GUJARAT TECHNOLOGICAL UNIVERSITY

Program Name: Bachelor of Engineering
Level: UG
Subject Code: BE04000261
Subject Name: Discrete Mathematics and Graph Theory

w. e. f. Academic Year:	2024-25
Semester:	4
Category of the Course:	Basic Science Course

Prerequisite:	Basic Algebra
Rationale:	This course introduces the basic concepts of discrete mathematics in the field of computer science. It covers sets, logic, functions, relations, graph theory and algebraic structures. These basic concepts of sets, logic, functions, relations and graph theory are applied to Boolean Algebra and logic networks, while the advanced concepts of functions and algebraic structures are applied to finite state machines and coding theory.

Course Outcomes:

Sr. No.	CO statement	Marks % Weightage
CO-1	Write an argument using logical notation and determine if the argument is or is not valid. To simplify and evaluate basic logic statements including compound statements, implications, inverses, converses, and contra positives using truth tables and the properties of logic. To express a logic sentence in terms of predicates, quantifiers, and logical connectives.	18%
CO-2	Understand the basic principles of sets and operations in sets and apply counting principles to determine probabilities. Be familiar with recurrence relations.	11%
CO-3	Understand the basic concept of functions like domain and range of a function, identify one-to- one function, perform the composition of functions and apply the properties of functions to application problems. Use the properties of algebraic structures and be familiar with algebraic systems, groups, subgroups, Rings and Fields.	22%
CO-4	Apply relations and to determine their properties. Understand partial orders with Hasse diagrams, and the fundamental properties and types of lattices to analyze mathematical structures.	22%
CO-5	Interpret different traversal methods for trees and graphs. Model problems in Computer Science using graphs and trees.	27%

Teaching and Examination Scheme:

Teaching / Learning Scheme (in Hours per semester)					Total Credits	Assessment Pattern and Marks					Total Marks
						Theory		Tutorial / Practical			
L	T	P	PBL	Total no of hours per semester		ESE (E)	PA / CA (M)	PA/ CA (I)	PBL (I)	ESE (V)	
45	30	00	15	90	3	70	30	0	30	0	130

Problem Based Learning (PBL) aims to accommodate learning beyond syllabus as per clause 9.4 of NBA manual.

Course Content:

Sr. No.	Content	Total Hrs	% of Weightage
1	**Propositional Logic**: Definition, statements & notation, truth values, connectives, statement formulas & truth tables, well-formed formulas, tautologies, equivalence of formulas, duality law, tautological implications, examples. **Predicate Logic:** Definition of predicates, statement functions, variables, quantifiers, predicate formulas, free & bound variables, the universe of discourse, examples, valid formulas & equivalences, examples	08	18
2	**Set Theory:** Basic concepts of set theory like inclusion, complement, intersection, union, Cartesian product, power set. **Counting and Combinatorics:** Basic counting principles, permutations and combinations, binomial coefficients and the binomial theorem, Pigeonhole principle, inclusion-exclusion principle. **Recurrence relations:** Definitions and simple examples only.	05	11
3	**Functions:** Basic concepts of functions like domain, range, surjective, injective, bijective, composition of functions, inverse of functions. **Algebraic Structures:** Algebraic structures with one binary operation- semigroup, monoid, group, subgroup, normal subgroup, coset, homomorphic subgroups, Lagrange's theorem, congruence relation and quotient structures. Algebraic structures with two binary operation- Ring, integral domain and field (definitions and simple examples only).	10	22
4	**Relations**: Definition, binary relation, representation, domain, range, universal relation, void relation, union, intersection, and complement operations on relations, properties of binary relations in a set: reflexive, symmetric, transitive, anti-symmetric relations, partition and covering of a set, equivalence relation, equivalence classes, compatibility relation, maximum compatibility block, composite relation, converse of a relation, transitive closure of a relation R in set X **Partial Ordering**: Definition, examples, simple or linear ordering, totally ordered set (chain), frequently used partially ordered relations, representation of partially ordered sets, Hesse diagrams, least & greatest elements, minimal & maximal elements, least upper bound (Supremum), greatest lower bound (Infimum), Well-ordered partially ordered sets (Posets). Lattice as Posets, complete, distributive, modular and complemented lattices, Boolean and pseudo Boolean lattices (definitions and simple examples only).	10	22
5	**Graphs:** Introduction, definition, examples; Nodes, edges, adjacent nodes, directed and undirected edge, directed graph, undirected graph, examples; Initiating and terminating nodes, loop (sling), distinct edges, parallel edges, multi-graph, simple graph, weighted graphs, examples; Isolated nodes, null graph, isomorphic graphs, examples; Degree, In-degree, out-degree, total degree of a node, examples; Subgraphs: definition, examples; Converse (reversal or directional dual) of a digraph, examples; Path: Definition, paths of a given graph, length of path, examples; Simple path (edge simple), elementary path (node simple), examples; Cycle (circuit), elementary cycle, examples; **Reachability:** Definition, geodesic, distance, examples; Properties of reachability, the triangle inequality; Reachable set of a given node, examples;	12	27

	Node base, examples **Connectedness:** Definition, weakly connected, strongly connected, unilaterally connected, examples; Strong, weak, and unilateral components of a graph, examples; Applications to represent resource allocation status of an operating system, and detection and correction of deadlocks; Matrix representation of graph: Definition, adjacency matrix, Boolean (or bit) matrix, examples; Determine number of paths of length n through adjacency matrix, examples; Path (Reachability) matrix of a graph, examples; Warshall's algorithm to produce path matrix, Flowchart. **Trees:** Definition, branch nodes, leaf (terminal) nodes, root, examples; Different representations of a tree, examples; Binary tree, m-ary tree, full (or complete) binary tree, examples; Converting any m-ary tree to a binary tree, examples; Representation of a binary tree: Linked-list; Tree traversal: Pre-order, in-order, post-order traversal, examples; Algorithms; Applications of List structures and graphs		

Suggested Specification table with Marks (Theory): (For B.E. only)

Distribution of Theory Marks					
R Level	U Level	A Level	N Level	E Level	C Level
30	**40**	**20**	**10**	**00**	**00**

R: Remembrance; U: Understanding; A: Application, N: Analyze and E: Evaluate C: Create and above Levels (Revised Bloom's Taxonomy)

Note: This specification table shall be treated as a general guideline for students and teachers. The actual distribution of marks in the question paper may vary slightly from above table.

Reference Books:

1. J. P. Tremblay and R. Manohar, Discrete Mathematical Structures with Applications to Computer Science, Tata McGraw-Hill,1997.
2. S. Lipschutz and M. L. Lipson, Schaum's Outline of Theory and Problems of Discrete Mathematics, 2nd Ed., Tata McGraw-Hill,1999.
3. K. H. Rosen, Discrete Mathematics and its applications, Tata McGraw-Hill, 6th Ed., 2007.
4. David Liben-Nowell, Discrete Mathematics for Computer Science, Wiley publication, July 2017. 5.
5. Eric Gossett, Discrete Mathematics with Proof, 2nd Edition,Wiley publication, July 2009.
6. Josheph A. Gallian, Contemporary Abstract Algebra, Chapman & Hall, 11th edition, July 2025.

Suggested MOOC Courses: https://onlinecourses.nptel.ac.in/noc26_cs68/preview

List of Tutorials:

Sr. No	Tutorials	CO

1	**Title: Propositional logic: Truth values, truth tables & tautology** **Objective:** To understand and apply the concepts of propositional logic, including Boolean formulas, generating truth tables, and determining a formula is a tautology, contradiction, or contingent. **Procedure:** Learn the foundation of propositional logic: truth values, truth tables and tautology through guided examples in the tutorials	**1**
2	**Title: Predicate logic evaluation over finite domain** **Objective:** To study and evaluate quantified formulas over a finite universe, and to clearly distinguish between free and bound variables. **Procedure:** Learn the Predicate logic evaluation over finite domain through examples, definitions, formula evaluation, and application-oriented exercises during the tutorial.	1
3	**Title: Set operations and verifying identities** **Objective:** To understand fundamental set concepts and perform operations such as union, intersection, difference, symmetric difference, and power set construction. **Procedure:** Learn about the universal set, types of sets, and standard set operations (union, intersection, complement, difference, symmetric difference, power set). Verify identities, including De Morgan's laws, through illustrative examples during the tutorial.	2
4	**Title: Counting: nPr, nCr, Pascal triangle & pigeonhole Principle** **Objective:** To understand and apply core combinatorial concepts including nPr, nCr , Pascal's Triangle (up to n=10), the Pigeonhole Principle, and to explore their computational and practical uses. **Procedure:** Work through examples involving permutations and combinations, construct Pascal's Triangle, apply the Pigeonhole Principle, and prove related identities in tutorial.	2
5	**Title: Recurrence relations: iteration & closed-form (characteristic equation)** **Objective:** To compute the terms for linear homogeneous recurrences with constant coefficients by iteratively and to derive closed form solutions using the characteristic polynomial (simple roots). **Procedure:** Solve the recurrence relation using illustrative examples.	2
6	**Title: Functions: mapping, composition & inverse** **Objective:** To understand and apply the concepts of functions, one-to-one and onto mappings, composition of functions and inverse functions. **Procedure:** Determine whether the given mapping is function or not, check one-to-one and onto function, find composition of functions, inverse of a function with the help of examples.	3
7	**Title: Permutations & group basics: cycle notation, inverse, subgroup generation** **Objective:** To represent permutations, convert them to cycle notation, compute inverses, and generate groups, subgroups through repeated composition. **Procedure:** Work through examples involving groups—such as abelian groups, cyclic groups, permutation groups, and subgroups—to illustrate these concepts and operations.	3
8	**Title: Normal Subgroup, Lagrange Theorem, Rings and Field** **Objective:** To understand normal subgroups, Lagrange's Theorem, and the basic concepts of rings and fields. **Procedure:** Work through examples involving normal subgroups, Lagrange's Theorem (including a demonstrated example), and fundamental operations in rings and fields.	3
9	**Title: Relations: Properties and examples** **Objective:** To understand and apply the concept of relation, their types, and their representations. **Procedure:** Understand relation, various types of relation, definitions, matrix and graphs representation of relations, properties of relations with the help of examples.	4

10	**Title: Transitive closure: Warshall algorithm & reachability matrix** **Objective:** To Implement Warshall algorithm to compute the transitive closure of a relations to understand reachability. **Procedure:** Apply Warshall's algorithm to the adjacency matrix of a given relation in order to obtain its transitive closure, complete different examples.	4
11	**Title: Partial orders & Hasse diagram generation** **Objective:** To understand partial orders and generate Hasse diagrams, identifying least, greatest, minimal, and maximal elements. **Procedure:** Check whether the given relation is a partial order and draw its Hasse diagram. Identify least, greatest, minimal, maximal elements, and chains using different examples.	4
12	**Title: Lattice: Distributive, Modular & complemented (small finite lattices)** **Objective:** To Understand small finite lattice defined as Poset, and if so, whether they are distributive, modular, complemented. **Procedure:** Check whether the given Poset is a lattice or not, and provide examples and Counter examples. Visualize each lattice using a Hasse diagram, and compute meet and join for different type of examples.	4
13	**Title: Graph basics: Adjacency matrix, degree, connected components** **Objective:** To understand graphs and their properties, different types of graphs, its elements, produce adjacency/incidence matrix, compute degree sequences, and find connected components (for undirected graphs) or strongly connected components (for directed graphs) **Procedure:** Introduce different types of graphs and their elements with examples, list its properties, compute common graph parameters, and demonstrate matrix-based representations and their properties through examples.	5
14	**Title: Graph algorithms: BFS/DFS, shortest path (unweighted), cycle detection** **Objective:** To understand and implement BFS and DFS traversals, reconstruct shortest paths in unweighted graphs, and detect cycles in both directed and undirected graphs. **Procedure: BFS:** Implement BFS using a queue to compute distances and parent nodes; reconstruct shortest paths from source to target with various graph examples. **DFS:** Implement DFS and use it to detect cycles, employing visited/recursion-stack techniques for directed graphs, illustrate with examples.	5
15	**Title: Weighted graphs: Dijkstra (nonnegative weights) & MST (Kruskal) + tree traversals** **Objective:** To implement Dijkstra's algorithm for shortest paths with non-negative weights, Kruskal algorithm using DSU for MST, and practice binary tree representations with pre/in/post order traversals. **Procedure:** 1. **Dijkstra:** Implement Dijkstra's algorithm for various weighted graphs and solve illustrative examples. 2. **Kruskal:** Implement Kruskal's algorithm with DSU for MST on different graphs and work through examples. 3. **Binary Trees:** Practice binary tree representations and perform pre/in/post-order traversals with examples.	5

List of suggested activities for Problem Based Learning:

Sr. No	Name of Activity	No. of Hours	Evaluation Criteria
1	Assignments	5 assignments × 1h each; Total = 5h	Based on completeness, correctness and submission of assignments
2	Technical Video-Based Learning – Watch online lectures/tutorials on topics covered	Duration = 4h, Report & Presentation = 1h; Total = 5h	Report / Presentation on key learning outcomes
3	Preparing Posters, Charts, Power Point Presentation or documents (using equation editor, Mathtype, Latex etc.) of all important definitions, theorems, formulae etc on topics of syllabus	Preparing, Designing and Presenting Visual Content = 5h	Assessment based on creativity, clarity, and presentation skills
4	Seminar / Presentation – Study and present a technical topic beyond syllabus	Study/Prep = 3h, Report = 1h, Presentation = 1h; Total = 5h	Based on technical depth, quality of report and presentation skills
5	Real-World Case Study problems based on syllabus	Study = 4h, Report prep = 1h; Total = 5h	Based on correctness, completeness and analysis of the report
6	Implementing the concepts of syllabus in programs	Completing at least five programs = 5h	Review based on the implementation, results, and its output
7	Modeling and Simulation on use of concept in Engineering Systems (developing models and performing simulations)	Model formulation (4h) + Simulation and result analysis (1h) = 5h	Evaluation based on model accuracy, computational efficiency, and interpretation of results

Note:

- The activities listed above are suggestive, and faculty members have the flexibility to select and modify them as needed.
- The total self-learning hours remain fixed at 15 hours, ensuring comprehensive coverage of topics of Discrete Mathematics.
- Faculty can adjust the distribution of hours across different activities while maintaining a balanced learning approach.
- All records pertaining to the evaluation and assessment of self-learning activities must be properly maintained and preserved at the institute level. These records should be made available to the university upon request.
- Institutes are encouraged to utilize digital platforms, such as Microsoft Teams, for effective record keeping and to ensure transparency in the evaluation and assessment of self-learning activities.

****