Program	Diploma Engineering (DE)	**Semester - 3**
Type of Course	Professional Core	
Prerequisite	Basic knowledge of C Programming	
Course Objective	This subject helps to understand fundamental concepts of the data structure. By understanding the various data structures and their applications, students can identify the problems, analyze different algorithms to solve the problem efficiently & choose the appropriate data structure to represent the data.	

Teaching Scheme (Contact Hours)				Examination Scheme				
				Theory Marks		Practical Marks		
Lecture	Tutorial	Lab	Credit	SEE	CIA	SEE	CIA	Total Marks
3	0	2	4	40	30	20	10	100

***SEE** - Semester End Examination, **CIA** - Continuous Internal Assessment (It consists of Assignments/Seminars/Presentations/MCQ Tests, etc.)*

Course Content

T - Teaching Hours | **W** - Weightage

Sr.	Topics	T	W
1	**Introduction of Data Structure & Linear Data Structures: Array & Strings** Data Structure, Types of data structures, Algorithm, Key features of an algorithm, Analysis Terms - Time and Space Complexity, Best case Time Complexity, Average Time Complexity, Worst Case Time Complexity **Array:** Array, Array Operations- Insertion, Deletion **String:** Pointer, String, String operations- Length, Copy, Concatenation, Append, Comparison, Reverse, Retrieve and Insert Substring, Convert Uppercase and Lowercase	9	20
2	**Linear Data Structures: Stack and Queue** **Stack:** Stack, Stack Operations – Push, Pop, Peep, Change, Display, Application of Stack, Polish notation, Conversion of Infix Expression to Prefix and Postfix Expression, Evaluation of Postfix Expression & Prefix Expression, Recursion **Queue:** Queue, Queue Operations- Enqueue and Dequeue, Limitation of Simple Queue, Circular Queue, Circular Queue Operations - Enqueue and Dequeue, Priority Queue, Application of Queue	10	20
3	**Linear Data Structure: Linked List** Structure, Dynamic Memory Allocation- Malloc and Free, Introduction of Linked List, Types of Linked Lists, Singly Linked List Insertion Operations- Beginning of List, End of List, Before Given Node, After Given Node, Singly Linked List Deletion Operations- Beginning of List, End of List, Given position, Singly Linked List- Searching a Node, Count Number of Nodes, Application of Linked List	10	20
4	**Non-Linear Data Structure: Tree** Introduction of Non-Linear Data Structure- Tree, Graph Notations - (Node, Edge, Isolated Node, Graph, Degree, In-Degree, Out-Degree, Leaf Node, Level, Path, Depth, Height, Weight, Sibling, Forest, M-array tree, Directed Edge, Ancestor nodes, Descendant nodes, Strict Binary Tree, Complete Binary Tree). General Tree, Conversion of General Tree to Binary Tree, Binary Search Tree Operations- Insertion, Deletion, Searching, Tree Traversal Algorithms- Preorder, Inorder, Postorder, Application of Tree	8	20
5	**Sorting and Searching**	8	20

Course Content

T - Teaching Hours | **W** - Weightage

Sr.	Topics	T	W
	Sorting: Introduction of Sorting, Bubble Sort, Selection Sort, Insertion Sort, Merge Sort, Quick Sort, Radix Sort **Searching:** Linear Search, Binary Search		
		Total	**45**	**100**

Suggested Distribution Of Theory Marks Using Bloom's Taxonomy

Level	Remembrance	Understanding	Application	Analyze	Evaluate	Create
Weightage	15	45	40	0	0	0

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

Course Outcomes

At the end of this course, students will be able to:	
CO1	**perform** basic operations on Array and String.
CO2	**implement** Stack, Queue, and Circular Queue algorithms.
CO3	**prepare** algorithms for Singly Linked List.
CO4	**practice** tree manipulation algorithms.
CO5	**apply** sorting and searching techniques.

Reference Books

1.	**Data and File Structures using C** By Reema Thareja | Oxford University Press
2.	**Data Structures using C & C++** By Aaron M. Tanenbaum | PHI Learning
3.	**Data Structures** By A Chitra, P T Rajan | Tata McGraw Hill | Latest
4.	**Classic Data Structures** By Samanta, Debasis | PHI Learning

List of Practical

1.	**Basic C Programs** 1. Write a program to calculate area of a Circle (A = πr2). (A) 2. Write a program to find whether a number is odd or even. (A) 3. Write a program to find factorial of a number. (Using Loop) (A) 4. Write a program to find power of a number using loop. (A) 5. Write a program to find factors of a given number. (B) 6. Write a program to check whether a number is prime or not. (B) 7. Write a program to find the sum of 1 + (1+2) + (1+2+3) + …+(1+2+3+4+….+n). (C) 8. Write a program to print Armstrong number from 1 to 1000. (C)
2.	**Advance Array Programs** 1. Write a program to read and display n numbers using an array. (A) 2. Write a program to calculate the sum of numbers from 1 to n. (A) 3. Write a program to insert an element into an array at a given position.(A) 4. Write a program to delete a given element from an array. (A)

	5. Write a program to find whether the array contains a duplicate number or not. (B) 6. Write a program to delete a number from an array that is already sorted in an ascending order. (C) 7. Write a program to delete duplicate numbers from an array. (C)
3.	**String Programs using Function and Pointer-Set-1** 1. Write a program to find the length of the given string. (A) 2. Write a program to Copy one string to another string. (A) 3. Write a program to perform the concatenation of two given strings. (A) 4. Write a program to append a second string at the end of the first string. (A) 5. Write a program to insert a given substring into an existing string. (B) 6. Write a program to retrieve the substring from the given string. (B)
4.	**String programs using Function and Pointer-Set-2** 1. Write a program to compare two strings. (A) 2. Write a program to display the reverse string of the given string. (A) 3. Write a program to convert given string into uppercase and lowercase. (A)
5.	**Implementation of Stack using Array** 1. Write a menu-driven program to implement the following operations on the Stack using an Array: (A) i. PUSH, POP ii. Display all elements of the stack iii. PEEP, CHANGE 2. Write a program to calculate the average value of the Stack elements. (B) 3. Write a program to find the minimum and maximum element from a Stack (B). 4. Write a program for the evaluation of postfix Expression using Stack. (C) 5. Write a program for the evaluation of prefix Expression using Stack. (C)
6.	**Implementation of Simple Queue using Array** 1. Write a menu-driven program to implement the following operations on the Queue using an Array: (A) i. ENQUEUE ii. DEQUEUE iii. Display all elements of the queue 2. Write a program to calculate the average value of the Simple Queue elements.(B) 3. Write a program to find the minimum and maximum elements from Simple Queue. (B) 4. Write a program to sort the elements of a Simple Queue in ascending order. (C)
7.	**Implementation of Circular Queue using Array** 1. Write a menu-driven program to implement the following operations on a circular queue using an Array: (A) i. ENQUEUE ii. DEQUEUE iii. Display all elements of the circular queue 2. Write a program to calculate the average value of the Circular Queue elements. (B) 3. Write a program to find the minimum and maximum elements from the Circular Queue. (B) 4. Write a program to sort the elements of a Circular Queue in descending order. (C)
8.	**Implentation of Singly Linked List** 1. Write a program to implement a node structure for a singly linked list. Read the data in a node, and print the node. (A) 2. Write a menu-driven program to implement the following operations on the singly linked list: (A) i. Insert a node at the beginning of the linked list. ii. Insert a node at the end of the linked list. iii. Delete a first node of the linked list. iv. Delete a last node of the linked list. v. Display all nodes.
9.	**Implementation of Singly Linked List** 1. Write a menu-driven program to implement the following operations on the singly linked list: (A)

	i. Insert a node before the given node of the linked list. ii. Insert a node after the given node of the linked list. iii. Delete a given specific node of the linked list. iv. Display all nodes. 2. Write a program to count the number of nodes in a singly linked list. (A) 3. Write a program to search given nodes in a singly linked list. (A) 4. Write a program to reverse a linked list. (B)
10.	**Implementation of Binary Search Tree** 1. Perform the following operation on the binary search tree: (Paper Work Only) (A) i. Insert Node-80, 30, 25, 35, 85, 26, 55, 84, 33, 90, 36, 60, 37, 8 ii. Root Node is 40. iii. Delete Node-33, 55, 85 iv. Write a preorder, postordert, inorder Traversal for resultant tree. 2. Write a menu-driven program to implement Binary Search Tree (BST) & perform the following operations: (B) i. Insert a node ii. Delete a node iii. Search a node iv. Inorder Traversal v. Preorder Traversal vi. Postorder Traversal
11.	**Implementation of Bubble Sort, Selection Sort, Insertion Sort** 1. Write a program to implement a Bubble using Array. (A) 2. Write a program to implement Selection Sort using Array. (A) 3. Write a program to implement Insertion Sort using Array. (A)
12.	**Implementation of Quick Sort, Merge Sort** 1. Write a program to implement a Quick Sort using Array. (A) 2. Write a program to implement a Merge Sort using Array. (B)
13.	**Implementation of Linear Search, Binary Search** 1. Write a program to implement a Linear Search using Array. (A) 2. Write a program to implement a binary search using Array. (A)