Pre-requisites:

1. Should be able to write code in any one OOP language (C++/Java/Python (C++ recommended)).
   Please do follow this link https://www.hackerrank.com/domains/cpp parallelly with the below
   a. Printing "Hello World".
   b. Common data types (int, double, char, bool, ...)
   c. Ranges of each data type
   d. Input variables of common datatypes
   e. Addition of two integers, print the result
   f. Trying every operators (+, -, /, *, %, ^, &, |, +=, -=, <<, >>, ...) for integers
   g. Operator precedence: Bodmas in programming
   h. If else condition, switch case, input an integer and print it is odd or even
   i. For, while, do-while loop, input N, print integers from 1 to N
   j. Arrays, String declaration
   k. Input N, Input N integers using loops in an array of N integers
   l. 2D arrays, matrix addition
   m. Functions, try implementing `int add(int x, int y)`
   n. Pass by value vs pass by reference
   o. Pointers, LinkedList (**optional**)
   p. Classes (**Try** making a matrix/queue/stack/deque datatype with operator overloading)
   q. C++ STL(vector, multiset, queue, sort(), next_permutation(), pbds, ...) GFG Youtube(Luv, Rachit Jain) (Topcoder Notes)
   r. **Try** implementing each and every container/algorithm on your own using OOP concepts (**optional**)
      Try and understand what is implemented in the libraries
   s. If you reach till here, give a pat on the back of your body :)
   t. Recursion (Fibonacci, Permutations, Combinations, Print all subsequences of an array)

Start:

1. Hackerrank > Practice > Problem Solving > Warmup + Implementation
2. HackerEarth CodeMonk
3. Basic Math: Sieve of Eratosthenes, Smallest Prime Factors, Modular Arithmetic, Modular Exponentiation, Euclidean Algorithm, Inverse Mod/Fermat's Theorem, NCR using Fermat's theorem, Totient Function
4. Binary Search, Two Pointers, Greedy Problems, String Pattern Matching (KMP, Rabin Karp Algorithm)
5. Basic data structures: Trie
6. Dynamic Programming(LCS, LIS, Knapsack), Graphs
7. Segment Trees

Resources:

1. CP-Algorithms (For the implementation of popular CP Algorithms)
2. HackerEarth Notes
3. Codeforces blogs (example)
4. Codechef editorials/discuss (example)
5. Codeforces Edu section - Binsearch, DSU, SegTree, 2pointers, suffix array (link)
6. Leetcode DP Blog
7. Errichto Youtube Channel: BinSearch

Where can I solve problems?

1. Codeforces (Frequent Contests with Editorials) (use this website to filter contests)
   a. Div2/Div3 Challenges
   b. Problem set with difficulty, tag, and topic sorted problems
   c. Gyms (Past ICPC contests)
2. Codechef (Regular Contests with editorials)
   a. Long Challenges (1st Friday each month, 10 days long)
   b. CookOff (2.5 hours)
   c. Lunchtime (3 hours)
3. Codedrills (Problems with editorials)
4. Atcoder (Beginner and Regular Contests track your progress here!)
5. A2OJ (For difficulty wise sorted codeforces problems)
6. CSES problem set (For topic wise problems)
7. SPOJ (For topic/tag wise problems)

Topic/Difficulty Wise Past Vjudge/Other Contests:

1. DP: Atcoder DP, VJ1
2. BinSearch: VJ1
3. SegTree: VJ1