Must do Interview questions gfgf-

Preparing for Prouct-Based Companies ? Check

<https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/>

**Topic :**

* [Arrays](https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/#arrays)
* [String](https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/#string)
* [Linked List](https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/#LinkedList)
* [Stack and Queue](https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/#stack)
* [Tree and BST](https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/#trees)
* [Heap](https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/#heap)
* [Recursion](https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/#recursion)
* [Hashing](https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/#hashing)
* [Graph](https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/#graph)
* [Greedy](https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/#Greedy)
* [Dynamic Programming](https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/#DP)
* [Divide and Conquer](https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/#DC)
* [Backtracking](https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/#BT)
* [Bit Magic](https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/#bits)

**Arrays :**

1. [Subarray with given sum](https://practice.geeksforgeeks.org/problems/subarray-with-given-sum/0)
2. [Count the triplets](https://practice.geeksforgeeks.org/problems/count-the-triplets/0)
3. [Kadane’s Algorithm](https://practice.geeksforgeeks.org/problems/kadanes-algorithm/0)
4. [Missing number in array](https://practice.geeksforgeeks.org/problems/missing-number-in-array/0)
5. [Merge two sorted arrays](https://practice.geeksforgeeks.org/problems/merge-two-sorted-arrays/0/)
6. [Rearrange array alternatively](https://practice.geeksforgeeks.org/problems/-rearrange-array-alternately/0/)
7. [Number of pairs](https://practice.geeksforgeeks.org/problems/number-of-pairs/0/)
8. [Inversion of Array](https://practice.geeksforgeeks.org/problems/inversion-of-array/0/)
9. [Sort an array of 0s, 1s and 2s](https://practice.geeksforgeeks.org/problems/sort-an-array-of-0s-1s-and-2s/0)
10. [Equilibrium point](https://practice.geeksforgeeks.org/problems/equilibrium-point/0)
11. [Leaders in an array](https://practice.geeksforgeeks.org/problems/leaders-in-an-array/0)
12. [Minimum Platforms](https://practice.geeksforgeeks.org/problems/minimum-platforms/0)
13. [Reverse array in groups](https://practice.geeksforgeeks.org/problems/reverse-array-in-groups/0)
14. [K’th smallest element](https://practice.geeksforgeeks.org/problems/kth-smallest-element/0)
15. [Trapping Rain Water](https://practice.geeksforgeeks.org/problems/trapping-rain-water/0)
16. [Pythagorean Triplet](https://practice.geeksforgeeks.org/problems/pythagorean-triplet/0)
17. [Chocolate Distribution Problem](https://practice.geeksforgeeks.org/problems/chocolate-distribution-problem/0)
18. [Stock buy and sell](https://practice.geeksforgeeks.org/problems/stock-buy-and-sell/0)
19. [Element with left side smaller and right side greater](https://practice.geeksforgeeks.org/problems/unsorted-array/0)
20. [Convert array into Zig-Zag fashion](https://practice.geeksforgeeks.org/problems/convert-array-into-zig-zag-fashion/0)
21. [Last Index of 1](https://practice.geeksforgeeks.org/problems/last-index-of-1/0)
22. [Spirally traversing a matrix](https://practice.geeksforgeeks.org/problems/spirally-traversing-a-matrix/0)
23. [Largest Number formed from an Array](https://practice.geeksforgeeks.org/problems/largest-number-formed-from-an-array/0)

**Solved the above?** [Go for some more Questions](https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/#More%20Questions%20on%20Arrays)

**Some More Questions on Arrays :**

1. [Find Missing And Repeating](https://practice.geeksforgeeks.org/problems/find-missing-and-repeating/0)
2. [Maximum Index](https://practice.geeksforgeeks.org/problems/maximum-index/0)
3. [Consecutive 1’s not allowed](https://practice.geeksforgeeks.org/problems/consecutive-1s-not-allowed/0)
4. [Majority Element](https://practice.geeksforgeeks.org/problems/majority-element/0)
5. [Two numbers with sum closest to zero](https://practice.geeksforgeeks.org/problems/two-numbers-with-sum-closest-to-zero/0)
6. [Nuts and Bolts Problem](https://practice.geeksforgeeks.org/problems/nuts-and-bolts-problem/0)
7. [Boolean Matrix Problem](https://practice.geeksforgeeks.org/problems/boolean-matrix-problem/0)
8. [Smallest Positive missing number](https://practice.geeksforgeeks.org/problems/smallest-positive-missing-number/0)
9. [Jumping Caterpillars](https://practice.geeksforgeeks.org/problems/jumping-caterpillars/0)

String :

Idea – To solve string questions think of these approaches to apply –

* Using map
* Using sliding window techniques
* Using DP if its variation of LCS or palindromic probllems

Almost all the questions based variations of above approaches !!!!!!

1. [Permutations of a given string](https://practice.geeksforgeeks.org/problems/permutations-of-a-given-string/0) -- notebook **(recursion+swapping**)
   1. Using recursion .
   2. For each level l , swap(arr[i], arr[l])
   3. Then permute(arr,n, l+1);
   4. Again swap(arr[i], arr[l]);
2. [Longest Palindrome in a String](https://practice.geeksforgeeks.org/problems/longest-palindrome-in-a-string/0) -- **using even, odd length palindromes** appoach
   1. 1st way using LCS
   2. Without using extra space O(1) space and O(n2) time
      * Find the max of all odd length palindromes by fixing I i.e
      * For each ‘i’ consider i1=i-1 , i2= i+1;
      * Find the max of all even length palindromes by fixing I i.e
      * For each ‘I’ consider i1=i-1, i2= I;
      * Then take max of (odd\_length, even\_length ) and return .
3. [Recursively remove all adjacent duplicates](https://practice.geeksforgeeks.org/problems/recursively-remove-all-adjacent-duplicates/0) -- using res string as stack way .
   1. Done iterative way ---
      * Firstly take one empty string res. // treat it as stack
      * For each char in string s , do
        1. If res size is not empty and str[i] matches with res.back() // last elemt on res string then
        2. do res.pop\_back() // pop the last element .
        3. else if s[i] matches s[i-1] then do nothing
        4. otherwise push char in res string .
        5. return res;
4. [Check if string is rotated by two places](https://practice.geeksforgeeks.org/problems/check-if-string-is-rotated-by-two-places/0)
   1. Just check for both cases Anticloswise rotated( shifted to left), Clockwise roated (shifted to right)
   2. Elements will shifted by +2 position so can do accordingly.
   3. ------ Other way ------
   4. Convert back rotated string to original and then compare with original string if equal return true.
5. [Roman Number to Integer](https://practice.geeksforgeeks.org/problems/roman-number-to-integer/0)

I 1 V 5 X 10 L 50 C 100 D 500 M 1000

* First put above standard symbols into map ‘m’ with its number.
* After that scan from last each char and check if
  + M[str[i]]< m[str[i+1] then sum-= m[str[i]]; as - XIV=14
  + Otherwise just add value to sum .

1. [Remove Duplicates](https://practice.geeksforgeeks.org/problems/remove-duplicates/0)

* Using visited[256] . // make use visited array. Very easy

1. [Form a Palindrome](https://practice.geeksforgeeks.org/problems/form-a-palindrome/0) --- variation of LCS

* Find the lcs of( str , reverse(str));
* Then just return ( string length - lcs)

1. [Longest Distinct Characters in the string](https://practice.geeksforgeeks.org/problems/longest-distinct-characters-in-string/0) ( length of longest substring without repeating characters)
   1. Approch one : all substrings + set ==🡺 O(n2).
   2. Just traverse string from left to right n keep track of distict chars Using map< character , it’s index>
   3. If char is already present in map then update ‘i ‘ as last index of repeating elemnt . so that we can start scanning from after that element and also clear the map. ( ex – aewergrththy --- handle this case).
   4. Keep taking res= max(res, map.size()).
2. [Implement Atoi](https://practice.geeksforgeeks.org/problems/implement-atoi/1)
   1. Scan string from left to right and do below for each char
   2. Just do res = res \*10 + (str[i]-‘0’) //initialize res=0;
   3. For taking sign also . check if first char is ‘-‘ then set flag=-1;
   4. Return (res \* flag) ------to return number with sign.
3. [Longest Common Prefix](https://practice.geeksforgeeks.org/problems/longest-common-prefix-in-an-array/0)

* Consider array of strings as 2D matrix and then check each columnwise , for each col its every row char should be same keep count++; or adding char to res.

**Solved the above?** [Go for some more Questions](https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/#More%20Questions%20on%20Strings)

**Some More Questions on Strings :**

1. [Most frequent word in an array of strings](https://practice.geeksforgeeks.org/problems/most-frequent-word-in-an-array-of-strings/0)
   * + Easy way using map<string,int> // string : frequency
     + But make sure same element from should not be consider again while taking res . so to avoid that once element’s frequency is read from map then decrease its freuencvy .
     + \*\*\*M[arr[i]]—after considering arr[i] --- for every i.
2. [CamelCase Pattern Matching](https://practice.geeksforgeeks.org/problems/camelcase-pattern-matching/0) ---
   * + 1. Usinhg hashing for each combination of camelcase
3. [String Ignorance](https://practice.geeksforgeeks.org/problems/string-ignorance/0)
   * 1. Increment the count of occurrence of current character in a hash table.
     2. Check if the count becomes odd, then print the current character, else not.
4. [Smallest window in a string containing all the characters of another string](https://practice.geeksforgeeks.org/problems/smallest-window-in-a-string-containing-all-the-characters-of-another-string/0)
   * + Using sliding window technique (notebook)
5. [Design a tiny URL or URL shortener](https://practice.geeksforgeeks.org/problems/design-a-tiny-url-or-url-shortener/0) (from integer id to url)
   * + Convert the ID to base 62 number then use map.
     + First store map-
     + // Map to store 62 possible characters
     + char map[] = "abcdefghijklmnopqrstuvwxyzABCDEF"

* "GHIJKLMNOPQRSTUVWXYZ0123456789";
  + - Then convert given integer id to base 62
* while (n)
* {
* // use above map to store actual character
* // in short url
* shorturl.push\_back(map[n%62]);
* n = n/62;
* }
  + - After this reverse shorturl string and return.
    - If we wanted to get back original id from url then –
* / Function to get integer ID back from a short url
* long int shortURLtoID(string shortURL)
* {
* long int id = 0; // initialize result
* // A simple base conversion logic
* for (int i=0; i < shortURL.length(); i++)
* {
* if ('a' <= shortURL[i] && shortURL[i] <= 'z')
* id = id\*62 + shortURL[i] - 'a';
* if ('A' <= shortURL[i] && shortURL[i] <= 'Z')
* id = id\*62 + shortURL[i] - 'A' + 26;
* if ('0' <= shortURL[i] && shortURL[i] <= '9')
* id = id\*62 + shortURL[i] - '0' + 52;
* }
* return id;
* }

1. [Remove common characters and concatenate](https://practice.geeksforgeeks.org/problems/remove-common-characters-and-concatenate/0)

* Using hashing ---map

1. [Geek and its Colored Strings](https://practice.geeksforgeeks.org/problems/geek-and-its-colored-strings/0)
2. [Second most repeated string in a sequence](https://practice.geeksforgeeks.org/problems/second-most-repeated-string-in-a-sequence/0)
   * Using map to store string and its count then find first\_max and second\_max from map---do the logic its easy one;

**Linked List :**

1. [Finding middle element in a linked list](https://practice.geeksforgeeks.org/problems/finding-middle-element-in-a-linked-list/1)
2. [Reverse a linked list](https://practice.geeksforgeeks.org/problems/reverse-a-linked-list/1)
3. [Rotate a Linked List](https://practice.geeksforgeeks.org/problems/rotate-a-linked-list/1)
4. [Reverse a Linked List in groups of given size](https://practice.geeksforgeeks.org/problems/reverse-a-linked-list-in-groups-of-given-size/1)
5. [Intersection point in Y shaped linked lists](https://practice.geeksforgeeks.org/problems/intersection-point-in-y-shapped-linked-lists/1/)
6. [Detect Loop in linked list](https://practice.geeksforgeeks.org/problems/detect-loop-in-linked-list/1)
7. [Remove loop in Linked List](https://practice.geeksforgeeks.org/problems/remove-loop-in-linked-list/1)
8. [n’th node from end of linked list](https://practice.geeksforgeeks.org/problems/nth-node-from-end-of-linked-list/1)
9. [Flattening a Linked List](https://practice.geeksforgeeks.org/problems/flattening-a-linked-list/1)
10. [Merge two sorted linked lists](https://practice.geeksforgeeks.org/problems/merge-two-sorted-linked-lists/1)
11. [Intersection point of two Linked Lists](https://practice.geeksforgeeks.org/problems/intersection-point-in-y-shapped-linked-lists/1)
12. [Pairwise swap of a linked list](https://practice.geeksforgeeks.org/problems/pairwise-swap-elements-of-a-linked-list-by-swapping-data/1)
13. [Add two numbers represented by linked lists](https://practice.geeksforgeeks.org/problems/add-two-numbers-represented-by-linked-lists/1)
14. [Check if Linked List is Palindrome](https://practice.geeksforgeeks.org/problems/check-if-linked-list-is-pallindrome/1)
15. [Implement Queue using Linked List](https://practice.geeksforgeeks.org/problems/implement-queue-using-linked-list/1)
16. [Implement Stack using Linked List](https://practice.geeksforgeeks.org/problems/implement-stack-using-linked-list/1)
17. [Given a linked list of 0s, 1s and 2s, sort it](https://practice.geeksforgeeks.org/problems/given-a-linked-list-of-0s-1s-and-2s-sort-it/1)
18. [Delete without head pointer](https://practice.geeksforgeeks.org/problems/delete-without-head-pointer/1)

**Stack and Queue :**

1. [Parenthesis Checker](https://practice.geeksforgeeks.org/problems/parenthesis-checker/0)
2. [Next larger element](https://practice.geeksforgeeks.org/problems/next-larger-element/0)
3. [Queue using two Stacks](https://practice.geeksforgeeks.org/problems/queue-using-two-stacks/1)
4. [Stack using two queues](https://practice.geeksforgeeks.org/problems/stack-using-two-queues/1)
5. [Get minimum element from stack](https://practice.geeksforgeeks.org/problems/get-minimum-element-from-stack/1)
6. [LRU Cache](https://practice.geeksforgeeks.org/problems/lru-cache/1)
7. [Circular tour](https://practice.geeksforgeeks.org/problems/circular-tour/1)
8. [First non-repeating character in a stream](https://practice.geeksforgeeks.org/problems/first-non-repeating-character-in-a-stream/0)
9. [Rotten Oranges](https://practice.geeksforgeeks.org/problems/rotten-oranges/0)
10. [Maximum of all subarrays of size k](https://practice.geeksforgeeks.org/problems/maximum-of-all-subarrays-of-size-k/0)

**Tree :**

1. [Print Left View of Binary Tree](https://practice.geeksforgeeks.org/problems/left-view-of-binary-tree/1)
2. [Check for BST](https://practice.geeksforgeeks.org/problems/check-for-bst/1)
3. [Print Bottom View of Binary Tree](https://practice.geeksforgeeks.org/problems/bottom-view-of-binary-tree/1)
4. [Print a Binary Tree in Vertical Order](https://practice.geeksforgeeks.org/problems/print-a-binary-tree-in-vertical-order/1)
5. [Level order traversal in spiral form](https://practice.geeksforgeeks.org/problems/level-order-traversal-in-spiral-form/1)
6. [Connect Nodes at Same Level](https://practice.geeksforgeeks.org/problems/connect-nodes-at-same-level/1)
7. [Lowest Common Ancestor in a BST](https://practice.geeksforgeeks.org/problems/lowest-common-ancestor-in-a-bst/1)
8. [Convert a given Binary Tree to Doubly Linked List](https://practice.geeksforgeeks.org/problems/binary-tree-to-dll/1)
9. [Write Code to Determine if Two Trees are Identical or Not](https://practice.geeksforgeeks.org/problems/determine-if-two-trees-are-identical/1)
10. [Given a binary tree, check whether it is a mirror of itself](https://practice.geeksforgeeks.org/problems/symmetric-tree/1)
11. [Height of Binary Tree](https://practice.geeksforgeeks.org/problems/height-of-binary-tree/1)
12. [Maximum Path Sum](https://practice.geeksforgeeks.org/problems/maximum-path-sum/1)
13. [Diameter of a Binary Tree](https://practice.geeksforgeeks.org/problems/diameter-of-binary-tree/1)
14. [Number of leaf nodes](https://practice.geeksforgeeks.org/problems/count-leaves-in-binary-tree/1)
15. [Check if given Binary Tree is Height Balanced or Not](https://practice.geeksforgeeks.org/problems/check-for-balanced-tree/1)
16. [Serialize and Deserialize a Binary Tree](https://practice.geeksforgeeks.org/problems/serialize-and-deserialize-a-binary-tree/1)

**Solved the above?** [Go for some more Questions](https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/#More%20Questions%20on%20Trees)

**Some more Questions on Trees :**

1. [Mirror Tree](https://practice.geeksforgeeks.org/problems/mirror-tree/1)
2. [Longest consecutive sequence in Binary tree](https://practice.geeksforgeeks.org/problems/longest-consecutive-sequence-in-binary-tree/1)
3. [Bottom View of Binary Tree](https://practice.geeksforgeeks.org/problems/bottom-view-of-binary-tree/1)
4. [Lowest Common Ancestor in a Binary Tree](https://practice.geeksforgeeks.org/problems/lowest-common-ancestor-in-a-binary-tree/1)
5. [Binary to DLL](https://practice.geeksforgeeks.org/problems/binary-tree-to-dll/1)

**Heap :**

1. [Find median in a stream](https://practice.geeksforgeeks.org/problems/find-median-in-a-stream/0)
2. [Heap Sort](https://practice.geeksforgeeks.org/problems/heap-sort/1)
3. [Operations on Binary Min Heap](https://practice.geeksforgeeks.org/problems/operations-on-binary-min-heap/1)
4. [Rearrange characters](https://practice.geeksforgeeks.org/problems/rearrange-characters/0)
5. [Kth largest element in a stream](https://practice.geeksforgeeks.org/problems/kth-largest-element-in-a-stream/0)
6. [Merge K sorted linked lists](https://practice.geeksforgeeks.org/problems/merge-k-sorted-linked-lists/1)
7. [Kth largest element in a stream](https://practice.geeksforgeeks.org/problems/kth-largest-element-in-a-stream/0)

**Recursion :**

1. [Flood fill Algorithm](https://practice.geeksforgeeks.org/problems/flood-fill-algorithm/0)
2. [Number of paths](https://practice.geeksforgeeks.org/problems/number-of-paths/0)
3. [Combination Sum – Part 2](https://practice.geeksforgeeks.org/problems/combination-sum-part-2/0)
4. [Special Keyboard](https://practice.geeksforgeeks.org/problems/special-keyboard/0)
5. [Josephus problem](https://practice.geeksforgeeks.org/problems/josephus-problem/1)

**Hashing :**

1. [Relative Sorting](https://practice.geeksforgeeks.org/problems/relative-sorting/0)
2. [Sorting Elements of an Array by Frequency](https://practice.geeksforgeeks.org/problems/sorting-elements-of-an-array-by-frequency/0)
3. [Largest subarray with 0 sum](https://practice.geeksforgeeks.org/problems/largest-subarray-with-0-sum/1)
4. [Common elements](https://practice.geeksforgeeks.org/problems/common-elements/0)
5. [Find all four sum numbers](https://practice.geeksforgeeks.org/problems/find-all-four-sum-numbers/0)
6. [Swapping pairs make sum equal](https://practice.geeksforgeeks.org/problems/swapping-pairs-make-sum-equal/0)
7. [Count distinct elements in every window](https://practice.geeksforgeeks.org/problems/count-distinct-elements-in-every-window/1)
8. [Array Pair Sum Divisibility Problem](https://practice.geeksforgeeks.org/problems/array-pair-sum-divisibility-problem/0)
9. [Longest consecutive subsequence](https://practice.geeksforgeeks.org/problems/longest-consecutive-subsequence/0)
10. [Array Subset of another array](https://practice.geeksforgeeks.org/problems/array-subset-of-another-array/0)
11. [Find all pairs with a given sum](https://practice.geeksforgeeks.org/problems/find-all-pairs-whose-sum-is-x/0)
12. [Find first repeated character](https://practice.geeksforgeeks.org/problems/find-first-repeated-character/0)
13. [Zero Sum Subarrays](https://practice.geeksforgeeks.org/problems/zero-sum-subarrays/0)
14. [Minimum indexed character](https://practice.geeksforgeeks.org/problems/minimum-indexed-character/0)
15. [Check if two arrays are equal or not](https://practice.geeksforgeeks.org/problems/check-if-two-arrays-are-equal-or-not/0)
16. [Uncommon characters](https://practice.geeksforgeeks.org/problems/uncommon-characters/0)
17. [Smallest window in a string containing all the characters of another string](https://practice.geeksforgeeks.org/problems/smallest-window-in-a-string-containing-all-the-characters-of-another-string/0)
18. [First element to occur k times](https://practice.geeksforgeeks.org/problems/first-element-to-occur-k-times/0)
19. [Check if frequencies can be equal](https://practice.geeksforgeeks.org/problems/check-frequencies/0)

**Graph :**

1. [Depth First Traversal](https://practice.geeksforgeeks.org/problems/depth-first-traversal-for-a-graph/1)
2. [Breadth First Traversal](https://practice.geeksforgeeks.org/problems/bfs-traversal-of-graph/1)
3. [Detect cycle in undirected graph](https://practice.geeksforgeeks.org/problems/detect-cycle-in-an-undirected-graph/1/)
4. [Detect cycle in a directed graph](https://practice.geeksforgeeks.org/problems/detect-cycle-in-a-directed-graph/1)
5. [Topological sort](https://practice.geeksforgeeks.org/problems/topological-sort/1)
6. [Find the number of islands](https://practice.geeksforgeeks.org/problems/find-the-number-of-islands/1)
7. [Implementing Dijkstra](https://practice.geeksforgeeks.org/problems/implementing-dijkstra-set-1-adjacency-matrix/1)
8. [Minimum Swaps](https://practice.geeksforgeeks.org/problems/minimum-swaps/1)
9. [Strongly Connected Components](https://practice.geeksforgeeks.org/problems/strongly-connected-components-kosarajus-algo/1)
10. [Shortest Source to Destination Path](https://practice.geeksforgeeks.org/problems/shortest-source-to-destination-path/0)
11. [Find whether path exist](https://practice.geeksforgeeks.org/problems/find-whether-path-exist/0)
12. [Minimum Cost Path](https://practice.geeksforgeeks.org/problems/minimum-cost-path/0)
13. [Circle of Strings](https://practice.geeksforgeeks.org/problems/circle-of-strings/0)
14. [Floyd Warshall](https://practice.geeksforgeeks.org/problems/implementing-floyd-warshall/0)
15. [Alien Dictionary](https://practice.geeksforgeeks.org/problems/alien-dictionary/1)
16. [Snake and Ladder Problem](https://practice.geeksforgeeks.org/problems/snake-and-ladder-problem/0)

**Greedy :**

1. [Activity Selection](https://practice.geeksforgeeks.org/problems/activity-selection/0)
2. [N meetings in one room](https://practice.geeksforgeeks.org/problems/n-meetings-in-one-room/0)
3. [Coin Piles](https://practice.geeksforgeeks.org/problems/coin-piles/0)
4. [Maximize Toys](https://practice.geeksforgeeks.org/problems/maximize-toys/0)
5. [Page Faults in LRU](https://practice.geeksforgeeks.org/problems/page-faults-in-lru/0)
6. [Largest number possible](https://practice.geeksforgeeks.org/problems/largest-number-possible/0)
7. [Minimize the heights](https://practice.geeksforgeeks.org/problems/minimize-the-heights/0)
8. [Minimize the sum of product](https://practice.geeksforgeeks.org/problems/minimize-the-sum-of-product/0)
9. [Huffman Decoding](https://practice.geeksforgeeks.org/problems/huffman-decoding-1/1)
10. [Minimum Spanning Tree](https://practice.geeksforgeeks.org/problems/minimum-spanning-tree/1)
11. [Shop in Candy Store](https://practice.geeksforgeeks.org/problems/shop-in-candy-store/0)
12. [Geek collects the balls](https://practice.geeksforgeeks.org/problems/geek-collects-the-balls/0)

**Dynamic Programming :**

1. [Minimum Operations](https://practice.geeksforgeeks.org/problems/find-optimum-operation/0)
2. [Max length chain](https://practice.geeksforgeeks.org/problems/max-length-chain/1)
3. [Minimum number of Coins](https://practice.geeksforgeeks.org/problems/-minimum-number-of-coins/0)
4. [Longest Common Substring](https://practice.geeksforgeeks.org/problems/longest-common-substring/0)
5. [Longest Increasing Subsequence](https://practice.geeksforgeeks.org/problems/longest-increasing-subsequence/0)
6. [Longest Common Subsequence](https://practice.geeksforgeeks.org/problems/longest-common-subsequence/0)
7. [0 – 1 Knapsack Problem](https://practice.geeksforgeeks.org/problems/0-1-knapsack-problem/0)
8. [Maximum sum increasing subsequence](https://practice.geeksforgeeks.org/problems/maximum-sum-increasing-subsequence/0)
9. [Minimum number of jumps](https://practice.geeksforgeeks.org/problems/minimum-number-of-jumps/0)
10. [Edit Distance](https://practice.geeksforgeeks.org/problems/edit-distance/0)
11. [Coin Change Problem](https://practice.geeksforgeeks.org/problems/coin-change/0)
12. [Subset Sum Problem](https://practice.geeksforgeeks.org/problems/subset-sum-problem/0)
13. [Box Stacking](https://practice.geeksforgeeks.org/problems/box-stacking/1)
14. [Rod Cutting](https://practice.geeksforgeeks.org/problems/cutted-segments/0)
15. [Path in Matrix](https://www.geeksforgeeks.org/find-the-longest-path-in-a-matrix-with-given-constraints/)
16. [Minimum sum partition](https://practice.geeksforgeeks.org/problems/minimum-sum-partition/0)
17. [Count number of ways to cover a distance](https://practice.geeksforgeeks.org/problems/count-number-of-hops/0)
18. [Egg Dropping Puzzle](https://practice.geeksforgeeks.org/problems/egg-dropping-puzzle/0)
19. [Optimal Strategy for a Game](https://practice.geeksforgeeks.org/problems/optimal-strategy-for-a-game/0)
20. [Shortest Common Supersequence](https://practice.geeksforgeeks.org/problems/shortest-common-supersequence/0)

**Divide and Conquer :**

1. [Find the element that appears once in sorted array](https://practice.geeksforgeeks.org/problems/find-the-element-that-appears-once-in-sorted-array/0)
2. [Search in a Rotated Array](https://practice.geeksforgeeks.org/problems/search-in-a-rotated-array/0)
3. [Binary Search](https://practice.geeksforgeeks.org/problems/binary-search/1)
4. [Sum of Middle Elements of two sorted arrays](https://practice.geeksforgeeks.org/problems/sum-of-middle-elements-of-two-sorted-arrays/0)
5. [Quick Sort](https://practice.geeksforgeeks.org/problems/quick-sort/1)
6. [Merge Sort](https://practice.geeksforgeeks.org/problems/merge-sort/1)
7. [K-th element of two sorted Arrays](https://practice.geeksforgeeks.org/problems/k-th-element-of-two-sorted-array/0)

**Backtracking :**

1. [N-Queen Problem](https://practice.geeksforgeeks.org/problems/n-queen-problem/0)
2. [Solve the Sudoku](https://practice.geeksforgeeks.org/problems/solve-the-sudoku/0)
3. [Rat in a Maze Problem](https://practice.geeksforgeeks.org/problems/rat-in-a-maze-problem/1)
4. [Word Boggle](https://practice.geeksforgeeks.org/problems/word-boggle/0)
5. [Generate IP Addresses](https://practice.geeksforgeeks.org/problems/generate-ip-addresses/1)

**Bit Magic :**

1. [Find first set bit](https://practice.geeksforgeeks.org/problems/find-first-set-bit/0)
2. [Rightmost different bit](https://practice.geeksforgeeks.org/problems/rightmost-different-bit/0)
3. [Check whether K-th bit is set or not](https://practice.geeksforgeeks.org/problems/check-whether-k-th-bit-is-set-or-not/0)
4. [Toggle bits given range](https://practice.geeksforgeeks.org/problems/toggle-bits-given-range/0)
5. [Set kth bit](https://practice.geeksforgeeks.org/problems/set-kth-bit/0)
6. [Power of 2](https://practice.geeksforgeeks.org/problems/power-of-2/0)
7. [Bit Difference](https://practice.geeksforgeeks.org/problems/bit-difference/0)
8. [Rotate Bits](https://practice.geeksforgeeks.org/problems/rotate-bits/0)
9. [Swap all odd and even bits](https://practice.geeksforgeeks.org/problems/swap-all-odd-and-even-bits/0)
10. [Count total set bits](https://practice.geeksforgeeks.org/problems/count-total-set-bits/0)
11. [Longest Consecutive 1’s](https://practice.geeksforgeeks.org/problems/longest-consecutive-1s/0)
12. [Sparse Number](https://practice.geeksforgeeks.org/problems/number-is-sparse-or-not/0)
13. [Alone in a couple](https://practice.geeksforgeeks.org/problems/alone-in-couple/0)
14. [Maximum subset XOR](https://practice.geeksforgeeks.org/problems/maximum-subset-xor/1)

**Mathematical :**

1. [Print the pattern](https://practice.geeksforgeeks.org/problems/print-the-pattern-set-1/1) (You only need to write function here)
2. [Print table](https://practice.geeksforgeeks.org/problems/print-table/0)(This is a full code problem. Please see sample codes [here](https://www.geeksforgeeks.org/how-to-begin-with-competitive-programming) before attempting the problem)
3. [Series AP](https://practice.geeksforgeeks.org/problems/series-ap/0)
4. [Series GP](https://practice.geeksforgeeks.org/problems/series-gp/0)
5. [Closest Number](https://practice.geeksforgeeks.org/problems/closest-number/0)
6. [Armstrong Numbers](https://practice.geeksforgeeks.org/problems/armstrong-numbers/0)
7. [Sum of digits of a number](https://practice.geeksforgeeks.org/problems/sum-of-digit-is-pallindrome-or-not/0)
8. [Reverse digits](https://practice.geeksforgeeks.org/problems/reverse-digit/0)
9. [Print the Kth Digit](https://practice.geeksforgeeks.org/problems/print-the-kth-digit/0)
10. [Binary number to decimal number](https://practice.geeksforgeeks.org/problems/binary-number-to-decimal-number/0)
11. [Jumping Numbers](https://practice.geeksforgeeks.org/problems/jumping-numbers/0)
12. [GCD of two numbers](https://practice.geeksforgeeks.org/problems/gcd-of-two-numbers/0)
13. [LCM of two numbers](https://practice.geeksforgeeks.org/problems/lcm-and-gcd/0)
14. [Add two fractions](https://practice.geeksforgeeks.org/problems/add-two-fractions/1)
15. [GCD of array](https://practice.geeksforgeeks.org/problems/gcd-of-array/0)
16. [Factorial of a number](https://practice.geeksforgeeks.org/problems/factorial/0)
17. [Compute nPr](https://practice.geeksforgeeks.org/problems/npr/0)
18. [Compute nCr](https://practice.geeksforgeeks.org/problems/ncr/0)
19. [Largest prime factor](https://practice.geeksforgeeks.org/problems/largest-prime-factor/0)
20. [Perfect Numbers](https://practice.geeksforgeeks.org/problems/perfect-numbers/0)
21. [Pair cube count](https://practice.geeksforgeeks.org/problems/pair-cube-count/0)
22. [Find Nth root of M](https://practice.geeksforgeeks.org/problems/find-nth-root-of-m/0)
23. [Prime Number](https://practice.geeksforgeeks.org/problems/prime-number/0)
24. [Sieve of Eratosthenes](https://practice.geeksforgeeks.org/problems/sieve-of-eratosthenes/0)
25. [Sum of all prime numbers between 1 and N](https://practice.geeksforgeeks.org/problems/sum-of-all-prime-numbers-between-1-and-n/0).
26. [Pairs of prime numbers](https://practice.geeksforgeeks.org/problems/pairs-of-prime-number/0)

Do this question in C-

Coding Questions:

1. You were given a list of strings which indicated at what time of the day an employee of a

company checked in and checked out of the company on a particular day. Input was in the

form of <Employee\_Name>, <In\_Time>, <Out\_Time> (comma separated).

Eg:

Alice,10:23:02,13:04:45

Bob,09:00:43,12:03:21

The time was always in HH:MM:SS 24-hours format. You were given a time of the day as

query <Query\_Time> Eg: 11:00:56. We have to determine how many numbers of employees

were present inside the company at the point in time (Query\_Time). In these examples, o/p

should be 2 (Reason: Alice and Bob both are inside the office at time 11:00:56).