



CLUSTER UNIVERSITY SRINAGAR
SYLLABUS (FYUP UNDER NEP 2020)
Offered by the Department of Information Technology,
Batch 2022, 2023, 2024

Offered by Department of Information Technology Batch 2022 and onwards
3rd Semester - Minor- Discrete Structures

Course Code: UGICT22N301
Credits: 6 (Theory: 4, Practical: 2)
Contact Hrs: 120 (Theory: 60, Practical: 60)

Max. Marks: 100
Theory External: 80; **Min Marks:** 32
Theory Internal (Continuous Assessment):
20 Marks, Min Marks: 08
Practical Experimental Basis = 30,
Min. Marks: 12
Practical Experimental (Continuous Assessment)
20 Marks, Min. Marks: 08

Course Objectives

The course aims to develop a strong foundation in mathematical reasoning and problem-solving using discrete structures. It enables students to understand logic, number theory, graphs, relations, and functions essential for computer science applications. Students learn to apply combinatorial and graph-theoretic concepts to model, analyze, and solve computational problems effectively.

Learning Outcomes:

After completing all four units, students will be able to:

1. Apply propositional and predicate logic to analyze and construct valid mathematical arguments.
2. Use number theory concepts and Boolean algebra to solve problems in modular arithmetic and digital logic design.
3. Represent and analyze graphs and trees, identify their properties, and apply algorithms for traversal and optimization.
4. Understand and apply relations, functions, and combinatorial techniques to solve discrete mathematical and computational problems.

Unit 1:

Propositional Logic: Applications, Propositional Equivalences, Predicates and Quantifiers, Nested Quantifiers, Rules of Inference. Proof Strategies: Introduction to Proofs, Proof Methods and Strategy.



CLUSTER UNIVERSITY SRINAGAR

SYLLABUS (FYUP UNDER NEP 2020)

Offered by the Department of Information Technology,
Batch 2022, 2023, 2024

Unit 2:

Number Theory Basics: Divisibility and Modular Arithmetic, Primes and Greatest Common Divisor (GCD), Least Common Multiple (LCM).

Boolean Functions, representing Boolean Functions, Logic Gates, Minimization of Circuits.

Unit 3:

Graphs and Graph Models, Terminology & Special types of Graphs, Representation of Graphs, Graph Isomorphism, Connectivity, Euler and Hamiltonian Paths.

Introduction to Trees, Tree Traversal, Spanning Trees, Minimum Spanning Trees, Planar Graphs, Graph Colouring

Unit 4:

Relations and Functions: Properties of Relations, Representations of Relations, Closures of Relations, Equivalence and Partial Order Relations. Posets, Hasse Diagrams, Lattices.

Functions: Types of Functions, Composition of Functions, Inverse Functions.

Combinatorics: Basic Counting Principles, Permutations and Combinations, Pigeonhole Principle, Inclusion–Exclusion Principle.

Recommended Textbooks:

1. Rosen, K. H. (2023). *Discrete Mathematics and its Applications* (9th ed., ISBN: 978-1260424718). New York, NY: McGraw-Hill.
2. Tremblay, J. P., & Manohar, R. (2020). *Discrete Mathematical Structures with Applications to Computer Science*. McGraw-Hill Education.
3. Liu, C. L. (2022). *Elements of Discrete Mathematics: A Computer-Oriented Approach* (4th ed.). McGraw-Hill.
4. Kolman, B., Busby, R. C., & Ross, S. (2019). *Discrete Mathematical Structures* (7th ed.). Pearson Education.
5. Yashwant Kanetkar. *Data Structures in C*. BPB Publications.



CLUSTER UNIVERSITY SRINAGAR
SYLLABUS (FYUP UNDER NEP 2020)
Offered by the Department of Information Technology,
Batch 2022, 2023, 2024

Lab Work (2 Credit: 60 Hours)

1. Implement a program to generate truth tables for given propositional formulas.
2. Write a program to determine if a given number is prime using different algorithms (e.g., brute force, sieve of Eratosthenes).
3. Implement functions to calculate the greatest common divisor (GCD) and least common multiple (LCM) of two numbers.
4. Create a program to perform arithmetic operations in modular arithmetic.
5. Develop a program to evaluate Boolean functions given input values.
6. Build a simulator for basic logic gates (AND, OR, NOT, XOR).
7. Implement a simple circuit minimization algorithm (e.g., Karnaugh maps).
8. Implement both adjacency matrix and adjacency list representations of graphs.
9. Find the degree of a vertex in a graph.
10. Find a path between two vertices in a graph.
11. Implement depth-first search (DFS) and breadth-first search (BFS) algorithms.
12. Determine if a graph is connected or disconnected.
13. Traverse a binary tree using pre-order, in-order, or post-order traversal.
14. Implement Dijkstra's algorithm to find the shortest path between two nodes.
15. Develop a program to check if two graphs are isomorphic.
16. Implement pre-order, in-order, and post-order traversals for binary trees.
17. Implement Kruskal's or Prim's algorithm to find a minimum spanning tree.
18. Implement a greedy algorithm for graph colouring.
19. Solve various number theory problems, such as finding prime factorization, modular exponentiation, and solving linear congruences.
20. Implement algebraic simplification techniques for Boolean expressions.
21. Write a program to represent a relation using an adjacency matrix and check if it is reflexive, symmetric, and transitive.
22. Implement a program to generate the Hasse diagram of a given partially ordered set.
23. Write a program to verify different types of functions such as one-to-one, onto, and bijective functions.
24. Solve problems on permutations and combinations using a programming language or calculator.
25. Implement the inclusion-exclusion principle to find the number of elements in the union of multiple sets.