**DEVIKTECH**

Curriculum for the course of

DATA STRUCTURE

Course Instructor –

**Learning Objectives**

This course will serve as a comprehensive introduction to various topics in data Structure. This course is a proper blend of theory, mathematical derivations and the practical hands on session for each and every concept. Throughout the course participants will learn and work on various data types, algorithms and it’s applications.

**Learning Outcomes**

At the end of the course participants should be able to -

* Solve problems related to graph, trees, lists.
* Solve time complexity and space complexity problems.

**Prerequisites**

There are no prerequisites required for the course. But a little knowledge of programming and high school mathematics is a benefit for candidate.

**Schedule**

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Topic** |  | **Total hours** |
| 1 | Introduction, Array, String, Recursion |  | 6 |
| 2 | Linked List, Stack, Queue |  | 6 |
| 3 | Searching, Sorting, Hashing |  | 6 |
| 4 | Trees- Basic Terminology, Binary Trees, Binary Trees Representation, Algebraic Expressions, Complete Binary Trees, Binary Search Tree etc. |  | 12 |
| 5 | Graphs - Multi-Graphs, Sequential representation of graphs, Adjacent Matrices, Traversal, Connected Component, Spanning Tree etc. |  | 6 |

**CURRICULUM**

1. **Introduction**
2. Basic Terminology
3. Algorithm Complexity
4. Time-Space trade-off
5. **Array**
6. Array Introduction
7. Algorithm Complexity
8. Time-Space trade-off
9. **Strings**
10. String Introduction
11. String operation
12. Practice Questions
13. **Recursion**
14. Recursion Definition
15. Finding the complexity of Recursion
16. Tower of Hanoi problem
17. Backtracking
18. Practice Question
19. **Linked List**
20. S​ingle Linked list
21. Introduction
22. Creating a Linked List
23. Traversing a Linked List
24. Adding a node in Front
25. Adding a node in last
26. Adding a node in middle
27. Deleting a node in frontDeleting a node in middle
28. Deleting a node in last
29. Reversing a linked list
30. Check whether a linked list is a Palindrome or not
31. Detect a loop in a linked list
32. Find middle element in a linked list
33. Doubly Linked List
34. Introduction
35. Insertion
36. Deletion
37. Reverse a linked List
38. Circular Linked List

Introduction

Traversal

1. **Stack**

​​Introduction

Operation on stack – Push and Pop

Array Representation of Stack

Linked Representation Of Stack

Queue using Stacks

Reverse a stack using recursion

1. **Application of stack**

Conversion Infix to Postfix

Infix to prefix

Postfix to Infix

Prefix to Infix

Prefix to Postfix

1. **Queues:**
2. Introduction
3. Operations on Queue:

Create

Add

Delete

Full

Empty

1. Reversing a Queue
2. Reversing a queue using recursion
3. Circular Queues
4. D-Queues
5. Priority Queues
6. **Searching**
7. ​Linear Search
8. Binary Search
9. Comparison and analysis
10. **Sorting**
11. ​​Bubble Sort
12. Insertion Sort
13. Selection Sort
14. Merge Sort
15. Quick Sort
16. Heap Sort
17. Comparison and analysis
18. **Hashing**
19. ​Introduction
20. Hash Table
21. Hash Functions
22. Hash Table Implementation
23. **Trees**
24. Basic Terminology
25. Binary Trees
26. Binary Trees Representation
27. Algebraic Expressions
28. Complete Binary Trees
29. Extended Binary Trees
30. Traversing Binary Trees
31. Construct Tree from given Inorder and Preorder traversals.
32. Construct a tree from Inorder and Level order traversals.
33. Check for Children Sum Property in a Binary Tree.
34. Check if a given Binary Tree is SumTree.
35. **Binary Search Tree(BST)**
36. Insertion
37. Deletion
38. Find the node with minimum value in a Binary Search Tree.
39. A program to check if a binary tree is BST or not.
40. Lowest Common Ancestor in a Binary Tree.
41. Complexity of Search algorithm
42. Path Length
43. **Graphs**
44. Basic Terminology
45. Representations
46. Graphs
47. Multi-Graphs
48. Sequential representation of graphs
49. Adjacent Matrices
50. Traversal
51. Connected Component
52. Spanning Tree
53. Minimum Cost Spanning Tree
54. **Alogorithm**
55. Analysis of Algorithm
56. Searching of Algorithm
57. Sorting of Algorithm
58. Greedy Algorithm
59. Dynamic Programming
60. Divide and Conquer
61. Backtracking. Branch and Bound