Competitive Programming Roadmap

Competitive programming (CP) involves solving well-defined algorithmic problems within a limited time frame. Here is a step-by-step roadmap to excel in CP, categorized into beginner, intermediate, and advanced levels.

Phase 1: Beginner Level

1. Understanding Basics

- Languages: Choose a programming language (C++, Python, or Java). C++ is widely used for its Standard Template Library (STL).
- Topics to Cover:
  - Input/Output (fast I/O methods like scanf/printf in C++ or BufferedReader in Java).
  - Loops, conditionals, arrays, strings.
  - Functions and recursion basics.
- Practice: Solve simple problems on platforms like HackerRank or Codeforces.

2. Basic Data Structures

- Arrays, Strings, Stacks, Queues, Linked Lists.
- Learn to implement these manually and use STL (C++) or Java Collections.
- Practice Problems: Solve problems tagged as "Easy" on LeetCode or GeeksforGeeks.

3. Learn Basic Algorithms

- Sorting algorithms: Bubble Sort, Merge Sort, Quick Sort.
- Searching: Binary Search.

- Two-pointer technique.

- Practice Problems: Start with simple implementation and pattern-based problems.

Phase 2: Intermediate Level

1. Core Data Structures

- Master advanced data structures:
  - Hash Maps/Hash Sets (unordered_map in C++, HashMap in Java).
  - Trees (Binary Trees, Binary Search Trees).
  - Graphs (Adjacency List, Adjacency Matrix).
  - Priority Queues/Heaps.
- Practice Problems: Solve medium-level problems on LeetCode and Codeforces.

2. Algorithms Mastery

- Dynamic Programming (start with Fibonacci and knapsack problems).
- Greedy algorithms (e.g., activity selection problem).
- Graph algorithms:
  - BFS, DFS.
  - Shortest path algorithms: Dijkstra's, Bellman-Ford.
  - Minimum Spanning Tree: Kruskal's, Prim's.
- Divide and Conquer.
- Backtracking (e.g., N-Queens, Sudoku Solver).

3. Problem-Solving Strategies

- Learn to debug effectively.

- Focus on writing clean and modular code.

- Participate in live contests on Codeforces, CodeChef, AtCoder.

- Practice: Solve a mix of implementation, greedy, and DP problems.

Phase 3: Advanced Level

1. Advanced Data Structures

- Segment Trees, Fenwick Trees (Binary Indexed Trees).

- Trie, Suffix Arrays.

- Disjoint Set Union (Union-Find).

- Practice Problems: Tackle problems requiring these structures on Codeforces (Div 2 and Div 1 contests).

2. Advanced Algorithms

- Advanced Dynamic Programming (e.g., DP with bitmasks).

- Network Flow Algorithms (Ford-Fulkerson, Edmonds-Karp).

- Computational Geometry (Convex Hull, Line Intersections).

- String Matching Algorithms (KMP, Z-algorithm).

3. Optimization Techniques

- Precomputations (e.g., prefix sums, modulo arithmetic).

- Space optimization in DP.

- Bit Manipulation.

- Practice Problems: Solve problems from SPOJ and advanced contests on Codeforces.

4. Participate in Competitions

- Regularly participate in ICPC-style contests, long challenges, and short contests.

- Join a CP community (e.g., Codeforces or Reddit's r/cscareerquestions).

General Tips

1. Consistent Practice

- Solve problems daily or follow a structured schedule.

- Focus on solving problems slightly above your current skill level.

2. Learn from Editorials

- After attempting a problem, study its editorial to understand the optimal solution.

3. Time Management

- Divide time between learning new topics and practicing.

- During contests, prioritize problems based on difficulty.

4. Maintain Notes

- Document algorithms, tricks, and key takeaways from problems.

- Maintain a list of mistakes to avoid repeating them.

5. Use Competitive Programming Tools

- Online judges: Codeforces, CodeChef, AtCoder, HackerRank, LeetCode.

- IDEs: VS Code, Sublime Text, or an online editor like CSES Problem Set.

- Books: "Competitive Programming 3" by Steven Halim or "Introduction to Algorithms" by Cormen

(CLRS).

Final Note

Competitive programming is a journey that requires perseverance, logical thinking, and consistent practice. Set realistic goals, track your progress, and enjoy the process of solving challenging problems!