TRAINING STRUCTURE
II YEAR ENGINEERING STUDENTS

CATEGORY	DETAILS
PROGRAM TITLE	**TechXcelerate Training Program**
PROGRAM DURATION	30 Days
BATCH COMPOSITION	2 Batches
BATCH 1	CSE Branch – Core Java + DSA + Problem Solving
BATCH 2	Circuit Branches – Java Fundamentals + DSA Basics + Problem Solving
TOTAL DURATION	30 Days
SLOT 1	18 Days • First 6 days – Full-day sessions (continuous) • Next 12 days – Half-day sessions
SLOT 2	12 Days - Stretch Training
OBJECTIVE	To build strong programming foundations, logical reasoning, and problem-solving skills among 2nd year students through practical learning.

TECHXCELERATE TRAINING PROGRAM

II YEAR ENGINEERING STUDENTS - CIRCUIT

SLOT 01 - 6 DAYS(6Hrs/day)

DAYS	TOPICS COVERED	ASSESSMENTS
DAY 1	Java Fundamentals - Basics to OOP Foundation - Data types, Variables, Operators, Control Flow, Arrays (1D & 2D), Methods & method parameters, Introduction to OOP (Classes, Objects, Constructors), Instance vs Static members, Access modifiers	1. Write 5 programs covering all data types and operators 2. Create a class with multiple constructors (3 variations) 3. Debug 4 programs with scope and access modifier issues 4. Design problem: Model a real-world entity using classes
DAY 2	Object-Oriented Programming - Core Concepts - Encapsulation (Getters/Setters), Inheritance (Single & Multi-level), Method Overriding, Polymorphism (Method overloading), Super keyword, This keyword, Object class methods (toString, equals, hashCode)	1. Create 2 inheritance hierarchies with proper encapsulation 2. Implement method overloading (minimum 4 methods) 3. Override Object class methods (3 programs) 4. Fix 5 programs with OOP principle violations
DAY 3	Advanced OOP & Exception Handling - Abstraction (Abstract Classes & Interfaces), Exception types, Try-catch-finally, Try-with-resources, Custom exceptions, Exception propagation, Throws keyword, Generics introduction (Wildcards, Type parameters)	1. Design abstract class hierarchies for 2 real-world scenarios 2. Create custom exception classes (3 variations) 3. Write 4 programs with proper exception handling 4. Practice: Implement generic methods (2 examples)
DAY 4	Collections Framework Fundamentals - Part 1 - List interface & implementations (ArrayList, LinkedList), Iteration methods (Iterator, forEach), Comparable interface, Basic Comparator usage, Sorting lists of objects, Common List operations	1. Create ArrayList programs for data management (3 scenarios) 2. Compare LinkedList performance vs ArrayList 3. Sort custom objects using Comparable (2 examples) 4. Solve list manipulation problems (5 programs)
DAY 5	Collections Framework Fundamentals - Part 2 - Set interface & implementations (HashSet, TreeSet), Map interface & implementations (HashMap, TreeMap), Hashtable basics, Iteration over Collections, Common operations, Choosing appropriate collection types	1. Use HashSet to solve duplicate problems (3 scenarios) 2. Implement frequency counting using HashMap (2 variations) 3. Compare performance: HashSet vs TreeSet 4. Design real-world scenarios using appropriate collections
DAY 6	Data Structures & Algorithms Basics - Algorithm Analysis (Big O notation), Arrays & Linked Lists operations, Linear & Binary Search, Introduction to sorting, Bubble & Selection Sort, Introduction to recursion, Recursive problems basics	1. Analyze complexity of 6 code snippets 2. Implement Linear & Binary Search with variations 3. Code 3 sorting algorithms and test them 4. Solve 5 beginner-level recursive problems

TECHXCELERATE TRAINING PROGRAM

II YEAR ENGINEERING STUDENTS - CIRCUIT

SLOT 01 - 12 DAYS(3Hrs/day)-Half Day

DAYS	TOPICS COVERED	ASSESSMENTS
DAY 1 & 2	Sorting & Searching Algorithms (3 hrs) - Bubble Sort, Selection Sort, Insertion Sort, Merge Sort, Quick Sort, Counting Sort, Radix Sort, Comparison of sorting algorithmsSearching Techniques (3 hrs) - Linear Search, Binary Search, Ternary Search, Interpolation Search, Jump Search, Binary Search variations & applications	1. Implement all 7 sorting algorithms 2. Analyze time complexity for each sort 3. Implement all 6 searching algorithms 4. Solve 8 sorting & searching problems 5. Performance comparison test on 100K elements
DAY 3 & 4	Data Structures - Stacks & Queues Basics (3 hrs) - Stack implementation (Array & LinkedList), Stack applications (Simple expression evaluation, Undo-Redo), Queue implementation (Array & LinkedList), Circular QueueLinked Lists Introduction (3 hrs) - Singly LinkedList, Doubly LinkedList, Circular LinkedList, LinkedList operations (Insert, Delete, Reverse), LinkedList problems	1. Implement Stack & Queue using both approaches 2. Solve 5 stack-based problems 3. Implement all 3 LinkedList types 4. Solve LinkedList problems (5 scenarios) 5. Compare performance: Array-based vs LinkedList-based structures
DAY 5 & 6	Binary Trees Basics (3 hrs) - Tree terminology, Binary Trees, Tree traversals (Inorder, Preorder, Postorder), Level-order traversal, Tree height & depth, Binary Search Trees fundamentalsBST Operations & Introduction to Graphs (3 hrs) - BST insertion & deletion, BST search, Balanced BST concept, Graph basics (Representation, DFS, BFS)	1. Implement Binary Tree with all traversals 2. Implement Binary Search Tree operations 3. Solve 6 tree-based problems 4. Implement graph representation (2 types) 5. Solve basic DFS & BFS problems (4 scenarios)
DAY 7 & 8	Graph Algorithms - Part 1 (3 hrs) - DFS detailed, BFS detailed, Cycle detection, Topological sorting, Connected componentsGraph Algorithms - Part 2 (3 hrs) - Shortest path (Dijkstra, BFS for unweighted), Minimum Spanning Tree (Kruskal, Prim), Connectivity problems	1. Implement DFS & BFS variants 2. Solve 6 graph traversal problems 3. Implement Dijkstra & MST algorithms 4. Solve real-world graph problems (4 scenarios) 5. Performance analysis of graph algorithms
DAY 9 & 10	Introduction to Dynamic Programming (3 hrs) - Recursion revisited, Overlapping subproblems, Optimal substructure, Memoization concept, Tabulation concept, Classic problems (Fibonacci, Factorial)Dynamic Programming Easy & Medium (3 hrs) - 0/1 Knapsack, Coin Change, Longest Common Subsequence, Longest Increasing Subsequence, Climbing Stairs, House Robber	1. Solve 10 classic DP problems 2. Convert recursion to memoization (3 problems) 3. Convert memoization to tabulation (3 problems) 4. Solve real-world DP problems (4 scenarios) 5. Practice: Mix of 15 DP problems
DAY 11 & 12	Advanced DP & Problem Solving Strategies (3 hrs) - 2D DP, DP on Grids (Path counting, Minimum path sum), Matrix Chain Multiplication, Edit Distance, Wildcard MatchingCompetitive Programming Basics & Comprehensive Review (3 hrs) - Time management in contests, Common pitfalls, Optimization techniques, Problem-solving approach, Comprehensive DSA revision	1. Solve 8 advanced DP problems 2. Practice 2D grid DP (5 problems) 3. Final Mock Test: 30 mixed DSA problems 4. Time-based contest simulation (2 hours) 5. One-on-one code reviews & optimization suggestions

TECHXCELERATE TRAINING PROGRAM

II YEAR ENGINEERING STUDENTS - CIRCUIT

SLOT 02 - 12 DAYS

DAYS	TOPICS COVERED	ASSESSMENTS
DAY 1	Advanced Problem Solving & Bit Manipulation - Bit fundamentals review, Bit manipulation tricks, Number system conversions, Bit DP introductionString Algorithms Introduction - String matching (Basic to KMP), Pattern matching, String transformations, Anagram & permutation problems	1. Solve 12 bit manipulation problems |2. Practice 8 string manipulation problems 3. Implement string matching algorithms (2 types) 4. Solve anagram & permutation problems (5 scenarios) 5. MCQ: 20 questions on bits & strings
DAY 2	Sliding Window & Two Pointers Consolidated - Sliding window patterns (Fixed, Variable size), Two pointers (Opposite ends, Same direction), Optimization using these techniquesIntroduction to System Design Thinking - Scalability concepts, Design patterns introduction, Trade-offs in design	1. Solve 12 sliding window problems 2. Solve 10 two-pointer problems 3. Optimize 5 brute-force solutions using these techniques 4. Design a simple system (LRU cache) 5. Code review: 4 optimized solutions
DAY 3	Recursion & Backtracking Fundamentals Revised - Recursion optimization, Tail recursion, Backtracking strategy, Common backtracking patternsRecursion Applications - Permutations, Combinations, Subsets, Partition problems	1. Solve 10 recursion problems with complexity analysis 2. Implement 5 backtracking solutions 3. Generate permutations & combinations (3 variations) 4. Solve partition & subset problems (6 scenarios) 5. Practice: 15 recursion & backtracking problems
DAY 4	Mathematical Algorithms & Modular Arithmetic - Prime numbers (Sieve, Primality testing), GCD/LCM, Modular arithmetic, Factorial & permutation moduloBasic Number Theory - Divisors, Euler's totient, Fermat's Little Theorem basics	1. Implement Sieve of Eratosthenes 2. Solve 10 prime & number theory problems 3. Solve modular arithmetic problems (8 scenarios) 4. Optimize factorial calculations with modulo 5. Practice: 20 mathematical algorithm problems
DAY 5	Trie & Tree Advanced Topics - Trie structure review, Trie applications (Autocomplete, Dictionary), Tree DP introductionAdvanced Tree Problems - LCA basics, Path sum problems, Tree serialization, Balanced tree concepts	1. Implement Trie structure with operations 2. Solve 8 Trie problems 3. Implement LCA algorithm 4. Solve 6 tree path problems 5. Tree serialization practice (3 formats)
DAY 6	Disjoint Set Union & Graph Union Basics - Union-Find fundamentals (Already learned, advanced revision), DSU applications, Cycle detectionAdvanced Graph Algorithms - Cycle detection in directed graphs, Bipartite checking, Connected components	1. Implement DSU with optimizations 2. Solve 10 DSU problems 3. Implement cycle detection (2 approaches) 4. Bipartite graph detection (3 scenarios) 5. Find connected components: 6 problems

TECHXCELERATE TRAINING PROGRAM

II YEAR ENGINEERING STUDENTS - CIRCUIT

SLOT 02 - 12 DAYS

DAYS	TOPICS COVERED	ASSESSMENTS
DAY 7	Topological Sorting & DAG Processing - Topological sort (DFS & BFS method), Longest/shortest path in DAG, Dependency resolutionScheduling & Activity Selection Problems - Task scheduling, Activity selection with constraints	1. Implement topological sort (2 methods) 2. Solve 8 topological sort problems 3. Find longest/shortest path in DAG (5 problems) 4. Solve scheduling problems (4 scenarios) 5. Practice: 12 DAG-based problems
DAY 8	Dynamic Programming Comprehensive Review - All DP patterns learned consolidated, 2D DP on grids, DP on Strings, Optimization techniquesProblem Solving on Mixed Topics - Combine multiple DSA concepts	1. Solve 12 DP problems (Mixed complexity) 2. Implement 4 complex DP solutions 3. Optimize DP for space (4 problems) 4. Convert between memoization & tabulation 5. Practice: 20 DP variations
DAY 9	Matrix & Grid Problems - Matrix traversal (All patterns), Rotations, Medians, Matrix DPSpatial Algorithms Introduction - Distance problems, Grid transformations	1. Solve 10 matrix problem variations 2. Implement matrix rotations (All types) 3. Solve matrix DP problems (5 scenarios) 4. Grid path & distance problems (6 problems) 5. Practice: Mixed matrix problems (15 problems)
DAY 10	Competitive Programming Techniques & Optimization - Common patterns in contests, Optimization strategies, Time management, Debugging techniquesCode Templates & Problem Classification - Common algorithm templates, Problem-solving framework	1. Analyze 6 time-limited problems 2. Create personal code templates 3. Practice time-bound contests (2 hours) 4. Solve contest problems with optimal solutions (10 problems) 5. Peer code review (2 solutions)
DAY 11	Final Review & Assessment Preparation - Comprehensive revision of all topics, Common mistakes review, Interview tipsMock Assessment & Feedback - Practice tests based on company patterns (TCS, Infosys, Cognizant)	1. Comprehensive test: 30 problems (All topics) in 3 hours 2. Performance analysis & weak areas identification 3. Targeted problem solving on weak areas (15 problems) 4. Mock interview: 1 round (45 minutes) 5. Individual feedback & improvement plan
DAY 12	Final Project & Viva - Mini project: Simple system design using DSA (Library management, Task management)Viva & Q&A Session - Concept clarity on difficult topics, Approach discussion, Interview preparation final tips	1. Mini project implementation & documentation 2. Project code walkthrough 3. Viva voce: Concept clarity verification 4. Final mock interview: System design + Algorithm problem 5. Career guidance & further learning roadmap 6. Certificate handover & course completion assessment

PROGRAM OUTCOMES

- Core OOP Principles & Practical Application - Master object-oriented concepts and apply them effectively in real-world projects with proper design patterns.
- Collections Framework Mastery & Practical Usage - Understand all Collection implementations and choose appropriate types for specific use cases based on performance needs.
- Data Structure Fundamentals & Implementation - Implement core data structures and understand their trade-offs for practical problem-solving scenarios.
- Standard Algorithm Knowledge & Application - Learn common algorithms (Sorting, Searching, DFS, BFS, Dijkstra's, basic DP) with proper complexity analysis and selection criteria.
- Problem-Solving Framework Development - Develop systematic approach to problem-solving: understand → identify structure → apply algorithm → optimize → validate.
- Service & Product Company Interview Preparation - Successfully clear technical interviews at TCS, Infosys, Cognizant, Accenture with master knowledge of commonly asked DSA problems.
- Real-World Project Development Skills - Build practical projects (Library Management, Task Management, File Systems) applying learned concepts end-to-end.
- Debugging & Code Optimization Abilities - Systematically identify and fix bugs, optimize inefficient code, and understand performance bottlenecks.
- Competitive Problem-Solving Introduction - Solve standard coding problems and participate in beginner to intermediate level coding contests confidently.
- Career Development & Continuous Learning Mindset - Build strong foundation for software development career with continuous learning habits and understanding of DSA's importance in software engineering.