Ayush Kumar 1 year ago

>1st set of basic stuffs
1.Pattern printing problems
2.time complexity analysis
3.linear search and circular array representation
4.palindrome and other numbers(perfect, Armstrong) for basic number problems
5.Simple Hashing Problem(frequency counting and stuffs)
6.Prefix Sum Problems(1D and 2D) {codeforces }
7.Sliding window technique(2 out of 5 contests)

Basics of number theory
1.Binary Search is a must(2/5 contests)
2. GCD of 2 numbers in logarithmic time(Euclidean and Extended Euclidean Algorithm)
3.linear Diphantine Equation
4. Checking prime in sqrt(n) complexity
5.Sieve of Eratosthenes(very imp to perform querry probs on the prime)
6.Segmented Sieve
7.Finding prime factorisation of a number in logn per querry
8.Euler Totent Function
9.Fermat Little Theorum
10.Wilson's Theorum
(gfg articles and hacker earth probs for 8,9,10)
tougher version of number theory
1.Finding x^n in log(n)
2. Modular Arithmetic
3.Modular Inverse of a number
3.Modular Exponentiation
4.Chinese Remainder Theorum
5.Factorial Modulo Mod
6.Finding nCr and nPr for queries (constant time)
7.Inclusion Exclusion Principle (combinatorics problems) {hackerearth has wonderful bunch} {codeforces}

1.learn about basic sorting algorithms (bubble, selection, insertion)
2. do problems which are constructive and have a lot of swapping terms in it.
3.solve problems related to two pointer approach.
4.Bit manupulation(left shift, right shift, xor, or, and,set bit,MSB, LSB etc..)
5.Power set of a given array or string using BIT
6.Number of subarrays with XOR as zero(not an algorithm but a must do problem)
(hackerearth has a very good tutorial on bit manipulation, coding blocks too has a proper video on bit) {for problems visit hackerearth}
7. Problems related to greedy algorithm tag
8.Kadane's algorithm and problems related to them
9.Job sequencing and activity selection problem
(after doing 8 and 9 go to codeforces and solve problems with greedy tag on it)
---

time to learn recursion
1.start with basic problems like finding factorial and all.
2.implement binary search
3.implement modular exponentiation using recurison
4.solve recursion problems like finding subset with given and others to get a stronggrip
5.Learn about merge sort and quick sort
6.solve problems related to merge sort
7. Do backtracking problems like sudoko and N Queen, it will help you when you want do DP Path problems
(for recursion you can refer to leetcode or gfg's practice where u will find recursion and backtracking problems)
After Recursion:
1. Meet in the middle algorithm and problems related to it.
2.Divide and conquer problems{highly recommended to use codeforces only for this}
3. Next greater element and next smaller element using stack
4. porblems related to parenthesis.
5.largest rectangular area in histogram. (concept is used in a lot of problems)
6.Problems related to Heap(Priority Queue) {although this gets under the greedy category but by priority queue will help you learn an inbuilt stl)

now the real advanced parts and the game decider
(imp guideline from 11:16 to 12:20)
String algorithms:
1.hashing on strings(learn and solve problem, understand when collision happens) {cpalgorithms has a wonderful article written on it)
{Spoj or codeforces}
2.Rabin Karp Algorithm(cpalgorithm has a wonderful blog on it)
3.Prefix Function
4.KMP Algorithm
5.Z-function
6.Manchers' Algorithm
(once you have wrapped up the above algorithms, solve a bunch of problems(25-30) on them from different platforms.)

Tree Algorithm :
1.Tree/Graph representation
2.DFS/BFS Traversal in Graph/Tree
3.Basic stuffs(diameter of tree, height of tree, level of tree)
4.Euler Tour of Tree(Learn and solve problems)
5.Finding LCA using Euler Tour(14:00){efficient solution uses segment trees)
6.Finding LCA using Binary Lifting.
7.Distance between two nodes.
8.Subtree Problems.
(SPOJ is highly recommended for trees and codeforces D and E problems also)

Graphs:
1.Connected Components.
2.Topological Sort.
3.Cycle Detection in Graph
4.Bipartite Check in graph
5.SCC using Kosaraju's algorithm
6.Dijkstra's Algorithm
7.Bellman Ford Algorithm
Sources for learning :hackerearth and cpalgorithm blogs
(25-30 problems on the avove topics,SPOJ and codeforces,hackerearth too)

Then:
1.Bridges In graphs
2.Articulation Point in a graph
3.Minimum Spanning Tree using Kruskal's Algo
4.Prim's Algorithm
5.0/1 BFS(a big saviour)
6.Learn Finding Bridges Online(cpalgorithms)
Solve problem

Dynamic Programming:
Imp instructions at 16:08
1.Solve the AtCoder Educational Contests on Dyanamic Programming.(all 26)
2.Solve problems from SPOJ(highly recommended,since it doesn't involve any other algorithms)
3. Google dynamic programming practice problem codeforces,u'll get a wonderful blog with a lot of problems on it.
After all the above standard DP,learn:
4.Understand how we write recurrence for Digit DP(codeforces blog)(digit dynamic progg) and solve problems
5. read about DP with Bitmasks and solve problems(hackerearth blog)
6.DP on trees(gfg articles,rachit jains video)
7.SOS DP(cpalgorithm blog)
Solve as much probs possible

1.Disjoint Set(cpalgorithms)
2.Offline Queries using Disjoint Set(colourful array problem from spoj)
3.Kruskal's Algorithm using disjoint set
Solve bunch of problems on avove

1.Sparse Table(not that imp)
2.Fenwick Tree and Binary Lifting on Fenwick Tree(read about range update trick also)
3.problems on fenwick tree

1.Matrix Exponentiation(problems)
2.Sqrt Decomposition Technique(gfg or cpalgorithms or codeforces)
3.Update and Querry Operations
4.Mo's algorithm(solve powerful array from codeforces)(codeforces blog)
5.Mo's algorithm on trees
6.Segment Tree(a must)(Range Queries and point Updates)
7.Lazy propagation on segment trees
---

then some optional and rare ones:
1.Sprague-Grundy Theorum
2.Flows and Related Problems
3.Heavy Light Decomposition
4.Convex Hull Algorithm
5.FFT/NTT
Show less