Course Outcomes: *After successful completion of the course, the students will be able to*

1. Apply mathematical logic to solve problems.
2. Learn the concept of sets, relations, functions and lattice.
3. Model and solve real world problems using graphs and trees.

Text Books:

1. Seymour Lipschutz and Marc Lars Lipson, **Discrete Mathematics**, Fourth Edition Schaum's Outline Series, McGraw Hill, 2022.
2. Kenneth H. Rosen, **Discrete Mathematics and Its Applications**, Seventh Edition, McGraw Hill, 2012.
3. Swapan K Sarkar, **A Textbook of Discrete Mathematics**, 9th Edition, S Chand & Co Ltd, 2016.

Reference Books:

1. D.J. Hunter, **Essentials of Discrete Mathematics**, Jones and Bartlett Publishers, 3rd Edition, 2008.
2. C.L. Liu, D.P. Mahopatra, **Elements of Discrete mathematics**, 2nd Edition, Tata McGraw Hill, 1985.
3. Deo N., **Graph Theory with Applications to Engineering and Computer Science**, PHI; 6th edition 2010.

Semester	**: II**
Course Type	**: DSC**
Course Code	**: CSCDSC151**
Name of the Course	**: Data Structure**
Learning level	**: Foundation or Introductory Course**
Credits	**: 3**
Contact Hours	**: 45**
Total Marks	**: 100**
End Semester Marks	**: 70**
Internal Marks	**: 30**

Course Objectives:

1. *Introduce the basic concepts and principles of data structures, including their definition, properties, and characteristics.*
2. *Familiarize students with the implementation of various data structures using programming languages, including arrays, linked lists, stacks, queues, trees, graphs, and hash tables.*
3. *How to analyze the time and space complexity of different data structures and algorithms, enabling them to make informed decisions regarding their selection and usage.*
4. *Cover various searching and sorting algorithms, including linear search, binary search, bubble sort, insertion sort, selection sort, merge sort, quicksort, and their analysis.*

5. *Cover the concepts of hashing, hash functions, collision resolution techniques, and the implementation and applications of hash tables.*

UNIT I

Arrays: Single and Multi-dimensional Arrays, Sparse Matrices (Array and Linked Representation).

Stacks: Implementing single / multiple stack/s in an Array; Prefix, Infix and Postfix expressions, Utility and conversion of these expressions from one to another; Applications of stack; Limitations of Array representation of stack.

Recursion: Developing Recursive Definition of Simple Problems and their implementation; Advantages and Limitations of Recursion.

UNIT II

Linked Lists: Singly, Doubly and Circular Lists (Array and Linked representation); Normal and Circular representation of Stack in Lists; Self Organizing Lists; Skip Lists. **Queues:** Array and Linked representation of Queue, De-queue, Priority Queues.

UNIT III

Trees: Introduction to Tree as a data structure; Binary Trees (Insertion, Deletion, Recursive and Iterative Traversals in Binary Search Trees); Threaded Binary Trees (Insertion, Deletion, Traversals); Height-Balanced Trees (Various operations on AVL Trees), Heap Tree.

UNIT IV

Searching and Sorting: Linear Search, Binary Search, and Comparison of Linear and Binary Search, Bubble Sort, Selection Sort, Insertion Sort, Merge Sort, Quick Sort, and Comparison of Sorting Techniques.

UNIT V

Hashing: Introduction to Hashing, Hash Table, Hash Key, Hash Function, Characteristics of Good Hash Functions, Types of Hash Functions, Collision, Resolving Collision by Open Addressing & closed Addressing: Linear probing, Quadratic probing, Double Hashing.

Course outcomes: *After successful completion of the course, the students will be able to*

1. *Demonstrate a solid understanding of various data structures, including arrays, linked lists, stacks, queues, trees and hash tables.*
2. *Develop proficiency in implementing data structures using programming languages, including creating and manipulating data structures through appropriate algorithms.*
3. *Apply analytical skills to analyze the time and space complexity of algorithms associated with different data structures, allowing for informed decision-making in algorithm selection.*
4. *Enhance critical thinking skills and problem analysis abilities by identifying the appropriate data structures and algorithms to solve given problems efficiently.*

Text Books:

1. Seymour Lipschutz, **Data Structures**, Schaum's Outline Series, TMH, 4th Edition, 2019.
2. Adam Drozdek, **Data Structures and Algorithms in C++**, Cengage Learning, 3rd Edition, 2012.
3. SartajSahni, **Data Structures, Algorithms and Applications in C++**, Universities Press, 2nd Edition, 2011.

Reference Books:

1. D.S Malik, **Data Structure using C++**, Cengage Learning, Second Edition, 2010.
2. Aaron M. Tenenbaum, Moshe J. Augenstein, YedidyahLangsam, **Data Structures Using C and C++**, PHI, 2nd Edition, 2009.
3. Robert L. Kruse, **Data Structures and Program Design in C++**, Pearson, 3rd Edition, 1999.

Semester	**: II**
Course Type	**: DSC**
Course Code	**: CSCDSC152**
Name of the Course	**: Lab on Data Structure**
Learning level	**: Foundation or Introductory Course**
Credits	**: 3**
Contact Hours	**: 90**
Total Marks	**: 100**
End Semester Marks	**: 70**
Internal Marks	**: 30**

Course Objectives:

1. *Help students apply the theoretical concepts of data structures in a practical setting. It should provide exercises and programming assignments that require students to implement and manipulate different data structures.*
2. *Enhancing students' programming skills by providing practical programming exercises.*
3. *It should encourage students to write code, debug, and test their implementations of data structures and associated algorithms.*

*This paper provides practical knowledge of data structure. List of laboratory programming assignments (not limited to these)***:**

1. Write a program to search an element from a list. Give users the option to perform Linear or Binary search. Use Template functions.
2. WAP using templates to sort a list of elements. Give users the option to perform sorting using Insertion sort, Bubble sort or Selection sort.
3. Implement Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list and concatenate two linked lists (include a function and also overload operator +).
4. Implement Doubly Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list.