

Milestone	Module	Topic	Sub Topic
DSA Sem 1	C --> CPP	Basics1	
		Basics2	
	Intro to Algorithms, TC and SC	Intro to Algorithms and Optimizations	DS/Algo Intro with Dijktras
			Factors Count Optimisation
		TC and SC Complexity 1	Calculating Iterations
			Comparing 2 Algorithms
			How to Calculate Big O
		TC and SC Complexity 2	Asymptotic Analysis - Big O
			Issue in BigO
			Importance of Constraints
			Why TLE Occurs ?
		Amortized Analysis	What is Amortized analysis?
			Types of Amortized Analysis
			Amortized Complexity Examples
	Array Basics	Simple Array 1	Count no of elements with atleast 1 ele > itself
			Pair sum = k
			Vector Intro, Pass by value and passby reference
		Simple Array 2	Rotate arr[] by k times and optimisation
			Count no of distinct elements
			Arrays of Vectors or Vector of Vectors
	Array Optimization Techniques	CarryForward 1	Leaders in Array
			Buy and Sell Stocks
			Ag Pairs
		CarryForward 2	Equilibrium Index
			Count GFG Pairs
			Count Triplets $i < j < k \ \&\& \ arr[i] < arr[j] < arr[k]$
		Precomputation : Pf Sums	Range Sum Queries
			0-1 Prefix Sum
		Subarrays 1 : Intro and Kadanes	Print all subarrays
			Print all subarray sums
			Print max subarray sum
		Subarrays 2 : Contribution and Sliding Window	Sum of all subarray sums
			Max subarray sum of len = k
		2D Mat : 1	Print row wise and column wise sum
			Identify matrix
			Diagonal Printing
		2D Mat : 2	Transpose
			Rotate 90 degrees
			Spiral Printing
		2D Mat : 3	Matrix multiplication
			Make zero

	Bit Manipulations	Bit Manipulation 1	Introduction to Number System
			Binary to Decimal and Decimal to Binary
			Addition of Binary Numbers
			Bitwise Operations
			Negative Numbers
			Ranges
			Importance of Constraints
		Bit Manipulation 2	Bit Wise Properties
			Single Number
			Left Shift
			Right Shift
			Basic Problems a. Check bit b. Count Set Bits c. Toggle Bit d. Set Bit
		Bit Manipulation 3	Single Element 1 (Every element 2 times and one unique element)
			Single Element 2 (Every element 3 times and one unique element)
			Single Number 3 (Every element 2 times and 2 unique elements).
		Bit Manipulation 4	Min XOR Pair
			Max AND Pair
		Bit Manipulation 5	Subsets and Subsequences Intro
			Check if there exists a subset with sum = k
		Bit Manipulation 6	Fast exponentiation using powers
			Sum of xor of all pairs
	Maths	Maths 1	% Modular Arithmetic
			Pairs with $(a+b)\%M = 0$
		Maths 2 : % Modular Arithmetic	For every number from 1 to N get frequency
			Prime or Not ?
			Prime from 1 to P using Seive
		Maths 3 : Primes	Count of factors for all numbers from 1 to P
			Segemented Sieve [If needed]
		Maths 4 : GCD	Calculate GCD
			Check if there exists a subsequence with GCD 1
			Remove an elements so that we get max GCD
	HashMaps/HashSets	Hashing 1 : Intro	Intro to HashMap and HashSet, Functions
			Given queries, find frequency of an element
			First repeating element in arr[]
			No. of Distinct Elements, <b>Teach multi set as well</b>
			Iterate on HashMap
		Hashing 2 : Subarray	Check if there exists a subarray with sum = 0
			Length of longest subarray with sum = 0
			Longest subarrays with equal 1's and 0's
		Hashing 3 : Counting Pairs	Count of Pair Sum

			Count of subarray with given sum.
		Hashing 4 : Interview Questions 1	Number of Distinct Character in every window of size K
			Longest Consecutive Sequence
Recursion	Recurion 1 : Intro and Basic Problems		Basics of Stack
			Function/ Call Stack
			Recursion Basics + 3 Steps of Recursion
			Sum of N numbers
			Factorial of a number
			Print 1 to N
			Print N to 1
			Print N to 1,1 to N
		Recurion 2 : More Problems	Sum of Digits of N
			Print 1 2 2 3 3 3 4 4 4 4 ...
			Fibonacci
			Power Function. 1. Pow(a,n) = Pow(a,n-1) * a 2. Pow(a,n) = Pow(a,n/2) * Pow(a,n.2)
Sorting	Recurion 3 : TC and SC		TC : Way1 : Based on no of function calls & SC
			TC : Way2 : Using recursive relation
		Recurion 4 : Towers of Hanoi	Towers of Hanoi
			Print arr[] using recursion
			Passing arr[]/Vector in C++
		Sorting 1	Bubble Sort
			Selection Sort
			Insertion Sort
			Merge 2 Sorted Arrays
			Merge 2 Sorted Subarrays in the same array
Searching & 2 Pointers	Searching 1 : In sorted arr[]	Sorting 2	Merge Sort
			Inversion Count
		Sorting 3	Count Sort
			Problems on count sort
		Sorting 4	Overriding Comperator and its problem
			Largest Element
			Sorting elements based on factors
			Search for the element K in sorted array.
			Search for floor of ele k in sorted arr[]
			Search first & last K in sorted array with repeated elements.
Searching & 2 Pointers	Searching 2 : In arr[]		Search in 2D sorted matrix
		Searching 2 : In arr[]	Search Peak Element.
			Single Element in a arr[] where every eleents repeats twice except 1
		Searching 3 : In Imaginary Search Space	Square root
Searching & 2 Pointers	Searching 4 : In Imaginary Search Spac		Painter's Partition Problem
		Searching 4 : In Imaginary Search Spac	Aggressive Cows

			When to apply Binary Search
		2 Pointers 1 : Intro and Basic questions	Pair sum = k
			Pair difference = k
			Closest pair with sum = k
		2 Pointers 2 : Interview questions 1	Minimize the triplet max - min diff
			Container with most water.
		2 Pointers 3 : Interview questions 2	Max consecutive 1's with atmost k flips
			Search k in a row-wise sorted matrix
		2 Pointers 3 : Interview questions 3	Increasing window problems
			Decreasing window problems
		Meet in the Middle	Count no of quadplets with sum = k
			More question on this technique
Strings	Basic Strings		First repeating character
			String concatenation TC
			More basic questions on strings
	Strings Interview Questions		Length of longest palindromic substring
			Length of longest substring with all distinct characters
Pattern Matching	String Pattern Matching 1		Either RabinKarp or KMP
	String Pattern Matching 2		Problems on Pattern Matching 1
	String Pattern Matching 3		Problems on Pattern Matching 2
Linked List	Linked List 1		Class and Object Basics
			Create/ Search / Insert if time permits
	Linked List 2		Insert x at position p
			Insert x in a sorted list
			Delete x in linked list
			Delete all occurrences of x
	Linked List 3		Reverse Linked List
			Middle of a linked list
			Check if given linked list is palindrome or not
	Linked List 4		Check if given linked list has cycle.
			Find Intersection of 2 Linked Lists
Advance Topics	Advanced Arrays 1		Continuous Sum Query
			Rain water trapped Using PrefixMax and Suffix Max
	Advanced Arrays 2		Product array puzzle Using PrexProd and Suffix Prod
			Majority Elements
	Advanced Arrays 3		Min swaps required to bring all elements $\leq k$ together
			First missing +ve Integer
	Advanced Arrays 4		Overlapping Intervals
			Insert in overlapping Intervals
			Next Permutation
	Advanced Arrays 5		$ arr[i] - arr[j]  +  i-j $ maxime

			Wave array
		Advanced Arrays 6	Submatrix sum query
			Max submatrix sum in row sorted matrix
		Advanced Sorting	Quick Sort
		Advanced Hashing	No of distinct pairs
			No of right angles triangles
			No of rectangles parallel to x-axis and y axis [If timer permits]
		Advanced TreeMap	TreeMap Introduction
			For every query get the nearest 1
		Advanced Searching 1	Search in rotated sorted arr[]
			Get 1st missing +ve integer
		Advanced Searching 2	Kth index element in arr[]
			Median of 2 sorted arr[]
			Median of N sorted arr[]
		Advanced Searching 3	Median of 2 sorted array in log(N+M) approach
		Advanced Strings	Check if 2 strings are anagrams of each other
			Length of smallest substring of B which contain all characters of A
			Count of Anagrams of A as substrings in B
		Advanced LinkedList 1	Doubly Linked List Introduction
			Insert a Node in DLL
			LRU Cache
		Advanced LinkedList 2	Reverse LinkedList in groups of size k
			Merge Two Sorted List.
			Merge Sort LinkedList
		Advanced LinkedList 3	Merge N Sorted Linked Lists
			Flatten Linked List
			Clone Linked List
Stacks and Queues	Stacks 1		Stack Intro and Implementation
			Balanced Parenthesis
			Double Character Trouble
	Stacks 2		Evaluate Postfix Expression
			Infix to Postfix Expression
		Stacks 3	Nearest Smaller & Greater Element on Left & Right.
	Queue 1		Largest Rectangle in a Histogram.
			Queue Introduction and implementation
			Reverse queue using stack
			Implement queue using stacks