# GUJARAT TECHNOLOGICAL UNIVERSITY

## Integrated MCA

### Year – 1 (Semester – I) (W.E.F. JULY 2018)

**Subject Name: Basic Mathematics**
**Subject Code: 2618604**

**Objective**

The objective of this course is to present the foundations of many basic mathematical topics used in Computer Science including RDBMS, Data Structures, Analysis of Algorithms, Theory of Computation, Cryptography, Artificial Intelligence, Statistics and others. This course will enhance the student's ability to think logically and mathematically.

**Prerequisites**: Binary number system, C Language

**Contents**:

| Sr. No. | Topics | Weightage Percentage |
|---|---|---|
| 1 | **Set Theory, Propositional & Predicate Logic:**<br>Set Theory: Basic Concepts of Set Theory: Definition, Two Methods to Describe (Represent) Sets; Examples, (Im)proper Subsets, Superset, Equality of Sets; Empty (Null) Set, Universal Set, Finite and Infinite Sets, Power Set; Operations on Sets: Union, Intersection, Complement, Venn Diagrams; Disjoint Sets, Various Laws: Identity, Idempotent, Commutative, Associative, Distributive, Absroption, DeMorgan; Difference (Relative Complement), Symmetric Difference of Two Sets; Cartesian Product; Power Set of a Set; Computer Representation of Sets; Examples; Theorems and Exercises (without Proof)<br><br>**Propositional Logic:** Definition, Statement (Proposition) & Notation, Truth Values, Connectives: Negation, Conjunction, Disjunction, Implication (condition), Bi-implication (Bi-conditional),<br>Truth Tables for all Connectives, Statement Formulas (Well-formed Formulas), Truth Tables, Tautologies, Contradiction, Logical Equivalence: Commutative Laws, Associative Laws, Distributive Laws, Absroption Laws, Idempotent Laws, Double Negation Law, DeMorgan's laws, Examples; Validity of Arguments, Some Valid Argument Forms: Modus Ponens, Modus Tollens, Disjunctive Syllogism, Dilemma, Equivalence of Formulas: Conjunctive Simplification, Disjunctive Addition, Conjunctive Addition, Examples and Exercises; Theorems (without Proof)<br><br>**Predicate Logic:** Definition of Predicates; Variables, Quantifiers: Universal Quantifiers, Existential Quantifiers; Free & Bound Variables; Negation of Predicates; Generalized DeMorgan's Laws; Valid Formulas and Equivalences; Additional Rules of Inference; Examples and Exercises; Theorems (without Proof) | **25%** |

| | | |
|---|---|---|
| **2** | **Proof Techniques, Matrices**<br><br>**Proof Techniques**: Direct Proof, Indirect Proof, Proof by Contradiction; Proving Bi-implications; Proving Equivalence Statements; Fallacies (Errors) in Proofs; Examples and Exercises<br><br>**Matrices**: Introduction; Representation of a Matrix; Equality of Matrices; Special Matrices: Rectangular / Square Matrices, Null (Zero) Matrix, Unit Matrix, Diagonal Matrices, Triangular Matrices; Sum and Difference of 2 Matrices; Multiplication of 2 matrices; Transpose of a Matrix, Symmetric Matrices; Boolean (Zero-One) Matrices, Boolean Join, Boolean Meet; Theorems and Exercises (without Proof) | **13%** |
| **3** | **Integers, Mathematical Induction**<br><br>**Integers**: Introduction, Basic Properties of Integers: Closure, Commutative Laws, Associative Laws, Identity Elements, Additive Inverse, Distributive Laws, Cancellation Laws; Well-ordering Principle; Division Algorithm: Quotient, Remainder, ***div*** and ***mod*** operators, Divisibility; Greatest Common Divisor (GCD); Euclidean Algorithm for Finding the GCD; Relative Prime; Least Common Multiples (LCM); Representation of Integers in Computer; Decimal, Binary, Octal, and Hexadecimal Representation; Operations on Binary Numbers: Addition, Subtraction; Theorems and Exercises (without Proof)<br><br>**Mathematical Induction**: Introduction; First Principle of Mathematical Induction; 3 Steps: Basis Step, Inductive Hypothesis, Inductive Step; Second Principle of mathematical Induction; Application: Loop Invariant (Program Correctness) | **12%** |
| **4** | **Relations and Functions**<br><br>**Relations**: Introduction, Binary Relation, Definition; Representation: Set of Ordered Pairs, Arrow Diagram, Matrix, Graph; Domain & Range of Relation; Universal Relation, Void Relation; Properties of a Relation: Reflexive, Symmetric, Transitive, Anti-symmetric, Irreflexive; Equivalence Relations, Partition, Block of Partition, Equivalence Classes and Partitions; Inverse of a Relation; Composition of Relations; Closures: Reflexive, Symmetric, Transitive; Theorems and Exercises (without Proof)<br><br>**Functions**: Introduction & Definition; Arrow Diagram of a Function; Domain, Co-domain (Target), Range of a Function; Special Function: Identity Function, Constant Function; One-to-One (Injective), Onto (Surjective), and One-to-One & Onto (Bijective) Functions; Composition of Functions; Inverse of a Function, Left-invertible & Right-invertible Functions; Floor & Ceiling Functions; Cardinality of a Finite Set; Theorems and Exercises (without Proof) | **25%** |
| **5** | **Graphs and Trees**<br><br>**Graphs**: Introduction, Definition; Initial & Terminal Nodes, Adjacent Nodes; Directed Edge, Undirected Edge, Directed Graph (Digraph), | **25%** |

| | | |
|---|---|---|
| | Undirected Graph, Mixed Graph; Loop (Sling); Distinct Edges, Parallel Edges; Multi-graph, Simple Graph; Weighted Graph; Isolated Nodes, Null Graph; Isomorphic Graphs; In-degree, Out-degree, Total-degree; Subgraphs; Reflexive, Symmetric, Transitive Digraphs; Paths, Length of Path of a Graph; Simple Path (Edge Simple), Elementary Path (Node Simple), Cycle (Circuit), Simple Cycle, Elementary Cycle; Path of Minimum Length (Geodesic), Distance between Two Nodes, Triangle Inequality; Reachability, Reachable Set of a Node, Reachable Set of a Set of Nodes, Node Base; Connected Graphs: Strongly, Unilaterally, Weakly Connected Graphs & Components; Matrix Representation of Graphs (Adjacency Matrix). In-degree, Out-degree of a Graph from Adjacency Matrix; An (A is Adjacency Matrix) to give Number of Paths of Length n; Path Matrix (Reachability Matrix) of a Graph; Warshall's Algorithm to Produce Path Matrix; Algorithm to Give Lengths of Minimum Paths; Theorems and Exercises (without Proof)<br><br>**Trees**: Introduction, Definition, Root, Branch Nodes, Leaf (Terminal Node); Different Representations of Trees; Forests, Subtrees; M-ary Tree, Full or Complete M-ary Tree; Binary Tree, Full (Complete) Binary Tree; Conversion of M-ary Tree to Binary Tree; Linked Allocation Technique to Represent Binary Tree in Computer. Traversal of Binary Tree: Pre-order, In-order, and Post-order Traversal; Theorems and Exercises (without Proof) | |

**Text Book:**


1. D. S. Malik & M. K. Sen, “Discrete Mathematics”, Cengage Learning (2004)
2. J. P. Tremblay and R.Manohar, “Discrete Mathematical Structures with Applications to Computer Science”, Tata McGraw-Hill (2010) – only for Unit-5 (Graphs & Trees).


**Reference Books:**


1. K. H. Rosen, “Discrete Mathematics and its applications”, Tata McGraw-Hill, 6th edition,
2. Bernard Kolmann & others, “Discrete Mathematical Structure”, Pearson Education, Sixth Edition
3. Edgar G. Goodaire, Michael M. Parmenter. “Discrete Mathematics with Graph Theory”, PHI
4. Ralph P Grimaldi & B V Ramana, “Discreet and Combinatorial mathematics: An Applied Introduction”, Pearson Education, 5th Edition (2018)
5. J. P. Tremblay and W. K. Grassman. “Logic and Discrete Mathematics”, Pearson Education


**Chapter-wise Coverage from the Text Book:**

Unit-1: Chapter-1
Unit-2: Chapters-1 & 4
Unit-3: Chapter-2
Unit-4: Chapters-3 & 5
Unit-5: Text Book-2: 5-1: 5-1.1 to 5-1.4; 5-2: 5-2.1.

**Accomplishment of Students after Completing the Course:**
Students will be able to understand various algorithms and implement them in C language. More specifically, they will be able to understand and apply the concepts of sets, logic, cross product of sets and relation, functions, matrices, and basic algorithms related with binary tree and graphs.

# Indicative Practical List

**List of Computer Lab Exercises (To be implemented in C Language)**

**Objectives**: To get an insight of data structures used in implementation of various basic mathematical concepts and algorithms based on these concepts.

**Prerequisites**: C Programming Language

**Advice (Note) to Teachers**:

1. The list of exercises given below is an indicative list. It is expected that students will be able to visualize how various mathematical concepts can be used to solve real-life problems through programming.
2. Some exercises have been labeled as "Mandatory" while other exercises have been marked as "Desirable". It is expected that all the students will do Mandatory exercises while bright students will additionally do Desirable exercises as well.

**List of Computer Lab Exercises**

**1. Mandatory Exercises: Set Theory and Logic**

(a) Start with a NULL set and add elements one-by-one: Use different ways of implementing sets and understand the pros and cons of each of these methods
(b) Given an element value, check whether it is a member of the set or not
(c) Find out the number of elements of a given set.
(d) Complement of a set; Union, Intersection
(e) Test whether a given set X is a subset of the set A or not.
(f) Test whether two given sets are equal or not
(g) Difference and Symmetric Difference of two sets
(h) Write functions for evaluating Logical-And, Logical-Or, Logical-Not, and Logical-XOR of given two Boolean values.

**Desirable Exercises: Set Theory & Boolean Logic**

(a) Find Cartesian Product of two given sets
(b) Find the Power Set of a given set
(c) Truth table of a Logical expression

**2. Mandatory Exercises: Matrices**

(a) Write a function to Create a Matrix of size m x n, and another function to Print a Matrix of size m x n.
(b) Print the Transpose of a given matrix A.
(c) Write a program to generate Null matrix of order (m x n) and Unit matrix of order n.
(d) Take as input two matrices, A & B and print A + B and A – B. First check whether it is possible to compute (A + B) and (A – B) or not.
(e) For a given square matrix A, print the Diagonal, Upper Triangular and the Lower Triangular Matrices of A.
(f) Evaluate Scalar Product of a Matrix A: For example, k A, where k is a constant (number)
(g) Take as input two matrices, A & B and print (A * B) and (B * A). First check which ones out of (A * B) and (A * B) are possible to compute.
(h) Given two matrices, determine whether one matrix is the inverse of the other matrix.

**Desirable Exercises – Matrices**

(a) For a given matrix A, find its inverse.

(b) Convert a given matrix with many zeros (0's) to Sparse matrix structure and vice versa.
(c) Evaluate the Value of the Determinant of the given Matrix.
(d) Find the Determinant of a square matrix and compute its value. Can it be done recursively?

**3. Mandatory Exercises: Integers**
(a) Write a program to generate all Prime Numbers between 1 and a given integer N.
(b) Given an integer N, test whether it is a Prime Number or a Composite Number.
(c) Write a program to find Factorial of a given number using (i) an iterative algorithm, and (ii) a recursive algorithm.
(d) Write a program to generate a set of Fibonacci Series up to (less than or equal to) a given number N using (i) an iterative algorithm, and (ii) a recursive algorithm.
(e) Evaluate Prime Factors of a given positive integer. (Assume that an array of prime numbers is available)
(f) Write a program to evaluate GCD (Greatest Common Divisor) of two positive integers using Euclidean algorithm
(g) Write Floor and Ceiling functions.
(h) Evaluate LCM (Least Common Multiple) of two positive integers.
(i) Convert a given Integer in decimal umber system into Binary, Octal, and Hexadecimal number systems.

**Desirable Exercises – Integers**
(a) Write a program using recursive algorithm to implement Tower of Hanoi problem.
(b) Convert a given Real Number with 3 decimal digits into Binary, Octal, and Hexadecimal.

**4. Mandatory Exercises: Binary Relation**
(a) Given the Boolean Matrix of a Binary Relation, determine whether the Relation is Reflexive and / or Symmetric.

**Desirable Exercises: Binary Relation**
(a) For a given Relation, generate a Boolean Matrix. Using Boolean Matrix, determine whether the Relation is Transitive
(b) Find the Partitions of the set using the given equivalence relation defined on the set.

**5. Desirable Exercises: Graphs & Trees**
(a) Take inputs of a graph and generate Adjacency Matrix of the graph. Print In-degree, Out-degree, and Total degree of each node.
(b) Represent a given binary tree using linked list structures. Print (i) Number of Leaf nodes, (ii) Intermediate nodes, and (iii) Total number of nodes.
(c) Given a binary tree represented as linked structure, traverse that binary tree using (i) Pre-order (or Depth-first) Traversal, (ii) In-order Traversal, (iii) Post-order Traversal using Recursive Algorithm
(d) Search whether an element is present in a given binary tree or not using Depth-first Search (DFS).

**Desirable Exercises: Graphs & Trees**
(a) Represent the given Adjacency Matrix of a graph as Sparse Matrix. Print In-degree, Out-degree, and Total degree of each node.
(b) Given a binary tree represented as linked structure, traverse that binary tree using (i) Pre-order Traversal, (ii) In-order Traversal, (iii) Post-order Traversal using Iterative Algorithm
(c) Use Linked List structure to represent threaded binary tree.
(d) Add a new node in a given AVL binary tree.
(e) Delete a specified node from a given AVL binary tree.

(f) Search whether an element is present in a given binary tree or not using Breadth-first Search (BFS).

**Reference Books:**

1. Kernighan and Ritchie, “C Programming”, Pearson Education
2. J. P. Tremblay and W. K. Grassman. “Logic and Discrete Mathematics”, Pearson Education,
3. ISRD Group, “Data Structures using C”, Tata McGraw Hill, 2006

**Reference Websites:**

1. www.uva.onlinejudge.org
2. www.cse.iitd.ernet.in/~bagchi/courses/discrete-book/fullbook.pdf

**Accomplishment of the student after completing the course:**

The student will be able to implement many of the concepts in C language. More specifically, the concept of Sets, Cross Product of Sets, Prime Numbers, Matrices, and basic algorithms related with Binary Tree and Graphs.