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 - CSE

SLOT 01 - 6 DAYS(6Hrs/day)

DAYS	TOPICS COVERED	ASSESSMENTS
DAY 1	Advanced OOP Concepts - Part 1 - Object References & Memory Management, Method Overloading vs Method Overriding, Polymorphism (Compile-time & Runtime), Upcasting & Downcasting, instanceof Operator, Super keyword deep dive, Abstraction Implementation Strategies	1. Solve 3 coding problems on method overloading with edge cases 2. Design a polymorphic hierarchy (minimum 3 classes) 3. MCQ Test: 10 questions on OOP concepts 4. Code Review: Identify polymorphism violations in provided code
DAY 2	Advanced OOP Concepts - Part 2 - Interface Design Patterns, Abstract Class vs Interface, Marker Interfaces, Sealed Classes (Java 17+), Composition over Inheritance, Object cloning (Shallow vs Deep), equals() & hashCode() implementation	1. Create interfaces with multiple implementations 2. Implement clone() method with deep copy 3. Debug 2 programs with incorrect equals() & hashCode() 4. Design problem: Create a scalable object hierarchy
DAY 3	Exception Handling & Java Advanced Features - Checked vs Unchecked Exceptions, Try-with-resources, Custom Exception hierarchies, Exception chaining, Annotations (@Override, @Deprecated, @FunctionalInterface), Generics fundamentals (Wildcards, Bounded types, Type erasure)	1. Write 4 custom exception classes for different scenarios 2. Convert traditional try-catch to try-with-resources (3 examples) 3. Implement generic classes and methods (2 complex examples) 4. Fix 5 programs with incorrect generic usage
DAY 4	Collections Framework Deep Dive - Part 1 - Comparable vs Comparator, List implementations (ArrayList, LinkedList, Vector), Internal implementations & time complexity, Sorting strategies, Searching algorithms (Binary Search), Custom Collections operations	1. Implement custom Comparator for 3 different scenarios 2. Performance test: ArrayList vs LinkedList (100K elements) 3. Write 4 sorting programs with custom objects 4. Problem: Find K closest elements in sorted array
DAY 5	Collections Framework Deep Dive - Part 2 - Set implementations (HashSet, LinkedHashSet, TreeSet), Map implementations (HashMap, LinkedHashMap, TreeMap), Hashtable vs HashMap, ConcurrentHashMap, Custom Hashmap implementation concepts	1. Solve 3 duplicate removal problems using different Set types 2. Design a cache system using LinkedHashMap 3. Compare performance: HashSet vs TreeSet (50K elements) 4. Create a frequency counter using Map (3 variations)
DAY 6	Fundamentals of Data Structures & Algorithms - Algorithm Analysis (Big O, Omega, Theta), Time & Space Complexity, Recurrence Relations, Master Theorem, Introduction to Arrays & Linked Lists, Searching (Linear & Binary), Sorting (Merge, Quick, Heap)	1. Analyze complexity of 8 code snippets 2. Implement 3 sorting algorithms and compare performance 3. Solve 4 problems on searching with different constraints 4. Practice Test: 10 algorithm analysis questions

TECHXCELERATE TRAINING PROGRAM

II YEAR ENGINEERING STUDENTS - CSE

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

DAYS	TOPICS COVERED	ASSESSMENTS
DAY 1 & 2	Data Structures - Stacks & Queues (3 hrs) - Stack ADT, Stack operations (Push, Pop, Peek), Stack applications (Expression evaluation, Parenthesis matching, Undo-Redo mechanisms), Implementation using Array & LinkedListQueues & Deques (3 hrs) - Queue ADT, Queue operations, Circular Queue, Deque operations, Priority Queue, Queue applications (BFS, LRU Cache)	1. Implement Stack using both Array and LinkedList 2. Solve 4 stack-based problems (Balanced parentheses, Next Greater Element, etc.) 3. Implement Circular Queue with all operations 4. Design LRU Cache using Deque 5. MCQ: 15 questions on Stacks & Queues
DAY 3 & 4	Data Structures - Trees - Part 1 (3 hrs) - Tree terminology & types, Binary Trees, Binary Search Trees (BST), Tree traversals (Inorder, Preorder, Postorder, Level-order), Height-balanced trees, Insertion & Deletion in BST AVL Trees & Rotations (3 hrs) - AVL Tree properties, Rotation techniques (LL, RR, LR, RL), Insertion & Deletion with balancing	1. Implement Binary Search Tree with all operations 2. Solve 5 tree traversal problems 3. Convert tree traversal outputs to identify tree structure 4. Implement AVL Tree insertion with rotations 5. Debug 4 BST violation programs
DAY 5 & 6	Data Structures - Advanced Trees (3 hrs) - Red-Black Trees (basic concepts), B-Trees, Segment Trees, Fenwick Trees (Binary Indexed Trees), Tree applications Graphs Introduction - Part 1 (3 hrs) - Graph terminology, Graph representations (Adjacency Matrix, Adjacency List), Graph types (Weighted, Unweighted, Directed, Undirected), Degrees in graphs	1. Implement Segment Tree with range queries 2. Create Fenwick Tree and perform prefix sum queries 3. Represent 5 different graphs using both representations 4. Analyze time & space complexity of graph representations 5. Solve 4 problems using appropriate tree structures
DAY 7 & 8	Graphs - Traversals & Algorithms - Part 1 (3 hrs) - DFS (Depth-First Search), BFS (Breadth-First Search), Applications of DFS & BFS, Cycle detection (Directed & Undirected), Topological sorting Graphs - Shortest Path Algorithms (3 hrs) - Dijkstra's Algorithm, Bellman-Ford Algorithm, Floyd-Warshall Algorithm, Shortest path in DAG	1. Implement DFS & BFS from scratch 2. Solve 6 graph traversal problems 3. Implement Dijkstra's & Bellman-Ford algorithms 4. Compare shortest path algorithms (performance analysis) 5. Solve real-world shortest path scenarios (3 problems)

TECHXCELERATE TRAINING PROGRAM

II YEAR ENGINEERING STUDENTS - CSE

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

DAYS	TOPICS COVERED	ASSESSMENTS
DAY 9 & 10	Graphs - Advanced Topics (3 hrs) - Minimum Spanning Tree (Kruskal's & Prim's), Union-Find (Disjoint Set Union), Graph connectivity problems, Strongly Connected Components Dynamic Programming Fundamentals (3 hrs) - Overlapping subproblems, Optimal substructure, Memoization vs Tabulation, Classic DP problems (Fibonacci, Knapsack, LCS, LIS)	1. Implement Kruskal's & Prim's algorithms 2. Solve MST problems with constraints 3. Implement Union-Find with path compression & union by rank 4. Solve 8 classic DP problems 5. Compare memoization & tabulation approaches
DAY 11 & 12	Dynamic Programming Advanced & Greedy Algorithms (3 hrs) - Multi-dimensional DP, DP on Grids, DP on Trees, Matrix Chain Multiplication, Edit Distance Greedy Algorithms (3 hrs) - Greedy strategy, Activity Selection, Fractional Knapsack, Huffman Coding, Interval Scheduling problems	1. Solve 6 advanced DP problems (2D grids, trees) 2. Optimize recursion to DP (3 problems) 3. Solve 4 greedy algorithm problems 4. Compare greedy vs DP for knapsack problem 5. Final Assessment: 20 mixed DSA problems (varying difficulty)

TECHXCELERATE TRAINING PROGRAM

II YEAR ENGINEERING STUDENTS - CSE

SLOT 02 - 12 DAYS

DAYS	TOPICS COVERED	ASSESSMENTS
DAY 1	System Design & Advanced Problem Solving - Part 1 - Bit Manipulation fundamentals, Bit operations, Number system conversions, XOR tricks, Bit manipulation applicationsInterview Preparation Introduction - Behavioral interview basics, Technical interview structure, STAR method, Company-specific interview patterns	1. Solve 15 bit manipulation problems 2. Optimize algorithms using bit operations (5 problems) 3. Write code for bit manipulation applications 4. Practice behavioral interview questions (5 scenarios) 5. Mock interview: 1 round
DAY 2	Advanced Bit Manipulation & String Algorithms - Advanced bit tricks, Single number variations, Power of numbers, String manipulation fundamentals, String matching (KMP, Rabin-Karp, Z-algorithm)Coding Best Practices - Clean code principles, Optimization techniques, Code documentation, Performance tuning	1. Solve 12 advanced bit problems 2. Implement KMP & Rabin-Karp algorithms 3. Solve 8 string matching problems 4. Code review: 3 complex algorithms 5. Performance optimization practice: 5 programs

TECHXCELERATE TRAINING PROGRAM

II YEAR ENGINEERING STUDENTS - CSE

SLOT 02 - 12 DAYS

DAYS	TOPICS COVERED	ASSESSMENTS
DAY 3	Sliding Window & Two Pointers Techniques - Sliding window approach, Window size optimization, Two pointers technique, Common patterns (Subarray sum, Container with most water, Merge sorted arrays)Hash Functions & Hashing - Hash function design, Collision handling, Load factor, Bloom filters, Hash table internal structure	1. Solve 10 sliding window problems 2. Solve 8 two-pointer problems 3. Design hash functions for custom objects (2 examples) 4. Implement hash table from scratch 5. Solve hashing-based problems (5 scenarios)
DAY 4	Recursion & Backtracking Advanced - Recursion tree analysis, Tail recursion, Recursion optimization, Backtracking paradigm, State space explorationBacktracking Applications - N-Queens, Sudoku solver, Permutations & Combinations, Subset generation, Letter combinations	1. Solve 8 recursion complexity analysis problems 2. Convert iterative to recursive & vice versa (4 problems) 3. Implement 5 classic backtracking problems 4. Optimize backtracking with pruning (3 problems) 5. Practice: 15 backtracking variations
DAY 5	Mathematical Algorithms & Number Theory - Prime checking (Sieve of Eratosthenes), GCD/LCM, Modular arithmetic, Fermat's Little Theorem, Factorial modulo, Combinatorics fundamentalsRange Queries & Segment Trees Deep Dive - Segment Tree advanced concepts, Lazy propagation, Persistent Segment Trees, Merge sort tree	1. Solve 10 number theory problems 2. Optimize prime checking (Large numbers) 3. Solve modular arithmetic problems (6 scenarios) 4. Implement Segment Tree with lazy propagation 5. Solve range query problems (8 problems)
DAY 6	Trie Data Structure & Advanced Tree Problems - Trie structure, Trie operations (Insert, Search, Delete), Trie applications (Autocomplete, Spell checker, IP routing)Advanced Tree Problems - LCA (Lowest Common Ancestor), Path Sum problems, Serialize & deserialize trees, Tree DP	1. Implement Trie with full operations 2. Solve 8 Trie-based problems 3. Implement LCA with different approaches (3 methods) 4. Solve 6 advanced tree problems 5. Tree serialization/deserialization (3 formats)
DAY 7	Disjoint Set Union (DSU) & Advanced Graph Problems - Union-Find with union by rank & path compression, Kruskal's algorithm revisited, Connected components detection, Cycle detection in undirected graphsGraph Coloring & Bipartite Graphs - Graph coloring problem, Bipartite graph detection, 2-coloring, K-coloring complexity	1. Implement DSU with optimizations 2. Solve 10 DSU-based problems 3. Find connected components & cycle detection (6 problems) 4. Implement bipartite checking (Multiple approaches) 5. Solve graph coloring problems (4 scenarios)

TECHXCELERATE TRAINING PROGRAM

II YEAR ENGINEERING STUDENTS - CSE

SLOT 02 - 12 DAYS

DAYS	TOPICS COVERED	ASSESSMENTS
DAY 8	Topological Sorting & DAG Algorithms - Topological sort (DFS & Kahn's), Longest path in DAG, Shortest path in DAG, Scheduling problems, Dependency resolutionNetwork Flow Basics - Flow networks, Ford-Fulkerson algorithm, Edmonds-Karp algorithm, Maximum flow applications	1. Implement topological sort (2 approaches) 2. Solve 8 DAG-based problems 3. Implement maximum flow algorithm 4. Solve flow network problems (5 scenarios) 5. Real-world network flow applications (3 problems)
DAY 9	Matrix Problems & Spatial Algorithms - Matrix traversal techniques (Spiral, Diagonal, Zigzag), Matrix rotation & transformation, Median in sorted matrixGeometry & Computational Geometry Basics - Distance calculations, Point in polygon, Line intersection, Convex hull	1. Solve 10 matrix traversal problems 2. Implement matrix rotation (All rotations) 3. Solve coordinate & geometry problems (6 scenarios) 4. Find convex hull (2 algorithms) 5. Practice: Mixed matrix & geometry problems (12 problems)
DAY 10	Advanced DP Patterns & Optimization - DP with probability, Digit DP, Profile DP, Convex hull optimization, Divide & conquer optimizationGame Theory Basics - Nim game, Grundy numbers, Winning & losing positions, Game state analysis	1. Solve 6 digit DP problems 2. Solve 5 probability DP problems 3. Implement convex hull optimization 4. Solve 4 game theory problems 5. Practice: 15 advanced DP variations
DAY 11	Competitive Programming Strategies & Optimization Techniques - Time complexity vs Space complexity trade-off, Premature optimization detection, Common TLE causes, Caching strategies, Parallel processing conceptsContest Problem Analysis - Problem classification, Approach selection, Stress testing, Debugging techniques, Code templates	1. Analyze 8 slow algorithms & optimize 2. Stress test 4 solutions 3. Debug 5 complex programs 4. Create personal code templates (DSA fundamentals) 5. Solve contest-style problems (15 mixed)
DAY 12	Comprehensive Assessment & Project Work - Capstone project: Design & implement an LLD system using DSA concepts (e.g., Uber backend, Chat system, File system)Mock Interviews & Final Assessment - 2 complete mock interviews (45 mins each), Code review & optimization, Interview feedback, Debugging session	1. Capstone project implementation (Full code with documentation) 2. Mock Interview 1: System design round 3. Mock Interview 2: Algorithm round (5 problems) 4. Code walkthrough & optimization 5. Final comprehensive test: 25 problems in 2 hours 6. Individual performance review & roadmap

PROGRAM OUTCOMES

- Advanced Object-Oriented Design Mastery - Design scalable, enterprise-level systems using advanced OOP principles, sealed classes, composition patterns, and SOLID-compliant hierarchies.
- Expert-Level Data Structure Implementation - Build custom implementations of all data structures (Stacks, Queues, Trees, Graphs, Tries, Segment Trees) from scratch with optimized complexity.
- Advanced Algorithm Optimization & Complexity Analysis - Master Big O notation, recurrence relations, and optimization techniques (bit manipulation, sliding windows, two pointers).
- System Design & Backend Architecture Thinking - Design scalable systems (LRU Cache, Chat Systems, Uber-like platforms) with proper trade-off analysis and caching strategies.
- Competitive Programming Excellence - Solve complex problems including network flow, convex hull optimization, digit DP, and game theory at competitive programming level.
- FAANG-Level Technical Interview Readiness - Successfully clear multiple rounds of technical interviews at top-tier tech companies with system design and optimization expertise.
- Graph Algorithms & Network Problem Mastery - Implement Dijkstra's, Bellman-Ford, Kruskal's, Prim's, topological sorting, and network flow algorithms for real-world problems.
- Dynamic Programming Expert-Level Problem Solving - Master all DP patterns (1D, 2D, digit DP, probability DP, DP on grids/trees) and recognize patterns in novel problems.
- Code Quality, Optimization & Peer Review Skills - Write production-grade code with proper documentation, conduct effective code reviews, and identify performance bottlenecks.
- Advanced Problem-Solving Mindset & Debugging Expertise - Develop expert approach to novel problems by combining multiple DSA concepts with rigorous debugging and optimization.