DATA STRUCTURES & ALGORITHMS - COURSE CONTENTS

This document describes the course contents of Data Structures & Algorithms

Invictus Engineers

3rd Floor, Surviba Towers, Plot No.247/3RT, Near Umesh Chandra Statue, Sanjeeva Reddy Nagar, Hyderabad, Telangana. 500038 **Email Us :** info@invictusengineers.com **Phone:** +91 99666 33097
www.invictusengineers.com

Table of Contents

DATA STRUCTURES & ALGORITHMS

COURSE CONTENTS

Week 01 – Introduction to Data Structures & Back to Arrays

Introduction to Data Structures

Applications of Data Structures

Types of Data Structures

A. **Physical vs Logical Data Structures**
B. **Linear vs Non-linear Data Structures**

Introduction to Arrays

Need for Arrays

1-D Arrays

- **Define and Initialize**
- **Memory Map of 1-DArray**
- **Traversing through the elements**
- **Copy operation**
- **Sum & Average of elements**
- **Min & Max of the elements**
- **Insert elements at given index**
- **Insert elements before/after given element**
- **Finding the Frequency of elements**
- **Shift & Rotate operations**

Data Structures & Algorithms

Course Content

1-D Arrays

Reversing the elements - efficient way

Finding Duplicate elements in unsorted Array

Sorting elements

A. Selection Sort and
B. Bubble Sort

Searching for elements – Linear Search

Finding Duplicate elements in sorted Array

Delete duplicate elements

Merging of elements from two arrays into one array

Set operations on Arrays

A. Union
B. Intersection and
C. Difference

2-D Arrays

Define and Initialize

Memory Map of 2-DArray

Traversing through the elements

A. Row by Row
B. Column by Column

Copy operation

Sum & Average of elements

A. Row by Row
B. Column by Column

2-D Arrays

Min & Max of the elements

A. Row by Row
B. Column by Column

Row-major and Column major representation

Operations on Matrices

A. Addition
B. Subtraction
C. Multiplication and
D. Transpose

Operations on Diagonal elements

Operations on Upper/Lower Triangle elements

Operations on Symmetric Matrices

Tri-diagonal and Tri-band matrices

Toeplitz Matrix

Sparse Matrix representation

Operations on Sparse Matrices

A. Addition
B. Subtraction
C. Multiplication and
D. Transpose

Polynomial Representation using Arrays

Operations on Polynomials

A. Addition
B. Subtraction and
C. Multiplication

Week 02 – Back to Functions - Time & Space Complexity

Basics of Functions

Scope Rules and Blocks

Calling and Called Functions

Prototyping and its necessity

Functions Returning Non-integers

Passing basic data type arguments

Call by Value - Limitations

Introduction to Call by Reference

Advantages of Call by Reference

Storage Classes - Classification

External Variables

Introduction to Recursion

How recursion works

Role of Stack in Recursion

Generalizing Recursion

Building the base conditions

Types of Recursion

i) Direct Recursion

A. **Tail Recursion**
B. **Head Recursion**
C. **Tree Recursion**
D. **Nested Recursion**

ii) Indirect Recursion

Passing 1-D and Multi-dimensional Arrays as arguments

Applying Recursion

A. **Printing natural numbers up to N**
B. **Sum of first N natural numbers**
C. **Factorial of a given number**
D. **Sum of digits**
E. **Fibonacci Series**
F. **Taylor Series**
G. **Horner's Rule**
H. **Towers of Hanoi**

Recursion vs Iteration

Introduction to Time and Space Complexity

A. **Best case**
B. **Worst case and**
C. **Average Case**

Measuring the Time & Space Complexity of given code snippets

Finding the Nth term of Fibonacci Series – Improved version

Role of Static & Global variables in Recursion

Deriving and Solving Recurrence Relations

Master's Theorem

Week 03 – A Tour to Structures & Pointers

Need for Structures

Structures – Terminology

Defining Structures and Structure variables

Implementing Structures

Operations on Structures

Copying Structure elements

Nested Structures

Using Bitfields in Structures

Passing Structures to Function – Call by Value

Applications of Structures

Limitations of fixed size arrays

Introduction to Dynamic Memory Allocation (DMA)

Stack vs Heap memory

Pointers & DMA

Implementing DMA functions

- **malloc()**
- **calloc()**
- **realloc()**
- **free()**

Pointers to Structures

DMA - 1D Arrays

DMA - 2D Arrays

DMA – Structures

DMA – Array of Structures

DMA - Nested Structures

Passing Structure variables to Functions – Call by Value

Passing Structure variables to Functions – Call by Reference

Returning Structure variables from Functions – By Value

Returning Structure variables from Functions – By Reference

Introduction to Self-referential structures

Applications of Self-referential structures

Week 04 – Searching & Sorting

Need for Searching & Sorting

Concept and Implementation of Linear Search

Concept and Implementation of Binary Search

Finding missing element using Binary Search

Find duplicate elements in sorted Array – Improved version

Selection Sort and its applications

Bubble Sort and its applications

Recursive Bubble Sort

Insertion Sort and its applications

Recursive Insertion Sort

Quick Sort and its applications

Recursive Quick Sort

Counting Sort and its applications

Shell Sort and its applications

Merge Sort and its applications

Recursive Merge Sort

Radix Sort and its applications

Pigeonhole Sort

Time & Space Complexity Analysis of Searching & Sorting techniques

Week 05 – Linked Lists & Applications

Self-referential Structures in play

Linked Lists – Introduction & Terminology

Types of Linked Lists

A. **Single Linked List**
B. **Circular Single Linked List**
C. **Double Linked List**
D. **Circular Double Linked List**

Representation of Linked Lists

Operations on Linked Lists

A. **CRUD operations**
B. **Merging of Linked Lists**
C. **Reversing Linked Lists**
D. **Ordered Linked Lists**
E. **Finding Duplicates in Linked Lists**
F. **Finding no. of nodes in linked list - efficient way**
G. **Finding Cycles in linked lists**

Sparse Matrix representation

Operations on Sparse Matrices

Polynomial Representation using Linked List

Operations on Polynomials

Week 06 – Stacks & Applications

Concept of Stack

Examples of Stack

Preliminary operations on Stack

Stack implementation

A. **Using Array**
B. **Using Linked List**

Implementation of two Stacks in Array

Applications of Stack

Reversing a String

Parenthesis matching problem

Introduction to Infix, Postfix and Prefix expressions

Conversion programs

A. **Infix to Postfix**
B. **Infix to Prefix**
C. **Postfix to Infix**
D. **Postfix to Prefix**
E. **Prefix to Infix**
F. **Prefix to Postfix**

Evaluation of Infix, Postfix and Prefix expressions

Validating the correctness of XML file content

Week 07 – Queues & Applications

Concept of Queue

Examples of Queue

Preliminary operations on Queue

Queue implementation

A. **Using Array**
B. **Using Linked List**

Types of Queues & Implementations

A. **Linear Queue – Pros & Cons**
B. **Circular Queues – Pros & Cons**
C. **Double-ended Queues – Pros & Cons**
D. **Priority Queues**
    1. **Sorted & Unsorted Arrays**
    2. **Sorted & Unsorted Linked Lists**

Implementing Queue using Stacks

Applications of Queues

A. **Simulation of Teller machine**
B. **Simulation of Doctor Appointment system**
C. **Calculation of Wait time in a Queue**
D. **Check if a Queue can be sorted into another Queue - using Stack**
E. **Reversing Queue using Recursion**
F. **Sorting Queue with no extra space**

Week 08 – Trees & Applications

Concept of Trees

Examples of Trees

Terminology

Types of Trees

Preliminary operations on Trees

Binary Trees & Properties

Implementation of Binary Tree

A. **Arrays**
B. **Linked List**

Binary Tree Traversal techniques

Operations on Binary Trees

Binary Search Tree & Properties

Implementation of Binary Search Tree

Operations on Binary Search Trees

Deletion from BST - All cases

Applications of BST

Expression Trees & Properties

Representation of Expression Trees

Concept of Heap

Implementation of Heap

Operations on Heap

Applications of Heap

Implementation of Expression Trees

Need for Balancing the Height of the Tree

AVL Trees & Properties

Rotations in AVL Trees

Implementation of AVL Trees

Operations on AVL Trees

Applications of BST

2-3 Trees

Operations on 2-3 Trees

Implementation of 2-3 Trees

2-3-4 Trees

Operations of 2-3-4 Trees

Implementation of 2-3-4 Trees

Red-Black Trees

Operations on Red-Black Trees

Week 09 – Greedy Algorithms & Applications

Concept of Greedy algorithms

Nature of Greedy Algorithms

Examples of Greedy Algorithms

Pros & Cons of Greedy Algorithms

Implementation of Greedy Algorithms

Applications of Greedy Algorithms

A. **Coin Change**
B. **Rod Cutting**
C. **Minimum Product Subset**
D. **Maximum Array Sum**
E. **Job sequencing**
F. **Activity Selection**
G. **No. of Railway Platforms**
H. **Huffman Encoding**
I. **Water Connection**
J. **Minimum Swaps for Bracket Balancing**
K. **Fractional Knapsack**

Week 10 – Divide & Conquer Algorithms – Applications

Concept of D & C algorithms

Nature of D & C Algorithms

Examples of D & C Algorithms

Quick Sort – Revisited

Merge Sort - Revisited

Pros & Cons of D & C Algorithms

Implementation of D & C Algorithms

Applications of D & C Algorithms

A. **Finding the Power**
B. **Inversion Count**
C. **Count of 0s**
D. **Missing element in Arithmetic Series**
E. **Polynomial Multiplication**
F. **Karatsuba's Fast Multiplication Algorithm**
G. **Strassen's Fast Matrix Multiplication Algorithm**
H. **The Tiling Problem**

Week 11 – Graphs & Applications

Concept of Graph

Terminology

Differences between Tree and Graph

Examples of Graph

Types of Graphs

Representation of Graph

A. **Adjacency Matrices**
B. **Adjacency List**
C. **Adjacency Multi list**

Preliminary operations on Graph

Implementation of Graph

Graph Traversing techniques

A. **Breadth First Search**
B. **Depth First Search**

Spanning Trees

Spanning Tree vs Graph

No. of possible Spanning Trees – Kirchhoff's Theorem

Minimum Cost Spanning Trees

Implementation of Prim's Algorithm

Implementation of Kruskal's Algorithm

Implementation of Warshall's algorithm

Implementation of Topological Sorting

Applications of Graphs

Week 12 – Hashing & Applications

Concept of Hashing

Need for Hashing

Examples of Hashing

Common Hashing functions

Address calculation techniques in Hashing

Collision and Resolution in Hashing

Implementation of Hashing

A. **Chaining**
B. **Linear Probing**
C. **Quadratic Probing**
D. **Double Hashing**

Load Factor and Rehashing

Applications of Hashing

A. **Highest Frequency element in an array**
B. **Finding if an array is subset of another array**
C. **Union and Intersection of two linked lists**
D. **Minimum delete operations in an array to make all the elements same**
E. **Check if two given sets are disjoint**

Week 13 – Dynamic Programming

Concept of Dynamic Programming

Nature of Dynamic Programming Problems

Pros & Cons of Dynamic Programming

Optimal vs Sub-optimal Solution

Introduction to Memoization – Nth Fibonacci Term

Property of Overlapping Subproblem

Tabulation vs Memoization

Applications of Dynamic Programming

A. **Coin Change**
B. **Rod Cutting**
C. **Tiling**
D. **Gold Mine**
E. **Friend Pairing**
F. **Subset Sum**
G. **Largest Divisible Pairs**
H. **Longest Common Sub-sequence**
I. **Shortest Common Super-sequence**
J. **Knapsack**

What next ???