**6+4**	**(CO5) Apply**

1. Implement a *k*-Nearest Neighbor algorithm to classify an instance of an appropriate dataset and compare the prediction performance. 2. Implement a locally weighted linear regression algorithm for a regression task considering an appropriate dataset. Self-Study: Cased-based Reasoning; Remarks on Lazy and Eager Learning		

Course outcomes:

The students will be able to

1. Apply the basic concepts learning and basic machine learning theory in machine learning problems.
2. Apply linear models for classification and regression tasks.
3. Apply decision trees and support vector machines in appropriate machine learning problems.
4. Apply ensemble learning to achieve relatively better model performance than individual ones.
5. Apply instance-based learning using *k*-Nearest Neighbour algorithm in classification and regression tasks.

Reference Books:

1. Aurélien Géron. *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow*, O'Reilly, 2025
2. Tom M. Mitchell. *Machine Learning*, McGraw Hill Education, Indian Edition, 2013

B.N.M. Institute of Technology

An Autonomous Institution under VTU, Approved by AICTE

Department of Artificial Intelligence and Machine Learning

SEMESTER – IV

Cloud Computing & Applications (PBL)
Credit: 2

Course Code	25AML146	CIA Marks	50
Teaching Hours/Week (L: T: P: J)	0:0:2:2	SEA Marks	50
Total Number of Lecture Hours	30	Exam Hours	03

Course Learning Objectives:

This course will enable students to

- Understand the fundamental concepts, architecture, deployment models, and service models of cloud computing along with virtualization technologies and cloud platforms.
- Develop and manage virtualized and cloud-based environments using tools such as VirtualBox, Google App Engine, AWS, and Microsoft Azure.
- Apply version control techniques and collaborative development practices using Git and GitHub repositories.
- Design, deploy, and manage containerized and orchestration-based applications using Docker, Docker Compose, and Kubernetes platforms.

Prerequisites:

Fundamental knowledge of CS, OS, networks, programming (Java/Python), web tech, databases, Linux, virtualization, and basic cloud platform concepts.

	Number of Hours	Bloom's Level
Task – 1		
Introduction, Cloud Infrastructure: Cloud computing, Cloud computing delivery models and services, Ethical issues, Cloud vulnerabilities, Cloud computing at Amazon, Cloud computing the Google perspective, Microsoft Windows Azure and online services, Opensource software platforms for private clouds, Cloud storage diversity and vendor lock-in. Cloud Deployment Models, Benefits and Challenges of Cloud Computing **Self-Study:** Explore one open-source cloud platform (e.g., OpenStack, Eucalyptus, NextCloud)	**4**	**Understand**
Task – 2		
Virtualization in Cloud Computing and Types : What is Virtualization? Why is Virtualization Important? Virtualization Example How does Virtualization Work, Virtual Machines (Cloud Instances) Types of Virtualization: Application Virtualization, Network Virtualization, Desktop Virtualization, Storage Virtualization, Server Virtualization, Data virtualization ; Benefits of Virtualization; Drawback of Virtualization; Characteristics of Virtualization; Virtualization vs Containerization Uses of Virtualization **Task for Execution:** Setting up a VIRTUALBOX/VMware Workstation, install a C Compiler in the virtual machine and execute a sample program **Self-Study:** Choose an organization or industry (e.g., banking, education, healthcare) and research how they use virtualization. Prepare a 1-page case study summary or short presentation on it.	**4**	**Apply**
Task – 3		
Introduction: What is Git? What is Git History? Why Use It? Where to use Git? Key Git Concepts: Repository, Clone, Stage, Commit, Branch, Merge, Pull, Push **Task for Execution :** Creating a GIT repository and executing the control system	**2**	**Apply**

commands to Clone, Commit, Push, Fetch, Pull, Checkout, Reset and Delete **Self-Study:** Simulate a merge conflict in a Git repo. Study how .gitignoreworks. Create a sample .gitignorefor a Python/Node.js/Java project.		
Task – 4		
What is Google App Engine? Features of App Engine, Advantages of Google App Engine. **Task for Execution** : 1. Install GOOGLE APP ENGINE and Host A Static Website On GOOGLE APP ENGINE **Self-Study:** Deep dive into the app.yaml configuration file. Understand the key fields: runtime, handlers, env_variables, automatic_scaling	**2**	**Apply**
Task – 5		
What is Cloudsim? Benefits of simulation over the actual deployment, Why use Cloudsim?, Cloudsim Architecture, Features of Cloudsim, Installation Steps, **Task for Execution:** 1. Simulate A Cloud Scenario Using Cloudsim and Run A FCFS Scheduling Algorithm. 2. Simulate A Cloud Scenario Using Cloudsim and Run A SJF Scheduling Algorithm. **Self-Study:** Research how researchers and industry professionals use CloudSim for: Testing Scheduling policies, Performance evaluation, Cost estimation.	**2**	**Apply**
Task – 6		
AWS Data Services: Identity Access Management- IAM, Elastic Cloud Compute- EC2, AWS RDS, AWS Glue, AWS Redshift, AWS EMR, AWS - Dynamo DB, AWS QuickSight, AWS SageMaker **GCP Data Services**: Compute Engine, Cloud Storage, Cloud SQL, Spanner, Datastore, Bigtable, BigQuery, Data Proc, DataFlow, Cloud Composer, Dataprep, Data Fusion, Cloud AutoML, Looker **Task for Execution:** To steup an AWS account and explore the services offered by AWS **Self-Study:** Use Amazon S3 to create a public bucket and host a static HTML website	**2**	**Apply**
Task – 7		
What is Docker? Why is Docker Popular? Key Components of Docker, what is a Dockerfile? What is Docker Architecture and How Docker Works? What is Docker Image? What is Docker Container? What is Docker Hub? Docker Commands **Task for Execution:** 1. Install Docker and Verify Installation 2. Execute the different docker commands like: run, pull, ps, stop, start, rm, RMI, images, exec, ports, push, build, restart, inspection 3. Create a Docker container to execute a Python program to find the sum of N numbers. 4. Create a simple calculator application in Python and execute it inside Docker container. **Self-Study:** Create two containers (e.g., a web server and a database) and connect them using Docker's bridge network.	**2**	**Apply**
Task – 8		
1. Build a Docker Image from a Simple Application 2. Run Multiple Containers Using Docker Compose 3. Create two Docker containers and establish communication between them using a bridge network. **Self-Study:** Build a CI/CD Pipeline with Docker and Jenkins	**2**	**Apply**
Task – 9		

Introduction to Kubernetes, what is Kubernetes? Key Terminologies, Benefits of using Kubernetes, Deploying and Managing Containerized Applications with Kubernetes, Use Cases of Kubernetes in Real-World Scenarios, Features of Kubernetes, Kubernetes v/s Docker, Architecture of Kubernetes, Key Components of Kubernetes, Application of Kubernetes **Task for Execution:** 1. Install and Configure Kubernetes on Ubuntu 2. Deploy a NGINX application in Kubernetes cluster and access it through browser. 3. Write a YAML file to deploy an Ubuntu pod in Kubernetes. **Self-Study:** Research what **Helm** is? Install Helm and deploy a chart (e.g., NGINX or WordPress).	**2**	**Apply**
Task – 10		
Introduction to Azure- Overview of Microsoft Azure , How Microsoft Azure work? Microsoft Azure architecture, Features of Azure, What is Microsoft Azure Used For? What are the various Azure Services and How does Azure Work? Azure for Disaster Recovery and Backup, Azure Competition How Azure can help in Business? What is Azure Cloud Shell? How to Access Azure Shell? How Azure Security Works? Difference between AWS, Google Cloud, and Azure. **Task for Execution:** 1. Setting up an Azure account 2. Creating an azure account and implement data load 3. Implementation of database operations like insert, delete, update using azure data factory. **Self-Study:** Create and deploy a simple **Web App** using App Services (e.g., Node.js or Python).	**2**	**Apply**
Task – 11		
What is Edge AI? Key Components of Edge AI, Benefits of Edge AI, Use Cases and Applications of Edge AI, Challenges and Limitations of Edge AI, Future Trends of Edge AI, Comparison of Edge AI & CloudAI. **Self-Study:** Research popular edge AI hardware: Raspberry Pi, NVIDIA Jetson Nano, Google Coral, Intel Neural Compute Stick, ESP32.	**2**	**Apply**
Mini Project	**4**	**Apply**

Course outcomes:

The students will be able to

1. Understand the fundamental concepts, architecture, deployment models, and service models of cloud computing along with virtualization technologies and cloud platforms.
2. Apply cloud platform services and virtualization tools to create, configure, and manage virtual machines, cloud resources, and web applications in AWS, Azure, and Google App Engine environments.
3. Apply version control techniques and collaborative development practices using Git and GitHub repositories.
4. Analyze and simulate cloud computing environments using CloudSim and implement scheduling algorithms for performance evaluation.
5. Develop and deploy containerized and orchestrated applications using Docker, Docker Compose, Kubernetes, and Edge AI technologies for modern cloud-native applications.

Reference Books

1. Cloud Computing: Theory and Practice, Dan C Marinescu Elsevier (MK), 2013.
2. Cloud Computing: Concepts, Technology & Architecture Thomas Erl, Ricardo Puttini, Zaigham Mahmood Latest (2023) Pearson Education
3. Computing Principles and Paradigms, RajkumarBuyya , James Broberg, Andrzej Goscinsk,i Willey, 2014.
4. Cloud Computing Implementation, Management and Security John W Rittinghouse, James F Ransome, CRC Press, 2013.

Web Links

1. https://www.w3schools.com/git/git_intro.asp?remote=github
2. https://www.geeksforgeeks.org/what-is-cloudsim/
3. https://www.geeksforgeeks.org/what-is-google-app-engine-gae/
4. https://www.tutorialspoint.com/apex/apex_overview.htm