**DSA with C++**

**1. Basics of Programming in C++**

* **Syntax and Data Types:** Variables, constants, types, operators
* **Control Structures:** if-else, switch-case, loops (for, while, do-while)
* **Functions:** Function definition, function calls, recursion
* **Pointers and Memory Management:** Pointers, dynamic memory allocation (new, delete), references

**2. Time and Space Complexity Analysis**

* **Big O Notation:** Time complexity, space complexity
* **Best, Worst, Average Case Analysis**
* **Amortized Analysis**

**3. Arrays and Strings**

* **1D Arrays, 2D Arrays**
* **Basic Array Operations:** Insertion, deletion, searching, sorting
* **Strings:** Manipulation of strings, common string algorithms (substring search, pattern matching)
* **Important Problems:** Reverse an array, find duplicates, sliding window problems

**4. Searching Algorithms**

* **Linear Search**
* **Binary Search**
* **Binary Search Variants:** Lower bound, upper bound, finding peak elements

**5. Sorting Algorithms**

* **Bubble Sort, Selection Sort, Insertion Sort**
* **Merge Sort, Quick Sort**
* **Heap Sort**
* **Counting Sort, Radix Sort, Bucket Sort**

**6. Recursion and Backtracking**

* **Basic Recursion:** Factorial, Fibonacci, Tower of Hanoi
* **Backtracking:** N-Queens problem, Sudoku solver, Rat in a maze
* **Recursion vs. Iteration**
* **Memoization (Dynamic Programming)**

**7. Linked Lists**

* **Singly Linked List:** Creation, traversal, insertion, deletion
* **Doubly Linked List**
* **Circular Linked List**
* **Common Problems:** Reverse a linked list, detect loops, merge two sorted lists

**8. Stacks and Queues**

* **Stacks:** Implementation using arrays or linked lists, applications (infix to postfix, balanced parentheses)
* **Queues:** Simple queue, circular queue, deque, priority queue
* **Applications of Stacks and Queues:** Next greater element, sliding window maximum

**9. Hashing**

* **Hash Tables:** Hash functions, collision resolution techniques (chaining, open addressing)
* **Applications of Hashing:** Frequency counting, finding duplicates, two-sum problem

**10. Trees**

* **Binary Trees:** Basic operations (insertion, deletion, traversal)
* **Binary Search Tree (BST)**
* **Tree Traversals:** In-order, pre-order, post-order, level-order
* **Balanced Trees:** AVL tree, Red-Black tree
* **Heaps:** Min-heap, max-heap, heapify, applications (priority queue, heap sort)

**11. Graphs**

* **Representation of Graphs:** Adjacency matrix, adjacency list
* **Graph Traversal:** Depth First Search (DFS), Breadth First Search (BFS)
* **Shortest Path Algorithms:** Dijkstra’s algorithm, Bellman-Ford algorithm
* **Minimum Spanning Tree (MST):** Kruskal's algorithm, Prim's algorithm
* **Topological Sort**
* **Strongly Connected Components (SCC)**
* **Applications of Graph Algorithms:** Detect cycle in a graph, bipartite check

**12. Greedy Algorithms**

* **Activity Selection Problem**
* **Fractional Knapsack Problem**
* **Huffman Coding**
* **Minimum Spanning Tree (Kruskal, Prim)**

**13. Dynamic Programming**

* **Basic Problems:** Fibonacci series, longest common subsequence, 0/1 knapsack
* **Advanced Problems:** Matrix chain multiplication, longest increasing subsequence, coin change problem
* **DP on Trees:** Diameter of a tree, longest path
* **Memoization vs. Tabulation**

**14. Divide and Conquer**

* **Merge Sort**
* **Quick Sort**
* **Binary Search**
* **Matrix Exponentiation**
* **Closest Pair Problem**

**15. Bit Manipulation**

* **Bitwise Operators:** AND, OR, XOR, left shift, right shift
* **Applications of Bit Manipulation:** Checking set/unset bits, counting bits, finding unique elements

**16. Advanced Topics**

* **Segment Trees:** Range queries, lazy propagation
* **Fenwick Tree (Binary Indexed Tree):** Point update, range queries
* **Trie (Prefix Tree):** Applications in searching strings, autocomplete
* **Disjoint Set Union (DSU) / Union-Find:** Applications in connected components, cycle detection
* **KMP Algorithm for Pattern Searching**

**17. Competitive Programming Techniques**

* **Fast Input/Output Techniques**
* **Optimized Recursion (Tail Recursion)**
* **Modular Arithmetic and Large Numbers**
* **Prime Number Algorithms:** Sieve of Eratosthenes, prime factorization

**18. Miscellaneous Algorithms**

* **Number Theory Algorithms:** GCD, LCM, modular exponentiation, Fermat’s little theorem
* **Game Theory Problems**
* **Geometry Algorithms:** Convex hull, line intersection

**Resources to Follow**

* **Books:**
  + "Introduction to Algorithms" by Cormen, Leiserson, Rivest, and Stein (CLRS)
  + "The Art of Computer Programming" by Donald Knuth
* **Online Resources:**
  + LeetCode, Codeforces, HackerRank, GeeksforGeeks
  + Competitive Programming platforms

**Practice**

* Regular practice on coding platforms
* Solving problems by difficulty level (easy -> medium -> hard)
* Participate in coding contests

By following this roadmap, you will build a strong understanding of DSA with C++.

4o