**Assignment Details**

**BTech Third for Swayam Students**

**Note:**

### **Instructions for DSA Practice Sheet (13 Oct – 10 Nov 2025) – For Third Year Students**

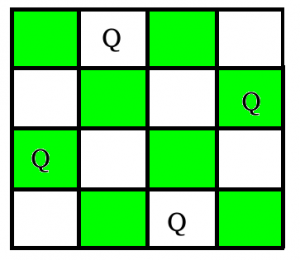
1. **These questions are mandatory** for all **Second Year B.Tech students** and must be completed sincerely.
2. **The syllabus for this sheet includes topics from Array to Linked List*.***
3. **All solutions must be uploaded on GitHub** under a properly named repository . Your sincerity will be measured by github profile on daily basis contributions on github.
4. Each problem must include **clear comments** explaining the logic and reasoning.
5. Every question should have **brute-force** and **optimized solutions** wherever possible.
6. Maintain a **DSA Practice Notebook** (hardcopy or digital) containing:
   * Problem statement
   * Step-by-step logic
   * Time and Space Complexity
   * Final code summary
7. Ensure all assigned problems are **completed and uploaded on or before given date.**
8. Use the **LeetCode links provided** in the sheet for practice and submission reference.
9. Maintain proper **naming conventions and folder structure**, such as Array/, String/, LinkedList/.
10. Focus on **clean, readable, and efficient coding style** — proper indentation, meaningful variable names, and no redundant code.

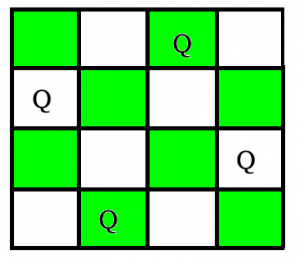
|  |  |  |
| --- | --- | --- |
| **DAY WISE DETAILS** | | |
| Date of Submission | Problem Statement Link |  |
| .13/10/2025 | Array I |  |
| [Reversing the array](https://www.naukri.com/code360/problems/reverse-the-array_1262298), [finding maximum and minimum elements,](https://leetcode.com/problems/removing-minimum-and-maximum-from-array/description/) [Running sum of 1d Array](https://leetcode.com/problems/running-sum-of-1d-array/description/), [count elements with maximum frequency](https://leetcode.com/problems/count-elements-with-maximum-frequency/description/) , [left/right rotate an array by k positions.](https://leetcode.com/problems/rotate-array/description/) | | |
| .13/10/2025 | Array II | . |
| Content Summary: [find element in an array](https://www.geeksforgeeks.org/searching-elements-in-an-array-array-operations/),  [Remove duplicate elements from an sorted array](https://leetcode.com/problems/remove-duplicates-from-sorted-array/description/), [find repeating element an array](https://takeuforward.org/data-structure/find-all-repeating-elements-in-an-array/), [find equilibrium element in an array](https://leetcode.com/problems/find-pivot-index/description/). | | |
| .13/10/2025 | Array’s Sorting and Time and space complexity Analysis | . |
| Content Summary: [Bubble sort](https://www.geeksforgeeks.org/bubble-sort-algorithm/), [selection sort](https://www.geeksforgeeks.org/selection-sort-algorithm-2/), [Insertion Sort and complexity Analysis](https://www.geeksforgeeks.org/insertion-sort-algorithm/) | | |
| .13/10/2025 | Array III | . |
| Content Summary: [union and intersection of sorted arrays](https://leetcode.com/problems/intersection-of-two-arrays/description/), [maximum subarray sum (Kadane’s Algorithm)](https://leetcode.com/problems/maximum-subarray/description/), [maximum product subarray(based on Kandane’s)](https://leetcode.com/problems/maximum-product-subarray/description/) , [majority Element (moore’s voting algorithm)](https://leetcode.com/problems/majority-element/description/) | | |
| .13/10/2025 | Strings I | . |
| Content Summary: [check given string is palindrome or not](https://leetcode.com/problems/valid-palindrome/description/), [count number of vowel and consonant](https://takeuforward.org/data-structure/count-number-of-vowels-consonants-spaces-in-string/), [remove character except alphabet](https://takeuforward.org/data-structure/remove-characters-from-a-string-except-alphabets/). | | |
| .14/10/2025 | String  II | . |
| Content Summary: [Calculate frequency of a character](https://takeuforward.org/data-structure/calculate-frequency-of-characters-in-a-string/), [print maximum occurring character in a string](https://takeuforward.org/data-structure/maximum-occurring-character-in-a-string/),  [Remove duplicate character from a string](https://www.geeksforgeeks.org/remove-duplicates-from-a-given-string/), [count number of word in a string](https://takeuforward.org/data-structure/count-the-number-of-words-in-a-given-string/) | | |
| .14/10/2025 | Recursion I | . |
| Content Summary: [find factorial](https://www.geeksforgeeks.org/program-for-factorial-of-a-number/), [find power of a number](https://www.geeksforgeeks.org/problems/power-of-numbers-1587115620/1?itm_source=geeksforgeeks&itm_medium=article&itm_campaign=practice_card), [(printing increasing, decreasing and Decreasing Increasing)](https://www.geeksforgeeks.org/problems/print-1-to-n-without-using-loops3621/1?itm_source=geeksforgeeks&itm_medium=article&itm_campaign=practice_card), [count digit](https://www.geeksforgeeks.org/problems/count-digits5716/0), [sum of array using recursion](https://www.geeksforgeeks.org/problems/sum-of-array2326/1?itm_source=geeksforgeeks&itm_medium=article&itm_campaign=practice_card) | | |
| .14/10/2025 | Recursion II | . |
| Content Summary: [find pivot index](https://leetcode.com/problems/find-pivot-index/description/), [remove duplicates](https://classroom.codingninjas.com/app/classroom/me/21313/content/451415/offering/6582572/problem/91?leftPanelTabValue=PROBLEM), [fibonacci number](https://www.geeksforgeeks.org/problems/nth-fibonacci-number1335/1?itm_source=geeksforgeeks&itm_medium=article&itm_campaign=practice_card),  [tower of hanoi with recursion tree presentation](https://www.geeksforgeeks.org/problems/tower-of-hanoi-1587115621/1), | | |
| .14/10/2025 | Matrix Problems I | . |
| Content Summary: [Spiral traversal](https://leetcode.com/problems/spiral-matrix/description/), [searching elements in a matrix](https://leetcode.com/problems/search-a-2d-matrix/description/), [Printing elements in sorted order.](https://leetcode.com/problems/sort-an-array/description/) | | |
| .14/10/2025 | Matrix Problems  II | . |
| Content Summary: [Finding median in row-wise sorted matrix](https://leetcode.com/problems/median-of-a-row-wise-sorted-matrix/description/), [identifying rows with maximum 1s](https://leetcode.com/problems/row-with-maximum-ones/description/) , [rotating matrices by 90 degrees.](https://leetcode.com/problems/rotate-image/) | | |
| .15/10/2025 | LinkedList  Introduction. | . |
| Content Summary: [add Node on any position](https://leetcode.com/problems/design-linked-list/description/), [delete Node from given position](https://leetcode.com/problems/delete-node-in-a-linked-list/description/), [search Node in a linked List](https://medium.com/@itsanuragjoshi/search-an-element-in-a-linked-list-iterative-and-recursive-data-structures-algorithm-0c3fad1cd1d6), [Count Node in linked List](https://www.geeksforgeeks.org/problems/count-nodes-of-linked-list/1?itm_source=geeksforgeeks&itm_medium=article&itm_campaign=practice_card) | | |
| . 15/10/2025 | LinkedList I | . |
| Content Summary: [reverse LinkedList](https://leetcode.com/problems/reverse-linked-list/description/), [find mid of the linkedList](https://leetcode.com/problems/middle-of-the-linked-list/description/), [Merge Two sorted LinkedList.](https://leetcode.com/problems/merge-two-sorted-lists/description/) | | |
| 15/10/2025 | LinkedList  II | . |
| Content Summary: [add two number](https://leetcode.com/problems/add-two-numbers/description/), [rotate list](https://leetcode.com/problems/rotate-list/description/), [remove duplicates from sorted list](https://leetcode.com/problems/remove-duplicates-from-sorted-list/description/) | | |
| 15/10/2025 | Stack Implementation | . |
| Content Summary: [Stack Implementation using Array](https://www.codechef.com/learn/course/stacks-and-queues/LSTACKS/problems/STACK03), [Next Greater Element](https://leetcode.com/problems/next-greater-element-i/description/) | | |
| 15/10/2025 | Stack I | . |
| Content Summary: [Smaller element on left](https://www.geeksforgeeks.org/problems/smaller-on-left20360700/1), [valid parentheses](https://leetcode.com/problems/valid-parentheses/description/), [Evaluate postfix expression](https://www.geeksforgeeks.org/problems/evaluation-of-postfix-expression1735/1?itm_source=geeksforgeeks&itm_medium=article&itm_campaign=practice_card) | | |
| 16/10/2025 | Stack  II | . |
| Content Summary: [min stack](https://leetcode.com/problems/min-stack/description/), [asteroid collision](https://leetcode.com/problems/asteroid-collision/description/), [stock span problem](https://leetcode.com/problems/online-stock-span/description/) | | |
| 16/10/2025 | Queue Introduction. | . |
| Content Summary: [Queue implementation using array](https://www.geeksforgeeks.org/problems/implement-queue-using-array/1), [Implement circular queue](https://leetcode.com/problems/design-circular-queue/description/), [queue using stack](https://leetcode.com/problems/implement-queue-using-stacks/description/) | | |
| 16/10/2025 | Advance Array-I | . |
| Content summary: [Two sum](https://leetcode.com/problems/two-sum/),  [Best time to buy and sell stocks](https://leetcode.com/problems/best-time-to-buy-and-sell-stock/), [Sort 0, 1 and 2(Dutch flag algorithm)](https://leetcode.com/problems/sort-colors/), | | |
| 16/10/2025 | Advance Array-II | . |
| Content Summary: [container with most water](https://leetcode.com/problems/container-with-most-water/), [merge sorted array](https://leetcode.com/problems/merge-sorted-array/), [trapping rain water](https://leetcode.com/problems/trapping-rain-water/) | | |
| 17/10/2025 | Binary Search-I | . |
| Content Summary: [lower bound](https://www.geeksforgeeks.org/implement-lower-bound/) , [upper bound](https://www.geeksforgeeks.org/upper_bound-in-cpp/), [koko eating bananas](https://leetcode.com/problems/koko-eating-bananas/), [first bad version](https://leetcode.com/problems/first-bad-version/) | | |
| 17/10/2025 | Binary Search-II |  |
| Content Summary: [Search in rotated sorted array](https://leetcode.com/problems/search-in-rotated-sorted-array/), [Search in rotated sorted array II](https://leetcode.com/problems/search-in-rotated-sorted-array-ii/), [aggressive cows](https://www.geeksforgeeks.org/problems/aggressive-cows/0) | | |
| 17/10/2025 | Binary Tree Introduction | . |
| Content Summary: Introduction of Tree, type of tree, [implementation of tree.](https://leetcode.com/problems/create-binary-tree-from-descriptions/description/) | | |
| 17/10/2025 | Binary Tree Traversal | . |
| Content Summary:  Tree Traversal, [preorder traversal,](https://leetcode.com/problems/binary-tree-preorder-traversal/) [inorder traversal](https://leetcode.com/problems/binary-tree-inorder-traversal/), [postorder traversal](https://leetcode.com/problems/binary-tree-postorder-traversal/), [level order traversal( Morris traversal ).](https://leetcode.com/problems/binary-tree-level-order-traversal/) | | |
| 17/10/2025 | Binary Tree-III. | . |
| Content Summary: [Height of the tree](https://leetcode.com/problems/maximum-depth-of-binary-tree/), [same tree](https://leetcode.com/problems/same-tree/), [symmetric tree](https://leetcode.com/problems/symmetric-tree/), | | |
| 18/10/2025 | Binary Tree-IV. | . |
| Content Summary: [diameter of tree](https://leetcode.com/problems/diameter-of-binary-tree/), [path sum](https://leetcode.com/problems/path-sum/), [print left/right view of Binary tree](https://leetcode.com/problems/binary-tree-right-side-view/). | | |
| 18/10/2025 | Binary Search Tree. | . |
| Content Summary: [Implementation of BST](https://mathcenter.oxford.emory.edu/site/cs171/probSetBst/), [check valid BST](https://leetcode.com/problems/validate-binary-search-tree/) | | |
| 18/10/2025 | Hashmap Introduction. | . |
| Content Summary:  [HashMap Implementation (operations put, get, containsKey, KeySet)](https://leetcode.com/problems/design-hashmap/description/) | | |
| 18/10/2025 | HashMap-II. | . |
| Content Summary: [Two Sum](https://leetcode.com/problems/two-sum/), [highest frequency character](https://www.geeksforgeeks.org/problems/maximum-occuring-character-1587115620/0), [missing number](https://leetcode.com/problems/missing-number/) | | |
| 18/10/2025 | HashMap-III. |  |
| Content Summary: [intersection of two arrays](https://leetcode.com/problems/intersection-of-two-arrays/), [set matrix zeros](https://leetcode.com/problems/set-matrix-zeroes/), [valid anagram](https://leetcode.com/problems/valid-anagram/) | | |
| 19/10/2025 | hashmap/Sliding window-technique Algorithm | . |
| Content Summary:[longest consecutive sequence](https://leetcode.com/problems/longest-consecutive-sequence/), [longest substring without repeating character](https://leetcode.com/problems/longest-substring-without-repeating-characters/), [bulls and cows](https://leetcode.com/problems/bulls-and-cows/description/?envType=problem-list-v2&envId=hash-table) | | |
| 19/10/2025 | hashmap/Sliding window-technique Algorithm | . |
| Content Summary: [largest subarray with 0 sum](https://www.geeksforgeeks.org/problems/largest-subarray-with-0-sum/1), [count of zero sum subarray](https://www.geeksforgeeks.org/problems/zero-sum-subarrays1825/1), [length of largest subarray with contiguous element](https://leetcode.com/problems/contiguous-array/) | | |
| 19/10/2025 | Priority Queue | . |
| Content Summary: [Implementation of Priority queue](https://www.geeksforgeeks.org/problems/operations-on-priorityqueue/1?itm_source=geeksforgeeks&itm_medium=article&itm_campaign=practice_card), [min and max Heap](https://www.naukri.com/code360/problems/build-heap_975375) | | |
| 20/10/2025 | priority Queue-II | . |
| Content Summary: [Inplace heap sort](https://www.csl.mtu.edu/cs2321/www/newLectures/09_Inplace_Heap_Sort.html), [kth largest element](https://leetcode.com/problems/kth-largest-element-in-an-array/), [kth smallest element](https://www.geeksforgeeks.org/problems/kth-smallest-element5635/1) | | |
| 20/10/2025 | priority Queue-III | . |
| Content Summary: [check max heap](https://www.geeksforgeeks.org/problems/is-binary-tree-heap/0), [top k frequent element](https://leetcode.com/problems/top-k-frequent-elements/), [sliding window maximum](https://leetcode.com/problems/sliding-window-maximum/) | | |
| 20/10/2025 | Sum up Binary tree and Binary search Tree | . |
| Content Summary: [sum of leaves](https://leetcode.com/problems/sum-of-left-leaves/), [top view](https://www.geeksforgeeks.org/problems/top-view-of-binary-tree/1), [bottom view](https://www.geeksforgeeks.org/problems/bottom-view-of-binary-tree/1), | | |
| 20/10/2025 | Sum up Hashmap / Sliding window technique. | . |
| Content Summary: [find all anagram in string](https://leetcode.com/problems/find-all-anagrams-in-a-string/description/), [isomorphic string](https://leetcode.com/problems/isomorphic-strings/) | | |
| 21/10/2025 | Bit Manipulation Introduction. | . |
| Content Summary: Introduction to AND, OR, XOR operations, [Count Set/unset Bits](https://www.geeksforgeeks.org/problems/count-total-set-bits-1587115620/1), [Toggle a given kth bit](https://www.geeksforgeeks.org/problems/toggle-bits-given-range0952/1), [check if nth bit is set or unset](https://www.geeksforgeeks.org/problems/check-whether-k-th-bit-is-set-or-not-1587115620/1),  [Check Power of Two/Four](https://leetcode.com/problems/power-of-two/description/). | | |
| 21/10/2025 | Bit Manipulation-II. | . |
| Content Summary:[Counting bits,](https://leetcode.com/problems/counting-bits/description/)  [Single Number 1](https://leetcode.com/problems/single-number/description/), [Single number 2,](https://leetcode.com/problems/single-number-ii/description/)  [Subsets using Bits ( power set problem)](https://leetcode.com/problems/subsets/description/) , [Find Missing number](https://leetcode.com/problems/missing-number/description/), [Duplicate Numbers.](https://leetcode.com/problems/find-the-duplicate-number/description/) | | |
| 21/10/2025 | Number theory basics. | . |
| Content Summary: [Sieve of Eratosthenes](https://leetcode.com/problems/count-primes/description/), [Modular Arithmetic](https://leetcode.com/problems/smallest-integer-divisible-by-k/description/), [Modular Exponentiation](https://leetcode.com/problems/super-pow/description/), [Chinese Remainder Theorem](https://leetcode.com/problems/super-pow/description/) | | |
| 21/10/2025 | Mathematical Algorithms. | . |
| Content Summary: [Euler’s Totient Function](https://cp-algorithms.com/algebra/phi-function.html), [Permutations and Combinations](https://leetcode.com/problems/permutations/description/), [Inclusion-Exclusion Principle,](https://cp-algorithms.com/combinatorics/inclusion-exclusion.html) [Catalan Numbers.](https://leetcode.com/problems/unique-binary-search-trees/description/) | | |
| 22/10/2025 | Advance Recursion. | . |
| Content Summary: [print all subset](https://leetcode.com/problems/subsets/description/), [permutation of a string](https://leetcode.com/problems/permutations/description/), [find all unique subset](https://www.geeksforgeeks.org/problems/subset-sum-ii/1?itm_source=geeksforgeeks&itm_medium=article&itm_campaign=practice_card) | | |
| 22/10/2025 | Backtracking I | . |
| Content Summary: [rat in maze](https://leetcode.com/problems/escape-a-large-maze/description/), [rat in a maze all path](https://leetcode.com/problems/the-maze-ii/description/), [N Queens](https://leetcode.com/problems/n-queens/description/) | | |
| 22/10/2025 | Backtracking-2 | . |
| Content Summary: [combination](https://leetcode.com/problems/combinations/description/), [combination sum](https://leetcode.com/problems/combination-sum/description/), [combination sum-2](https://leetcode.com/problems/combination-sum-ii/description/) | | |
| 23/10/2025 | Backtracking-3 | . |
| Content Summary: [generate parentheses](https://leetcode.com/problems/generate-parentheses/description/), [subset-2](https://leetcode.com/problems/subsets-ii/description/), [sudoku solver](https://leetcode.com/problems/sudoku-solver/description/) | | |
| 23/10/2025 | Greedy I | . |
| Content Summary: [assign cookies](https://leetcode.com/problems/assign-cookies/description/), [array partition](https://leetcode.com/problems/array-partition/description/), [can place flower](https://leetcode.com/problems/can-place-flowers/description/), [lemonade change](https://leetcode.com/problems/lemonade-change/description/) | | |
| 23/10/2025 | Greedy-II. | . |
| Content Summary: [Activity selection](https://www.geeksforgeeks.org/problems/activity-selection-1587115620/1), [minimum platform](https://www.geeksforgeeks.org/problems/minimum-platforms-1587115620/1), [coin change](https://leetcode.com/problems/coin-change/description/) | | |
| 23/10/2025 | Greedy-III. | . |
| Content Summary: [max chunk to make sorted](https://leetcode.com/problems/max-chunks-to-make-sorted/description/), [max chunk to make sorted-2](https://leetcode.com/problems/max-chunks-to-make-sorted-ii/description/), [0/1 knapsack](https://www.geeksforgeeks.org/problems/0-1-knapsack-problem0945/1). | | |
| 24/10/2025 | Graph Introduction and representation. | . |
| Content Summary: Introduction,Representation using adjacency matrix and [adjacency list](https://www.geeksforgeeks.org/problems/print-adjacency-list-1587115620/1) | | |
| 24/10/2025 | Graph-Traversal Algorithm. | . |
| Content Summary: [Graph Traversal BFS(Breadth first search)](https://www.geeksforgeeks.org/problems/bfs-traversal-of-graph/1) and [DFS(Depth first search)](https://www.geeksforgeeks.org/problems/depth-first-traversal-for-a-graph/1?itm_source=geeksforgeeks&itm_medium=article&itm_campaign=practice_card) | | |
| 24/10/2025 | Graph-III | . |
| Content Summary : [Connected Components](https://neetcode.io/problems/count-connected-components), [Detecting Cycles in Graphs](https://www.geeksforgeeks.org/problems/detect-cycle-in-a-directed-graph/1?itm_source=geeksforgeeks&itm_medium=article&itm_campaign=practice_card) | | |
| 25/10/2025 | Graph Problems-IV. | . |
| Content summary: [find if path exist(has path)](https://leetcode.com/problems/find-if-path-exists-in-graph/description/), [print all path from source to destination](https://leetcode.com/problems/all-paths-from-source-to-target/description/), [Number of Island](https://leetcode.com/problems/number-of-islands/description/) | | |
| 25/10/2025 | Advanced Graph. | . |
| Content summary: [Number of Provinces](https://leetcode.com/problems/number-of-provinces/description/), [Flood Fill](https://leetcode.com/problems/flood-fill/description/),  [Number of closed islands.](https://leetcode.com/problems/number-of-closed-islands/) | | |
| 25/10/2025 | Minimum Spanning Tree algorithms. | . |
| Content summary: [Prim’s Algorithm](https://cp-algorithms.com/graph/mst_prim.html),  [Kruskal's algorithm](https://cp-algorithms.com/graph/mst_kruskal.html). | | |
| 26/10/2025 | Shortest Path Algorithm. | . |
| Content summary: [Dijkstra algorithm](https://cp-algorithms.com/graph/dijkstra.html), [Bellman ford algorithm](https://cp-algorithms.com/graph/bellman_ford.html). | | |
| 26/10/2025 | Trie | . |
| Content summary: what is trie DS, use of trie, hashmap vs trie, [implementation(representation,  insert node, search node)](https://leetcode.com/problems/implement-trie-prefix-tree/description/) | | |
| 26/10/2025 | Trie-II | . |
| Content Summary: [delete node](https://www.geeksforgeeks.org/trie-delete/), [application of trie](https://www.geeksforgeeks.org/applications-advantages-and-disadvantages-of-trie/), [count word in trie](https://www.geeksforgeeks.org/counting-number-words-trie/), [word break](https://leetcode.com/problems/word-break/), | | |
| 26/10/2025 | Huffman coding | . |
| Content Summary: [huffman coding algorithm](https://www.geeksforgeeks.org/problems/huffman-encoding3345/1?itm_source=geeksforgeeks&itm_medium=article&itm_campaign=practice_card), decompression in huffman coding | | |
| 27/10/2025 | Dynamic programming | . |
| Content Summary: [introduction of dynamic programing](https://cp-algorithms.com/dynamic_programming/intro-to-dp.html), covering of prerequisites for dynamic programming, discussion of memoization and tabulation using [fibonacci number](https://leetcode.com/problems/fibonacci-number/)/ any problem | | |
| 27/10/2025 | Dynamic programming-II (both approach memoization and tabulation) | . |
| Content Summary: [staircase](https://leetcode.com/problems/climbing-stairs/), [min cost climbing stairs](https://leetcode.com/problems/min-cost-climbing-stairs/), [counting bits](https://leetcode.com/problems/counting-bits/), [perfect square](https://leetcode.com/problems/perfect-squares/) | | |
| 28/10/2025 | Dynamic programming-III(both approach memoization and tabulation) | . |
| Content Summary: [house robber](https://leetcode.com/problems/house-robber/), [house robber 2](https://leetcode.com/problems/house-robber-ii/), [goldmine](https://www.geeksforgeeks.org/problems/gold-mine-problem2608/1), [path with maximum gold](https://leetcode.com/problems/path-with-maximum-gold/) | | |
| 28/10/2025 | Dynamic programming-IV | . |
| Content Summary: [unique path-1](https://leetcode.com/problems/unique-paths/), [unique path-2](https://leetcode.com/problems/unique-paths-ii/), [minimum path sum](https://leetcode.com/problems/minimum-path-sum/), [target sum subset (DP)](https://leetcode.com/problems/target-sum/) | | |
| 29/10/2025 | Dynamic programming-V. | . |
| Content Summary: [coin change](https://leetcode.com/problems/coin-change/), [coin change-2](https://leetcode.com/problems/coin-change-ii/), [jump game](https://leetcode.com/problems/jump-game/), [jump game 2](https://leetcode.com/problems/jump-game-ii/) | | |
| 29/10/2025 | Dynamic programming-VI. | . |
| Content Summary: [longest common subsequences(recursive, memoization tabulation)](https://leetcode.com/problems/longest-common-subsequence/), [longest palindrome substring,](https://leetcode.com/problems/longest-palindromic-substring/) | | |
| 30/10/2025 | Dynamic programming-VII. | . |
| Content Summary: [longest palindrome subsequences](https://leetcode.com/problems/longest-palindromic-subsequence/), [palindromic substring](https://leetcode.com/problems/palindromic-substrings/) | | |
| 31/10/2025 | Dynamic programming-VIII. | . |
| Content Summary: [wildcard matching problem](https://leetcode.com/problems/wildcard-matching/), [egg dropping problem](https://leetcode.com/problems/super-egg-drop/) | | |
| 01/11/2025 | Revised Day on DP | . |
| Content Summary: [regular expression matching problem](https://leetcode.com/problems/regular-expression-matching/) | | |
| 01/11/2025 | Segment tree |  |
| Content Summary: [what is segment tree, what kind of problem we solve using segment tree,](https://cp-algorithms.com/data_structures/segment_tree.html) [max in interval](https://leetcode.com/discuss/interview-question/algorithms/199350/how-to-find-the-largest-number-in-an-interval), | | |
| 02/11/2025 | Segment tree-II/ range query | . |
| Content Summary: [sum of interval](https://cp-algorithms.com/data_structures/segment_tree.html#sum-queries), [sum of ranges(lazy propagation)](https://www.hackerearth.com/practice/notes/segment-tree-and-lazy-propagation/) | | |

**(Completion Date: 03/11/2025 to 10/11/2025)**

**1- N Queen Problem**

The N Queen is the problem of placing N chess queens on an N×N chessboard so that no two queens attack each other. Queen can attack horizontally, vertically and diagonaly.For example, following are two solutions for 4 Queen Problem





<https://www.geeksforgeeks.org/printing-solutions-n-queen-problem/?ref=lbp>

**2-K Battle Ship**

You are given NxM matrix. You have K battle ships. You are also given 3 parameters for each of the battle ship - rowAttack, columnAttack and diagonalAttack. Each telling how many cells in a row/column/diagonal(on each direction) the ship can attack. Your task is find whether the ships can be placed in the board such that no one attacks each other.

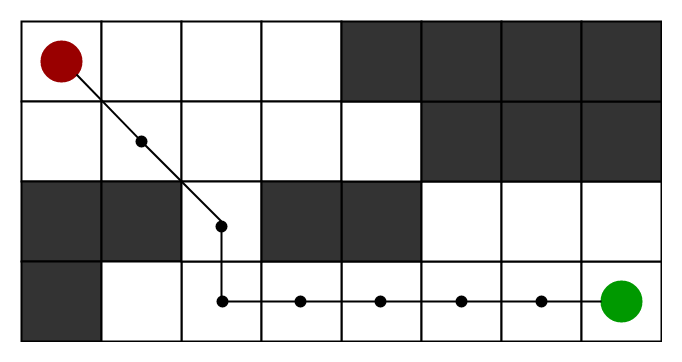
To shed more light on the attacks, say if the ship is placed at (5,5) and if it's column attack range is 3, then it can attack any ship from (5,2) to (5,8).

Link <https://stackoverflow.com/questions/23726791/variation-of-n-queens>

**3- A\* Search Algorithm**

To approximate the shortest path in real-life situations, like- in maps, games where there can be many hindrances.

We can consider a 2D Grid having several obstacles and we start from a source cell (coloured red below) to reach towards a goal cell (coloured green below)



**What is A\* Search Algorithm?**  
A\* Search algorithm is one of the best and popular technique used in path-finding and graph traversals.

Informally speaking, A\* Search algorithms, unlike other traversal techniques, it has “brains”. What it means is that it is really a smart algorithm which separates it from the other conventional algorithms. This fact is cleared in detail in below sections.  
And it is also worth mentioning that many games and web-based maps use this algorithm to find the shortest path very efficiently (approximation).

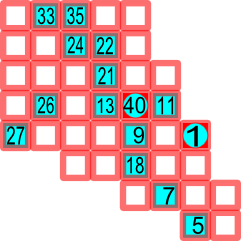


<https://www.geeksforgeeks.org/a-search-algorithm/>

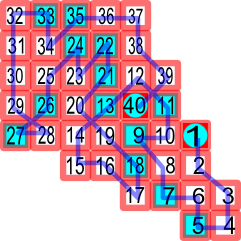
**4- Hidato Puzzle**

The rules are:

* You are given a grid with some numbers placed in it. The other squares in the grid will be blank.
  + The grid is not necessarily rectangular.
  + The grid may have holes in it.
  + The grid is always connected.
  + The number “1” is always present, as is another number that is equal to the number of squares in the grid. Other numbers are present so as to force the solution to be unique.
  + It may be assumed that the difference between numbers present on the grid is not greater than lucky 13.
* The aim is to place a natural number in each blank square so that in the sequence of numbered squares from “1” upwards, each square is in the [wp:Moore neighborhood](https://en.wikipedia.org/wiki/Moore_neighborhood) of the squares immediately before and after it in the sequence (except for the first and last squares, of course, which only have one-sided constraints).
  + Thus, if the grid was overlaid on a chessboard, a king would be able to make legal moves along the path from first to last square in numerical order.
  + A square may only contain one number.
* In a proper Hidato puzzle, the solution is unique.



Solution



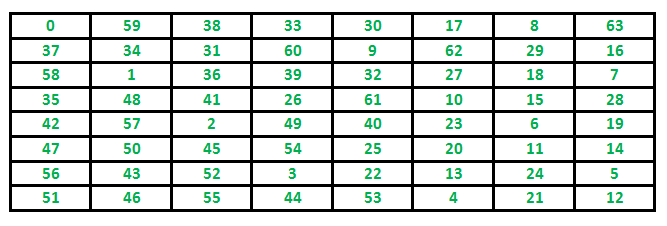
<https://rosettacode.org/wiki/Solve_a_Hidato_puzzle>

**5- Knight Tour Problem**

*the knight is placed on the first block of an empty board and, moving according to the rules of chess, must visit each square exactly once.*

**Path followed by Knight to cover all the cells**

Following is chessboard with 8 x 8 cells. Numbers in cells indicate move number of Knight.



<https://www.geeksforgeeks.org/the-knights-tour-problem-backtracking-1/>

**6- Numberix Puzzle**

Numbrix puzzles are similar to [Hidato](https://rosettacode.org/wiki/Solve_a_Hidato_puzzle). The most important difference is that it is only possible to move 1 node left, right, up, or down (sometimes referred to as the [Von Neumann neighborhood](https://en.wikipedia.org/wiki/Von_Neumann_neighborhood)). Published puzzles also tend not to have holes in the grid and may not always indicate the end node. Two examples follow:

0 0 0 0 0 0 0 0 0

0 0 46 45 0 55 74 0 0

0 38 0 0 43 0 0 78 0

0 35 0 0 0 0 0 71 0

0 0 33 0 0 0 59 0 0

0 17 0 0 0 0 0 67 0

0 18 0 0 11 0 0 64 0

0 0 24 21 0 1 2 0 0

0 0 0 0 0 0 0 0 0

Solution

49 50 51 52 53 54 75 76 81

48 47 46 45 44 55 74 77 80

37 38 39 40 43 56 73 78 79

36 35 34 41 42 57 72 71 70

31 32 33 14 13 58 59 68 69

30 17 16 15 12 61 60 67 66

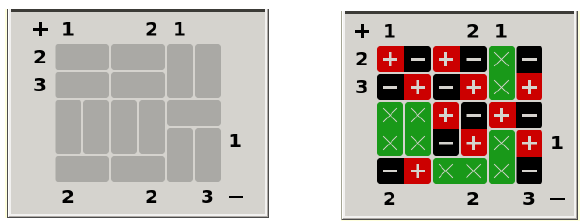
29 18 19 20 11 62 63 64 65

28 25 24 21 10 1 2 3 4

27 26 23 22 9 8 7 6 5

**7- Magnet Puzzle**

The puzzle game Magnets involves placing a set of domino-shaped magnets (or electrets or other polarized objects) in a subset of slots on a M X N board so as to satisfy a set of constraints. For example, the puzzle on the left has the solution shown on the right



Each slot contains either a blank entry (indicated by ‘x’s), or a “magnet” with a positive and negative end. The numbers along the left and top sides show the numbers of ‘+’ squares in particular rows or columns. Those along the right and bottom show the number of ‘-’ signs in particular rows or columns. Rows and columns without a number at one or both ends are unconstrained as to the number of ‘+’ or ‘-’ signs, depending on which number is not present. In addition to fulfilling these numerical constraints, a puzzle solution must also satisfy the constraint that no two orthogonally touching squares may have the same sign (diagonally joined squares are not constrained).

You are given top[], bottom[], left[], right[] arrays indicates the count of + or – along the top(+), bottom(-), left(+) and right(-) edges respectively. Values of -1 indicate any number of + and – signs. Also given matrix rules[][] contain any one T, B, L or R characters. For a vertical slot in the board, T indicates its top end and B indicates the bottom end. For a horizontal slot in the board, L indicates left end and R indicates the right end.

input : M = 5, N = 6

top[] = { 1, -1, -1, 2, 1, -1 }

bottom[] = { 2, -1, -1, 2, -1, 3 }

left[] = { 2, 3, -1, -1, -1 }

right[] = { -1, -1, -1, 1, -1 }

rules[][] = { { L, R, L, R, T, T },

{ L, R, L, R, B, B },

{ T, T, T, T, L, R },

{ B, B, B, B, T, T },

{ L, R, L, R, B, B }};

Output : + - + - X -

- + - + X +

X X + - + -

X X - + X +

- + X X X -

Input : M = 4, N = 3

top[] = { 2, -1, -1 }

bottom[] = { -1, -1, 2 }

left[] = { -1, -1, 2, -1 }

right[] = { 0, -1, -1, -1 }

rules[][] = { { T, T, T },

{ B, B, B },

{ T, L, R },

{ B, L, R } };

Output : + X +

– X –

+ – +

– + –

**8- Rat In a Maze**

A Maze is given as N\*N binary matrix of blocks where source block is the upper left most block i.e., maze[0][0] and destination block is lower rightmost block i.e., maze[N-1][N-1]. A rat starts from source and has to reach the destination. The rat can move only in two directions: forward and down.  
In the maze matrix, 0 means the block is a dead end and 1 means the block can be used in the path from source to destination. Note that this is a simple version of the typical Maze problem. For example, a more complex version can be that the rat can move in 4 directions and a more complex version can be with a limited number of moves.



Solution



**9- Print all shortest route in a grid**

Given a matrix of N\*M order. Find the shortest distance from a source cell to a destination cell, traversing through limited cells only. Also you can move only up, down, left and right. If found output the distance else -1. s represents ‘source’ d represents ‘destination’ \* represents cell you can travel 0 represents cell you can not travel This problem is meant for single source and destination.

<https://www.techiedelight.com/print-triplets-array-sum-less-equal-given-number/>

**10- Print longest route in a grid**

<https://www.techiedelight.com/find-longest-possible-route-matrix/>

**11- Find total number of unique path in maze from source to destination**

<https://www.techiedelight.com/find-total-number-unique-paths-maze-source-destination/>

**12- Subset Sum Problem**

Subset sum problem is to find subset of elements that are selected from a given set whose sum adds up to a given number K. We are considering the set contains non-negative values. It is assumed that the input set is unique (no duplicates are presented)

<https://www.geeksforgeeks.org/subset-sum-backtracking-4/>

**13- Coin Change Problem**

Given a value N, if we want to make change for N cents, and we have infinite supply of each of S = { S1, S2, .. , Sm} valued coins, how many ways can we make the change? The order of coins doesn’t matter.

For example, for N = 4 and S = {1,2,3}, there are four solutions: {1,1,1,1},{1,1,2},{2,2},{1,3}. So output should be 4. For N = 10 and S = {2, 5, 3, 6}, there are five solutions: {2,2,2,2,2}, {2,2,3,3}, {2,2,6}, {2,3,5} and {5,5}. So the output should be 5.

<https://www.geeksforgeeks.org/coin-change-dp-7/>

**14- Cutting a Rod**

Given a rod of length n inches and an array of prices that contains prices of all pieces of size smaller than n. Determine the maximum value obtainable by cutting up the rod and selling the pieces. For example, if length of the rod is 8 and the values of different pieces are given as following, then the maximum obtainable value is 22 (by cutting in two pieces of lengths 2 and 6)

length | 1 2 3 4 5 6 7 8

--------------------------------------------

price | 1 5 8 9 10 17 17 20

And if the prices are as following, then the maximum obtainable value is 24 (by cutting in eight pieces of length 1)

length | 1 2 3 4 5 6 7 8

--------------------------------------------

price | 3 5 8 9 10 17 17 20

<https://www.geeksforgeeks.org/cutting-a-rod-dp-13/>

.

**15- Print all triplets in an array with sum is less than or equal to given number**

<https://www.techiedelight.com/print-triplets-array-sum-less-equal-given-number/>

**16- K Partition Probem**

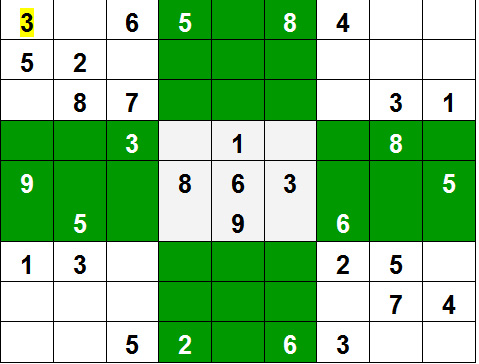
<https://www.techiedelight.com/k-partition-problem-print-all-subsets/>

**17- find ways to calculate a target from array use addition and substraction operator**

<https://www.techiedelight.com/find-ways-calculate-target-elements-array/>

**18- Sudoku Problem**

Given a partially filled 9×9 2D array ‘grid[9][9]’, the goal is to assign digits (from 1 to 9) to the empty cells so that every row, column, and subgrid of size 3×3 contains exactly one instance of the digits from 1 to 9.



<https://www.geeksforgeeks.org/sudoku-backtracking-7/>

**19- longest common subsequence**

Given two sequences, print the longest subsequence present in both of them.

**Examples:**  
LCS for input Sequences “ABCDGH” and “AEDFHR” is “ADH” of length 3.  
LCS for input Sequences “AGGTAB” and “GXTXAYB” is “GTAB” of length 4.

<https://www.geeksforgeeks.org/printing-longest-common-subsequence/>

**20- longest palindromic subsequence**

Given a sequence, print a longest palindromic subsequence of it.  
**Examples :**

Input : BBABCBCAB

Output : BABCBAB

The above output is the longest

palindromic subsequence of given

sequence. "BBBBB" and "BBCBB" are

also palindromic subsequences of

the given sequence, but not the

longest ones.

Input : GEEKSFORGEEKS

Output : Output can be either EEKEE

or EESEE or EEGEE, ..

**21- longest increasing subsequence**

The Longest Increasing Subsequence (LIS) problem is to find the length of the longest subsequenceof a given sequence such that all elements of the subsequence are sorted in increasing order.

For example, the length of LIS for {10, 22, 9, 33, 21, 50, 41, 60, 80} is 6 and LIS is {10, 22, 33, 50, 60, 80}.

<https://www.geeksforgeeks.org/construction-of-longest-increasing-subsequence-using-dynamic-programming/>

**22- Text Segmentation**

Given a valid sentence without any spaces between the words and a dictionary of valid English words, find all possible ways to break the sentence in individual dictionary words.

**Example**

Consider the following dictionary

{ i, like, sam, sung, samsung, mobile, ice,

cream, icecream, man, go, mango}

Input: "ilikesamsungmobile"

Output: i like sam sung mobile

i like samsung mobile

Input: "ilikeicecreamandmango"

Output: i like ice cream and man go

i like ice cream and mango

i like icecream and man go

i like icecream and mango

<https://www.geeksforgeeks.org/word-break-problem-using-backtracking/?ref=lbp>

**23- Edit Distance**

Given two strings str1 and str2 and below operations that can performed on str1. Find minimum number of edits (operations) required to convert ‘str1’ into ‘str2’.

1. Insert
2. Remove
3. Replace

All of the above operations are of equal cost.  
 **Examples:**

Input: str1 = "geek", str2 = "gesek"

Output: 1

We can convert str1 into str2 by inserting a 's'.

**24- Edit Distance**

Given two strings str1 and str2 and below operations that can performed on str1. Find minimum number of edits (operations) required to convert ‘str1’ into ‘str2’.

1-Insert

2-Remove

All of the above operations are of equal cost.  
 **Examples:**

Input: str1 = "geek", str2 = "gesek"

Output: 1

We can convert str1 into str2 by inserting a 's'.

**25- Optimal BST**

Given a sorted array keys[0.. n-1] of search keys and an array freq[0.. n-1] of frequency counts, where freq[i] is the number of searches to keys[i]. Construct a binary search tree of all keys such that the total cost of all the searches is as small as possible.

Let us first define the cost of a BST. The cost of a BST node is level of that node multiplied by its frequency. Level of root is 1.

**Examples:**

Input: keys[] = {10, 12}, freq[] = {34, 50}

There can be following two possible BSTs

10 12

\ /

12 10

I II

Frequency of searches of 10 and 12 are 34 and 50 respectively.

The cost of tree I is 34\*1 + 50\*2 = 134

The cost of tree II is 50\*1 + 34\*2 = 118

<https://www.geeksforgeeks.org/optimal-binary-search-tree-dp-24/>

<https://www.techiedelight.com/find-optimal-cost-to-construct-binary-search-tree/>

**26- Graph Coloring**

Given an undirected graph and a number m, determine if the graph can be coloured with at most m colours such that no two adjacent vertices of the graph are colored with the same color. Here coloring of a graph means the assignment of colors to all vertices.  
**Input-Output format:**  
***Input:***

1. A 2D array graph[V][V] where V is the number of vertices in graph and graph[V][V] is adjacency matrix representation of the graph. A value graph[i][j] is 1 if there is a direct edge from i to j, otherwise graph[i][j] is 0.
2. An integer m which is the maximum number of colors that can be used.

***Output:***  
An array color[V] that should have numbers from 1 to m. color[i] should represent the color assigned to the ith vertex. The code should also return false if the graph cannot be colored with m colors.

**Example:**

**Input:**

graph = {0, 1, 1, 1},

{1, 0, 1, 0},

{1, 1, 0, 1},

{1, 0, 1, 0}

**Output:**

Solution Exists:

Following are the assigned colors

1 2 3 2

**Explanation:** By coloring the vertices

with following colors, adjacent

vertices does not have same colors

<https://www.geeksforgeeks.org/m-coloring-problem-backtracking-5/>

**27- find all topological ordering in DAG**

<https://www.techiedelight.com/find-all-possible-topological-orderings-of-dag/>

**28- find all path between two vertices in graph**

<https://www.techiedelight.com/find-path-between-vertices-directed-graph/>

**29- hamiltonian cycle**

<https://www.techiedelight.com/print-all-hamiltonian-path-present-in-a-graph/>

**30- find minimum number possible by doing atmost k swaps**

<https://www.techiedelight.com/find-minimum-number-possible-k-swaps/>

**31- generate list of possible words from given matrix**

<https://www.techiedelight.com/generate-list-of-possible-words-from-a-character-matrix/>

**32- All combinaation of elements satihfying given constraints**

<https://www.techiedelight.com/find-combinations-of-elements-satisfies-given-constraints/>

**33- find all binary string from a given wildcard pattern**

<https://www.techiedelight.com/find-binary-strings-can-formed-given-wildcard-pattern/>

**34- find all possibe permutaton of given string**

<https://www.techiedelight.com/find-permutations-given-string/>