



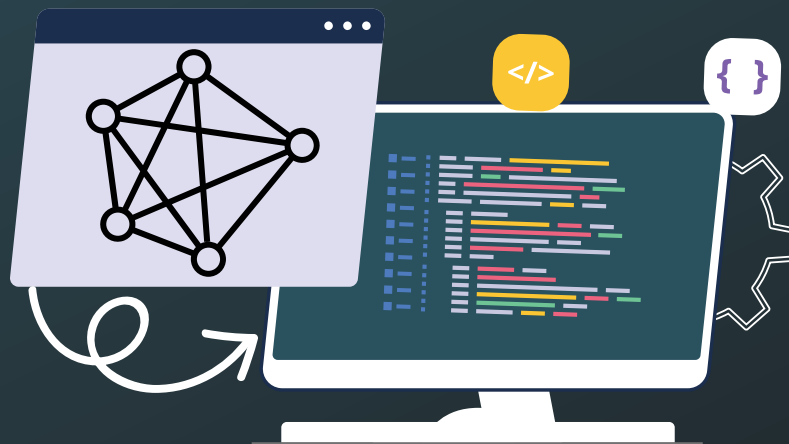
GeeksforGeeks

# DSA to Development

---

## A Complete Guide

---



Detailed  
Course Syllabus

## C++ BASICS

- Background Introduction
- Why do we need Programming Languages
- C++ Introduction
- C++ Standards and Implementation
- Writing First Code in C++
- Comments in C++
- Practice Problems

## VARIABLES AND DATA TYPES

- Variable in C++ and Naming Rules
- Data Types in C++
- Range of Data Types
- Global variable and scope
- sizeof in C++
- static in C++
- const in C++
- auto in C++
- Literals in C++
- Type Conversion in C++
- Swap two numbers
- Practice Problems

## INPUT OUTPUT IN C++

- Input in C++ (cin)
- Output in C++ (cout)
- Buffering in C++
- Escape Sequence in C++
- IO Manipulation
- Floating Point: Default Print Format, Manipulating Default format, fixed and scientific
- Practice Problems

## OPERATORS

- Arithmetic Operators
- Comparison Operators
- Logical Operators

- Assignment Operators
- Operator Precedence and Associativity
- Bitwise Operators
- Day Before N days
- Sum of N Natural Numbers
- Last Digit of a Number
- Practice Problems

## FLOW CONTROL

- If Else
- Nested If Else
- Switch
- Ternary
- Even Odd
- Largest of 3 numbers
- Leap Year
- Calculator Program
- Practice Problems

## FUNCTION

- Introduction to function
- Function Definition and Declaration
- Inline function
- Function Overloading
- First Digit of a Number
- Prime Factorization
- Practice Problems

## LOOPS

- For Loop
- While Loop
- Do while loop
- Continue
- Break
- Square Pattern
- Triangle Pattern

- Inverted Triangle Pattern
- Pyramid pattern
- Count Digits of a Number
- Factorial of a Number
- Check for Prime
- All Divisors of a Number
- GCD of Two Numbers
- LCM of Two Numbers
- Fibonacci Numbers
- Check for prime
- Table of a Number
- **Practice Problems**

## ARRAY

- Introduction to Arrays
- Declaration and initialization of an array
- Size of an Array
- Array Traversal
- Check if Array is Sorted
- Count Distinct in an array
- Sum of an array
- Average of an array
- Maximum in an array
- **Practice Problems**

## REFERENCES

- References in C++
- Function Parameters & References
- Range based for Loop & References
- Const & R value References
- **Practice Problems**

## POINTERS

- Address and Dereference Operators
- Introduction to Pointers
- Application of Pointers
- Function Pointer

- Array Parameter and Pointers
- Pointers vs Arrays
- Null in C++
- nullptr in C++
- Pointer Arithmetic
- Dynamic Memory Allocation
- Practice Problems

## STRINGS

- Introduction to strings
- C style String in C++
- String class in C++
- Operation on strings
- String Comparison
- getline() in c++
- String Traversal
- Reverse a string
- Palindrome
- Pattern Searching
- Practice Problems

## STRUCTURE AND UNION

- Struct in C++
- Struct vs Class in C++
- Structure (Pointer Array and Argument)
- Structure Alignment and Padding in C++
- Union in C++
- Complex Number using C++
- Practice Problems

## VECTORS

- Introduction to vectors
- Vector Declaration
- Operation on Vectors
- Get Smaller Elements
- Separate Even Odd
- Practice Problems

## MULTIDIMENSIONAL ARRAY

- Multidimensional Array in CPP
- Passing 2D array in as argument in CPP
- Transpose of a Matrix
- Matrix Multiplication
- Practice Problems

## TEMPLATES IN C++

- Template in C++
- Function Template in C++
- Class Template in C++
- Practice Problems

## OBJECT ORIENTED PROGRAMMING IN C++

- Object Oriented Programming in C++
- Constructor and Destructor
- This Pointer
- Static Member in C++
- Inheritance
- Virtual Functions
- Multiple Inheritance
- Operator Overloading
- Friend Function
- Practice Problems

## EXCEPTION HANDLING

- Exception Handling in C++
- Try throw and Catch in Exception Handling
- Stack Unwinding in Exception Handling
- User Defined Exception
- Practice Problems

## ADVANCED

- Smart Pointer Introduction

- unique\_ptr, shared\_ptr and weak\_ptr in C++
- Function Pointers
- Passing function as Parameters
- Lambda Expression
- Lambda Expression Examples
- Capture List in Lambda Expression
- Practice Problems
- Practice Problems

## INTRODUCTION TO STL

- Templates in C++
- Importance of STL
- STL Containers and its Classifications
- Iterators
- Practice Problems

## SIMPLE CONTAINERS

- Pairs
- Practice Problems

## SEQUENCED CONTAINERS

- Vector
- Forward\_list and List
- Dequeue
- Practice Problems

## CONTAINER ADAPTERS

- Stack
- Queue
- Priority Queue
- Practice Problems

## ASSOCIATIVE CONTAINERS

- Set & Multiset
- Map & Multimap
- Unordered\_set
- Unordered\_map
- Practice Problems

## STL ALGORITHMS

- Non-Mutating STL Algorithms
- Mutating STL Algorithms
- Practice Problems

## MISCELLANEOUS

- C++ string Class
- Manipulating STL
- builtin\_popcount(), builtin\_popcountll()
- Practice Problems

## JAVA BASICS

- Background Part 1 (IO Devices, CPU and Memory)
- Background Part 2 (Computer Organization)
- Why Do We Need Programming Languages
- Java Introduction (Features, Applications and Working of JVM, etc)
- Writing First Program in Java
- Practice Problems

## VARIABLES AND DATA TYPES

- Variables in Java
- Non Primitive Types
- Wrapper Classes
- Autoboxing and Unboxing
- Swap two Variables
- Type Conversion in Java
- Practice Problems



## INPUT AND OUTPUT IN JAVA

- Output in Java (print() and println())
- Input in Java (Scanner, Reader & Writer)
- Escape Sequences in Java
- Practice Problems

## OPERATORS

- Operators in Java (Arithmetic, Logical and Bitwise Operators)
- Arithmetic Progression nth Term in Java
- Geometric Progression nth Term in Java
- Sum of Natural numbers
- Find Last Digit in Java
- Practice Problems

## FLOW CONTROL

- if, else and elif in Java
- Switch Statement
- Even-Odd in Java
- Largest of three in Java
- Leap Year in Java
- Calculator Program in Java
- Practice Problems

## LOOPS

- Loops In Java
- While Loops in Java
- For Loop In Java
- ForEach Loop in Java
- Table of A Number
- Break In Java
- Continue In Java
- Nested Loop in Java
- Square Pattern in Java
- Printing Triangle Pattern in Java
- Inverted Triangle in Java

- Pyramid Pattern in Java
- Count Digits
- Factorial in Java
- GCD in Java
- LCM in Java
- Fibonacci Numbers in Java
- Check for Prime in Java
- All Divisors in Java
- Optimizations of All Divisors and Prime
- Practice Problems

## FUNCTIONS

- Functions in Java (Introduction and Working)
- Applications of Functions
- Command Line Arguments
- Find First Digit in Java
- Prime Factorization
- Practice Problems

## ARRAY

- Introduction
- a[] vs []a
- Working of arrays and types
- Multidimensional Arrays in Java
- Find Average
- Check if Array is Sorted
- Count Distinct Elements
- Practice Problems

## STRING

- Strings in Java
- Escape Sequences and Raw Strings
- StringBuffer and StringBuilder
- StringBuffer and StringBuilder Methods
- Pattern Searching in Java
- Check For Palindrome In Java
- Reverse A String in Java
- Practice Problems

## CLASSES AND OBJECTS

- Classes in Java
- Objects in Java
- Constructors and Types
- This Reference
- Access Specifiers
- Static and Final
- Practice Problems

## JAVA OOPS

- Class and Objects in Java
- Encapsulation in Java
- Access Modifiers in Java
- This Reference
- Final Keyword
- Static Members
- Constructors
- Inheritance in Java
- Super keyword in Java
- More on Java Inheritances
- Polymorphism in Java
- Method Overriding in Java
- More on Method Overriding
- Abstraction in Java
- Abstract Classes in Java
- Interface in Java
- Interfaces vs Abstract Classes
- OOP Quiz | Part 1
- OOP Quiz | Part 2
- Practice Problems

## ADVANCED

- File I/O
- MultiThreading
- Exception Handling
- BigInteger
- Practice Problems

## COLLECTIONS OVERVIEW

- Introduction to Java Collections Framework
- Collections hierarchy
- Generics and much more
- Practice Problems

## JAVA LAMBDA EXPRESSIONS

- Introduction to Lambda Expressions and ways to use them
- Introduction to Method References and examples
- Syntax of Lambda Expressions and much more
- Practice Problems

## JAVA STREAMS

- Introduction to Streams in Java
- Various Applications of Streams
- The Stream hierarchy and methods and much more
- Practice Problems

## ARRAY LIST

- Introduction to List Interface
- Using List Iterator
- Introduction to ArrayLists and much more
- Practice Problems

## LINKED LIST

- Introduction and implementation of LinkedList in Java
- Problems with a video explanation and much more
- Practice Problems

## STACK

- Introduction to Stack
- Implementation
- Methods and much more
- Practice Problems

## QUEUE

- Introduction to Queue Interface
- Implementation and usage
- Methods and much more
- Practice Problems

## DEQUE

- Introduction to Deque
- Implementation and usage
- ArrayDeque and much more
- Practice Problems

## PRIORITY QUEUE

- Introduction to PriorityQueue
- Implementation and usage
- Methods and much more
- Practice Problems

## HASHSET AND LINKEDHASHSET

- Introduction to HashSet
- Introduction to LinkedHashSet
- Implementation and usage and much more
- Practice Problems

## TREESET

- Introduction to TreeSet
- Implementation and usage
- Methods and much more
- Practice Problems

## HASHMAP AND LINKEDHASHMAP

- Introduction to HashMap
- Introduction to LinkedHashMap
- Implementation and usage and much more
- Practice Problems

## TREEMAP

- Introduction to TreeMap
- Implementation and usage
- Methods and much more
- Practice Problems

## STRING

- Introduction to Strings
- Introduction to StringBuilder and StringBuffer
- Implementation and usage and much more
- Practice Problems

## COMPARATOR AND COMPARABLE

- Introduction to Comparable Interface
- Introduction to Comparator Interface
- Methods of Comparator Interface and Examples on it and much more
- Practice Problems

## ARRAYS CLASS

- Introduction to Arrays and the Arrays Class
- Implementation and usage
- Traversal and much more
- Practice Problems

## COLLECTIONS CLASS

- Introduction to Collections Class
- Methods like fill(), reverse(), binarySearch(), max(), min(), frequency() and much more
- Practice Problems

## SORTING

- Introduction to sorting in Java, Arrays.sort(), Collections.sort() and much more
- Practice Problems

## PYTHON BASICS

- Background Part 1 (IO Devices, CPU and Memory)
- Background Part 2 (Computer Organization)
- Why Do We Need Programming Languages
- Python Introduction
- Python Standard and Implementations
- How Python Programs Are Executed
- Python Programming Terminology
- Python Installation and First Program
- Comments in Python

## VARIABLES AND DATA TYPES

- Variables in Python
- Swap two Variables
- Id() in Python
- Type() in Python
- List Introduction
- Tuples in Python
- Set in Python
- Dictionary in Python
- Type Conversion in Python

## INPUT AND OUTPUT IN PYTHON

- print() in Python
- input() in Python

## OPERATORS

- Arithmetic Operators in Python
- Logical Operators in Python
- Identity Comparison Operators in Python
- Membership Test Operators in Python
- Bitwise Operators in Python Part 1
- Bitwise Operators in Python Part 2
- Arithmetic Progression nth Term in Python
- Geometric Progression nth Term in Python
- Sum of Natural numbers
- Find Last Digit in Python

## FLOW CONTROL

- if, else and elif in Python
- Even-Odd in Python
- Largest of three in Python
- Leap Year in Python
- Calculator Program in Python

## FUNCTIONS

- Functions in Python
- Applications of Functions
- How Functions Work?
- Default Arguments
- Keyword Arguments
- Variable Length Arguments
- Parameter Passing in Python
- Returning Multiple Values in Python
- Global Variables in Python
- Find First Digit in Python
- Prime Factorization

## LOOPS

- Loops In Python
- While Loops in Python
- range() in Python
- For Loop In Python
- Table of A Number
- Break In Python
- Continue In Python
- Nested Loop in Python
- Square Pattern in Python
- Printing Triangle Pattern in Python
- Inverted Triangle in Python
- Pyramid Pattern in Python
- Count Digits
- Factorial in Python
- GCD in Python
- LCM in Python
- Fibonacci Numbers in Python
- Check for Prime in Python
- All Divisors in Python
- Optimizations of All Divisors and Prime



## STRING

- Strings in Python
- Escape Sequences and Raw Strings
- Formatted String in Python
- String Operations Part (1)
- String Operations Part (2)
- String Comparison in Python
- Pattern Searching in Python
- Check For Palindrome In Python
- Reverse A String in Python
- Decimal to Binary Conversion
- Binary to Decimal Conversion

## LIST

- Slicing (List,Tuple And String)
- Get Smaller Elements
- Separate Even and Odd
- Comprehensions in Python
- Average or Mean of A List
- Count Distinct Elements in A List
- Check if a list is Sorted

## RESUME BUILDING

- Resume Building

## PROGRAMMING LANGUAGES

- **C++** : Introduction and Basic I/O, Variables, Different Errors, Operators, Loops, Arrays, String, Functions, Pointers, Dynamic Memory Allocation, Exception Handling and Smart Pointers
- **Java** : Introduction and Basic I/O, Variables , Operators, Loops, Exception Handling, Arrays, String , Immutable Strings, ArrayList , BigInteger

## OBJECT ORIENTED PROGRAMMING

- Classes and Objects
- Inheritance and Polymorphism : Overloading and Overriding
- Abstraction and Encapsulation
- Access Modifiers
- Friend and Virtual functions in C++
- static, final, this and super keywords and Interfaces in Java

## DATA STRUCTURES (BASICS)

### Analysis of Algorithms:

- Growth of functions
- Asymptotic Notations Omega, Theta,
- Recursion Tree Method
- Space Complexity

### Arrays:

- Insertion, Deletion, Updation, Shifting
- Reversal, Sort Check, Maximum, Minimum

### Recursion

- Introduction to Recursion
- Tail Recursion
- Natural Number Check Using Recursion
- Palindrome Check Using Recursion
- Sum of Digits, Rod Cutting and Subsets
- Tower of Hanoi

### Hashing:

- Introduction to Hashing
- Direct Address Table
- Collision Handling
- Chaining
- Open Addressing
- Double Hashing
- Chaining Vs Open Addressing

### String:

- Introduction to Strings

## Searching:

- Linear Search
- Binary Search (Iterative and Recursive)

## Sorting:

- Stability in Sorting Algorithm
- Bubble Sort
- Selection Sort
- Insertion Sort
- Quick Sort
- Different Partition Schemes in QuickSort
- Merge Sort
- Lomuto Partition
- Hoare Partition
- Heap Sort
- Counting Sort
- Radix Sort
- Bucket Sort

## Linked List:

- Drawback of Arrays
- Introduction to Linked List and Implementation
- Traversal, Insertion and Deletion
- Sorted Insertion in Linked List
- Reversal of Linked List (Iterative and Recursive)
- Finding Middle
- Remove Duplicate from Sorted Linked List

## Circular Linked List:

- Traversal
- Insertion (Head, End)
- Deletion (Head, Kth Node)

## Doubly Linked List:

- Traversal
- Insertion (Head, End)
- Deletion (Head, End)
- Reversal
- Circular Doubly Linked List

## Stack:

- Introduction to Stack Data Structure
- Implement using array
- Implementation using Linked List
- Stack Applications

## Queue:

- Introduction to Queue Data Structure
- Implementation using array
- Implementation using Linked List.

## Deque:

- Introduction to Deque Data Structure.
- Implementations using Array
- Implementation using Linked List

## Tree:

- Implementation
- Traversals: preorder, postorder, inorder, level order(Iterative & Recursive)
- Binary Tree: Height, Size, Maximum
- Print Nodes at K Distance

## BST:

- Implementation
- Search
- Insertion
- Deletion
- Floor and Ceil in BST in CPP and Java
- Self Balancing BST
- AVL Tree (Introduction and applications)
- Red-Black Tree (Introduction and applications)
- Applications of BST

## Heap:

- Implementation
- Insert
- Heapify and Extract in Heap
- Decrease Key, Delete and Build Heap

## LIBRARIES

### C++ STL

#### Introduction to STL

- i) Introduction and Application
- ii) Iterators
- iii) Templates
- iv) Function Templates
- v) Class Templates

#### ◦ Pairs in CPP STL

- i) Introduction
- ii) Problem(With Video Solutions): Sorting an array according to another array
- iii) Practice Problems
  - (1) This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

#### ◦ Vectors in CPP STL

- i) Introduction
- ii) Vector Declaration
- iii) More functions of Vectors
- iv) Time Complexities of different operations and passing Vectors to function
- v) Internal Working of Vectors
- vi) Problems(With Video Solutions):
  - (1) Vector and Vector of Pairs
  - (2) Keeping track of previous indexes after sorting a Vector

#### ◦ Forward\_list and list

- i) Forward List in C++ STL
- ii) List in C++ STL
- iii) Problems(With Video Solutions):
  - (1) Josephus Problem using List in STL
  - (2) Design a Data Structure with Insert/Replace/Print operations
- iv) Practice Problems
  - (1) This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned.

- Deque

- i) Introduction
- ii) Problems(With Video Solutions):
  - (1) Sliding Window Maximum
  - (2) Design a Data Structure with Min/Max operations in  $O(1)$  time
- iii) Practice Problems
  - (1) This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned.

- Stack

- i) Introduction and Various Operations
  - (1) push()
  - (2) pop()
  - (3) top()
  - (4) size()
  - (5) empty()
- ii) Problems(With Video Solutions):
  - (1) Reverse items using Stack
  - (2) Balanced Parenthesis
  - (3) Stock Span Problem
  - (4) Previous Greater Elements
  - (5) Next Greater Elements
- iii) Practice Problems
  - (1) This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned.

- Queue

- i) Introduction and Various Operations
  - (1) push()
  - (2) pop()
  - (3) front()
  - (4) back()
  - (5) empty()
  - (6) size()
- ii) Problems(With Video Solutions):
  - (1) Reverse first K items in a Queue
- iii) Practice Problems
  - (1) This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned.

- Priority Queue

- i) Introduction and Various Operations
  - (1) push()
  - (2) pop()
  - (3) top()
  - (4) empty()
  - (5) size()
  - (6) Creating Min Heap based Priority Queue
- ii) Problems(With Video Solutions):
  - (1) Sort an array using Priority Queue
  - (2) K Largest Elements in an array
  - (3) Buy maximum items with given money
  - (4) Find K most frequent elements
- iii) Practice Problems
  - (1) This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned.

- Set & MultiSet

- i) Set in C++ STL
  - (1) Introduction and Implementation
  - (2) insert()
  - (3) begin()
  - (4) end()
  - (5) rbegin()
  - (6) rend()
  - (7) erase()
  - (8) clear()
  - (9) find()
  - (10) Internal Working
  - (11) Time Complexities
- ii) Problems on Set(With Video Solutions):
  - (1) Design a Data Structure that supports the below operations:
  - (2) insert()
  - (3) delete()
  - (4) search()
  - (5) getFloor()
  - (6) getCeiling()
- iii) Multiset in C++ STL with few operations
- iv) Practice Problems
  - (1) This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned.

- Map and MultiMap

- i) Introduction to Map
  - (1) insert()
  - (2) operator()
  - (3) size()
  - (4) empty()
  - (5) clear()
  - (6) begin()
  - (7) end()
  - (8) Internal Working
  - (9) Time Complexities
- ii) Problem:
  - (1) Design a data structure for item prices. The operations are add(), find(), findGreater(), findSmaller() and printSorted()
  - (2) Count greater elements for every array element.
- iii) Multimap in C++ STL with few functional operations
- iv) Problem(With Video Solutions):
  - (1) Design a data structure for prices with duplicates allowed. The operations are add(), find(), findGreater(), findSmaller() and printSorted
- v) Practice Problems
  - (1) This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned.

- Unordered\_set

- i) Introduction to Set
  - (1) insert()
  - (2) begin()
  - (3) size()
  - (4) end()
  - (5) clear()
  - (6) find()
  - (7) Internal Working
  - (8) Time Complexities
- ii) Problems(With Video Solutions):
  - (1) Print Unique Elements of Array
  - (2) Print duplicate elements of the array
- iii) Practice Problems
  - (1) This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned.



- **Unordered\_Map**

- i) Introduction
- ii) Problems(With Video Solutions):
  - (1) Design a DS for storing user balance
  - (2) Find Winner of Election
- iii) Practice Problems
  - (1) This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned.

- **Non Mutating STL Algorithms**

- i) Explanation along with Time Complexities of
  - (1) max\_element()
  - (2) min\_element()
  - (3) accumulate()
  - (4) count()
  - (5) find()
  - (6) binary\_search()
  - (7) lower\_bound()
  - (8) upper\_bound()
  - (9) rotate()
  - (10) fill()
  - (11) is\_permutation()
  - (12) rand()
- ii) Practice Problems
  - (1) This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned.

- **Mutating STL Algorithm**

- i) Explanation along with Time Complexities of
  - (1) sort()
  - (2) reverse()
  - (3) next\_permutation()
  - (4) prev\_permutation()
  - (5) make\_heap()
  - (6) merge()
- ii) Problems(With Video Solutions):
  - (1) The Thief problem
  - (2) Fractional knapsack problem
  - (3) Chocolate Distribution problem
  - (4) Sort array elements by frequency

## Java Collections

### ◦ Collection Overview

- i) Introduction to Java Collections Framework
- ii) Collections hierarchy
- iii) Generics
- iv) Wildcards
- v) toArray() Methods
- vi) Collections Interface
- vii) Iterators
- viii) Collections Bulk operations
- ix) Iterating through Collections

### ◦ Java Lambda Expressions

- i) Introduction to Lambda Expressions and ways to use them
- ii) Introduction to Method References and examples
- iii) Syntax of Lambda Expressions
- iv) Practice Problems
  - (1) Practice problems on Lambda Expressions

### ◦ Java Streams

- i) Introduction to Streams in Java
- ii) Various Applications of Streams
- iii) The Stream hierarchy and methods
- iv) Examples on Streams
- v) Practice Problems
  - (1) Practice problems on Streams

### ◦ ArrayList

- i) Introduction to List Interface
- ii) Using List Iterator
- iii) Introduction to ArrayLists
- iv) Implementation
- v) ArrayList Methods
- vi) Traversal
- vii) Problems with video explanation
  - (1) List of smaller elements
- viii) Practice Problems
  - (1) Practice problems on implementation, iterator, methods, and using ArrayList to solve dsa problems

- **Linked List**

- i) Introduction and implementation of LinkedList in Java
- ii) Problems with video explanation
  - (1) Josephus Problem using LinkedList
  - (2) Design a DS for remove and print
- iii) Practice Problems
  - (1) Practice problems on implementation, traversal, and use of LinkedList,

- **Stack**

- i) Introduction to Stack
- ii) Implementation
- iii) Methods
- iv) Traversal
- v) Problems with video explanation
  - (1) Reverse order of items
  - (2) Check for balanced parentheses
  - (3) Stock span
  - (4) Previous greater element
  - (5) Next greater element
- vi) Practice Problems
  - (1) Practice problems on implementation, methods, and using Stacks to solve dsa problems

- **Queue**

- i) Introduction to Queue Interface
- ii) Implementation and usage
- iii) Methods
- iv) Traversal
- v) Problems with video explanation
  - (1) Reverse first k items
- vi) Practice Problems
  - (1) Practice problems on implementation, methods, and using Queue to solve dsa problems

- **Deque**

- i) Introduction to Deque
- ii) Implementation and usage
- iii) ArrayDeque
- iv) Methods
- v) Traversal
- vi) Practice Problems
  - (1) Practice problems on implementation, methods, and using ArrayDeque to solve dsa problems

- PriorityQueue

- i) Introduction to PriorityQueue
- ii) Implementation and usage
- iii) Methods
- iv) Traversal
- v) Problems with video explanation
  - (1) Purchasing maximum items
  - (2) K largest elements
  - (3) Find k most frequent
  - (4) Find k most frequent in Linear time
- vi) Practice Problems
  - (1) Practice problems on implementation, methods, and using PriorityQueue to solve dsa problems

- HashSet and LinkedHashSet

- i) Introduction to HashSet
- ii) Introduction to LinkedHashSet
- iii) Implementation and usage
- iv) Methods
- v) Traversal
- vi) Problems with video explanation
  - (1) Print distinct elements
  - (2) Print repeating elements
- vii) Practice Problems
  - (1) Practice problems on implementation, methods, and using HashSet to solve dsa problems

- TreeSet

- i) Introduction to TreeSet
- ii) Implementation and usage
- iii) Methods
- iv) Traversal
- v) Problems with video explanation
  - (1) Ceiling on right
  - (2) Count greater element
- vi) Practice Problems
  - (1) Practice problems on implementation, methods, and using TreeSet to solve dsa problems

- **HashMap and LinkedHashMap**

- i) Introduction to HashMap
- ii) Introduction to LinkedHashMap
- iii) Implementation and usage
- iv) Methods
- v) Traversal
- vi) Problems with video explanation
  - (1) DS for balance
  - (2) Print frequencies in order
- vii) Practice Problems
  - (1) Practice problems on implementation, methods, and using HashMap to solve dsa problems

- **TreeMap**

- i) Introduction to TreeMap
- ii) Implementation and usage
- iii) Methods
- iv) Traversal
- v) Problems with video explanation
  - (1) Design a data structure for item prices
  - (2) Design a data structure for item prices with duplicates allowed
- vi) Practice Problems
  - (1) Practice problems on implementation, methods, and using TreeMap to solve dsa problems

- **String**

- i) Introduction to Strings
- ii) Introduction to StringBuilder and StringBuffer
- iii) Implementation and usage
- iv) Methods
- v) Traversal
- vi) Problems with video explanation
  - (1) Pangram checking
  - (2) Pattern searching
  - (3) Find one extra character
- vii) Practice Problems
  - (1) Practice problems on implementation, methods, and using Strings to solve dsa problems

- **Comparator and Comparable**

- i) Introduction to Comparable Interface
- ii) Introduction to Comparator Interface
- iii) Methods of Comparator Interface and Examples on it
- iv) Practice Problems
  - (1) Practice problems on using Comparator to sort effectively

- **Arrays Class**

- i) Introduction to Arrays and the Arrays Class
- ii) Implementation and usage
- iii) Methods like
  - (1) fill()
  - (2) BinarySearch()
  - (3) equals()
  - (4) mismatch()
  - (5) compare()
  - (6) asList()
  - (7) toString()
- iv) Traversal
- v) Practice Problems
  - (1) Practice problems on implementation and methods

- **Collections Class**

- i) Introduction to Collections Class
- ii) Methods like fill(), reverse(), binarySearch(), max(), min(), frequency()
- iii) Practice Problems
  - (1) Practice problems on methods

- **Sorting**

- i) Introduction to sorting in Java
- ii) Arrays.sort()
- iii) Collections.sort()
- iv) Comparable Interface
- v) Problems with video explanation
  - (1) The thief problem
  - (2) Chocolate distribution problem
  - (3) Keep indices after sorting
  - (4) Sort an array according to other
  - (5) Sort students by marks
  - (6) Sort elements by frequency
  - (7) Sort elements by frequency in Linear Time
- vi) Practice Problems
  - (1) Practice problems on various sorting algorithms, and comparator sort

## DATA STRUCTURES (ADVANCED)

### Mathematics

- Count Digits
- Palindrome Numbers
- Factorial of Numbers
- GCD of Two Numbers
- LCM of Two Numbers
- Check for Prime
- Prime Factors
- Sieve of Eratosthenes
- Computing Power

### Bit Magic

- Bitwise Operators in CPP(Part 1)
- Bitwise Operators in CPP(Part 2)
- Bitwise Operators in JAVA(Part 1)
- Bitwise Operators in JAVA(Part 2)
- Bitwise Operators in JAVA(Part 3)
- Check Kth bit is set or not
- Count set bits
- Power of Two
- One Odd Occurring
- Two Odd Occurring
- Power Set using Bitwise

### Recursion

- Josephus Problem
- Subset Sum Problem

### Arrays

- Kadane's Algorithm
- Shuffling Algorithms
- Sliding Window
- Prefix Sum Technique
- Video Solutions for some standard and complex problems
- More Problems for Practice.

## Matrix

- Multidimensional Array in CPP and Java
- Search, Transpose and Rotate
- Pattern Traversal: Snake, Spiral, Boundary
- Video Solutions for some standard and complex problems
- More Problems for Practice.

## Searching

- Two Pointer Approach
- Video Solutions for some standard and complex problems
- More Problems for Practice.

## Sorting

- Union And Intersection of Sorted Arrays
- Inversions Count
- Tail Call elimination Quick Sort
- Cycle Sort
- Merge of Overlapping Intervals
- Overview of Sorting Algorithms
- Video Solutions for some standard and complex problems
- More Problems for Practice.

## Hashing

- Double Hashing
- Find frequencies of array
- Count Distinct element in Every Window
- Intersection and Union via Hashing
- Frequencies of Array Elements
- Distinct Elements in Window
- Counting Occurences
- Check for a Pair with given Sum
- Longest Consecutive Subsequence
- Subsequence Problems
- Subarray Problems
- Video Solutions for some standard and complex problems
- More Problems for Practice.

## Strings

- Creation, Updation
- Reverse, Pangram, Case conversion
- Validation, Length
- Palindrome Check
- Overview of Pattern Searching



- Pattern Matching Algorithms: Rabin Karp Algorithm, KMP Algorithm
- Rotations Check of two Strings
- Anagram
- Video Solutions for some standard and complex problems
- More Problems for Practice.

## Linked List

- Doubly Linked List
- Circular Linked List
- Loop in Linked List (Detection and Removal)
- Loop Detection Algorithms
- Union and Intersection of LinkedLists
- Reverse in Groups
- LRU Cache Design
- Palindrome LinkedList
- Video Solutions for some standard and complex problems
- More Problems for Practice.

## Stack

- Infix, Postfix, Prefix (Introduction)
- Infix to PostFix (Simple Solution)
- Infix to PostFix (Efficient Solution)
- Evaluation of Postfix
- Infix to Prefix (Simple Solution)
- Infix to Postfix (Efficient Solution)
- Evaluation of Prefix
- Implementing Two Stacks in Single Array
- Implementing K stacks in Single Array
- Largest Rectangular Area in Histogram
- Design a Stack that supports getMin() operation
- Video Solutions for some standard and complex problems
- More Problems for Practice.

## Queue and Deque

- Stack using Queue
- Reversal
- Maximum of all Subarrays of Size K
- Generate numbers using given digits
- Design a data structure with min/max operations
- Video Solutions for some standard and complex problems
- More Problems for Practice.

## Tree

- Line By Line Level Order Traversal
- Printing Left, Right, Top and Bottom Views
- Binary Tree to Doubly Linked List
- Binary Tree from Inorder and Postorder Traversal
- Maximum Width
- Child Sum Property
- Convert Binary Tree to Doubly LinkedList
- Burning a Tree from Leaf
- Diameter
- LCA
- Serialize and Deserialize
- Count Nodes in Complete Binary Tree
- Video Solutions for some standard and complex problems
- More Problems for Practice

## Binary Search Tree

- Top View
- Bottom View
- Vertical Sum
- Vertical Traversal
- Fix BST With Two Nodes Swapped
- Check For BST
- Video Solutions for some standard and complex problems
- More Problems for Practice.

## Heap

- Heap Sort
- Video Solutions for some standard and complex problems
- More Problems for Practice.

## Graph

- Graph Representation: Adjacency List
- Adjacency List Implementation in CPP
- Adjacency List Implementation in Java
- Adjacency List and Matrix Comparison
- Breadth First Search and application
- Depth First Search and application
- Detect Cycle in Undirected Graph
- Detect Cycle in Directed Graph

- Topological Sorting
- Shortest Path Problems
- Prim's Algorithm Introduction and Implementation in CPP and Java
- Dijkstra's Algorithm Introduction and Implementation in CPP and Java
- Bellman Ford Algorithm
- Kosaraju's Algorithm
- Articulation Point
- Bridges in Graph
- Tarjan's Algorithm
- Video Solutions for some standard and complex problems
- More Problems for Practice.

## Greedy Algorithm

- Introduction
- Activity Selection Problem in CPP and Java
- Fractional Knapsack in CPP and Java
- Job Sequencing Problem
- Huffman Coding
- Video Solutions for some standard and complex problems
- More Problems for Practice.

## BackTracking

- Concept of Backtracking
- Problems: Rat In Maze, N Queen, Sudoku
- More Problems for Practice

## Dynamic Programming

- Introduction
- Memoization
- Tabulation
- LCS and its variations
- Coin Change
- KnapSack
- LIS and its variations
- Egg Drop Puzzle
- Subset Sum
- Matrix Chain Multiplication
- Palindrome Partitioning
- Video Solutions for some standard and complex problems
- More Problems for Practice.

## Trie

- Introduction
- Insert, Search, Delete
- Video Solutions for some standard and complex problems
- More Problems for Practice

## Segment Tree

- Introduction
- Construction
- Range and Update Query
- More Problems for Practice

## Disjoint-Set

- Introduction
- Union-Find
- Union By Rank
- Path Compression
- Kruskal's Algorithm
- More Problems for Practice

## DATA STRUCTURE (BASICS)

### • ANALYSIS OF ALGORITHMS

- Analysis of Algorithms (Background)
- Asymptotic Analysis
- Order of Growth
- Best, Average and Worst Cases
- Asymptotic Notations
- Big O Notation
- Omega Notation
- Theta Notation
- Analysis of Common Loops
- Analysis of multiple loops
- Analysis of Recursion
- Recursion Tree method for solving recurrences
- More example recurrences
- Upper bound using Recursion tree method
- Space Complexity

### • MATHEMATICS

- Python DSA - Count Digits
- Palindrome Number
- Factorial of a number
- Trailing Zeros in Factorial
- GCD and HCF of two numbers
- LCM of two numbers
- Check for Prime
- Prime Factors
- All Divisors of a Number
- Sieve of Eratosthenes
- Computing Power
- Iterative Power

### Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

- **LIST**

- List (Dynamic Sized Array) Introduction
- Working of List in Python
- Average or Mean of a List
- Separate Even and Odd
- Get Smaller Elements
- Slicing (List,Tuple And String)
- Comprehensions in Python
- Largest Element in a List
- Second Largest Element in a list
- Check if a list is Sorted
- Find the only Odd
- Reverse a List in Python
- Remove duplicates from sorted array
- Move Zeros to End
- Leaders in an Array problem
- Frequencies in a sorted array
- Left Rotate a List by one

## **Practice Problems**

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

## • RECURSION

- Applications of Recursion
- Writing Base Cases in Recursion
- Tail Recursion
- Practice For Recursion (Part 1)
- Practice For Recursion (Part 2)
- Print N to 1 using Recursion in Python
- Print 1 to N using Recursion in Python
- Sum of Natural Numbers Using Recursion
- Sum Of Digits Using Recursion
- Palindrome Check using Recursion

### Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

## • SEARCHING

- Binary Search in Python
- Recursive Binary Search in Python
- Analysis of Binary Search
- Index of first occurrence in a sorted array
- Index of Last Occurrence
- Count Occurrences in a Sorted Array
- Count 1s in a Sorted Binary Array

### Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

## • SORTING

- Sorting in Python
- List Sort in Python
- Sorted in Python
- Stability in Sorting Algorithm
- Bubble Sort
- Selection Sort
- Insertion Sort in Python
- Merge Sort Algorithm
- Merge Two Sorted Arrays
- Merge Subarrays

- Count inversions in Array
- Merge Sort Analysis
- Quick Sort Introduction
- Partition a Given Array
- Lomuto Partition
- Hoare's Partition
- Quick Sort using Lomuto Partition
- Quick Sort using Hoare's Partition
- Analysis of Quick Sort
- Space Analysis of Quick Sort
- Heap Sort

## **Practice Problems**

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

## **• HASHING**

- Introduction to Hashing
- Hashing Application
- Direct Address Table
- Hashing Functions
- Collision Handling
- Chaining
- Implementation of Chaining in Python
- Open Addressing
- Double Hashing
- Implementation of Open Addressing in Python
- Chaining vs Open Addressing
- Set in Python
- Dictionary in Python
- Count Distinct Elements in a List
- Frequencies of array elements

## **Practice Problems**

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned



## • STRING

- Strings in Python
- Escape Sequences and Raw Strings
- Formatted String in Python
- String Comparison in Python
- String Operations Part (1)
- String Operations Part (2)
- Reverse A String in Python
- Check if string is rotated
- Check For Palindrome In Python
- Check if a String is Subsequence of Other
- Check for Anagram in Python
- Leftmost Repeating Character
- Leftmost Non-Repeating Element
- Reverse words in a string

### Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

## • LINKED LIST

- Video - Problems with Array Data Structure
- Linked List Introduction in Python
- Simple Linked List Implementation in Python
- Applications of Linked List
- Traversing a Linked List in Python
- Search in Linked List
- Insert At The Bigenning of Linked list in Python
- Insert at The End Of Linked List
- Insert at Given Position in Singly Linked list
- Delete First Node Of Linked List in Python
- Delete Last Node of Linked List
- Delete a node with pointer given to it
- Sorted Insert Linked List in Python
- Middle of Linked List
- Nth Node From end of Linked List
- Remove duplicates from a sorted Singly Linked List
- Reverse a Linked List In Python
- Recursive Reverse A Linked List (Part 1)
- Recursive Reverse A Linked List (Part 2)

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

### • CIRCULAR LINKED LIST

- Circular Linked List in Python
- Circular Linked List (Advantages & Disadvantages)
- Circular Linked List traversal
- Insert at the Beginning of Circular Linked List
- Insert at The End of A Circular Linked List
- Delete Head of circular Linked List
- Delete Kth Node of Circular Linked List

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

### • DOUBLY LINKED LIST

- Doubly Linked List in Python
- Singly Vs Doubly Linked List (Advantages & Disadvantages)
- Insert at the Beginning of DLL in Python
- Insert at the End of DLL in Python
- Delete Head of A Doubly Linked List
- Delete Last Node of DLL in Python
- Reverse A Doubly Linked List in Python

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

### • STACK

- Stack Data Structure
- Stack in Python
- Linked List Implementation of Stack in Python
- Stack Applications
- Check for Balanced Parenthesis in Python

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

### • QUEUE

- Queue in Python
- Queue Data Structure
- Application of Queue Data structure
- Implementation of Queue using Array
- Linked List Implementation of Queue in Python

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

### • DEQUE

- Deque Introduction
- Deque Applications
- Deque in Python
- List Implementation of Deque in Python
- Linked List Implementation of Deque

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

### • TREE

- Tree Data Structure
- Application of Tree
- Binary Tree in Python
- Tree Traversal
- Inorder Traversal in Python
- Preorder Traversal in Python
- Postorder Traversal in Python
- Height of Binary Tree
- Print Node at K distance
- Level Order Traversal
- Size of Binary Tree in Python
- Maximum in Binary Tree

- Iterative Inorder Traversal
- Iterative Preorder Traversal
- Iterative Preorder Traversal (Space Optimized)

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

## • BINARY SEARCH TREE

- Binary Search Tree(Background)
- Binary Search Tree(Introduction)
- Search in BST in Python
- BST insert in Python
- BST Delete in Python
- Floor in BST (Problem and Solution Idea)
- BST Floor in Python
- Ceiling in BST in Python
- Self Balancing BST
- AVL Tree
- Red Black Tree
- Applications of BST

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

## • HEAP

- Binary Heap Introduction
- Heap Python Implementation (Introduction)
- Binary Heap Insert
- Binary Heap (Extract min and Heapify)
- Decrease Key and Delete Operations
- Build Heap
- Heap Sort
- Heapsort in Python

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

## DATA STRUCTURE - ADVANCED

### • BIT MAGIC

- Bitwise Operator in Python - Part 1
- Bitwise Operator in Python - Part 2
- Check Kth bit is set or not
- Count Set Bits
- Power of Two
- One Odd Occuring
- Two Odd Occuring
- Power Set Using Bitwise

#### Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

### • LIST

- Left Rotate by d Places
- Maximum difference
- Stock Buy & Sell Part 2
- Trapping Rainwater
- Maximum Consecutive 1s
- Longest even odd subarray
- Majority element
- Minimum Consecutive flips
- Sliding Window Technique
- Maximum subarray sum
- Maximum circular sum subarray
- Prefix Sum Technique

#### Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

### • RECURSION

- Subset of a given string
- Printing all Permutations
- Tower of Hanoi in Python
- Josephus Problem in Python
- Subset sum problem
- Rope Cutting Problem

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

### • SEARCHING

- Search in Sorted Rotated Array
- Median of two sorted arrays
- Repeating Elements Part (1)
- Repeating Elements Part (2)
- Allocate Minimum Pages (Naive Method)
- Allocate Minimum Pages (Binary Search)

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

### • SORTING

- Tail Call Elimination in Quick Sort
- Kth Smallest
- Minimum Difference in an Array
- Chocolate Distribution Problem
- Sort an array with two types of element
- Sort an array with three types of elements
- Merge overlapping intervals
- Meeting the maximum guests
- Counting Sort
- Cycle Sort
- Bucket Sort
- Radix Sort
- Overview of sorting algorithm

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

### • HASHING

- Intersection of two arrays
- Union of two unsorted arrays
- Pair with given sum in unsorted array

- Subarray with 0 sum in Python
- Check for Palindrome Permutation
- Subarray with given sum
- Longest Subarray with equal number of 0s and 1s
- Longest common span with same sum in binary array
- Longest Consecutive Subsequence
- Longest Subarray with given sum
- More than n/k Occurrences ( $O(nk)$  solution)

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

### • STRING

- Overview of Pattern Searching
- Pattern Searching in Python
- Naive Pattern Searching
- Improved Naive Pattern Searching for Distinct
- Rabin Karp Algorithm
- KMP Algorithm (Part 1 : Constructing LPS Array)
- KMP Algorithm (Part 2 : Complete Algorithm)
- Anagram Search
- Lexicographic rank of a String
- Longest Substring With Distinct Characters

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

### • LINKED LIST

- Reverse a linked list in groups of size k
- Detect loop using floyd's cycle detection algorithm
- Detect and remove loop in linked list
- Intersection Point of two linked list
- Segregate even odd nodes of linked list
- Pairwise swap nodes of linked list
- Clone a linked list using a random pointer
- LRU Cache Design
- Merge two sorted linked lists
- Palindrome Linked List

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

### • STACK

- Two stacks in an array
- K Stacks in an array
- Previous Greater Element
- Next Greater Element
- Stock span problem
- Largest Rectangular Area in a Histogram (Part 1)
- Largest Rectangular Area in a Histogram (Part 2)
- Largest Rectangle with all 1's
- Stack with getMin() in O(1)
- Design a stack with getMin() in O(1) space
- Infix to Postfix (Simple Solution)
- Infix to Postfix (Efficient Solution)
- Evaluation of Postfix
- Infix to Prefix (Simple Solution)
- Infix to Prefix (Efficient Solution)
- Evaluation of Prefix

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

### • QUEUE

- (Queue and Deque)
- Queue Implementation using Circular List
- Implementing stack using queue
- Reversing a Queue
- Generate numbers with given digits
- Design a data structure with min/max operations
- Maximums of all subarrays of size k
- First Circular Tour

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned



## • TREE

- Level Order Traversal by Line - Part 1
- Level Order Traversal by Line - Part 2
- Check for Balanced Binary Tree
- Maximum Width of Binary Tree
- Convert Binary Tree to Doubly Linked List
- Construct Binary Tree from Inorder and Preorder
- Tree Traversal in Spiral Form
- Diameter of a Binary Tree
- LCA of Binary Tree (Part 1)
- LCA of Binary Tree (Part 2)
- Burn a Binary Tree from a Leaf
- Count nodes in a Complete Binary Tree
- Serialize and Deserialize a Binary Tree

### Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

## • BINARY SEARCH TREE

- Ceiling on the left side in an array
- Find Kth Smallest in BST
- Check for BST
- Fix BST with Two Nodes Swapped
- Pair Sum with Given BST
- Vertical Sum in a Binary Tree
- Vertical Traversal of binary tree
- Top View of Binary Tree
- bottom view of binary tree

### Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

## • HEAP

- Sort K Sorted Array
- Purchase Maximum Items
- K Largest Elements
- K Closest Elements
- Merge K Sorted Arrays
- Median of a Stream

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

### • GRAPH

- Introduction to Graph
- Graph Representation (Adjacency Matrix)
- Graph Representation (Adjacency List)
- Graph Adjacency List Representation in Python
- Adjacency Matrix and List Comparison
- Breadth First Search in Python
- BFS for Disconnected Graph
- Connected Components in an Undirected Graph using BFS
- Applications of BFS
- Depth First Search
- DFS For Disconnected Graph
- Connected Components in an Undirected Graph using DFS
- Applications of DFS
- Shortest Path in an Unweighted Graph
- Detect Cycle in Undirected Graph
- Detect Cycle in a Directed Graph (Part 1)
- Topological Sorting (Kahn's BFS Based Algorithm)
- Detect Cycle in a Directed Graph (Part 2)
- Topological Sorting (DFS Based Algorithm)
- Shortest Path in DAG
- Prim's Algorithm/Minimum Spanning Tree
- Implementation of Prim's Algorithm
- Dijkstra's Shortest Path Algorithm
- Implementation of Dijkstra's Algorithm
- Kosaraju's Algorithm Part 1
- Kosaraju's Algorithm Part 2
- Bellman Ford Shortest Path Algorithm
- Articulation Point
- Bridges in Graph
- Tarjans Algorithm
- Kruskal's Algorithm

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

## • GREEDY

- Introduction to Greedy Algorithms
- Activity Selection Problem
- Activity selection
- Fractional Knapsack
- Fractional Knapsack in Python
- Job Sequencing Problem
- Huffman Coding (introduction)
- Huffman Algorithms
- Python Implementation of Huffman coding

### Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

## • BACKTRACKING

- Concept of backtracking
- Rat In a Maze
- N Queen Problem
- Sudoku Problem

### Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

## • DYNAMIC PROGRAMMING

- Introduction to DP
- Dynamic Programming Memoization
- Dynamic Programming Tabulation
- Longest Common Subsequence (Part 1)
- Longest Common Subsequence (Part 2)
- Variation of LCS
- Coin Change Count Combinations
- Edit Distance Problem
- Edit Distance Problem DP solution
- Longest Increasing Sub sequence Problem
- Longest Increasing Subsequence  $O(n \log n)$
- Variation of LIS (Part 1)
- Variations of LIS (Part 2)
- Maximum Cuts

- Minimum coins to make a value
- Minimum Jumps to reach the end
- 0-1 knapsack problem
- 0-1 knapsack problem DP Solution
- Optimal Strategy for a Game
- Egg Dropping Puzzle - Part 1
- Egg Dropping Puzzle - Part 2
- Count BSTs with n keys
- Maximum sum with no two consecutive
- Subset sum problem
- Subset Sum Problem (DP Solution)
- Matrix Chain Multiplication
- Matrix Chain Multiplication (DP Solution)
- Palindrome Partitioning
- Allocate Minimum Pages (Naive Method)
- Allocate Minimum Pages (DP Solution)

## **Practice Problems**

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

### **• TRIE**

- Trie Data Structure (Introduction)
- Trie (Representation, Search and Insert)
- Trie Delete
- Count Distinct Rows in a Binary Matrix

## **Practice Problems**

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

### **• SEGMENT AND BINARY INDEXED TREES**

- Segment Tree (Introduction)
- Constructing Segment Tree
- Range Query on Segment Tree
- Update Query on Segment Tree
- Binary Indexed Tree (Introduction)
- Binary Indexed Tree (An Example Problem)
- Binary Indexed Tree (Prefix Sum)
- Binary Indexed Tree (Prefix Sum Implementation)
- Binary Indexed Tree (Update Operation)

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

### • DISJOINT SET

- Disjoint Set Introduction
- Find and Union Operations on Disjoint Sets
- Union by Rank
- Path Compression
- Kruskal's Algorithm

## Practice Problems

This track contains many practice problems for the users which are considered important and must-do as far as Data Structure and Algorithm is concerned

## BASICS OF JAVASCRIPT

- Course Introduction
- Setup IDE - VS Code Installation
- First program - Hello Geeks
- JavaScript Variables
- Difference between var let and const
- Variable Naming Convention
- Data Types
- Concatenation and Template Literal
- Arithmetic Operators
- Type conversion
- ReadlineSync

## JAVASCRIPT FUNDAMENTALS

- Comparison Operators
- Conditional Statements
- Nested Conditional Statement
- Ternary Operator - Part 1
- Ternary Operator - Part 2
- Logical Operator Part 1
- Conditional Statement Exercise

- Logical Operator Part 2
- Logical Operator Part 3
- Nullish Coalescing
- Exercise - For Loop
- Nested For Loop
- Loops - for loop
- Loops - while loop
- Exercise - While Loop
- Error Handling - try and catch

## FUNCTIONAL PROGRAMMING

- Function declaration
- Anonymous Function & Function Expression
- Return and undefined
- Arrow Function
- Function Exercise

## STRING METHODS

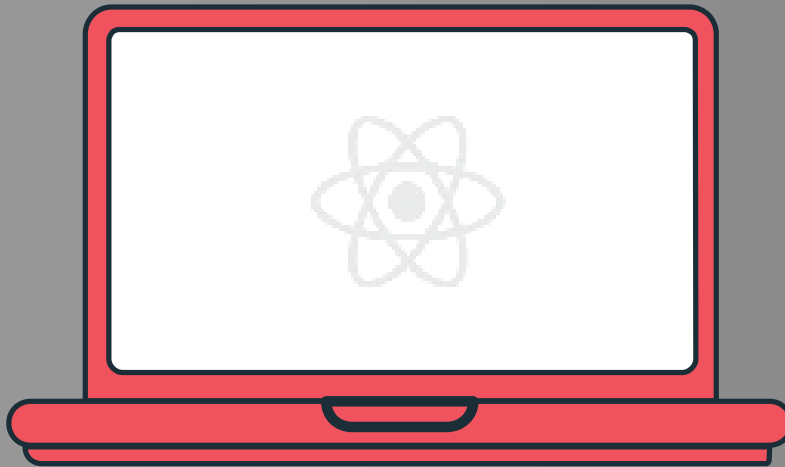
- Iterating over String
- String Method - charAt & charCodeAt
- String Method - indexOf
- String Method - includes
- String Method - toUpperCase & toLowerCase
- String Method - substring
- String Method - trim

## ARRAYS

- Iterating Over Array
- Copy By Reference
- Array Method - Push & Concat
- Array Method - pop, splice
- Array Method - includes
- Array Method - sort
- split and join
- Spread Array
- Destructuring Array

# Full Stack Development with React & Node JS

**LIVE** 



Detailed  
Course Syllabus

## SESSION 01: INTRODUCTION & BASICS OF VCS

- Introduction to Full Stack Development
- Community Bonding
- What and Why VCS?
- Intro to Git & Git Commands using Command Line Interface
- Git vs. GitHub
- Git WorkFlow & Contribution Guidelines
- Pull Requests in GitHub and Access to Source Code

## SESSION 02: BASICS OF FRONT END TECH

- Introduction to Front End
- Semantic HTML
- CSS Basics
- Working with Chrome DevTools
  - CSS Layouts: Display Property
  - CSS Layouts: Float Property
  - CSS Layouts: Position Property
- HTML Forms & Inputs
- Basics of Responsive Website Design
- Responsive Layouts & Mobile First Design with Flexbox

## SESSION 03: INTRODUCTION TO JAVASCRIPT

- Programming in JavaScript
- Variables and Data Types
- Functions
- Working with Strings & Numbers
- Application Logic & Loops
- Introduction to Arrays & Objects
  - Working with Arrays
  - Working with Objects
- Variable Types & Scoping
- ES 7 and Beyond New Features

## SESSION 04: MINI FACEBOOK PROJECT

- Introduction to Mini Facebook Project
- Motivation and Objectives
- Tech Stack
- Project Organisation & Plan
- Visual Design Walkthrough



## SESSION 05: INTRODUCTION TO REACT JS

- Setting up your Dev Environment
- Introduction to Node & NPM
- Create React App Boilerplate & Introduction to JSX
- Introduction to SCSS & Adding SCSS Support
- Organising Code & Creating a Hello World Application

## SESSION 06: REACT COMPONENTS AND STATES

- React Basics: Components with States & Props
- Class Components vs. Functional Components
- Passing Static & Dynamic Data between Components
- Routing Support using React Router
- Event Handling

## SESSION 07: WORKING WITH DATA

- Setting States and Props
- Introduction to AJAX & Asynchronous Calls
- Introduction to Axios & Service Calls
- Loading Data into State
- Fetching Data using API Calls
- Using Helper Functions

## SESSION 08: UNDERSTANDING API'S AND CRUD OPERATIONS

- Introduction to REST API
- Creating a Mock API Server
- Understanding CRUD & HTTP Verbs
- Implementing CRUD Endpoints

## SESSION 09: CONDITIONAL RENDERING

- Introduction to Conditional Rendering using React
- Handling Undefined and Unknown Data
- Completely Componentising Repeats
- Building the App based on Bootstrap
- Content Validation using React

## SESSION 10: NODE JS BASICS

- Introduction to Node JS & Express JS
- Basic Hello World REST API in Express JS
- Creating modular Routers in Express JS
- Using POST Variables with Body Parser
- Using in-memory storage to access data

## SESSION 11: INTRODUCTION TO REST APIS

- What are REST APIs
- Understanding the different HTTP Verbs and where to use them
- Setting up Routing
- Middleware and Error Handling
- Creating a Mock REST API

## SESSION 12: INTRODUCTION TO DATABASES

- Understanding Databases
- Bird's Eye View of different Databases
- Introduction to MongoDB
- Introduction to PostgreSQL
- Introduction to sqlite3

## SESSION 13: INTRODUCTION TO SESSION HANDLING

- What is User Session?
- Using a Session middleware
- Handling Sessions and Authentication
- Using NodeMailer to send out Emails
- Multifactor Authentication

## SESSION 14: SETTING UP OUR OWN REST API

- Using GET method to list and fetch a record
- Using POST method to create a record
- Using PUT & PATCH methods to update a record
- Using DELETE method to delete single and multiple records
- Access Control Allow Origin and CORS 6. Other interesting HTTP Verbs

## **SESSION 15: CONNECTING OUR REST API TO FRONTEND**

- Introduction to AJAX calls
- Using Fetch
- Using Axios and understanding difference between fetch and Axios
- Creating & implementing Service Calls
- Consuming service calls and updating components

## **SESSION 16: STYLING AND COMPONENTISATION**

- Implementing Routers
- Creating multiple pages
- Using Partial
- Adding Bootstrap Styles
- Creating Components and using in Pages
- Migrating Class Components to Functional Components
- Using Hooks
- useState and useEffect

## **SESSION 17: BUILDING THE HOME PAGE**

## **SESSION 18: BUILDING THE PROFILE & POSTS PAGE WITH CRUD**

## **SESSION 19: DEPLOYING AND HOSTING THE APP**

- Introduction to Deployment
- Introduction to CI & CD pipelines
- Looking at different deployment methods
- Deploying using GitHub Pages (React JS)
- Deploying using Netlify (Node JS)
- Other deployment methods

## **SESSION 20: HEADS UP FOR TYPESCRIPT & GRAPHQL**

- Basics of TypeScript
- Where and how to use them?
- Basics of GraphQL
- Use-cases of GraphQL
- Related technologies and next steps

# JAVA

## Backend Development

**LIVE**



Detailed  
Course Syllabus

## Week 01

### Session 01 Java OOPS Fundamentals

- Understand the fundamentals of Java OOPS concepts like Object, Class, Inheritance, Polymorphism, Abstraction, and Encapsulation
- Learn how to handle exceptions in Java
- Work with Java Collections for efficient data management

### Session 02 Java 8 Functional Interfaces

- Master the Singleton Design Pattern
- Learn the differences between Abstract Classes and Interfaces with practical examples
- Explore Functional Interfaces, Lambda Expressions
- Use Generics and Streams to write efficient code

## Week 02

### Session 03 Multithreading & HashMap

- Delve into the workings of HashMap
- Understand the concepts of Multithreading, such as Thread creation, Thread Groups, and Thread Join
- Learn to differentiate between Sequential and Parallel Streams for effective task execution

### Session 04 Maven for Project Management

- Understand the need for Maven
- Learn to work with POM.xml, explore different Maven Repositories and their types
- Understand the Maven Lifecycle for efficient project management

## Week 03

### Session 05 Spring Boot Basics

- Learn the basics of Server and Client models
- Introduction to Spring Boot
- How to run application as a Server
- Understand Embedded Servers like Jetty and Tomcat
- Manage Logging Levels in Spring Boot
- Work with Spring profiles and terminal commands

## Session 06 REST API & Spring MVC

- Gain knowledge about REST API, HTTP Requests and Responses
- learn to work with POSTMAN and CURL for API testing
- Understand Annotations and Lombok
- Explore the Spring MVC framework

### Week 04

## Session 07 Spring IOC & Dependency Injection

- Learn about the Spring IOC container, Dependency Injection, and Enums
- Understand the target of an Annotation
- Configure Beans using @Configuration and @Bean annotations

## Session 08 Java Database Connectivity (JDBC)

- Understand the differences between In-Memory and Disk Storage
- Learn to connect a Spring Boot application with a Database Server
- Create Request Classes
- Perform validations using JDBC

### Week 05

## Session 09 JPA & Hibernate

- Understand the need for an abstraction layer between DAO and Database
- Learn about JPA (Java Persistence API), Hibernate, Entity Classes, Annotations, JPA Repository, and ResponseEntity

## Session 10 JPQL & Minor Project 1

- Explore custom queries using JPQL (Java Persistence Query Language)
- Learn about relationships in JPA
- Work on a minor project and create a project flowchart

### Week 06

## Session 11 Minor Project 1 (Continued)

- Continue working on Minor Project 1
- Understand project HLSD
- Data Modelling concepts

## **Session 12**    **Unit Testing with JUnit & Mockito**

- Learn to change the path of the local repository (.m2)
- Parse CSV files with Spring Boot
- Understand the importance of Unit Testing with JUnit and Mockito

### **Week 07**

## **Session 13**    **Redis & Caching**

- Get introduced to Redis
- Learn the differences between Cache and Cookie
- Understand Server Cache vs Browser Cache
- Work with Local Redis Server
- Online Centralized Redis Server for efficient caching

## **Session 14**    **Spring Security & Authentication**

- Learn about Spring Security and its terminologies
- Perform Basic Authentication using System Generated Credentials
- Understand Authorization with In-Memory
- Database user Authentication

### **Week 08**

## **Session 15**    **Minor Project 2**

- Work on Minor Project 2
- Gain insights into project overview, project HLSD, and project Data Modelling

## **Session 16**    **OAuth 2 & Github Integration**

- Introduction to OAuth2
- OAuth 2.0 concepts
- Learn the workflow of OAuth 2.0
- Explore Scopes and Consent
- Implement Github OAuth2 integration with Spring Boot

## Week 09

### **Session 17**    **Kafka Message Queue Introduction**

- Learn about Message Queues and their types
- Types of Message Queues
- Get introduced to Kafka Message Queue for efficient messaging systems

### **Session 18**    **Kafka Integration with Spring Boot**

- Integrate Kafka with Spring Boot
- Learn about the Consumers and Producers Model
- Understand Kafka Topics and Events for effective message handling

## Week 10

### **Session 19**    **Major Project Part 1**

- Start working on major project
- Project Overview
- Project HLSD
- Project Data Modelling

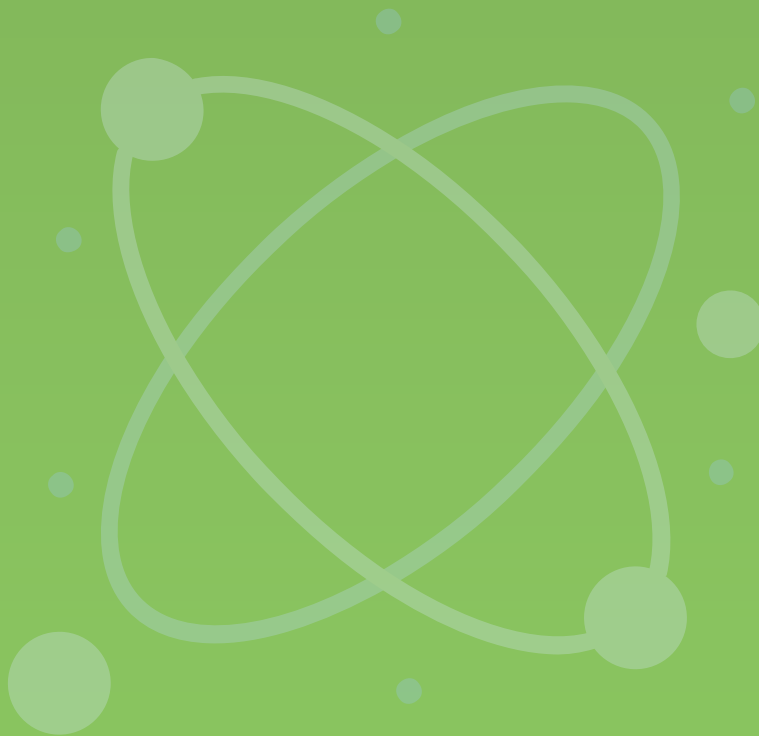
### **Session 20**    **Major Project Part 2**

- Major Project Continued
- Project Queries
- Career Guidance



# Data Science Training Program

**LIVE**



Detailed  
Course Syllabus

## Week 1: Introduction to Data Science and Machine Learning

### Day 1: Introduction to Data Science and Machine Learning

- Overview of Data Science
- Fundamentals of Machine Learning
- Types of Machine Learning Algorithms
- Applications of Data Science and Machine Learning
- Introduction to Python programming language
- Setting up Python environment
- Running the first Python program

### Day 2: Introduction to Python

- Basic syntax and data types in Python
- Working with variables and operators
- Using built-in functions and libraries

### Day 3: Data Manipulation with Python

- Lists, tuples, and dictionaries
- Conditional statements and loops
- User-defined functions
- Functions such as map, filter, lambda

## Week 2: Working on Data with Python

### Day 1: Data Cleaning and Preprocessing with Pandas

- Import and Exporting data

- Handling missing data
- Removing duplicates and dealing with Outliers
- Cleaning and adjustments in data

## **Day 2: Exploratory Data Analysis (EDA) with Pandas**

- Descriptive Statistics and data summarization
- Grouping and Aggregating data
- SQL-like operation in data

## **Day 3: Data Visualization with Matplotlib**

- Creating Basic Plots (line plots, scatter plots, histograms)
- Customizing and Styling Visualizations
- Creating Informative and Aesthetically Pleasing Visualizations
- Pair plots, Heatmaps, and Advanced Plotting technique

## **Week 3: Data analysis using Python**

### **Day 1: Knowing about the data**

- Types of analysis - univariate, Multivariate analysis
- Creating Hypothesis of Project Approach
- Introduction to Statistical tests and Hypothesis testing
- Implementing Statistical tests in Python

### **Day 2: Final Project and Case Studies**

- Participants work on a real-world data analysis project
- Applying learned Python skills to analyze and visualize data

## Day 3: Data Preprocessing and Exploration

- Feature Engineering / Selection

## Week 4: Introduction to math in data science

### Day 1: Probability and Statistics

- Basic Probability Concepts
- Random Variables
- Probability Distributions
- Statistical Inference
- Hypothesis Testing
- Regression Analysis

### Day 2: Statistical Analysis with Python

- Measures of central tendency and dispersion
- Hypothesis testing and confidence intervals
- Correlation and regression analysis
- ANOVA and chi-square tests

### Day 3: Statistical Analysis with Scipy

- Introduction to statistical tests and hypothesis testing
- Implementing statistical tests in Python

## Week 5: Starting with Machine Learning - Supervised Learning

### Day 1: Supervised learning

- Classification
- Naive Bayes , KNN Classifier- with Practical

## Day 2: Decision tree, Logistic regression

- Decision tree, Logistic Regression- with Practical

## Day 3: SVM

- SVM with Practical

## Week 6: Classification and Regression

### Day 1: Linear regression

- Linear regression - Practical

### Day 2: Types of linear regression

- Lasso & Ridge

### Day 3: Non-linear regression

- Non-linear regression

## Week 7: Unsupervised learning model

### Day 1: Types of Clustering model

- Types of clustering model and evaluation

### Day 2: K-means algo + Agglomerative

- K-means algo

## Day 3: Associative rules

- Market Basket analysis

## Week 8: Advanced models and Optimization

### Day 1: Ensemble Models

- Ensemble Techniques - Stacking, Boosting and Bagging

### Day 2: Ensemble models

- Random forest, XGBoost Classifier, and Regressor

### Day 3: Hyperparameter Tuning

- Hyperparameter tuning

## Week 9: Model Evaluation, Optimization and Validation

### Day 1: Cross-Validation

- Types of cross-validation

### Day 2: Hyperparameter Tuning

- Hyperparameter Tuning

### Day 3: Model Selection Method

- Model selection method

## Week 10: Introduction to Advanced Topics in Machine Learning

## Day 1: Advance Models

- Reinforcement Learning
- Time Series Analysis

## Day 2: Advance models

- Anomaly Detection
- Recommendation Systems

## Day 3: Computer Vision

- Operation on Images
- Working with Image Classification

## Week 11: Introduction to Advanced Topics in Machine Learning

### Day 1: Natural Language Processing

- Text based operations
- Working on text data

### Day 2: Deep Learning

- Neuron, Neural Network and why DNN

### Day 3: Deep Learning

- ANN, CNN, RNN, TRANSFER LEARNING

## Week 12: Projects

### Day 1: Case Studies and Projects

- Real-life Applications of Machine Learning
- Project Frameworks
- Project Implementation and Execution
- Project Presentation and Documentation
- Project Covered: Predict the income of an individual based on its social and financial attributes – supervised learning

## **Day 2: Project Covered:**

- Market Basket Analysis – unsupervised learning

## **Day 3: Deployment of model using Streamlit**

- Deployment of model using Streamlit