**DISCRETE MATHEMATICAL STRUCTURES - I**

**COURSE OUTCOMES**

At the end of the course, the student will develop ability to

1. Apply the mathematical logic, predicate rules to design an abstract system for theorem proof.
2. Apply mathematical foundations, algorithmic principles in modelling and design in computer based system.
3. Understand sets, relations, functions, connectives, truth tables, and discrete structures..
4. Apply the concepts of graph theory in solving practical engineering problems.
5. Develop the ability to solve problems involving recurrence relations and generating functions
6. Visualize and simplify situations using graphs and trees as tools.

**UNIT I**

**Mathematical Logic**

Statements and notations, Connectives, Well formed formulas, Truth Tables, tautology, equivalence implication, Normal forms.

**Predicates**

Predicative logic, Free and Bound variables, Rules of inference, Consistency, proof of contradiction.

**UNIT II**

**Set Theory**

Introduction, Sets and Elements, Subsets, Venn Diagrams, Set Operations, Power Sets, Partitions

**Relations**

Introduction, Product Sets, Relations, Pictorial Representatives of Relations, Composition of Relations, Types of Relations, Closure Properties, Equivalence Relations, compatibility and Partial Ordering Relations

**UNIT III**

**Ordered Sets**

Ordered Sets, Hasse Diagrams of Partially Ordered Sets, Supremum and Infimum, Isomorphic (Similar) Ordered Sets, Well-Ordered Sets, Lattices and its Properties

**Functions:** Introduction, Functions, One-to-One, Onto and Bijective Functions, Invertible Functions, Recursive Functions.

**UNIT IV**

**Techniques of Counting**

Introduction, Basic Counting Principles, Permutations, Combinations, The Pigeonhole Principle and its applications, The Inclusion–Exclusion Principle, Combinations with Repetitions, Binomial and Multinomial Theorems

**UNIT V**

**Recurrence Relation**

Generating Functions, Function of Sequences Calculating Coefficient of generating function, Recurrence relations, Solving recurrence relation by substitution and Generating functions.

**UNIT VI**

**Graph Theory**

Representation of Graph, Basic Concepts, Basic types of Graphs and their properties, types of paths, Isomorphism and Sub graphs, Multi graphs,  Euler circuits, Hamiltonian graphs, Chromatic Numbers, DFS, BFS, Trees, Spanning Trees, Planar Graph, Prim’s and Kruskal’s Shortest Path

**TEXT BOOKS**

1. Seymour Lipschutz, Lipson Marc, “Discrete Mathematics”, Tata Mcgraw Hill, ISBN-100070669120
2. Trembly J.P. and Manohar .P, “Discrete Mathematical Structures with Applications to computer Science”, TMH,ISBN-10: 0074631136

**REFERENCE BOOKS**

1. Ralph. P.Grimaldi “Discrete and Combinational Mathematics- An Applied Introduction”, 5th Edition Pearson Education,ISBN:9780201726343
2. BernandKolman, Roberty C. Busby, Sharn Cutter Ross, “Discrete Mathematical Structures”, Pearson Education / PHI.
3. J.L. Mott, A. Kandel, T.P. “Discrete Mathematics for Computer Scientists and Mathematicians”, Baker Prentice Hall.

**WEB LINKS**

1. <http://nptel.iitm.ac.in>
2. http://www.math.northwestern.edu/~mlerma/courses/cs310-05s/
3. http://highered.mheducation.com/sites/0073383090/student\_view0/applications\_of\_discrete\_mathematics.html
4. <http://www.mhhe.com/math/advmath/rosen/r5/student/ch01/weblinks.html>

**MICROPROCESSORS AND MICROCONTROLLERS**

**COURSE OUTCOMES**

Students will be able to

1. Outline the history of computing devices (remember)

2. Describe the architecture of 8086 microprocessor (understand)

3. Develop assembly level programs for microprocessor and microcontroller (apply)

4. Compare between microprocessors and microcontrollers (analyze)

5. Design and implement microcontroller-based embedded system (create)

**UNIT I**

**Introduction:** Evolution of Microprocessors, 8085 MPU architecture.

**8086 Family Architecture:** Organization of 8086 CPU, Concept of Memory Segmentation, Physical and logical addressing, Addressing Modes, Instruction set: Data transfer, arithmetic, logical, string and control transfer instructions.

**UNIT II**

**Assembly Language Programming**

Assemble directives, simple Assembly language Programming of 8086 ondata transfer, arithmetic, logical, string and branching. Procedures, macros, time delays, Assembly Language Development tools(Linker ,Loader Debugger etc)

**UNIT III**

**Interfacing With 8086**

8255 PPI, interfacing, interfacing of switches, LEDs, ADC, DAC and Stepper motor. Interrupt structure of 8086, 8259 PIC, need for DMA, 8257 DMA Controller.

**UNIT IV**

**8051 Microcontroller**

8051 Architecture, pin diagram, addressing modes, instruction set: data transfer, arithmetic, logical, control transfer instructions. Assembly language Programming.

**UNIT V**

**8051 Microcontroller**

Timers and counter, Programing Timers and counters I/O ports, Serial port, Interrupts and Interrupts Programing in Assembly Language

**UNIT VI**

**Interfacing**: LEDs, switches, LCD, 7 Segment display and keyboard

**TEXT BOOKS**

1. D.V. Hall, “Microprocessors and Interfacing”, TMGH, 2nd Edition, 2006,

2. Muhammed Ali Mazidi, “The 8051 Microcontrollers and Embedded Systems”, Pearson,

New Delhi.

**REFERENCE BOOKS**

* + - 1. A.K. Ray and K.M. Bhurchandani, “Advanced Microprocessors and Peripherals”, TMH, 2nd Edition, 2006.

1. Kenneth J Ayala, “The 8051 Microcontroller”, Cengage Learning, 3rd Edition.
2. Brey, “Advanced Microprocessors”, Prentice Hall of India, New Delhi.

**SMART SYSTEM DESIGN**

**COURSE OUTCOMES**

Students will be able to

Understand, illustrate and apply the system design process to address a need

Create a smart system design using system design principles.

Differentiate between different microcontrollers

1. Select appropriate sensors and actuators based on the design requirements
2. Estimate the power requirements for circuits
3. Implement a complete smart system using Arduino Microcontroller

**UNIT I**

**Introduction to Systems Thinking**

Definition of System, Design, User, Needs, Tasks and Environment. Relation between User, needs, tasks and environment. Need statement.

**UNIT II**

**System Design**

Introduction to Smart system design, Key elements of Smart system design, Architectural design – System structure and behavior,

Logical design-Abstract representation of data flow, inputs and outputs,

Physical design-Verification of input, output and process requirements.

**UNIT III**

**Arduino Microcontroller**

Introduction to Arduino controller, Block diagram, pin map, Arduino programming. Signal Processing and Conditioning:Rectifiers, Filters, Regulators, Amplifying signals using OP Amps.

**UNIT IV**

**Sensors**

Characteristics of Sensors – Static and Dynamic, Classification – Analog Sensors (Force, displacement, temperature, LDR), Digital Sensors (Photo sensors, proximity sensor),

**UNIT V**

**Mechanical Drives**

Gears, Belt and Chain Drives, Bearings. Selection of Mechanical

**Electrical Actuation systems**

Relays, Solenoids, Solid State Switches – Diodes, Transistors, Thyristors and Triacs, fundamentals of DC and AC Motors, Stepper motor. Speed, position and direction control of motors.

**UNIT VI**

Project Testing and validation –Defining the test protocol. Product validation. Product delivery. Product Documentation

**TEXT BOOKS**

1. Clarence de Silva, “Sensors and Actuators. CRC Press. 2016.
2. W. Bolton, “ Mechatronics: Electronic Control Systems in Mechanical and Electrical Engineering. Pearson Education Asia.

**REFERENCE BOOKS**

1. D. Patranabi. Sensors and Transducers. PHI Learning. 2003.
2. Alciatore and Histand. Introduction to Mechatronics and Measurements. Tata McGraw Hill. 2012.

**DATA STRUCTURES**

**COURSE OUTCOMES**

At the end of the course, the students will develop ability to

1. Explain how arrays, records, linked structures, stacks, queues, trees, and graphs are represented in memory and used by algorithms.
2. Compare and contrast the benefits of dynamic and static data structures implementations.
3. Develop and Evaluate programs that use arrays, records, linked structures, stacks, queues, trees and graphs.
4. Demonstrate different methods for traversing trees.
5. Design and implement an appropriate hashing function for an application.
6. Discuss the computational efficiency of the principal algorithms for sorting, searching and hashing.

**UNIT I**

**Basic Concepts of Data Structures:** Data objects and Structures, Algorithm Specification-Introduction, Recursive algorithms, Data Abstraction, Performance analysis- time complexity and space complexity, Asymptotic Notation-Big O, Omega and Theta notations, Complexity Analysis Examples, Introduction to Linear and Non-Linear data structures.

Representation of single, two dimensional arrays, sparse matrices-array and linked representations.

**UNIT II**

Linear list ADT-array representation and linked representation, Singly Linked Lists-Operations-Insertion, Deletion, Circularly linked lists-Operations for Circularly linked lists, Doubly Linked Lists- Operations- Insertion, Deletion.

Stack ADT, definition, array and linked implementations, applications-infix to postfix conversion, Postfix expression evaluation, recursion implementation, Queue ADT, definition, array and linked Implementations, Circular queues-Insertion and deletion operations.

**UNIT III**

Trees – definition, terminology, Binary trees-definition, Properties of Binary Trees, Binary Tree ADT, representation of Binary Trees-array and linked representations, Binary Tree traversals, Threaded binary trees, Priority Queues –Definition and applications, Max Priority Queue ADT-implementation-Max Heap-Definition, Insertion into a Max Heap, Deletion from a Max Heap.

**UNIT IV**

Searching - Linear Search, Binary Search, Hashing-Introduction, hash tables, hash functions, Overflow Handling, Comparison of Searching methods.

Sorting-Insertion Sort, Selection Sort, Radix Sort, Quick sort, Heap Sort, Merge sort, Comparison of Sorting methods.

**UNIT V**

Graphs–Definitions, Terminology, Applications and more definitions, Properties, Graph ADT, Graph Representations- Adjacency matrix, Adjacency lists, Graph Search methods - DFS and BFS, Complexity analysis,

**UNIT VI**

Search Trees-Binary Search Tree ADT, Definition, Operations- Searching, Insertion and Deletion, Balanced search trees-AVL Trees-Definition and Examples only, B-Trees- Definition and Examples only, Red-Black Trees-Definitions and Examples only, Comparison of Search Trees.

**TEXT BOOKS**

1. Data structures, Algorithms and Applications in C++, 2nd Edition, Sartaj Sahni, Universities Press.
2. Data structures and Algorithms in C++, Adam Drozdek, 4th edition, Cengage learning.

**REFERENCE BOOKS**

1. Data structures with C++, J. Hubbard, Schaum’s outlines, TMH.
2. Data structures and Algorithms in C++, M.T. Goodrich, R. Tamassia and D. Mount, Wiley India.
3. Data structures and Algorithm Analysis in C++, 3rd edition, M. A. Weiss, Pearson.
4. Classic Data Structures, D. Samanta, 2nd edition, PHI.

**DIGITAL LOGIC DESIGN**

**COURSE OUTCOMES**

At the end of the course, the students will develop ability to

1. Discuss fundamental concepts in the design of digital circuits and systems.
2. Discuss and have a working knowledge of Boolean algebra and its application to combinational logic circuits.
3. Manipulate and design basic combinational operators (and, or, not, etc) and sequential circuits.
4. Manipulate and design combination of operators to form higher level functions (multiplexer, counter) and memory element (flip-flop).
5. Explain the basic components of the Von Neumann computer architecture.
6. Prepare and make professional presentations relevant to the course material.

**UNIT I**

**Digital Systems and Binary Numbers**

Digital Systems, Binary Numbers, Binary Numbers, Octal and Hexadecimal Numbers, Complements of Numbers, Complements of Numbers, Signed Binary Numbers, Arithmetic addition and subtraction

**UNIT II**

**Concept of Boolean algebra**

Basic Theorems and Properties of Boolean algebra, Boolean Functions, Canonical and Standard Forms, Minterms and Maxterms,

**UNIT III**

**Gate level Minimization**

Map Method, Two-Variable K-Map, Three-Variable K-Map, Four Variable K-Maps. Products of Sum Simplification, Sum of Products Simplification, Dont Care Conditions, NAND and NOR Implementation, Exclusive? OR Function

**UNIT IV**

**Combinational Logic**

Introduction, Analysis Procedure, Design Procedure, Binary Adder Subtractor, Decimal Adder, Binary Multiplier, Decoders, Encoders, Multiplexers, HDL Models of Combinational Circuits

**UNIT V**

**Synchronous Sequential Logic**

Introduction to Sequential Circuits, Storage Elements: Latches, Storage Elements: Flip?Flops, Analysis of Clocked Sequential Circuits, Mealy and Moore Models of Finite State Machines

**UNIT VI**

**Registers and Counters**

Registers, Shift Registers, Ripple Counters, Synchronous Counters, Ring Counter, Johnson Counter, Ripple Counter

**TEXT BOOKS**

1. Digital Design, 5/e, M.Morris Mano, Michael D Ciletti, PEA.
2. Fundamentals of Logic Design, 5/e, Roth, Cengage.

**REFERENCE BOOKS**

1. Digital Logic and Computer Design, M.Morris Mano, PEA.
2. Digital Logic Design, Leach, Malvino, Saha, TMH.
3. Modern Digital Electronics, R.P. Jain, TMH.

**MICROPROCESSORS AND MICROCONTROLLERS LAB**

**COURSE OUTCOMES**

1. Write assembly level programs on arithmetic operations using various addressing modes.
2. Apply the concepts of assembly level programming on sorting and code conversions.
3. Design interfacing of various I/O devices to microprocessor.
4. Apply the concept of serial communication for transmission of serial data.
5. Verify the ports, timers, and interrupts operation in 8051 microcontrollers.
6. Design and implement microcontroller-based embedded system.

**LIST OF EXPERIMENTS**

**I. Microprocessor 8086**

1. Introduction to Assembler.
2. Arithmetic operation – Multi byte Addition and Subtraction, Multiplication and Division – Signed and unsigned Arithmetic operation, ASCII – arithmetic operation.
3. Logic operations – Shift and rotate – Converting packed BCD to unpacked BCD, BCD to ASCII conversion.
4. By using string operation and Instruction prefix: Move Block, Reverse string, Sorting, Inserting, Deleting, Length of the string, String comparison.
5. DOS/BIOS programming: Reading keyboard (Buffered with and without echo) – Display characters, Strings.

**II. Interfacing**

1. ADC/DAC 2. Stepper Motor 3. Traffic Light 4. Keyboard

**III. Microcontroller 8051**

1. Programming on arithmetic operations

2. Reading and writing on a parallel port.

3. Timer in different modes

4. Serial communication implementation.

5. Interfacing: switches, LEDs, LCD.

**DATA STRUCTURES LAB**

**COURSE OUTCOMES**

At the end of the course, the students will develop ability to

1. Describe how arrays, records, linked structures, stacks, queues, trees, and graphs are represented in memory and used by algorithms.
2. Compare and contrast the benefits of dynamic and static data structures implementations.
3. Develop and Evaluate programs that use arrays, records, linked structures, stacks, queues, trees and graphs.
4. Demonstrate different methods for traversing trees.
5. Design and implement an appropriate hashing function for an application.
6. Discuss the computational efficiency of the principal algorithms for sorting, searching and hashing.

Write a C++ programs to implement recursive and non recursive i) Linear search ii) Binary search

Write a C++ programs to implement

1. Bubble sort
2. Selection sort
3. Quick sort
4. insertion sort

Write a C++ programs to implement the following using an array.

* 1. Stack ADT b) Queue ADT

Write a C++ programs to implement list ADT to perform following operations

a) Insert an element into a list.

b) Delete an element from list

c) Search for a key element in list

d) Count number of nodes in list

Write C++ programs to implement the following using a singly linked list.

Stack ADT b) Queue ADT

Write C++ programs to implement the deque (double ended queue)

ADT using a doubly linked list and an array.

Write a C++ program to perform the following operations:

a) Insert an element into a binary search tree.

b) Delete an element from a binary search tree.

c) Search for a key element in a binary search tree.

Write C++ programs for implementing the following sorting methods:

Merge sort

Heap sort

Write C++ programs that use recursive functions to traverse the given binary tree in

a)Preorder

b) inorder and

c) postorder.

Write a C++ program to perform the following operations

a)Insertion into a B-tree

b) Deletion from a B-tree

Write a C++ program to perform the following operations

a)Insertion into an AVL-tree

b) Deletion from an AVL-tree

Write a C++ program to implement all the functions of a dictionary (ADT)