DSE-01 (b): Discrete Structures

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Discrete Structures	4	3	0	1	Class XII with Mathematics	NIL

***Learning objectives*:**

1. To teach students how to think logically and mathematically.
2. To stress on mathematical reasoning and describe different ways in which mathematical problems could be solved.
3. To cover four thematic areas: mathematical reasoning, combinatorial analysis, discrete structures, and mathematical modelling.
4. To touch upon topics like logic, proofs, set theory, counting, probability theory (the discrete part of the subject), graph theory, trees, Boolean algebra, and modelling computation.

Learning Outcomes:

1. *Relate mathematical concepts and terminology to examples in the domain of Computer Science.*
2. *Model real world problems using various mathematical constructs.*
3. *Use different proofing techniques; construct simple mathematical proofs using logical arguments.*
4. *Divide a problem or a proof into smaller cases.*
5. *Formulate mathematical claims and construct counterexamples.*

UNIT-I **(7 hours)**

Sets, Functions, Sequences and Summations, Relations. Sets: Set Operations, Computer Representation of Sets, Countable and Uncountable Set, Principle of Inclusion and Exclusion, Multi-sets; Functions: One-to-one and Onto Functions, Inverse Functions and Compositions of Functions, Graphs of Functions Sequences and Summations: Sequences, Special Integer Sequences, Summations; Relations: Properties of Binary Relations, Equivalence relations and Partitions, Partial Ordering Relations and Lattices.

UNIT-II **(8 hours)**

Logic and Proofs. Propositional Logic, Propositional Equivalences, Use of first-order logic to express natural language predicates, Quantifiers, Nested Quantifiers, Rules of Inference, Introduction to Proofs, Proof Methods and Strategies, Mathematical Induction.

UNIT-III **(7 hours)**

Number Theory. Division and Integers, Primes and Greatest Common Divisors, Representation of Integers, Algorithms for Integer Operations, Modular Exponentiation, Applications of Number

Theory.

UNIT-IV **(8 hours)**

Combinatorics/Counting. The Pigeonhole Principle, Permutations and Combinations, Binomial Coefficients, Generalized Permutations and Combinations, Generating Permutations and Combinations.

UNIT-V **(10 hours)**

Graphs and Trees. Graphs: Basic Terminology, Multigraphs and Weighted Graphs, Paths and Circuits, Eulerian Paths and Circuits, Hamiltonian paths and Circuits, Shortest Paths, Spanning Trees, Graph Isomorphism, Planar Graphs; Trees: Trees, Rooted Trees, Path Lengths in Rooted Trees.

UNIT-VI **(5 hours)**

Recurrence. Recurrence Relations, Generating Functions, Linear Recurrence Relations with Constant Coefficients and their solution.

References

1. *C.L. Liu & Mahopatra, Elements of Discrete mathematics. 3rd edition. Tata McGraw Hill. 2008.*
2. *Kenneth R., Discrete Mathematics and Its Applications. 6th edition. Mc Graw Hill. 2006.*

List of practicals (30 Hours)

1. Write a Program to create a SET A and determine the cardinality of SET for an input array of elements (repetition allowed) and perform the following operations on the SET:
a) ismember (a, A): check whether an element belongs to set or not and return value as true/false.
b) powerset(A): list all the elements of power set of A.

2. Create a class SET and take two sets as input from user to perform following SET Operations:
a) Subset: Check whether one set is a subset of other or not.
b) Union and Intersection of two Sets.
c) Complement: Assume Universal Set as per the input elements from the user.
d) Set Difference and Symmetric Difference between two SETS
e) Cartesian Product of Sets.

3. Create a class RELATION, use Matrix notation to represent a relation. Include functions to check if the relation is Reflexive, Symmetric, Anti-symmetric and Transitive. Write a Program to use this class.

4. Use the functions defined in Ques 3 to check whether the given relation is:
a) Equivalent, or
b) Partial Order relation, or
c) None

5. Write a Program to implement Bubble Sort. Find the number of comparisons during each pass and display the intermediate result. Use the observed values to plot a graph to analyse the complexity of algorithm.

6. Write a Program to implement Insertion Sort. Find the number of comparisons during each pass and display the intermediate result. Use the observed values to plot a graph to analyse the complexity of algorithm.

7. Write a Program that generates all the permutations of a given set of digits, with or without repetition. (For example, if the given set is {1,2}, the permutations are 12 and 21). (One method is given in Liu)

8. Write a Program to accept the truth values of variables x and y, and print the truth table of the following logical operations:
a) Conjunction f) Exclusive NOR
b) Disjunction g) Negation
c) Exclusive OR h) NAND
d) Conditional i) NOR
e) Bi-conditional

9. Write a Program to store a function (polynomial/exponential), and then evaluate the polynomial. (For example store f(x) = 4n3 + 2n + 9 in an array and for a given value of n, say n = 5, evaluate (i.e. compute the value of f(5)).

10. Write a Program to represent Graphs using the Adjacency Matrices and check if it is a complete graph.

11. Write a Program to accept a directed graph G and compute the in-degree and out-degree of each vertex.