Amazon Final Interview Questions | All Combined 2021 | SDE & New Grad

Amazon Final Interview Questions

A copy of my personal incomplete list of questions I found on Amazon's Interview process from end of 2020 - 2021.
Purposely very verbose so as to allow to find the original post (via google search).
Feel free to comment missing questions so that I add them to the list.

Onsite

All Combined 2021 | SDE & New Grad

Tips

- I feel like to pass the onsite interview, you must demonstrate stong leadership principles and communication skills. The problems weren't too hard so I'm guessing they're looking to see if you can clearly explain your thought process.
- I personally think I will not get the offer because from my experience you either have to get everything perfect or you will fail. For me I can only say 2/4 interviews went perfectly thus I probably will get rejected. Alas, I do have another final round interview this August so I have another chance.
- One Leetcode Medium Question. Rather than the actual code, they were mostly interested in how I approach the problem and I had to speak aloud about my thought process in solving the problem. Heard back within 3 business days that they would like to extend the offer. If you practice Leetcode with full concentration then there is no reason not to clear the interviews at big companies. With regards to LP, if you have 2 scenarios which you can explain in detail for each of the LP principles, you are golden. All the best for everyone who is preparing.
- Finally, one last piece of advice. Please, PLEASE, avoid remembering the exact questions, but rather focus on learning the actual patterns, data structures, and logic behind each exercise.
- Experience -
    1. Initial 20 minutes they will ask you LP questions only. Its not like they will ask just one or two questions on LP.

2. Next 20 minutes will be dedicated to coding. Links to coding platform will be shared. They wont be able to compile the code, but will see the way you are coding. One thing common in all rounds were they wanted me to write the code in OOD format. It was not like i just wrote the method and its done. They wanted me to write the object creation, function calls etc. I was not done with full implementation in round 1 and round 3, they stopped me like 2 minutes before 20 minutes are completed and asked me to write the OOD format for whole code.

NOTE - I was not able to complete full implementation but i explained the approach very well, step by step. Its okay to go back and forth sometimes, just tell them i am thinking out loud, i am making connections to find optimal way :)

3 Last five minutes were just if i have any questions for them.

- Finally I would suggest anyone who is appearing for the Amazon technical interview to focus on proper naming of variables and code readability and reusabality aspects as well.

Questions

Old https://leetcode.com/discuss/interview-question/488887/amazon-final-interview-questions-sde1 Amazon Final Interview Questions | SDE1 - Google Docs

- Number of Islands,
- Level Order (and Zigzag) Tree Traversal,
- Word Series (Word Ladder 1 & 2, Word Break 1 & 2, Word Search 1 & 2).
- "parking lot" OOD question
- Bloomberg | Software Engineering Intern | London | January 2020 [Offer] - LeetCode Discuss
- Amazon Dublin Ireland | Onsite - SDE New Grad 2021 | Income Calculator - LeetCode Discuss
- What are Design patterns? example and explanation? https://leetcode.com/discuss/interview-experience/1060893/amazon-sde-new-grad-2021-dublin-experience-offer
- Round2 Onsite: This was a class design. In the beginning, there were 2 LP questions then he asked how a **hash table** is implemented as a starter and then asked

to **design a file system** and implement search on the filesystem based on certain parameters (like file size creation date etc)

https://leetcode.com/discuss/interview-question/609070/Amazon-OOD-Design-Unix-File-Search-API

Round 3: Bar raiser Basically asked LP questions and asked to **Design a task scheduler** (in memory) and then implement a function that **finds the time a certain task takes to complete** (given that there exist tasks that have to be executed before it)

Round 4: Simple Number of islands question along with LP questions

- 1 OOP questions variant of **UNIX command**, **Game of Life Question** and **Iterator Question**.
- Remove all duplicate numbers from a list.
- https://leetcode.com/problems/min-stack/ However, it came with the constraint that we would be needing extra O(n) space. I was also asked to implement exception handling to deal with situations when you call pop() or getMin() on an empty stack.
- First round: 2-3 behavioral questions followed by a coding question. The question was **given a binary tree find a subtree within the binary tree that adds up to a certain target sum**.
- Second round: 2-3 behavoral questions followed by an OOD question. The question was a bit vague and I had a hard time understanding their english but essentially it was to design a wharehouse class that had certain constraints such that the particular wharehouse could only store certain products.
- Fourth round: 2-3 behavioral questions. This was another coding round, which involved taking a string that say said "This sweater cost $40 dollars." The objective was to take that number and apply a 20% discount and return the string back with the updated price, i.e., "This sweater cost $32 dollars.".
- This round was interesting since it was not any of the Leetcode questions. I was given certain conditions for a **valid transaction** and I had to write a function that will determine if the transaction is valid or not. The goal of this round was to see how I can write a maintainable and scalable code. For example, how can we add more conditions for a valid transaction and still not alter the main function very much. I had LP questions as part of this round also.

- Find the count of a given sequence that appears in an array. There can be any amount of characters between the numbers of the sequence. ex 4,5,6,2 would have a count of 1.

sequence 4,6,2

array [3, 4, 4, 6, 7, 8 , 2, 6, 9, 2]

count of sequences: 6
https://leetcode.com/discuss/interview-question/437362/amazon-sde-count-of-sequences-within-an-array

- 2 LP´s and Medium exercise on Trie Structure (make sure you know how to use a Trie, the Explore section here has a really good card on it). Really similar to an Amazon tagged exercise.
- 2 LP´s, then started with an extremely simple exercise, and afterwards, the interviewer started adding complexity, to see how I adapted my solution so that it scaled. The interviewer was testing my knowledge on Object Oriented Design, which I use in my everyday work, but had not prepared for the interview. This was almost a disaster, my solution was far from good.
- Course Schedule II -
- Given huge database of sentences, write a class to find most frequently used words
- Questions on data structures like array list, linked list, hash table, binary search tree. Differences between these data structures and when would you use which one. Inheritance and Composition.
- Coding: Design a Tic Tac Toe Game
- Given a graph and destination D, find shortest path between all nodes. Not given any graph implementation, and input is a list of edges ([1,2,3] = Node 1 to Node 2, distance 3). Had to implement my own adjacency list and "nodes" (Dijkstra's Algorithm)
- Similar to this question, but exclude the combined word (ex. instead of [['car','super', 'supercar'], the answer is [['car','super'],)https://leetcode.com/discuss/interview-question/314550/amazon-onsite-interview-concatenated-words
- https://leetcode.com/problems/copy-list-with-random-pointer/
- https://leetcode.com/problems/longest-happy-prefix/
- https://leetcode.com/problems/insert-interval/

- Design a system to generate and apply coupons for e-commerce site based on product and its category.
- https://leetcode.com/problems/lfu-cache/
- Find the next value from the target node using BST
- https://leetcode.com/problems/string-to-integer-atoi/
- Longest common prefix among an array of strings - but the longest between any two (instead of the longest in all)
- https://leetcode.com/problems/boundary-of-binary-tree/
- https://leetcode.com/problems/integer-to-roman/
-
    -

K most frequent items(item_id) in a stream. Input is a stream of (item_id,timestamp).

This I was able to easily solve using hashmap and min-heap

Then, Interviewer asked me to find frequent items in last hour given a timestamp.

For example, Given time 6:07 PM, Return k most frequent items from 5:07 PM to 6:07 PM.

Gave an answer using queue, hashmap and min-heap, Not an optimal answer, But it worked.

- Three sum
- Round 1 - 2 interviewers https://leetcode.com/discuss/interview-experience/1060893/Amazon-SDE-new-grad-2021-Dublin-Experience-or-Offer/854173
    - Implement queue using stack
    - Find loop in single linkedlist
    - find missing element in an array
    - Explain any sorting algorithm
    - Time and space complexity for the above problems
    - Difference similarity between list, linkedlist and array
    - Implementation of Hashmap

    - Difference between set, hashmap, list
- Round 2 - 1 inerviewer
    - 2 LP questions
    - What are Design patterns? example and explanation?
    - Best coding practices
    - https://leetcode.com/problems/integer-to-english-words/
    - Time complexity for the above problem
- Round 3 - 1 interviewer
    - Print matrix spirally. Time and space complexity for the above problem
- Given an array of large numbers and a number K, give an algorithm to search for a triplet whose sum is K. You can modify the array. You can print any triplet whose sum is K.
- After basic introduction, the interviewer asked me questions about Hashmaps. such as time complexity and how to handle collisions. And he asked me how would I implement a structure to store contact informations(phones numbers, names and area codes). I think I did ok in this part, and the interviewer agreed with me on how I would use a hashmap to do so.
- https://leetcode.com/problems/valid-parentheses/
- Merge List + find Top K Elements , I solved this using a Heap to merge the Lists, and a later variation involved using BFS, basically the question was , Recommend a user features that are used by friends and friends of friends N - level deep. Given an api which can give you user's friends and friends most used features.
- https://leetcode.com/problems/number-of-islands/
- https://leetcode.com/problems/knight-probability-in-chessboard/
- Given a positive integer N, and a starting point 1 one can perform any of the following 2 operations in one step: https://leetcode.com/discuss/interview-experience/906535/amazon-sde-i-bangalore-october-2020-offer
    - Add 1 to the number
    - Double the number

The task is to find the minimum number of steps to reach N (desired complexity O(log n)).

- Given a set of N numbers (both positive and negative) sorted in increasing order with A[i] < A[j] for all i<j, find whether there exists an index i (i = 1 .. N) such that A[i] = i. If multiple answers are present return any one of them. (desired complexity O(log n)).
- Give the design of an automated valet parking system with the following specifications:
    - There are 3 parking areas (each of different sizes) for 3 different vehicle sizes - small, medium and large.
    - Small one can accommodate only small vehicles, medium can accommodate small and medium vehicles and similarly for the large one.
    - Design a system which issues a parking ticket to a vehicle entering the lot with the optimal parking space allotted to it. For eg., if a medium vehicle arrives and both medium and large parking areas have vacant spaces, the vehicle should be allotted the medium slot.
    - Also design a syntax for the token ID which is generated when each vehicle enters the lot. The ID should be uniquely able to determine the details of the slot where the vehicle is parked for smooth parking and un-parking.
    - Provide the class design of the same.
- **Virtual Onsite Round 3 (Bar Raiser, Coding + CS Fundamentals)**

1. Given a set of N integers find the kth largest contiguous subarray sum.
2. Questions on OS like describe the boot up process of OS.
3. Difference between caching, paging and buffering.
4. Difference between stack and heap memory, calloc and malloc, etc.
5. Things to keep in mind while designing a software product (scalability, memory leaks, deadlocks).

- Design a restaurant table booking system with the following specifications:
    - The restaurant has X tables of size 2, Y tables of size 3 and Z tables of size 4.
    - There are 3 slots from 6:30 pm to 11 pm, each of 1.5 hrs.
    - Restaurant can take bookings upto 7 days in advance.
    - No two parties will share a table.
    - A pack of size N should always be assigned the largest table available.

  - Wastage of seats should be minimised.
  - Provide the database structure to support the above constraints (space should be optimised).
  - Provide an algorithm to allot table to an incoming reservation with the following specs:
    - Allot table(s) to a group of N people for the ith day and the jth slot.
  - Provide pseudo code for the algorithm stated above.

- Search for smallest element in sorted rotated list
- Tell me about a situation when you had a tight deadline and what are the sacrifices you made to achieve the deadline.
- TTL LRU Cache. https://leetcode.com/discuss/interview-question/284925/ttl-lru-cache
- Tell me about a time you couldn't finish your task within the given deadline.
- flatten a hierarchical linked list.
- A variant of topological sort in a graph.
- Find smallest +ve missing integer from given array without extra space.

(follow up : Array can contain both positive and negative numbers).

Ex : ar [1,9,8] output :2

- You have locking suitcase system (the one in which there will be number codes). Find the minimum number of steps to reach from given number to target number. You can move any place digit at a time. (for n-digit codes, discuss complexity). Ex : start : 0-0-1Target : 1-0-0 min steps = 2 .....changing an xth place digit by b will be counted as b steps. follow -up...what if you are not allowed to use certain numbers(not digits but whole number)(given as array)Ex: [100,234,115]
- Implement a custom collection class with add, remove and getRandomOperations in O(1) time
- https://leetcode.com/discuss/interview-question/1586723/amazon-sde-3-interview-question
- https://leetcode.com/discuss/interview-question/system-design/685338/microsoft-onsite-design-the-t9-predictive-text-algorithm-and-system (not sure)

- https://kobejean.github.io/algorithms/2020/03/08/the-award-budget-cuts-problem/
- Poker Cards Game

As we all know that poker cards have four suites: Spades, Hearts, Clubs and Diamonds with figures from 1 to 13.

Now you are given a set of poker cards, you can pick any one card as the first card. And except for the first card, you can only pick the card that has the same suit or figure with the previous one.

Return the max number of cards you can.

For example: [(H, 3), (H, 4), (S, 4), (D, 5), (D, 1)], it returns 3 as follows: (H,3)-->(H,4)-->(S,4)

https://leetcode.com/discuss/interview-question/1030942/Amazon-Onsite-Inteview-Poker-Cards-Game

- Given a game board| B ||A | C | D||E | F | G ||H | I | J|Rules of movinga. you can not move to same row.b. you can not move in same column.Given a starting point and number of moves tell the number of possible options https://leetcode.com/discuss/interview-question/1170079/amazon-onsite-imdb-team
- Given an array of integers(pos/neg) in sorted order, return an array of elements square in sorted order.
- Given an array of wine prices, any given year you can sell a bottle of wine only from either of the ends. Bottle of wine increases every year. Find max profit after selling all.
- Given a uni-directed graph with numbers find maximum root to leaf sum with using only internal data structure.
- https://leetcode.com/problems/binary-tree-maximum-path-sum/
- https://leetcode.com/problems/word-break/

Given a dictionary of words and a string, state if the string if broken into multiple words consists of dictionary words.

I explained him the standard solution using BFS which is O(n) time ans O(n) space.However, the interviewer was deeply interested with me solving the question via making a Graph and then solving it.

https://leetcode.com/discuss/interview-question/600412/Amazon-onsite

- A very similar question to this , same concept of BFS will applyGiven a 2D grid, each cell is either a zombie 1 or a human 0. Zombies can turn adjacent (up/down/left/right) human beings into zombies every hour. Find out how many hours does it take to infect all humans?https://leetcode.com/discuss/interview-question/411357/ Amazon | OA 2019 | 🧟 Zombie in Matrix - LeetCode Discuss
- given a list of unique strings, if the last char at string A match first char at string B then you can append them together: good+dog -> goo**d**og . Now return the longest possible string (length of concatenated String, not the string number). https://leetcode.com/discuss/interview-question/250623/Onsite-question-for-Amazon-String-concatenation(Updated)/246655
- Convert BST to Doubly Linked List without deforming tree and without using extra space except used for creating List. So this shouldn't be done inplace.Time and Space Complexity of my solutions -: O(n) & O(1) respectively
- You are given a subarray which has only 0's and 1's , Maximise the subarray containing 1's and in this you can only flip one 0 , tell the index of that 0Similar to thisfind-zeroes-to-be-flipped-so-that-number-of-consecutive-1s-is-maximizedon GFGTime and Space Complexity of my solutions -: O(n) & O(1) respectively
- Given 2 large files say file1, file2 consisting of strings of customerid - find the unique customers present in those files. Basically unique words (ex. customer1, customer 3, customer5 and so on)!

My approach - Solved using HashSet but interviewer told it won't work as input is very large and Set would exceed Memory limit. I told maybe something like divide and conquer but wasn't able to come up with the actual solution.

- For SDE I found Amazon mostly ask Graph Medium questions for example:

323. Number of Connected Components in an Undirected Graph

547. Number of Provinces

- http://leetcode.com/problems/generate-parentheses
- This Question I have not seen in any platform. So I will be giving the question description. We have been given a string with parenthesis and characters in between. The parenthesis can be invalid also if it's invalid we have to return -1 and if it's valid then we need to find the maximum depth of the brackets.For eg. let s = "((((X))(((Y)))))" Output will be 5 in this caseThe string can have characters in between the brackets and it can be without characters in between also. I gave him the approach and after the initial discussion, he told me to optimize the space for

validity checking of the brackets. After more discussion, I came with an optimized approach and he told it to code it. I wrote the code along with the time and space complexity.

- https://leetcode.com/discuss/interview-question/995825/amazon-onsite-sql-library
- implement business logic for amazon products with friends. Writing a function like you would in a work style
- for the given binary tree find the distance or number of hop required to move from one node to another.
- https://leetcode.com/problems/frog-jump SDE 2
- https://leetcode.com/problems/sliding-puzzle/ SDE 2
- Write a program to print the sum of two roman numbers
- https://leetcode.com/problems/best-time-to-buy-and-sell-stock If there is a tie, then return the minimum days to finish the transaction. Example: [1 2 3 1 3], we can buy at index 0 and sell at 2, the profit is 2. but we can also buy at index 3 and sell at 4 which only takes 2 days. So return [3, 4]
- describe the differences between Set, Map, Array, List and which data structure should be used with different scenarios
- https://leetcode.com/problems/k-similar-strings/
- We are given a list of cars, and the direction of the road they are in. We need to find the order in which cars leave the intersection. The list of cars have the cars at the intersection in order. However, if two cars are in opposite direction, for example East/West or North/South, they can leave the intersection even though other cars arrived before it.
- Given a list of inputs in the form of a string array, for example [["Item: Bread", "Id:1"], ["Item: Meat", "Id:3"], ["Item: Sauce", Id:2], ["Item: Bread", "Id:4"]], we need to arrange the input string in order and output the result. The interviewer said do not worry about parsing the string. I gave the solution of sorting based on Id and then getting the result. It was very unclear what the interviewer was expecting. Also, I made the mistake of having only screen, and the interviewer said he was nodding to my questions, and I was writing code in another window. This was a very uncomfortable interview.

- Write an API for Linux find command. It was this exact question: https://leetcode.com/discuss/interview-question/369272/Amazon-or-Onsite-or-Linux-Find-Command
    - Medium OOD question. Very similar to linux find command question (https://leetcode.com/discuss/interview-question/369272/Amazon-or-Onsite-or-Linux-Find-Command) but for validation of purchases. If you are good at applying SOLID principles, this should be solvable. The question was in Java, I spent a couple of minutes to convert it to Python.
- https://leetcode.com/problems/russian-doll-envelopes/
- https://leetcode.com/problems/best-time-to-buy-and-sell-stock-iii/
- A variation of meeting rooms: https://leetcode.com/problems/meeting-rooms/Given a list of meeting rooms and a meeting reservation request. Find if the reservation request can be fulfilled or not.Follow up: What if the list of meeting rooms is sorted?**Edit**: I found a similar one here: https://leetcode.com/problems/my-calendar-i/
- Coding: create stack with O(1) push, pop and min.
- K most frequent items(item_id) in a stream. Input is a stream of (item_id,timestamp).

Then, Interviewer asked me to find frequent items in last hour given a timestamp.

For example, Given time 6:07 PM, Return k most frequent items from 5:07 PM to 6:07 PM.

Gave an answer using queue, hashmap and min-heap, Not an optimal answer, But it worked.

- https://leetcode.com/problems/integer-to-roman/
- https://leetcode.com/problems/reorder-data-in-log-files/
- Design and implement a playlist of a user's most recently played songs on Amazon music. It should support the following operations: getSong and addSong.

getSong – user should be able to search and get the song from the playlist if the song exists.

addSong – add the song into the playlist if the song is not already present.

- Design a Parking Lot , Which takes assigns a ticket number a when car arrives and a vacant parking slot is available and rejects if not, Empty the parking slot when car departs. There were different types of cars and parking spots(small,medium and large).

- https://leetcode.com/problems/longest-common-subsequence/
- https://leetcode.com/problems/longest-consecutive-sequence/
- https://leetcode.com/problems/trapping-rain-water/
- Merge k sorted linked list & array. (Highly scalable solution was expected)
- Implementaion of coin change variation of dynamic programming. -
- https://leetcode.com/problems/design-in-memory-file-system/
- https://leetcode.com/problems/copy-list-with-random-pointer/
- https://leetcode.com/problems/product-of-array-except-self/
- https://leetcode.com/problems/median-of-two-sorted-arrays/
- https://leetcode.com/problems/rectangle-overlap/
- https://leetcode.com/problems/design-tic-tac-toe/
- https://leetcode.com/problems/longest-consecutive-sequence/solution/
- https://leetcode.com/problems/concatenated-words/
- https://leetcode.com/problems/word-search-ii/
- https://leetcode.com/problems/making-file-names-unique/
    - Follow up question - Assume the tree as a Binary Search Tree & was asked to solve it.
- https://leetcode.com/problems/find-all-anagrams-in-a-string/
- Largest pair sum that is less than or equal to k in an array
- https://leetcode.com/discuss/interview-question/369272/Amazon-or-Onsite-or-Linux-FindCommand
- https://leetcode.com/problems/rotting-oranges/
- https://leetcode.com/problems/all-paths-from-source-to-target/
- https://leetcode.com/discuss/interview-question/algorithms/124715/amazon-is-cheese-reachable-in-the-maze
- Implement operations for an AutoComplete feature. **New Grad 2021**

1. InsertWords(words) - Given a stream of words, store the words

2. CheckPrefix(prefix) - Returns if the prefix exists

3. SearchPrefix(prefix) Given a prefix string, return words starting with the prefix string.Eg: Insert Words {car, cart, carpool, bus, apple, cargo}SearchPrefix (car) -> car, cart, carpool, cargoFollow up questions – SearchPrefix() return in sorted order/ top k results

- Isomorphic Strings **New Grad 2021**
- Design Unix Find command - https://leetcode.com/discuss/interview-question/609070/Amazon-OOD-Design-Unix-File-Search-API **New Grad 2021** Loading... (leetcode.com)
- Design and implement a playlist of a user's most recently played songs on Amazon music. It should support the following operations: getSong and addSong.

getSong – user should be able to search and get the song from the playlist if the song exists.

addSong – add the song into the playlist if the song is not already present.

- In the logistic floor robots entered and exited, how many robot exist at any specific time. interviewer was not very specific about the question and I had to question a lot to understand and construct the problem statement.
- https://leetcode.com/problems/jump-game-ii/
- Sum of all binary search trees in a tree. So, basically, the interviewer wanted to create a tree and then if there was a BST, add the value of all the nodes in the tree. **SDE2**
- OOP/Coding: Design a game where there are multiple players, you can either add a new player, update their score or display Leadership Board **SDE2**
- Design phone bill calculator
- [K maximum sum combinations from two arrays] - GeeksfGeeks question
- There are N trees in Jon's backyard and the height of tree i is h[i]. Jon doesn't like the appearance of it and is planning to increase and decrease the height of trees such that the new heights are in strictly increasing order. Every day he can pick one tree and increase or decrease the height by 1 unit. Heights can be 0 or even negative (it's a magical world). https://leetcode.com/discuss/interview-question/1171979/a-good-question-of-amazon-sde

- This problem is a variant of closest pair sum. You'll be given two arraysarr1 = { {1, 2000}, {2, 3000}, {3, 4000} }arr2 = { { 1, 5000 }, {2, 3000} }the first element of every pair represents *id* and the second value represents the *value*.and a target x = 5000Find the pairs from both the arrays whose vaue add upto a sum which is less than given target and should be close to the target.

Output for the above example:{ {1, 2} } // Note that the output should be in id's

I cleared all the test cases for the first problem (will share), but for this problem, I couldn't clear 3 test cases which were of handling duplicates!In the online assesment this problem was sort of tagged as approximate solution, does anyone know what that means?Awaiting for the result!

- https://leetcode.com/problems/find-median-from-data-stream/
- Variation of https://leetcode.com/problems/concatenated-words/

Given a large list of words. Some of these are compounds, where all parts are also in the List. Write a method that will find all combinations where one word is a composite of two or more words from the same list and return them.

https://leetcode.com/discuss/interview-question/545748/amazon-subsidiary-phone-concatenated-words

- Given two number in the form of sting, we have to perform two operations: Remove all the zeroes from the number and then add the two numbers.
- Given two numbers in the form of LinkedList we have to add the numbers. But there was a condition that we cannot reverse the linkedlist.
- https://leetcode.com/problems/asteroid-collision/
- https://leetcode.com/problems/best-time-to-buy-and-sell-stock-with-cooldown/
- https://leetcode.com/problems/minimum-knight-moves/
- Group Product Id pairs into Categories ****https://leetcode.com/discuss/interview-question/690707/Amazon-or-Phone-Interview-or-Group-Product-Id-pairs-into-Categories
- https://leetcode.com/problems/word-break-ii/
- Search for smallest element in sorted rotated list
- https://leetcode.com/problems/binary-tree-right-side-view/

Facebook interview experiences - All Combined from LC - Till Date 07-Jun-2020

1. Phone Screen:

1. https://leetcode.com/problems/find-all-anagrams-in-a-string/
2. A string / array problem involving distinct characters and window
3. https://leetcode.com/problems/shortest-bridge/
4. https://leetcode.com/problems/partition-equal-subset-sum/
5. https://leetcode.com/problems/valid-palindrome-ii/
6. https://leetcode.com/problems/kth-smallest-element-in-a-bst/
7. You are given a m*n grid. You are asked to generate k mines on this grid randomly. Each cell should have equal probability of k / m*n of being chosen. Your algorithm should run in O(m) time.
8. https://leetcode.com/problems/continuous-subarray-sum/
   (Given a list of positive numbers and a target integer k, write a function to check if the array has a continuous subarray which sums to k.)
9. https://leetcode.com/problems/verifying-an-alien-dictionary/
10. https://leetcode.com/problems/alien-dictionary/
11. https://leetcode.com/problems/course-schedule/
12. https://leetcode.com/problems/interval-list-intersections/
13. https://leetcode.com/problems/minimum-remove-to-make-valid-parentheses/
14. https://leetcode.com/problems/plus-one/
15. https://www.***.org/find-index-maximum-occurring-element-equal-probability/***
16. https://leetcode.com/problems/range-sum-of-bst/
17. Similar strings ("face", "eacf") returns true if only 2 positions in the strings are swapped. Here 'f' and 'e' are swapped in the example.
18. https://leetcode.com/problems/number-of-connected-components-in-an-undirected-graph

19. https://leetcode.com/problems/add-binary/
20. Given two binary search trees how do we merge everything so it prints inorder. The answer I gave was to run inorder on both trees and use "merge" from merge-sort.
21. https://leetcode.com/problems/valid-palindrome
22. https://leetcode.com/problems/add-strings
23. https://leetcode.com/problems/serialize-and-deserialize-binary-tree/
24. https://leetcode.com/problems/lowest-common-ancestor-of-a-binary-tree
25. https://leetcode.com/problems/smallest-subtree-with-all-the-deepest-nodes/
26. https://leetcode.com/problems/binary-tree-paths
27. https://leetcode.com/problems/minimum-window-substring
28. How to remove duplicates from a list
29. https://leetcode.com/problems/maximum-subarray
30. https://leetcode.com/problems/valid-parentheses
31. https://leetcode.com/problems/merge-intervals
32. https://leetcode.com/problems/task-scheduler/
33. https://leetcode.com/problems/clone-graph/

2. Coding Round 1:

1. https://leetcode.com/problems/insert-interval/
2. https://leetcode.com/problems/convert-a-number-to-hexadecimal/
3. https://leetcode.com/problems/rotate-array/
4. https://leetcode.com/problems/k-closest-points-to-origin/
5. https://leetcode.com/discuss/interview-question/124759/
6. https://leetcode.com/problems/product-of-array-except-self
7. https://leetcode.com/problems/find-all-anagrams-in-a-string/
8. https://leetcode.com/problems/minimum-window-substring/
9. https://leetcode.com/problems/closest-binary-search-tree-value/
10. https://leetcode.com/problems/insert-delete-getrandom-o1/

11. https://leetcode.com/problems/fraction-to-recurring-decimal/
12. https://leetcode.com/problems/powx-n
13. https://leetcode.com/problems/subarray-sum-equals-k
14. https://leetcode.com/problems/best-time-to-buy-and-sell-stock
15. https://leetcode.com/problems/best-time-to-buy-and-sell-stock-iii
16. https://leetcode.com/problems/best-time-to-buy-and-sell-stock-iv
17. https://leetcode.com/problems/add-and-search-word-data-structure-design
18. https://leetcode.com/problems/sudoku-solver/
19. https://leetcode.com/discuss/interview-question/338948/Facebook-or-Onsite-or-Schedule-of-Tasks
20. https://leetcode.com/problems/binary-tree-maximum-path-sum
21. https://leetcode.com/problems/maximum-subarray
22. https://leetcode.com/problems/move-zeroes
23. https://leetcode.com/problems/valid-number
24. https://leetcode.com/problems/first-bad-version/

3. Coding Round 2:

1. https://leetcode.com/problems/valid-number/
2. You have an API to check if is it possible to move left, right, up, down and one more method to check if current position is the last one. Find the shortest way to the last position. You don't have any data structure - only API.
3. https://leetcode.com/problems/serialize-and-deserialize-binary-tree/
4. https://leetcode.com/problems/group-shifted-strings/
5. https://leetcode.com/problems/task-scheduler/
6. Calculate tax if Salary and Tax Brackets are given as list in the form
   [[10000, 0.3],[20000, 0.2], [30000, 0.1], [null, .1]]
   null being rest of the salary
7. Is there a way to reach (0,0) from a mXn matrix to (m-1,n-1) position and give the path.

8. https://leetcode.com/problems/simplify-path/
9. n-ary Tree with each node having a boolean flag. Traverse all the nodes with only boolean flag = True. Return the total distance traveled from root to all those nodes.
10. https://leetcode.com/problems/product-of-array-except-self/
11. https://leetcode.com/discuss/interview-question/432086/Facebook-or-Phone-Screen-or-Task-Scheduler/394783
12. https://leetcode.com/problems/find-all-anagrams-in-a-string
13. https://leetcode.com/problems/is-graph-bipartite
14. https://leetcode.com/problems/merge-sorted-array
15. https://leetcode.com/problems/maximum-subarray
16. https://leetcode.com/problems/serialize-and-deserialize-binary-tree
17. https://leetcode.com/problems/remove-invalid-parentheses/
18. https://leetcode.com/problems/subarray-sum-equals-k/
19. https://leetcode.com/problems/binary-tree-level-order-traversal/
20. https://leetcode.com/problems/longest-increasing-path-in-a-matrix/
21. https://leetcode.com/problems/custom-sort-string
22. https://leetcode.com/problems/read-n-characters-given-read4
23. https://leetcode.com/problems/remove-invalid-parentheses
24. https://leetcode.com/problems/palindrome-permutation
25. https://leetcode.com/problems/max-consecutive-ones-iii
26. https://leetcode.com/problems/range-sum-of-bst
27. https://leetcode.com/problems/exclusive-time-of-functions
28. https://leetcode.com/problems/search-in-rotated-sorted-array/
29. https://leetcode.com/problems/search-in-rotated-sorted-array-ii/

4. Design:

1. Design Google search

2. Some question related to caching and balancing. Not exactly the "design twitter" type of question, but expect to talk about different components, latency, throughput, consistency and availability.
3. A remote server is not responding. Debug the issue. Needed to cover entire TCP/IP stack(fragmentation/icmp/etc) + machine metrics (vmstat,iostat,strace etc). Describe virtual memory in terms of demand paging.
4. 2 machines are connected, suddenly 1 machine is responding slowly. Why ?
   We had a good discussion in which we discussed everything under the sun, from NFS being bad to Networking being wrong to Kernel running out of resources(buffer-cache/inodes/virtual memory). Interviewer was interested to know the commands that i would use (strace, lsof, readlink, cat /proc/pid etc).
5. Copy some resource from N sources to M sinks. where N could be < 10 and M could be 10k/Millions etc.
6. Design File Storage System. Like Dropbox, Google Drive
7. Not any fancy one like design Twitter or Uber. More on scheduling service side and i designed using SQL appraoch. Discussed concuurency issues, Table schemas, composite keys etc.
8. Design recommendation of celebrities to user on Instagram
9. Design search for Twitter
10. Design a Content publishing site with privacy restrictions.
11. System Design of Uber. He liked my design. He was really nice guy, i felt he was interested in my success.
12. Design a type ahead features for a website. We discussed various data structures, advantage /disadvantages. Lot of different cases, scenario to handle etc.
13. Design instagram client side.
14. Design a leetcode contest, leadership board system
15. Design Instagram
16. Design keyword search in FB Posts
17. There are music providers like spotify, apple music etc. Design a service for these providers to display top 10 songs played by each user. Was aked to write ER tables and API's.

18. Design a system like Hacker Rank for a programming contest and their ranking.

5. Behavioral:

1. Work experience, past projects, standard "tell me about a time" questions, hypothetical scenario questions
2. Usual stuff around things that I am proud of/ projects that I regret etc
3. Tell me about your current role
4. Tell me about a projects you are proud of
5. Tell Me About A Time When You Had To Give Someone Difficult Feedback. How Did You Handle It?(What kind of feedback you give ?)
6. Tell me about a time when you had a conflict with a manage and how you resolved it
7. What's the most difficult/challenging problem you have had to solve?
8. Which environment is best to you to work ?
9. Tell about best decision in your life from childhood ? Decision that changed your life
10. On which topics you want improve? What are doing to impoving on that topics ? Did you try build project on that topics ?

Facebook Interviews Non Premium Questions

- 1. Valid Parentheses
- 2. Search in Rotated Sorted Array
- 3. Sudoku Solver
- 4. Pow(x, n)
- 5. Maximum Subarray
- 6. Merge Intervals
- 7. Insert Interval
- 8. Valid Number
- 9. Plus One
- 10. Add Binary
- 11. Simplify Path

- 12. Minimum Window Substring
- 13. Search in Rotated Sorted Array II
- 14. Merge Sorted Array
- 15. Binary Tree Level Order Traversal
- 16. Best Time to Buy and Sell Stock
- 17. Best Time to Buy and Sell Stock III
- 18. Binary Tree Maximum Path Sum
- 19. Valid Palindrome
- 20. Fraction to Recurring Decimal
- 21. Best Time to Buy and Sell Stock IV
- 22. Rotate Array
- 23. Course Schedule
- 24. Design Add and Search Words Data Structure
- 25. Kth Smallest Element in a BST
- 26. Lowest Common Ancestor of a Binary Tree
- 27. Product of Array Except Self
- 28. Binary Tree Paths
- 29. First Bad Version
- 30. Move Zeroes
- 31. Serialize and Deserialize Binary Tree
- 32. Remove Invalid Parentheses
- 33. Longest Increasing Path in a Matrix
- 34. Insert Delete GetRandom O(1)
- 35. Convert a Number to Hexadecimal
- 36. Add Strings
- 37. Partition Equal Subset Sum
- 38. Find All Anagrams in a String

- 39. Continuous Subarray Sum
- 40. Subarray Sum Equals K
- 41. Task Scheduler
- 42. Exclusive Time of Functions
- 43. Valid Palindrome II
- 44. Is Graph Bipartite?
- 45. Custom Sort String
- 46. Smallest Subtree with all the Deepest Nodes
- 47. Shortest Bridge
- 48. Range Sum of BST
- 49. Verifying an Alien Dictionary
- 50. K Closest Points to Origin
- 51. Interval List Intersections
- 52. Max Consecutive Ones III
- 53. Minimum Remove to Make Valid Parentheses

✅ Helpful list of LeetCode Posts on System Design at Facebook, Google, Amazon, Uber, Microsoft

Note: If you find this post helpful, kindly upvote it. Thanks a lot everyone!

Hi Everyone, I have compiled a list of helpful LeetCode posts that I found on System Design Interviews at different companies. Let me know in the comments in case I have missed any posts. I will add them as edits to this post. Hopefully, this list will grow and help everyone with their interview preparation 🙂

Facebook

1. https://leetcode.com/discuss/interview-question/1002218/Facebook-or-Google-or-Top-System-Design-Interview-Questions-(Part-1)
2. https://leetcode.com/discuss/interview-question/1042229/Facebook-or-Google-or-Top-System-Design-Interview-Questions-(Part-2)
3. https://leetcode.com/discuss/interview-question/719253/Design-Facebook-%3A-System-Design-Interview
4. https://leetcode.com/discuss/interview-question/124657/Facebook-or-System-Design-or-A-web-crawler-that-will-crawl-Wikipedia

Google

1. https://leetcode.com/discuss/interview-question/system-design/496042/Design-video-sharing-platform-like-Youtube
2. https://leetcode.com/discuss/interview-question/system-design/733520/Design-YouTube-Very-detailed-design-with-diagrams
3. https://leetcode.com/discuss/interview-question/system-design/318811/Google-or-System-design-or-Design-a-translation-service-like-Google-Translate
4. https://leetcode.com/discuss/interview-question/system-design/692383/Google-or-Onsite-or-Design-a-organization-pharmacy-shop-with-managers

Amazon

1. https://leetcode.com/discuss/interview-question/341980/Amazon-or-System-Design-or-System-to-capture-unique-addresses-in-the-entire-world
2. https://leetcode.com/discuss/interview-question/system-design/124557/Amazon's-"Customers-who-bought-this-item-also-bought"-recommendation-system
3. https://leetcode.com/discuss/interview-question/system-design/344524/Amazon-or-Design-a-JobTask-Scheduler
4. https://leetcode.com/discuss/interview-question/373887/Amazon-or-System-Design-or-A-configuration-management-system

Uber

1. https://leetcode.com/discuss/interview-question/124673/Design-a-Location-Sharing-Android-Application

2. https://leetcode.com/discuss/interview-question/124542/Design-Uber-Backend
3. https://leetcode.com/discuss/interview-question/system-design/124558/Uber-or-Rate-Limiter

Microsoft

1. https://leetcode.com/discuss/interview-question/system-design/685310/Microsoft-virtual-or-Design-distributed-counter
2. https://leetcode.com/discuss/interview-question/system-design/680047/How-will-you-store-millions-of-subscribers-list-(assume-it-as-email-id)
3. https://leetcode.com/discuss/interview-question/system-design/598634/Microsoft-or-Onsite-or-System-Design-or-SDE-2

Others (Does not fall under any particular company):

1. https://leetcode.com/discuss/general-discussion/1105898/System-Design%3A-Introduction-to-Distributed-Systems-or-Designing-a-highly-available-system
2. https://leetcode.com/discuss/general-discussion/1114279/System-Design%3A-Introduction-to-Distributed-Systems-Pt.-2-or-Design-Highly-available-System
3. https://leetcode.com/discuss/general-discussion/1082786/System-Design%3A-Designing-a-distributed-Job-Scheduler-or-Many-interesting-concepts-to-learn
4. https://leetcode.com/discuss/general-discussion/1035779/System-Design-Reading-Resources
5. https://leetcode.com/discuss/general-discussion/901324/My-System-Design-Interview-Checklist-A-Gateway-to-FAANGs
6. https://leetcode.com/discuss/interview-question/124658/Design-URL-Shortening-service-like-TinyURL
7. https://leetcode.com/discuss/interview-question/1061256/Tips-on-System-design-from-a-20%2B-YOE-Engineer
8. https://leetcode.com/discuss/interview-question/system-design/566057/Machine-Learning-System-Design-%3A-A-framework-for-the-interview-day

9. https://leetcode.com/discuss/interview-question/system-design/547669/Algorithm-you-should-know-before-system-design

10. https://leetcode.com/discuss/interview-question/system-design/691010/System-Design-for-Mobile-App-Developers

Design Facebook : System Design Interview

Design a simple model of Facebook where people can add other people as friends. In addition, where people can post messages and that messages are visible on their friend's page. The design should be such that it can handle 10M of people. There may be, on an average 100 friends each person has. Every day each person posts around 10 messages on an average.

Use Case

1. A user can create their own profile.
2. A user can add other users to his friend list.
3. Users can post messages to their timeline.
4. The system should display posts of friends to the display board/timeline.
5. People can like a post.
6. People can share their friends post to their own display board/timeline.

Constraints

1. Consider a whole network of people as represented by a graph. Each person is a node and each friend
   relationship is an edge of the graph.
2. Total number of distinct users / nodes: 10 million
3. Total number of distinct friend's relationship / edges in the graph: 100 * 10 million
4. Number of messages posted by a single user per day: 10
5. Total number of messages posted by the whole network per day: 10 * 10 million

Basic Design

Our system architecture is divided into two parts:

1. First, the web server that will handle all the incoming requests.
2. The second database, which will store the entire person's profile, their friend relations and posts.

First, three requirements creating a profile, adding friends, posting messages are written some information
to the database. While the last operation is reading data from the database.

The system will look like this:

1. Each user will have a profile.
2. There will be a list of friends in each user profile.
3. Each user will have their own homepage where his posts will be visible.
   A user can like any post of their friend and that likes will reflect on the actual message shared by his
   friend.

If a user shares some post, then this post will be added to the user home page and all the other friends of
the user will see this post as a new post.

Bottleneck

A number of requests posted per day is 100 million. Approximate some 1000 request are posted per second. There will be an uneven distribution of load so the system that we will design should be able to
handle a few thousand requests per seconds.

Scalability

Since there is, a heavy load we need horizontal scaling many web servers will be handling the requests.
In doing this we need to have a load balancer, which will distribute the request among the servers.
This approach gives us a flexibility that when the load increases, we can add more web servers to handle
the increased load.
These web servers are responsible for handling new post added by the user. They are responsible for generating various user homepage and timeline pages. In our case, the client is the web browser, which is rendering the page for the user.

We need to store data about user profile, Users friend list, User-generated posts, User like statues to the
posts.
Let us find out how much storage we need to store all this data. The total number of users 10 million. Let
us suppose each user is using Facebook for 5 to 6 years, so the total number of posts that a user had produced in this whole time is approximately 20,000 million or 20 billion. Let us suppose each message
consists of 100 words or 500 characters. Let us assume each character take 2 bytes.

```
Total memory required = 20 * 500 * 2 billion bytes.

= 20,000 billion bytes

= 20, 000 GB
= 20 TB

1 gigabyte (GB) = 1 billion bytes

1000 gigabytes (GB) = 1 Terabytes
```

Most of the memory is taken from the posts and the user profile and friend list will take nominal as compared with the posts. We can use a relational database like SQL to store this data. Facebook and twitter are using a relational database to store their data.

Responsiveness is key for social networking site. Databases have their own cache to increase their performance. Still database access is slow as databases are stored on hard drives and they are slower than RAM. Database performance can be increased by replication of the database. Requests can be distributed between the various copies of the databases.

Also, there will be more reads then writes in the database so there can be multiple slave DB which are
used for reading and there can be few master DB for writing. Still database access is slow to we will use
some caching mechanism like Memcached in between application server and database. Highly popular
users and their home page will always remain in the cache.

There may be the case when the replication no longer solves the performance problem. In addition, we
need to do some Geo-location based optimization in our solution.
Again, look for a complete diagram in the scalability theory section.
If it were asked in the interview how you would store the data in the database. The schema of the database can look like:

Table Users:

```
1. User Id

2. First Name

3. Last Name

4. Email

5. Password

6. Gender

7. DOB

8. Relationship
```

Table Posts:

```
1. Post Id

2.  Author Id

3.  Date of Creation

4.  Content
```

Table Friends:

```
1. Relation Id

2. First Friend Id

3. Second Friend Id
```

Table Likes:

```
1. Id

2. Post Id

3. User Id
```

Facebook | System Design | A web crawler that will crawl Wikipedia

We need to deploy the same software on each node. We have 10,000 nodes, the software can know about all the nodes. We have to minimize communication and make sure each node does equal amount of work.

Encountered this question in facebook.

My solution:
Intial Proposal:
Start out with the root page, and based on a hash function decide if that page is for you (the node) or not.

What if one node fails or does not work?
When encountering a page, ping to see whether the node handling that page is online, if not

use a secondary hash function to determine alternate handler. If you are the alternate handler, handle the page, if not ping to see if the alternate handler is online. Do this for al hash functions you decide to have, more hash functions means less impact by failure of one node.

How do you know when the crawler is done?
Pass around a timestamp of the last crawled page. If the timestamp gets back to you without changing, then you are done.

Additional questions on the requirements:

1. We need to check whether a link has been visited or not, to avoid loops. Where can we keep the "history" of visited links? Should each server keep a copy of the pages locally and some replicas on other servers? If that's the case, maybe we need to ask other servers (who should handle the link based on the hash) whether they have already visited that page. This seems to be quite chatty so we may want to use a local bloom filter (updated periodically) to cut down on the traffic.
2. For a naive solution, the crawling is complete if the current page is crawled and no new links in the work queue. For the distributed case, we need to make sure the work queue on each and every server is empty after certain timeout (the max allowed to complete the parsing of one page). Would this be good enough answer?

Behavioral/Leadership questions for interviews

Behavioral round questions are very difficult to answer because these questions are scenario based. The interviewer will take you back in your history and will ask questions from there. Sometimes it becomes difficult to recall important points so it is must that we should be well prepared for it before appersing for the interview.

Below are some commonly asked behavioral round questions. I have tried to cover most of the questions that may be useful. Please comment if I missed any important questions which you think should be added.

- **Have you ever faced any challenges/conflicts with colleagues.**
- **Challenge with manager when you had to disagree with your manager and manager agreed with your point.**
- **Challenge with manager when you had to agree with manager even though he was wrong.**

- What you could have done for above so that you could have convinced your manager?
- What is your biggest strength and weakness?
- Situation when you had to agree with your junior.
- Your improvement areas.
- Recent improvements which helped your in professional and personal life.
- Describe the project that you had the most trouble with. What would you have done differently?
- Talk about a time where you had to make a decision in a lot of ambiguity.
- What was the biggest takeaway from your current job that you'll carry with you throughout your career?
- How do you keep your team engaged?
- Give an example of how you set your goals and achieve them.
- Tell me about a time when you solved a conflict at work.
- Give examples of situations where you have shown effectiveness, empathy, adaptability, and humbleness.
- Why do you want to change jobs? Why now?
- Share an example of how you were able to motivate employees or co-workers.
- What do you do when the requirement from the stakeholder is vague?
- How do you make a case for your vision and opinion?
- How do you stay organized?
- Give an example of an occasion when you used logic to solve a problem.
- What do you do if you don't know the solution for a certain problem and nobody can help at the moment?
- Have you handled a difficult situation with a coworker? How?
- How do you experiment?
- What websites do you spend a lot of time on?
- What do you do if you disagree with your boss?

- Talk about a time when you failed.
- Where do you see yourself in next 5/10 years?
- How do you know a feature you have built is working ?
- Tell me about a time when your suggestions brought positive changes/impact to the team.

Tell me about a situation where you had a conflict with someone on your team. What was it about? What did you do? How did they react? What was the outcome?
Give an example of when you saw a peer struggling and decided to step in and help. What was the situation and what actions did you take? What was the outcome?
Tell me about a time you committed a mistake?
Tell me about a time when your earned your teammate's trust?
Tell me about a time when you couldn't meet your deadline?
Tell me about a time when your teammate didn't agree with you? What did you do?
Tell me about a time when you invented something?
Tell me about a time when you took important decision without any data?
Tell me about a time when you helped one of your teammates?
Have you ever been in a situation where you had to make a choice among a few options, but did not have a lot of time to explore each option
Have you ever failed at something? What did you learn from it?
name time when you went out of your way to help someone?
Time when you came up with novel solution.
Received negative feedback from manager and how you responded.
Time when you went above and beyond your job responsibilities.
Time when you did not have enough data and had to use judgement to make decision.
Time when you helped someone in their work.
Time when you helped someone grow in career and it benefited them.
Time when you helped someone grow but did not benefit them.
Time when you were 75% through a project and realized you had the wrong goal.
Time when your team members were not supporting something but you pushed and went for a more optimal solution.
Time when you pushed back a decision from your management for better long term benefits.
Time when you failed to meet your commitment
Tell me about yourself. Tell me about a project you're working on.
Time when you were working on a project on a time constraint
Time when you didn't meet a deadline
Time when you needed help from somebody

Tell me about yourself.
Tell me about a time you had to help a team member struggling with a task.
Tell me about a time you faced an obstacle and how you overcame it.
Tell me about one of your projects?
Tell me about one of your projects so the same as the first guy.
Tell me a time you took some on some risk
Have you ever gone out of your way to help a peer? (ownership)
Have you ever had to make a tough decision without consulting anybody? (bias for action)
asked me about my past projects that I've worked on and gave me detailed explaination about the Internship.
Tell me about a time when you learned new technologies
Tell me about a time when you took a decision on your own without the manager's prior approval
Tell me about a time you had multiple solutions and you had to select an optimal one
Tell me about a time when you innovated and exceeded the expectation
Tell me about a time where you had to make a decision based on limited information and how it impacted the outcome.
Tell me about a time where you had limited time and how it impacted
Tell me about a time where you did not know something and how you tackled it(Something related to it)
first one was about handling a tight deadline, second is setbacks on projects?
Handling a tight deadline
How would you help a new employee who is facing technical difficulties?
disagree and commit and ownership LPs.
Tell me about your yourself (the general icebreaker).
Tell me about tim when you faced a difficult challenge.
Tell me about a time when you needed help from someone during a project.
Tell me about a time when you thought of an unpopular idea.
Tell me about a time when you had to decide upon something without consulting your superior.
Tell me about a time when you had to face tight time constraints during a project.
Tell me about yourself.
Tell me about a time when you did not meet your deadlines for a project.
Tell me about a time when you had conflicting ideas with your teammates and how did you resolve them?
a project you're proud of
a time when you faced a setback initially but still achieved the goal.
a time when you had to cut corners to meet a deadline

"Tell me about a time when you felt under pressure that you wouldn't be able to get something done or had to take a pivot at the last minute"

Facebook Phone Interview Questions

https://leetcode.com/problems/product-of-array-except-self/

https://leetcode.com/problems/leftmost-column-with-at-least-a-one/

https://leetcode.com/problems/employee-free-time/

https://www.*.org/lowest-common-ancestor-in-a-binary-tree-set-2-using-parent-pointer/

https://leetcode.com/problems/subarray-sum-equals-k/

https://leetcode.com/problems/copy-list-with-random-pointer/

https://leetcode.com/problems/serialize-and-deserialize-binary-tree/

https://leetcode.com/problems/word-break-ii/

Validate Single Binary Tree

- https://leetcode.com/discuss/interview-question/347374/

Task Scheduler

- https://leetcode.com/discuss/interview-question/673575/Facebook-or-Phone-or-Task-Scheduler
- https://leetcode.com/problems/task-scheduler/

https://leetcode.com/problems/target-sum/

https://leetcode.com/problems/generate-parentheses/

https://leetcode.com/problems/nth-digit/

https://leetcode.com/problems/insert-delete-getrandom-o1/

https://leetcode.com/problems/insert-delete-getrandom-o1-duplicates-allowed/

https://leetcode.com/problems/accounts-merge/

https://leetcode.com/problems/valid-word-abbreviation/

https://leetcode.com/problems/candy-crush/

https://leetcode.com/problems/koko-eating-bananas/

https://leetcode.com/problems/binary-tree-right-side-view/

https://leetcode.com/problems/restore-ip-addresses/

https://leetcode.com/problems/powx-n/

https://leetcode.com/problems/russian-doll-envelopes/

https://leetcode.com/problems/walls-and-gates/

https://leetcode.com/problems/best-time-to-buy-and-sell-stock/

https://leetcode.com/problems/find-largest-value-in-each-tree-row/

https://leetcode.com/problems/add-strings/

https://leetcode.com/problems/combination-sum/

https://leetcode.com/problems/maximum-swap/

Dot Product of Sparse vectors
https://leetcode.com/discuss/interview-question/124823/

- **Find an efficient way to represent a vector (1,1,1,1,1,1,22,2,2,2,2,2,2,2,2,3,4,4,5,6,6,7,7,7,8,8,8,9,9,9,99,9,,1,1,1,1,1,1,2,3,34, 3,4,,3,3,3,3....)**
- **Use the representation you come up with to compute dot product of two vectors**
    - **Ex: If you come up with MyDataStructure to represent a vector, then your function signature will be**
      int dotProduct(MyDataStructure vector1, MyDataStructure vector2)
      // dot product of two vectors [1,2,3,4] and [5,6,7,8] is 1 * 5 + 2 * 6 + 3 * 7 + 4 * 8
    - **Take advantage of your "efficient" representation to compute the dot product faster.**

https://leetcode.com/problems/random-pick-with-weight/

Some questions are the closest that it can get to the actual question. Like Russian Doll envelopes or Task Scheduler.

Amazon Behavioral questions | Leadership Principles | LP

1.https://interviewgenie.com/blog-1/category/Amazon+interviews
2.https://www.youtube.com/channel/UCw0uQHve23oMWgQcTTpgQsQ/playlists
3.https://medium.com/@scarletinked/are-you-the-leader-were-looking-for-interviewing-at-amazon-8301d787815d

Tell me about a situation where you had a conflict with someone on your team. What was it about? What did you do? How did they react? What was the outcome?

Give an example of when you saw a peer struggling and decided to step in and help. What was the situation and what actions did you take? What was the outcome?
Tell me about a time you committed a mistake?

Tell me about a time when your earned your teammate's trust?

Tell me about a time when you couldn't meet your deadline?

Tell me about a time when your teammate didn't agree with you? What did you do?

Tell me about a time when you invented something?

Tell me about a time when you took important decision without any data?

Tell me about a time when you helped one of your teammates?

Have you ever been in a situation where you had to make a choice among a few options, but did not have a lot of time to explore each option

Have you ever failed at something? What did you learn from it?

name time when you went out of your way to help someone?

Time when you came up with novel solution.
Received negative feedback from manager and how you responded.
Time when you went above and beyond your job responsibilities.
Time when you did not have enough data and had to use judgement to make decision.
Time when you helped someone in their work.
Time when you helped someone grow in career and it benefited them.
Time when you helped someone grow but did not benefit them.
Time when you were 75% through a project and realized you had the wrong goal.
Time when your team members were not supporting something but you pushed and went for a more optimal solution.
Time when you pushed back a decision from your management for better long term

benefits.
Time when you failed to meet your commitment

Tell me about yourself. Tell me about a project you're working on.

Time when you were working on a project on a time constraint

Time when you didn't meet a deadline

Time when you needed help from somebody

Tell me about yourself.
Tell me about a time you had to help a team member struggling with a task.
Tell me about a time you faced an obstacle and how you overcame it.

Tell me about one of your projects?
Tell me about one of your projects so the same as the first guy.
Tell me a time you took some on some risk

Have you ever gone out of your way to help a peer? (ownership)
Have you ever had to make a tough decision without consulting anybody? (bias for action)
asked me about my past projects that I've worked on and gave me detailed explaination about the Internship.

Tell me about a time when you learned new technologies
Tell me about a time when you took a decision on your own without the manager's prior approval
Tell me about a time you had multiple solutions and you had to select an optimal one

Tell me about a time when you innovated and exceeded the expectation

Tell me about a time where you had to make a decision based on limited information and how it impacted the outcome.

Tell me about a time where you had limited time and how it impacted

Tell me about a time where you did not know something and how you tackled it(Something related to it)

first one was about handling a tight deadline, second is setbacks on projects?

Handling a tight deadline
How would you help a new employee who is facing technical difficulties?

disagree and commit and ownership LPs.

Tell me about your yourself (the general icebreaker).
Tell me about tim when you faced a difficult challenge.
Tell me about a time when you needed help from someone during a project.

Tell me about a time when you thought of an unpopular idea.
Tell me about a time when you had to decide upon something without consulting your superior.
Tell me about a time when you had to face tight time constraints during a project.

Tell me about yourself.
Tell me about a time when you did not meet your deadlines for a project.
Tell me about a time when you had conflicting ideas with your teammates and how did you resolve them?

a project you're proud of
a time when you faced a setback initially but still achieved the goal.
a time when you had to cut corners to meet a deadline

"Tell me about a time when you felt under pressure that you wouldn't be able to get something done or had to take a pivot at the last minute"

I will update the list regularly
Hope this helps

Source: LeetCode Interview experiences!!!!

What are the main features of OOP?

This is a common interview question, which seems to be simple and is also easy to be ignored. However without preparation, it can be difficult for interviewer to give a clear and impressive answer.

- **Abstraction**
  We try to obtain abstract view, model or structure of real life problem, and reduce its unnecessary details. With definition of properties of problems, including the data which are affected and the operations which are identified, the model abstracted from problems can be a standard solution to this type of problems. It is an efficient way since there are nebulous real-life problems that have similar properties.

- **Encapsulation**
  Encapsulation is the process of combining data and functions into a single unit called class. In Encapsulation, the data is not accessed directly; it is accessed through the functions present inside the class. In simpler words, attributes of the class are kept private and public getter and setter methods are provided to manipulate these attributes. Thus, encapsulation makes the concept of data hiding possible. (Data hiding: a language feature to restrict access to members of an object, reducing the negative effect due to dependencies. e.g. "protected", "private" feature in C++)

- **Inheritance**
  The idea of inheritance is simple, a class is based on another class and uses data and implementation of the other class. And the purpose of inheritance is Code Reuse.

- **Polymorphism**
  Polymorphism is the ability to present the same interface for differing underlying forms (data types). With polymorphism, each of these classes will have different underlying data. A point shape needs only two co-ordinates (assuming it's in a two-dimensional space of course). A circle needs a center and radius. A square or rectangle needs two co-ordinates for the top left and bottom right corners and (possibly) a rotation. An irregular polygon needs a series of lines.

Ref:
[1] http://www.geeksforgeeks.org/commonly-asked-oop-interview-questions/
[2] http://gd.tuwien.ac.at/languages/c/c++oop-pmueller/tutorial.html
[3] https://stackoverflow.com/questions/16014290/simple-way-to-understand-encapsulation-and-abstraction
[4] https://stackoverflow.com/questions/1031273/what-is-polymorphism-what-is-it-for-and-how-is-it-used

The most important OOP features are these:
• Abstraction
• Encapsulation and data hiding
• Polymorphism
• Inheritance
• Reusable code

suggest looking at the book 'Clean Architecture' by Robert Martin. He explains it very nicely.

I will attempt to paraphrase in this post.

Usually in OOP we think of four things, Abstraction, Encapsulation, Inheritance, and Polymorphism. If you were to compare a non-OOP language versus an OOP language, like C and C++. There is only one thing that a non-OOP doesn't have.

Abstraction and Encapsulation both C and C++ have, actually in C++ show private variables in header files, so you lose some Encapsulation. Even other languages like python and java don't have header files, so we don't even follow enough Encapsulation.

For inheritance and Polymorphism, you can do that in both C and C++. C just had to use some trickier with pointers and creating two separate classes both with the same named methods and variables in the correct order to mimic the affect of them. 'Dangerous' but doable.

Now what, we've exhausted all options? What C++ has that C doesn't have are abstract classes. Why are abstract classes that important? Abstract classes allow for dependency inversion which was difficult for programmers to perform due to limitation of the language.

Google Online Assessment Questions

US/EU

- Min Amplitude [New Grad] 🆕
- Ways to Split String [New Grad] 🆕
- Maximum Time ⭐ ⭐ ⭐ [Intern]
- Min Abs Difference of Server Loads ⭐ [Intern]
- Most Booked Hotel Room ⭐ ⭐ ⭐ [Intern]
- Minimum Domino Rotations For Equal Row [New Grad]
- Time to Type a String
- Maximum Level Sum of a Binary Tree
- Min Number of Chairs
- K Closest Points to Origin
- Odd Even Jump
- License Key Formatting

- Unique Email Addresses
- Fruit Into Baskets
- Min Days to Bloom
- Fill Matrix
- Decreasing Subsequences
- Max Distance
- Stores and Houses

Repeatedly Asked Microsoft Onsite Questions | DS + LLD + HLD

Low Level Design Questions:

1. **Design Parking Lot**

- Single Entry and Exit Gates
- Multiple Entry and Exit Gates

2. **Design Multiplayer Sudoku Game**

- Players have to Rotate the Pin in order to decide the turn.
- Player who is able to fill the last empty cell will win.

3. **Design LRU (Static + Dynamic Input Flow)**

- Follow Up : Implement LFU with Least Changes in the previous design (LRU)

4. **Design In Memory Cache**
5. **Design Snake and Ladder Game**

- How would you change the design if wild cards are allowed.

6. **Design Exception Class With Test Cases**

- Note: Don't forget to use Singleton Pattern Here.

7. **Design Money Splitter**
8. **Design Notification Service**

9. **Design Message Queue**
10. **Design a Terminal/Command Prompt**

High Level Design Questions:

1. **Design Whatsapp**
2. **Design Uber Backend**
3. **Design Tiny URL**
4. **Design Dream11**
5. **Design Leetcode**
6. **Design MS Teams**
7. **Design Dominos/PizzaHut**
8. **Design Big Bazaar**
9. **Design Zomato/Swiggy**
10. **Design Docker**

DS/ALGO Questions:

1. **Reverse Vowels in the string.**

- Using Constant space

2. **Longest Palindromic Substring.**
3. **Find the largest island in Matrix of 0s and 1s**
4. **Longest Repeating Subsequence**
5. **Compress and Uncompress a Tree/Graph/LinkedList**
6. **All Variations of Jump Game**
7. **All Variations of Stocks Buy and Sell DP**

8. **Word Search 1/2**
9. **Median of 2 sorted arrays**
10. **Kth largest/smallest element in the rowwise and columnwise sorted Matrix**

Amazon Final Interview Questions | SDE1

I've compiled a list of questions people have been asked for Amazon who have had 3 virtual interviews for SDE 1:

- Two Sum (#1)
- Median of Two Sorted Arrays * (#4)
- Longest Palindromic Substring (#5)
- String to Integer (atoi) (#8)
- Integer to Roman (#12)
- Roman to Integer (#13)
- Valid Parentheses (#20)
- Merge K Sorted Lists (#23)
- Valid Sudoku (#36)
- Combination Sum (#39)
- Permutations (#46)
- Merge Intervals (#56)
- Rotate List (#61)
- Minimum Path Sum (#64)
- Word Search (#79)
- Validate Binary Search Tree(#98)
- Same Tree ~ (#100)
- Symmetric Tree ~ (#101)
- Binary Tree Level Order Traversal (#102)
- Convert Sorted List to Binary Search Tree (#109)

- Populating Next Right Pointers in Each Node (#116)
- Best Time to Buy and Sell Stock (#121)
- Word Ladder II (#126)
- Word Ladder (#127)
- LRU Cache (#146)
- Min Stack (#155)
- Number of Islands (#200)
- Course Schedule (#207)
- Course Schedule II (#210)
- Add and Search Word - Data structure design (#211)
- Word Search II (#212)
- Integer to English Words (#273)
- Game of Life (#289)
- Find Median from Data Stream (#295)
- Longest Increasing Subsequence (#300)
- Design Tic-Tac-Toe (#348)
- Pacific Atlantic Water Flow (#417)
- Minesweeper (#529)
- Diameter of Binary Tree (#543)
- Reorganize String (#767)

**not required to solve*
~solve both iteratively and recursively

Amazon Leadership Principle (Behavioral) Questions

1. A time when you faced a problem that had multiple solutions
2. When did you take a risk, make a mistake or fail? How did you respond?
3. What did you do when you needed to motivate a group of individuals?
4. How have you leveraged data to develop a strategy?
5. A time when a team member didn't meet your expectations on a project
6. Tell me about a time you failed and what you learned from it
7. The last time you had to apologize to someone
8. A time when you were 75% through a project, and you had to pivot strategy
9. How do you show your customer Obsession?
10. How do you wow your customers?
11. How do you develop client relationships?
12. How do you understand your customer's needs?
13. Describe a time when you made a judgement call with limited information
14. Describe a time when you improved a process with limited budget
15. Describe a problem you solved. What was the root cause of the problem?
16. An example of when you had to push back to HQ or challenged a decision
17. Provide an example of when you had personally demonstrated ownership
18. Tell me about a time you went above and beyond
19. When you took on something outside your area of responsibility. Outcome?
20. An example of when you saw a peer struggling and decided to step in and help
21. Describe a situation where you disagreed with a supervisor?
22. Tell me about a time you recovered from a difficult situation
23. Tell me about a situation where you had to solve a difficult problem
24. Describe a project or idea (not necessarily your own) that was implemented primarily because of your efforts

25. Do you feel you work well under pressure? If so, describe a time when you have done so...
26. Give me an example of when you motivated others
27. Tell me about a time where you had to delegate tasks during a project
28. Give me an example of when you showed initiative and took the lead
29. Tell me about a time when you missed an obvious solution to a problem
30. Tell me about your proudest professional accomplishment
31. Tell me about a time you handled a crisis
32. Tell me about a time you had to deal with ambiguity
33. Are you easy to get along with?
34. Do you collaborate well?
35. Tell me about a team project you worked on
36. Describe a time when you had to bring two departments together to work more efficiently with each other
37. Tell me about a time you stepped up into a leadership role
38. Tell me about a time you led an important meeting
39. How do you motivate other people to work with you?
40. Describe your style in dealing with irate customers
41. Describe a difficult decision you had to make in your business life and how you went about doing it
42. Tell me about a time you made a mistake that affected a client adversely and how you coped with it
43. Tell me about a time when you disagreed with a rule or approach
44. Describe a situation when you were able to use persuasion to successfully convince someone to see things your way
45. Tell me about a time when you were asked to take sides regarding another employee and you remained neutral
46. We've all had to work with people that don't like us. How do you deal with someone that doesn't like you?

47. Give me an example of a time when you set a goal and were able to meet or achieve it
48. Describe a time when you went beyond your job description to save your company time and money
49. Give an example of a situation in which you took specific steps to further your career
50. Talk about a time when you had to work closely with someone whose personality was very different from yours
51. Give me an example of a time you faced a conflict while working on a team. How did you handle that?
52. Describe a time when you struggled to build a relationship with someone important. How did you eventually overcome that?
53. We all make mistakes we wish we could take back. Tell me about a time you wish you'd handled a situation differently with a colleague?
54. Tell me about a time you needed to get information from someone who wasn't very responsive. What did you do?
55. Describe a time when it was especially important to make a good impression on a client. How did you go about doing so?
56. Give me an example of a time when you did not meet a client's expectation. What happened, and how did you attempt to rectify the situation?
57. Tell me about a time when you made sure a customer was pleased with your service
58. When you're working with a large number of customers, it's tricky to deliver excellent service to them all. How do you go about prioritizing your customers' needs?
59. Tell me about a time when you were under a lot of pressure. What was going on, and how did you get through it?
60. Describe a time when your team or company was undergoing some change. How did that impact you, and how did you adapt?
61. What is the most difficult decision you ever took in software?
62. Tell me about a time you had to be very strategic in order to meet all your top priorities
63. Describe a long-term project that you managed. How did you keep everything along in a timely manner?

64. Sometimes it' jut not possible to get everything on your to-do lit done. Tell me about a time your responsibilities got a bit overwhelming. What did you do?
65. Tell me about a time you set a goal for yourself. How did you go about ensuring that you would meet your objective?
66. Describe a time when you were the resident technical expert. What did you do to make sure everyone was able to understand you?
67. Tell me about a time when you had to rely on written communication to get your ideas across to your team.
68. Tell me about a successful presentation you gave and why you think it was a hit
69. Describe a time when you saw some problem and took the initiative to correct it rather than waiting for someone else to do it
70. Tell me about a time when you worked under close supervision or extremely loose supervision. How did you handle that?
71. Give me an example of a time you were able to be creative with your work. What was exciting or difficult about it?
72. Tell me about a time you were dissatisfied in your work. What could have been done to make it better?
73. How do you handle a challenge? Give an example.
74. Describe a decision you made that wasn't popular and how you handled implementing it
75. Share an example of how you were able to motivate employees or co-workers
76. Tell me about the toughest decision you've had to make in the past six months
77. Tell me about a major mistake you made, and what you did to correct it
78. Tell me about the last time a customer or coworker got upset with you
79. Tell me about a time you knew you were right, but still had to follow directions or guidelines
80. Tell me about the lat time your workday ended before you were able to get everything done
81. Tell me about a time you had to raise an uncomfortable issue with your boss
82. Tell me about a goal you failed to achieve

83. Tell me about a time you worked on a team with individuals from different cultural backgrounds
84. Describe a time when you found it difficult to work with someone from a different background
85. Give an example of a situation where you had to take into account the sensitiveness of different parties
86. Tell me about a time you observed culturally insensitive behavior on the job
87. What experiences have you had with recruiting, hiring, training, and/or supervising a diverse workforce?
88. Tell me about a time recently when you had to take someone's cultural perspective into account in dealing with themselves
89. When have you worked the hardest?
90. Tell me about a time when you had to juggle multiple important projects
91. What was your biggest mistake and what did you learn from it?
92. Tell me about a decision that you regret
93. What's your greatest professional regret?
94. Describe a situation in which you found a creative way to overcome an obstacle
95. Tell me about a time when you came up with a new approach to a problems
96. What's the most innovative new idea that you have implemented?
97. Tell me about a two improvements you have made in the last six months
98. Describe a time when you anticipated potential problems and developed preventive measures
99. Tell me about a time when you had to analyze information and make a recommendation
100. Tell me about a time when you demonstrated leadership skills
101. When have you delegated efficiently?
102. Describe a time when you led by example
103. Who have you coached or mentored to achieve success?

104. Describe a project that required input from people at different levels in the organization

105. Share a rewarding team experience

106. Tell me about a time when you worked with a difficult team member

107. Give me an example of a team project that failed

108. Tell me about a time you stepped up into a leadership role

109. Let's say you need something important from a coworker and that person isn't responding. How would you deal with this?

110. Tell me about a time when you had to manage multiple responsibilities. How did you handle it?

111. Let's say you're working on a major project and you'r in the weeds. How do you find your way out?

112. We all deal with difficult customers from time to time. Tell me about a challenging client-facing situation and how you handled it.

113. Sometimes we have an all-hands-on-deck situation that may require everyone to work extra hours. How would you handle that?

114. Everyone starts somewhere. Talk about a time when you were new on the job and had a lot to learn. How did you manage?

115. I'm interested in how you recharge when you're not working. What do you do with your downtime?

116. Has your manager/supervisor ever asked you to do something that you were uncomfortable with? How did you handle this?

117. Tell me about a time you witnessed a fellow employee do something you didn't think was appropriate

118. Tell me about a time you made a mistake at work. How did you deal with it?

119. How have you reacted to a colleague who regularly lets the team down?

120. Tell me about a time you had to work at a fast pace for an extended period of time. How did you maintain your work pace?

121. Tell me about the longest time it took you to conclude a deal with a customer

122. Describe a situation when you had to overcome a number of obstacles to achieve an objective

123. Describe what you do to control mistakes in your work

124. Describe a situation in which you had to schedule your activities to meet a work objective

125. Describe a new idea or suggestion that you made to your supervisor recently

126. Describe a situation when you negotiated with others in your organization to reach agreement

127. Tell me about a time you had to quickly adjust your work priorities to meet changing demands

128. Tell me about a time you had to make decision under pressure to meet a deadline

129. Describe a time when you were faced with a stressful situation and used your coping skills

130. Describe a time that you demonstrated the ability to be an effective team member

131. Describe a time when you sacrificed short term goals for long term success

132. Tell me about a time when you had to choose between technologies for a project

133. Tell me about a time when you disagreed with your hiring manager. What did you do?

134. Tell me about a time when you had to coach an employee

135. Tell me about a time you heard difficult feedback and how you handled it

136. Describe a time when you had to convince the leaders of your team despite their disagreement. What did you do to convince them?

137. Tell me about a time when you were not able to meet a time commitment. What prevented you from meeting it? What was the outcome and what did you learn from it?

138. Describe an example of when you took risk and failed

139. Tell me about a time when you overcame an obstacle and delivered results

140. Tell me about a time when you disagreed with your boss

141. What is the most significant impact of your work inside a team?

142. Tell me about a time when you helped improve an internal work process. How you did it and what was the result?

143. What is the best invention or idea you had in the past two years?

144. Tell me about a time you exceeded your expectations

145. Describe a project that you are particularly proud of. How did it impact your company? What challenges did you encounter and how did you solve them?

146. Tell me about a time you had a conflict at work

147. Tell me about a time when you took the lead on a difficult project

148. Tell me about a time when you received negative feedback from your manager. How did you respond?

149. How would you handle a project that is expected to be behind schedule

150. Describe a time when you received criticism and how you handled it

151. Tell me about a time when you had to make a judgement call without having time to refer to a manager/supervisor

152. Give me an example where you strongly held an opinion and you were the outlier

153. Why Amazon?

154. Explain a situation where you faced a problem and how did you deal with it?

155. Tell me about a time you made a decision without consulting your manager

156. What's your greatest strength?

157. Tell me about a time when you were curious; what were your expectations and what did you learn from them?

A guide for Behavioral round

My interactions with many students preparing and giving interviews have made me realise that students do not prepare enough for behavioral questions in general. They treat these categories of questions as secondary but fail to realise that a good leadership answer can change the interviewers perspective towards them as a prospective new hire. A good response can be a decisive factor if there is confusion in the mind of interviewers regarding your technical prowess. It is extremely important to realise that apart from assessing the technical abilities, the interviewers also assess your ability to understand the culture and the values of the company. They want to see how you tackle different situations and how well you can gel into the company as a prospective employee. Hence, this category demands a thorough preparation. While this seems like a wastage of time at first, the benefits are immense. A good preparation of only about a week can prepare you for the long term. Once you get a knack for answering these questions, not enough practise is required to maintain this skill, and you can ace the leadership round of pretty much every company. Companies like Amazon stress too much on their leadership principles. In fact, for interns, the final round is more of a behavioural than a technical round. Hence, good preparation is required.

A general mindset of students is to know just the intricacies of the projects they have worked on in college and the company and knit a story during the interview. While that is a plausible approach, the resultant answers are usually not well structured and fail to impress the interviewers. To them, instead of the technologies and the intricacies of the project, the approach, the mindset, and the learnings are far more important, which we as interviewees sometimes fail to address. I agree that there are so many different types of leadership questions that it is virtually impossible to prepare for all such questions. But the good news is, just like DSA, we can segregate these questions into categories and then smartly target those categories. I will be talking about some patterns that I have seen, some tips I have found useful, and then address how to structure your answers.

The categories:

- **Data**: Many times, the interviewer wants to know how you handle the data. Data is unarguably one of the major factors to devise solutions, and the interviewer wants to know how your decision making is affected by different amounts of data. Your approach when you have say, a large amount of data, just enough data, or not enough data available tells a lot about your problem-solving skills. While answering such questions, you need to make sure that you leverage the data to come up with a robust and scalable solution that does not exploit much of the company's resources. You should talk about general trends, observations, insights, and

outliers.
Related questions:

  - An approach you took when you had enough data available.
  - An approach you took when you had less data available.
  - A choice you made when there were multiple solutions available.
  - A time when you leveraged data to develop a solution.

- **Decision**: This category includes questions where you have to take key decisions based on different situations. Those decisions can be instinctive, quick, risky, etc. The interviewer wants to know how you handle pressure and take important decisions when a lot is at stake. Make sure your decisions are sound, scalable, and have solid data backing.
  Related questions:
  - A time when you took a quick/instinctive decision and succeeded.
  - A time when you took a risky decision and failed.
- **Innovation and hard work**: This is a general category of positive questions where the interviewer wants to know more about how you think and how curious you are to learn more about new technologies and culture. Curiosity drives innovation which along with hard work improves your chances of succeeding at a company. It would be great if you have some projects where you have solved a problem using a not so usual approach and solved a pain point of the customers. The qualities you want to show while answering such questions are curiosity, hard work, inquisitiveness, and modesty.
  Related questions:
  - A time when you devised a simple solution to a complex problem.
  - A time when you raised the bar.
  - A time when you solved a pain point.
  - A time when you influenced a change by asking questions.
  - A time when you invented something.
  - A time when you did more than what was required.
  - A time when you did something for customer satisfaction.

- **Deadline**: It is a very important category of questions. Working in a company, deadlines keep the engineers on their toes. Many times to fulfil the deadlines, the quality of the code might get compromised. Sometimes a last-minute bug or a late dependency resolution results in a deadline shift. Instead of playing the blame game, you need to take ownership of the mistakes and ensure that the customer impact is minimal. Some viable reasons for missing the deadlines are late dependency resolution, short-term sacrifices for long-term gain like developing a scalable and a more robust solution, an insufficient case study resulting in incorrect time estimates. An important aspect of this category is learnings. What steps did you take to mitigate the damage (for instance, keeping the manager in the loop instead of informing him at the last moment that the deadline might be missed), and what measures did you take so that such a thing is never repeated. How you learnt from your mistakes and grew as an engineer is something you should address.
  Related questions:
  - A time when you missed a deadline
  - A time when you had to shift a deadline.
- **Mistakes**: It is a category of negative questions that should be answered with care. Just like the deadline category, you should take ownership of the mistakes committed and focus more on how you mitigated the damage. Ensure that the customer impact is minimal. You can also talk about why the approach you took seemed right then and what went wrong. Understanding where you went wrong and the steps you took to avoid such mistakes from happening in the future is crucial.
  Related questions:
  - A time when you made a mistake.
  - A time you used the wrong approach when there was a better one.
  - Something you did you thought was right, but it failed.
  - A situation where you used external client/organizations plugin/adapter, which was faulty.
- **Motivation**: Just like innovation, it is a positive category of questions that in a way is coupled with innovation. While innovation is mostly about your tangible impact on the project, motivation is more about your psychological impact on the team. You can motivate others around you by being a leader or even an intern. Having a sense of empathy, collaboration, taking initiative, helping fellow employees, and leading by example are some of the qualities you should try to portray while answering such questions. In this category, apart from projects, you can also talk about organising

team-bonding activities to improve the cohesion between team members.
Related questions:

  - A time when you took an initiative.
  - A time when you lead a project.
  - A time when you boosted the morale of the team.

- **Conflict**: Conflicts are a common phenomenon in any group project. There might be conflicts about work division, who committed a mistake, conflicts about technology choice, and designs. It is important to resolve the conflicts as soon as possible in the most practical way taking into account the well being of the project and eventually the customers. We need to understand that it's not a personal battle between engineers about who is right but what's best for the company. Many times, there is no correct answer, but an answer that suits a particular situation in the best possible way. A balance between the company's resources and customer satisfaction needs to be established.
  Related questions:
  - A time when you handled a crisis.
  - A time when you resolved a group conflict.

The approach:

First of all, it seems intimidating to come up with situations/ stories/ projects for so many different questions I have mentioned above. What you need to understand is that you only need to be thorough with 3-4 projects you have worked upon in your company/ college. Most of the categories can be addressed by talking about a single example. For instance, describing the same project, you can talk about innovation, raising the bar, or say missing the deadline. It is always better to talk about different projects answering different questions as most companies will not ask more than three questions. Other questions would be the follow-ups. You just need to decide which project fits the best in the above categories. For all the questions, you need to be well-versed with the situations/ projects you would be talking about, and everything else would be easy following the category guidelines I have mentioned above and the structure and tips I would be writing below. Try to incorporate the company's ideologies, principles, and values in your answers. It is a bonus to know how the company's goals are aligned with yours.

Coming to the structure, all of us might have heard about the STAR method of answering leadership questions promoted by Amazon. I believe that, with minor tweaks, it is a very

structured way of answering leadership questions irrespective of the company.
The STAR methodology:

- **Situation**: Like first impressions, the beginnings are important. A good beginning acts as a hook and decide how the rest of your answer unfolds. A good beginning immediately grabs the interviewer's attention which increases his involvement, which in turn boosts your confidence while answering. I should mention that students are not very good at describing their projects. You need to understand that there are a plethora of software technologies and domains. Hence, it is hard to be well-versed with everything even for the interviewers. Since you have worked extensively in a particular domain, it does not mean the interviewer has the required expertise in it as well. Students rush into the technicalities of the project instead of discussing the managerial aspects of it. In this part, we should address why the project started, why the project was important, what was your role in the project, and what was the impact of the project. After that, you start setting the pretext. Depending on the question, start talking about say the scope of innovation, a situation where a morale boost of the team was required, a situation where you needed to abide by the deadline or a key decision was to be made.
- **Task**: The tasks are just like the JIRA tasks. In this part, you need to talk about what was needed to be done and not how. Focus on your tasks and not that of the entire team. Focus on the goals needed to be accomplished in order to address what was asked in the question.
- **Action**: In this section, you talk about the "how". How the task was completed and what actions you took. In this section, you talk about the technical details of your tasks. You also talk about how your technical and personal actions impacted the team and the project. Again focus on your actions and not that of the team's.
- **Result**: In this section, you talk about the quantitative and qualitative results of your actions. How your actions given a particular situation impacted the project. Talk about the accomplishments, the metrics and the consequences of your actions.
- **Conclusion**: To me, conclusions are the second most important part after the beginning. This is where you want to leave a lasting positive impact on the interviewer. You essentially wrap up your answer and try to show your positive outlook towards the entire situation as a whole. Both positive and negative situations pose a scope for learning and you should discuss it with the interviewer. While positive situations reinforce your faith in the right process, facing negative situations with the right mindset helps you grow as an engineer and as a person.

Important tips:

- Properly structure your answers. Like a good essay, your answers should have a beginning, a middle, and an end. The STAR format can be thought of as the SBI approach. You first talk about the **Situation**, followed by your **Behaviour** towards it, concluded by the **Impact**.
- Focus on the details. Be as descriptive as possible. Include who was involved, the specifics of the project, and key metrics.
- Focus on "I". You want to talk more about your ideas, approach, solutions, skills than that of the team as a whole.
- Don't shy away from failures. Take ownership of the mistakes committed and focus on the learnings.
- If the interviewer is asking follow-ups, listen carefully as to what specifics they want to know.
- Be clear about what you expect from that particular company, and how it will help you achieve your goals. Identify the values and results that resonate with you.
- **1. Can you tell me about a project/software/feature that you worked on turned out to be a failure. What was that project/software/feature and why did it fail?**
- **2. Give me a specific example of a time when you sold your supervisor or team on an idea or concept. How did you convince them? What was the result?**

Interview Preparation for Beginners - [DS | Algorithms | OS | System Design]

Hey Guys,

This is gonna be a short intro post about how to prepare well for interviews and start with your coding journey. I have 3 years of experience in working closely with various products and during this pandemic I lost my job. I started leetcoding since a month or two and have already landed my dream job at a very good gaming company with a decent compensation. it would not have been possible without the leetcode community for which I will always be grateful for and this is my way to give something back. I hope it helps to those who are just starting out and don't know where to start. I have prepared a plan which I have been following regorously and It's been working out quite good for me.

This curriculum covers most of the concepts which are asked in the interviews. So there are main 5 big components that one should always be familier with. I have attached various links to problems which you can solve after learning the concepts.

1. Data Structures
2. Algorithms
3. Concept problems and Maths
4. Networks and OS concepts
5. System Design

Click Here - 12 weeks

https://docs.google.com/document/d/1wUCqhVHydWiDk6FJdFLSMpgigNrGcs4OFZg0Wa7JGEw/edit#heading=h.qg7xbkkowf3z

When you are done with one chapter, you can mark it as done so you can keep track of things. Just right click the bullet point and select the check mark.

I am still in the process of adding things as in when I am able to. If you wish to edit the document, please comment it and I will be happy to add resources there. I wish you all the best for your hardwork. These are unprecedented times and having no job literallly sucks.

Just remember, you are not alone in this journey. You will get the job that you have always wanted, we all just need a little push. Another thing I wanna add in here is about interviews going bad because of several reasons. Please do not get discouraged by all these experiences, I myself have gone through a lot of rejections early on when I started prepering, look at it as a learning curve and move on from that. Interviews do not justify your talent or hardwork whatsoever, So be gentle with yourself and keep preparing. Stay Safe!

A guide for dummies (like me)

A little bit about myself.

I am an electronics undergraduate from New Delhi, and I started programming at the end of my sophomore year, as electronics has a very limited career scope in my country, I decided to switch my field to CS. I am mostly self-taught.
When I began my journey on April 19, my knowledge was limited to writing Hello world program in C language and being in electronics, my coursework was hefty enough itself.
Now almost about 12 months later, I am currently working at a large telecom company as a software intern. There is nothing impressive in being an intern in a tech company, but every year many people with and without CS background start on their own to establish a career in tech.
Now, there is an ocean of information available on the web about how to crack the coding interview,

but I got lost somewhere in it. I had to do it all myself, find the relevant material, practice the questions, study new concepts along with managing my coursework of electronics. I wished someone can tell me exactly what to study and where.

Hence, I have designed this guide using mine and my friend's experiences, who also happen to have a non-CS background.
I am no expert at this topic, but everyone can still benefit a lot from this guide.
It is my give back to the awesome community I have found here at LeetCode.

I have summarized the guide in tasks, and you can complete them in the time limit prescribed. Time limit is actually how long I took to complete these tasks, you may require less time as I was not studying it full time.

Task 1: Weapon of Choice (approx. 20 days)

To start with, you have to learn a programming language.
Think of it as your weapon of choice, with which you would slay the demons of interview questions.
There are many available choices in the market, most popular here on LeetCode are
- **C++/Java** closely followed by **Python** and then other languages follow.
Most people of non-CS background start with **Python**, as it is the most popular option.
But I think it is a poor choice for your first language, as you can get away without knowing many important concepts like pointers, memory allocation, pass by value etc.

Hence my advice is to start with C++, as it is fast, it will be much simple to debug and you can easily learn almost any other language without much effort after C++.

I learnt C++ by this simple book that I found in my library - **Let us C++**
No need to buy it, you can use any **good** book you can find. If you prefer, you can also use these free online resources to learn C++.

1. **Learn C++ from Scratch course**
2. **Codecademy C++ section**

But, just learning a programming language, although important, is not enough to get a job as a software developer, most companies have an interview process that tests a candidate on Data Structures and Algorithms along with fundamentals such as DBMS, OS and System Design. Along with that, you must also build your Resume, which is a document created by a person to present their background, skills, and accomplishments.

I have limited the scope of this article to Data structures and Algorithms which comprise around 80% of an interview for junior roles and a separate post would be covering later topics.

Task 2: Mastering the basics (approx. 40 days)

After you have selected your weapon of choice, and have learned how to use it, for the next forty days, we are going to learn the basics of solving questions.
Data structures and algorithms, which are vital for solving the programming questions, are bread and butter of any interview preparation.

Data structures you are going to need are -

- Arrays
- Stacks
- Queues
- Linked Lists
- Hash Tables
- Trees and BST
- Graphs
- Disjoint sets

An algorithmic paradigm is a framework which underlies the design of a class of algorithms. (Don't worry if you did not understand the above statement, just note that you would be using these)

- Backtracking
- Branch and bound
- Brute-force search
- Divide and conquer
- Dynamic programming
- Greedy algorithm
- Prune and search

Imp. note - There are many more DS and Algo out there other than these like AVL trees, Rolling Hash etc, and you may find some of them in your interview also, but be assured chances of that happening is minuscule, and reward/effort ratio is very low, hence I advise you to skip them.

Although many resources can be used to learn the basic DSA, like CLRS, LeetCode Explore section etc. But if you ask me, to save your time and effort, just head over to Abdul Bari courses, and thank me later.
He is a person I can vouch for and his style of explaining is intuitive and lucid. Watch a few videos of him explaining algorithms on YouTube, to understand what I mean.
He has a paid Udemy course for data structures and a free YouTube playlist for algorithms. This course is the only paid thing I am going to recommend in this guide, just because it is so good.
You can check the algorithm section first by going to YouTube and typing Abdul Bari algorithms and find the playlist. (Sorry, could not add the link, as there was some problem with the alignment of video.)

After watching any video, implement the code in your local IDE or if you have not set it up yet, do not worry, you can implement code here as well - **LeetCode Playground**.

After watching 10-12 videos, you will realize the importance of data structures.
You can get the Udemy course of Abdul Bari here - **Udemy Abdul Bari**

If you are not able to buy, you can refer to any or all of these -

1. Programiz DSA section [Best alternative in this list].
2. LeetCode Explore [May not cover all topics]
3. Coursera DSA course [Audit for free, and financial aid available].
4. CLRS book [Remember, you aim is to crack the coding interview, not do a PhD, only study relevant topics, if any at all].
5. Data structures through C++ book [Not have read it personally, but recommended by a friend].

Now after you have completed all the important data structures and successfully implemented them on the playground or local IDE, go back and complete remaining YouTube videos of algorithms.

You would have known by now, that implementing data structures and operations like sorting is very time consuming. Fortunately for us, every language (except C maybe) has built-in features or libraries that you should learn to quickly do mundane tasks like sorting or defining stack and queue.
In C++, we have STL for these, which you can learn from here or here.

Task 3: Starting your practice (approx. 60 days)

Going back to our weapon analogy, you have successfully learned to use your weapon and also learnt the basic skills of a fight, but now is the time to get your hands dirty on some real opponents.
Before we start, I want you to understand that first few days are going to be full of struggle, it was surely for me, I sat for several hours on easy questions, scratching my hairs and wondering, why am I so dumb.
But it too shall pass, believe me, if you can get through the first few days, you have won half the battle.

Now, I want you to start by following this or any other roadmap for yourself. Solving questions haphazardly is not a good way to improve when you are starting.
I think LeetCode has all that you would require for practice, and you do not need to go anywhere else, but still, you can solve a few questions on HackerRank and HackerEarth to get accustomed to those platforms, as most companies use them for online coding round (at least in India).

On LeetCode you can select the tag of your choice and sort the questions by their acceptance rate as it reflects a much better grading of questions than just Easy, Medium or Hard. After that solve at least 15-20 questions in every section.
You should give every question a good try, which means uninterrupted attention for 45 minutes if you can find a probable solution, try to dry run it on a few test cases. Only after that touch your keyboard.
Make this a habit, believe me, you would be tempted to start typing, just so to start, but resist that urge and only type, when you fully understand your idea.
Also, always sit with a pen and paper ready, to enforce this habit.
If you cannot figure out a question, even after an hour of thinking, see the idea used by top rated posts on discussion, and then implement it yourself.

Road map to solving questions of different topics, can be done in any other order, but follow this one if you have none other.

```
Recursion -> Linked list -> Stack -> Queue -> Two pointers -> Sliding-window ->
hashing -> sorting -> binary search -> trees -> BST -> Heaps -> Graph basics -> BFS -
> DFS -> backtracking -> greedy -> Dynamic programming -> advanced graph -> Union
find -> bit manipulation.
```

Most people (including me) find DP a difficult topic, as some questions are not very intuitive. The only solution to it is practice. I have compiled a list of awesome resources at the bottom, so do not forget to check it out.

Task 4: Getting better (approx. 100 days)

If you have reached this level, you probably would not need my guidance any further.
Still, here are my 2 cents - Start solving without knowing the topic beforehand. Solve random questions and also take part in LeetCode contests. You can also start giving mock interviews on LeetCode.
If you still feel insecure about your DP skills, do not worry, just keep grinding, every day you will get better.

Some useful bookmarks that I have, check them out after you are done with above tasks.

- Best article ever written on DP (use it when revising DP)
- An article on RSQ, read if you have time
- A monster compilation of all common Interview questions
- Syllabus for competitive programming (not relevant for interview preparation)
- Fast IO for C++
- Big O cheat sheet
- For brushing up C++ before coding interview

Some interesting stuff

- Meet the top coders of the world here, here and here
- When I am sad, I watch this.

That's all for now, if you find any mistakes or broken links, please write in comments.
If you have read this far, please upvote.
Thanks!

If you have any interesting bookmarks that you would like to share, please write in comment, I will add it.

Here is a list of awesome resources I found in the comment section -

- Coding university, brought to you by at794
- DSA at codechef, all thanks to naveenverma
- DS playlist by WilliamFiset, originally posted by shubhamrajvanshi

Tips for answering few tricky behavioural interview questions

I am trying to compile a list of tricky behavioural questions and sensible answers for the same.
Feel free to add your inputs in Comments section.

Please upvote if these tips help.
I will keep adding more points as and when I come across the same.

Happy interviewing !!

These questions are very commonly asked in HR/behaviourial/managerial rounds of companies.

The reality is - in such rounds, it is important to sound **diplomatic** and **politically correct**. **Yes, it might sound like faking at times**, but that is precisely what is needed.

As candidates, we definitely need to appear diplomatic and mature - specially for senior positions (say 5+ years of experience). In these rounds, these soft skills matter way more than your stellar coding rounds or design rounds.

0. The MOST IMPORTANT POINT - Resume DrillDown - Know your **resume** inside out. **Each and every word written in your resume** can be questioned, it can be about your technical skills, it can be about your hobbies or some award won by you. Be very very sure about everything which you have written in your resume.

Even a HR person may ask question about some fancy technical term(s) mentioned in your skill set. You need to be ready to explain technical terms at a high level in 2-3 sentences. And do not under-estimate the HR - I have personally known people working in HR who have a background in Computer Science, and they are equally capable to grill you technically as well.

1. Why do you want to quit your current job ?

Always frame the answer around things like - not learning anything new as product is in a saturated state, and unable to switch teams due to internal policies. NEVER EVER talk about bad boss, politics etc - it will backfire bigtime.
We can also talk about your interest in the technology on which the company (who is interviewing you) is working on.

In case your current company is recently acquired by another bigger company, then you can always talk diplomatically about uncertainity in the environment in your company, several key people leaving after acquisition, product being discontinued or put in maintenance mode etc.

2. Why do you want to join us ?

This is a similar question to first one. Only difference is - for this question - we need to be able to talk about 1-2 core products created by that company, the tech stack, and being able to express our interest in the same. A sample answer maybe - *I came to know about your company because of product X, and I found it a very interesting concept*.
I would love to get an opportunity to learn new things by working on this product. Also, use this opportunity to try to get insights into any new features planned for the product in next version.

Expressing interest in their technology is 50% job done for behavioral interviews.
MOST IMPORTANT PART - you must SOUND genuinely passionate when answering this question. Practise rehearsing your lines in front of a mirror for this question - it may help.

3. What has been the biggest failure of your career till now ?
OR
What is the most critical feedback received from your boss in your entire career ?

Even if there is no such thing in reality (for people with say less than 3 years of experience), do craft a reasonable story.
Talk about a time when you caused a major feature break, or a serious bug missed by you which came through customer etc.
You can add that it caused a lot of chaos and I received very critical feedback from my boss saying - "*I did not expect this mistake from you*".
Then talk about some corrective actions which you took, like enhancing test automation, or getting into TDD mode etc and this helped in minimizing future recurrences of such nature.
YES, talking about corrective actions is the most important part here.

If possible, support with data like - this enhancement in our process helped in preventing at least 20 show stopper bugs from slipping in.

4. What does your typical day at work look like ?

Talk about your routine like - we have daily status meetings as first thing in the morning, then we get onto development/debugging work, customer calls etc. Try to compress in 3-4 sentences at max.
Again, SOUND passionate and energetic when answering this question.

Here the interviewer may ask a follow-up question - **what do you do in your free time at work**?

Possible answer - I prefer to read about other components of the product on which I have not worked, and try to understand how to debug/troubleshoot those pieces of code.
Or else - my company has a huge repository of learning resources - I try to learn new things etc

Basic motive is to showcase being eager to learn.

Another flavor of above question - *describe your current role*.
In this case, give an overview of your current responsibilities, like I am developer/lead on this project, working in a team of X number of people. I am responsible for X, Y and Z parts of the project. Be honest about whether you are a hands-on coder or a hybrid of coding and team management or a pure manager.
In case you are more on the management side, then be prepared to justify how do you keep your technical skills up-to-date.

5. Explain about your current product/project at a high level.

Here, we need to be very cautious. NO - never disclose any confidential details.
Try to explain the product using a high level architectural/block diagram (if interviewer allows) or else prepare a crisp 5-6 sentence description of the same.

6. What would you like to improve at your current workplace ?
OR
What do you dislike/hate at your current job/workplace ?

Again a trap - if you badmouth your current employer or use any words like **hate** or **dislike**, it is guaranteed to go against you.

Talk about generic things like - sometimes code reviews take a long time due to senior developers being busy - probably that can be streamlined.
Or say - we should invest more in enhancing the test automation infrastructure, which often takes a backseat due to various constraints

Basically, try to stay around technical things, and avoid talking about poor cafeteria or no free cab pickup-drop services etc.
The motive is to show your passion towards work related things and not focus on secondary things like cafeteria or cabs or playgrounds etc.

7. Are you happy at your current job ?

This is also a big trap.
If you talk only about goody-goody positive things, then this question will be immediately followed by - **if you like your job, then why are you looking around for another job**?

So, answer it diplomatically around point 1 of this post. Talk about good things like - I have gotten to learn a lot.

If by any chance your current job is your first job - then definitely talk about things like - I learnt the ways of working of industry in this job and hence this job will always be special for me etc etc.

Then talk about negatives - again in polished way - now the product has really evolved and hence there is not much productive work to do other than customer support or maintenance etc. Or else say that I have worked for a really long time on same technology stack, and want to expand my breadth by getting a chance to work on other technologies.

You may also borrow few points listed in **"Why do you want to quit your current job"** above in this post.

8. How would you react if you get to know that your boss is younger than you (or has less years of experience than you) ?

This one is a *make-or-break cultural fit question*. Always answer diplomatically saying that age/years etc are irrelevent for me. What matters for me is what can I learn from someone irrespective of whether that person is junior or senior to me. I can learn a few things from my boss for sure, and if it turns out that I am more experienced or older than my boss, then I am sure I can be a valuable resource for my boss in terms of helping out with critical decision making.
My experience will definitely be an asset for my boss in such cases.

Throw in examples of successful sports team of any sport which you like, there will be several examples where the captain of a successful sports team is younger than a senior important veteran player of the team. If you can give that type of analogy, your answer will sound more genuine/natural.

9. What would you do if you find your senior or boss doing something unethical or violating a company policy ?

Always talk about that you would gently point out to that person directly, and request that person to follow the correct process. In case the violations continue, then I would like to know about the violation reporting policy of your company.

Here, you can turnaround the interview by **cross-questioning** your interviewer - "**by the way, can you give a brief insight into policy violation reporting mechanism which exists in this company ?**"

10. Who has been your ideal/best boss ?

Talk about any boss who was a strong decision maker, always stood by the team, and openly praised contributions by team members, and also clearly guided the team members towards career growth.

Yes, such bosses are very rare in real life - but even if you did not have one such boss in your real life, do talk along these lines.

Nobody is going to conduct an enquiry about whether you really had a boss who had all these qualities or not.
Just be careful - **not to take any person's name** while talking about these qualities. If explicitly asked, just politely say that I want to respect the privacy of that person, hence won't take any names.

You can talk like - "*Yes, I have been very fortunate to work under one/two such boss(es) earlier in my career. I learnt a lot from him/her/them. I always found that person to be always supportive of the team, and that person was the only person who actually told me the difference between a Software Engineer and a Senior Software Engineer in clear measurable terms etc etc*."

11. Describe the worst boss you have ever worked under.

NEVER NEVER EVER start describing that one bad boss you may have encountered. Be very very diplomatic here.
One slip of tongue in answering this one - and you are already rejected as "*being a poor culture fit*".

You can say something like this and create a reasonable story like this - "*Well, nobody is perfect - everyone is good in some things and not so good in other things. Regarding my bosses, I am not able to think of anyone who can be labelled as worst as such. I learnt something or other from each and every boss I worked under.*
If I have to explicitly describe the shortcomings of any particular boss, then I will not exactly call it a shortcoming, but I feel it was his extradordinary passion towards his work. I had a boss who often used to get so carried away in work when working with us in brainstorming/discussion sessions that we would often end up getting very delayed for lunch.
Initially I found it irritating, but with the passage of time, I realised that he was a person of extreme dedication and passion towards his work. The more I worked with him, the more I learnt from him.
And coming to this lunch time issue, I hesitatingly brought up this point once with him privately, and he happily agreed to have working lunch in such cases.

So, in a nutshell - there are no bad bosses - it is just that their way of working/thinking is drastically different from ours."

Prepare 2-3 stories of this type, just in case the interviewer adds a follow-up like - "**any other irritating bosses you worked under**".

So, the whole idea is to **mould a highly negative conversation towards a positive direction**.

12. If you are say 5+ years experienced and mention that you yourself are an interviewer - then you may be asked to **describe your strategy of taking interviews** - **stuff like what type of questions you ask, and what are you trying to judge based on that question**.

In such case, be ready with 2-3 questions and present them as your **pet questions**.
Be ready with justification like - by asking this tricky coding problem, I try to judge upto what extent the candidate can think, can the candidate come up with different approaches etc.
By asking this question Y - I try to get a feel as to whether the candidate will be a decent cultural fit in our company or not.

Be prepared for a **follow-up** like - "**if you have to look at your own interview rounds going on right now in this company, what do you think the interviewers were trying to judge about you.**"
Try to answer on similar lines staying close to job description. You can say - "as this job is for a Senior Software Engineer, so it is expected that the interviewer would like to be fully convinced about my coding abilities. So, it is quite natural that I was asked X, Y, Z. In the current round, I believe you are trying to take a call as to whether people would like to work with me as a teammate, and hence you are trying your best not to er in judging me as a person."

13. What is your greatest weakness ?
No, NO, NO - please do not talk about being impatient or pushing your team hard etc etc. These are all very very well-known answers.

Talk about something more genuine and possibly not related to work - I am not so good at remembering road directions, or maybe people whose first/second language is not English may feel that occasionally I speak too fast.

But be prepared to answer - **what are you doing to overcome this weakness** ?
Possible answers to above examples may be - I am trying to get better at remembering road directions by observing the landmarks more closely when I pass through any road
or

I consciously try to speak at a slower pace and in a neutral accent when interacting with people whose first/second language may not be English. I am actually trying to learn more about different cultures across the globe to be sensitive to their requirements.
If you talk about culture, be prepared to talk about a few interesting facts about culture of few countries - e.g. using left hand to give/take something in Asian countries may be considered offensive, tipping is not considered good in some countries but is a natural expectation in some other countries etc.

Being culturally sensitive is very very important for working in any MNC.

14. Why should we hire you?
OR
Why do you think you are a good fit for this role/job ?

Again a cliche question. Just talk about your passion for the role, talk about how your skills are a perfect match for this role, definitely do highlight any not-so-common achievement of yours (may be you have been a state level player, or maybe you appeared on a TV show - use it to market your people skills, maybe you won multiple scholarships in your career).
And at the end say - I strongly believe these qualities of mine make me a worthy candidate for this role.
Mark the wording - worthy - NOT perfect. Be very cautious as to *NOT to sound arrogant*.

15. [For fresh graduates/postgraduates only] What is your favorite subject and why ?
Be prepared with 2-3 names, because sometimes the interviewer may dig like - **second favorite subject** ?

It is upto you whether you want to stay with routine subjects like Algorithms, Data Structures etc or choose something else like Machine Learning, AI, Computer Graphics, Computer Networks.
To answer the **why** part, NEVER EVER say things like because I always score good grades/marks in this subject. You must be able to give a solid justification about the real world applicability of the concepts taught in those subjects.

Some examples maybe - Data structures is the basic foundation of any software which we use in real life.
Or
Machine learning is the basic concept behind so many useful features like recommended news, purchase suggestions on e-commerce sites etc
Or
Computer Graphics is the very reason we get to see such amazing 3D/4D movies like <name any of your favorite 3D/4D movies>

Also, a word of caution here - maybe you studied a subject named Advanced Algorithms or say Advanced Concepts in Computer Networks - AVOID the word **advanced** if you can. Because, using **the word advanced may open up a pandora box** and you may get bombarded with several questions about concepts which you may not even have heard of !!

16. [For fresh graduates/postgraduates only] Which subject(s) do you HATE and why ?
This one is a very tricky question to answer, and you need a solid justification for the same.

You should NEVER use the word HATE but say something like I am not very much interested in this subject X.

X can be some compulsory elective course which is not exactly related to Computer Science but is a part of your college/university curricullum. You can justify by saying that I have tried very hard to understand where would I need to apply the concept taught in that subject, in the field of software engineering.

By any chance, if your least favorite subject is related to Computer Science, then the only possible way to justify it could be - I have never gotten any chance to **practically apply** the concepts taught in that subject. There are indeed some subjects which are taught only theoretically - if that is the case then you can give this justification.

Or even better - you can also say that there is no subject which I would put in the category of dislike/hate, because every subject has its own importance.

17. [For fresh graduates/postgraduates only] Tell me about your major/degree project.
Be prepared to explain each and everything in thorough detail.
Try to structure your answer like this

Motive - what inspired you to take up this project? Justify with some real world application - max 2-3 sentences
Initial exploration - what all did you read up about it - max 2-3 sentences
Implementation details - preferably a flow/block diagram (if possible), and 5-6 sentence summary explanation
What did you learn from this task - I learnt how to actually implement a Classifier/NeuralNetwork, I learnt a new tool/language named X etc. (2-3 sentences at max)

Q. How do you give constructive feedback to your peers.
A. I have a very open relationship with my peers, and if I feel that the task that they are working on or the code that they have written is not up to the mark, I always discuss the same with them. I usually do this by giving them suggestions like maybe you can try doing this task in this way, or sharing links with them which might help them in understanding how they could have done their current task in a better way.

If they are not performing well, then I like to have a frank discussion with them regarding if they are going through some problems which are hindering their performance? If there's any way in which I can help them, then I try to do the same. Like sometimes someone isn't performing well because they are not enjoying the task at hand, at that time it becomes a pointer for me that maybe those kinds of tasks need to be distributed to other members of the team who might enjoy that type of task more.

I am not sure if my answer is right or now, but that's what I usually practice at work, and shared the same with the interviewer. Let me know what you guys think.

5 flavors of singleton!

There are several flavors:

1. **Eager initialization** :

```
public class Singleton {

        private static Singleton instance = new Singleton(); //eagerness

        private Singleton() {

                //init

        }



        public static Singleton getInstance() {

                return instance;

        }

}
```

2. **Lazy initialization** : (Consider this if init is **resource heavy** and if it is required in **only some application flows**)

```
public class Singleton {
        private static Singleton instance;
        private Singleton() {
                //init
        }

        public static Singleton getInstance() {
                if(instance == null) { //laziness
                        instance = new Singleton();
                }
                return instance;
        }
}
```

3. **Thread safe Lazy initialization** : (In the lazy init construct above, if there are two or more threads accessing the getInstance method, it might lead to the **creation of spurious multiple instances** of the singleton!)

```
public class Singleton {
        private static Singleton instance;
        private Singleton() {
                //init
        }

        public static synchronized Singleton getInstance() { //thread safety
                if(instance == null) {
                        instance = new Singleton();
                }
                return instance;
```

```
    }
}
```

4. **Double checked locking** : (Note that in the above approach, the race condition can be reached only **once** in the entire application lifecycle, ie when **instance is null**! What we are doing is introducing a LOT of overhead wth the *synchronized* keyword. So we need a way to ensure that the locking protection only applies **once**)

```
public class Singleton {
        private volatile static Singleton instance;
        private Singleton() {
                //init
        }

        public static Singleton getInstance() {
                if(instance == null) {
                        synchronized(this) { //only lock the FIRST time
                                if(instance == null) { //The "double" check
                                        instance = new Singleton();
                                }
                        }
                }
                return instance;
        }
}
```

5. **Using an enum!** : (The simplest way to define a singleton! And guess what! Enums are **lazily initialized** by the JVM, ie they are instantiated the first time they are accessed! The **creation of an enum is thread safe** too, the **JVM** ensures that! :D)

```
public enum Singleton {
    INSTANCE;

}
```

35 behavioral questions asked in 95% of Amazon interviews with examples

Team / time management (positive & negative) https://youtu.be/CQG4Ui0oAmk

1. Tell me about a time when you were not able to meet a time commitment. What prevented you from meeting it? What was the outcome and what did you learn from it?
2. Describe a long-term project that you managed. How did you keep everything moving along in a timely manner?
3. Give me an example of a time when you set a goal and were able to meet or achieve it Adaptation https://youtu.be/ys7fLcH5gpg
4. Tell me about a time you had to quickly adjust your work priorities to meet changing demands. Team / decision https://youtu.be/3NExTeMnobU
5. an example when you had to push back to HQ or challenged a decision
6. Tell me about the toughest decision you've had to make in the past six months
7. Tell me about a decision that you regret. Team / leadership https://youtu.be/Tg6BVRTsuic
8. What did you do when you needed to motivate a group of individuals?
9. Tell me about a time you stepped up into a leadership role Team / communication & negotiation https://youtu.be/UJXkaide9bU
10. Do you collaborate well?
11. Describe a situation when you negotiated with others in your organization to reach agreement. Team / coworkers https://youtu.be/ZsxkoZdyEcw
12. We've all had to work with people that don't like us. How do you deal with someone that doesn't like you?
13. We all make mistakes we wish we could take back. Tell me about a time you wish you'd handled a situation differently with a colleague.
14. The last time you had to apologize to someone Team / conflict https://youtu.be/Zz8iQ852YMs
15. Give me an example of a time you faced a conflict while working on a team. How did you handle that?
16. Tell me about a time when you received negative feedback from your manager. How did you respond? Problem solving https://youtu.be/2XxCUain1IU
17. Tell me about a time when you missed an obvious solution to a problem
18. A time when you faced a problem that had multiple possible solutions
19. Tell me about a time when you came up with a new approach to a problem.

20. Describe a time when you anticipated potential problems and developed preventive measures.
21. Describe a situation in which you found a creative way to overcome an obstacle. Strategy / data https://youtu.be/oRGKHTiM29E
22. How have you leveraged data to develop a strategy?
23. a time when you were 75% through a project, & you had to pivot strategy
24. Tell me about a time when you had to choose between technologies for a project
25. Tell me about a time you had to deal with ambiguity Innovation https://youtu.be/774ovkE2y5I
26. What's the most innovative new idea that you have implemented? Ownership principle https://youtu.be/Rn3EjvukTkI
27. Describe a time when you sacrificed short term goals for long term success
28. Provide an example of when you personally demonstrate ownership. Strength / weakness https://youtu.be/q8fK73QjmbA
29. What's your greatest strength
30. Biggest weakness Clients https://youtu.be/OFEUzyB12rE
31. We all deal with difficult customers from time to time. Tell me about a challenging client-facing situation and how you handled it.
32. How do you show customer obsession? Failure https://youtu.be/eaUUeFoB9CQ
33. Tell me about a time you recovered from a difficult situation
34. Tell me about a time you failed and what you learned from it
35. Why Amazon https://youtu.be/H_KGM0i9jkA

Based on what we discussed, how do I stack against other candidates? Any concerns about my ability to do the job? What do I need to do to get the job? What makes a great candidate for this role?

Self Practice: Behavioral Questions

List of typical behavioral interview questions. Taken from Udacity Data Science Interview Preparation.

1. Getting to Know You

- What motivates you at work?
- Describe what your preferred supervisor—employee relationship looks like.
- What two or three things are most important to you in your work?

2. Knowledge & Interests

- What do you think are the most pressing issues in this field?
- What challenges does this position present for you?

- What do you think it takes to be successful in this organization?
- What do you know about our company?

3. Readiness & Experience

- What is your greatest strength/weakness?
- Tell me about a problem you have encountered and how you dealt with it?
- Tell me about a mistake you made and what you learned from it.
- What experience do you have in this field? How have you prepared yourself to switch fields?

4. Goals, Motivation & Values

- Why do you think you will like this field?
- Describe a time when you saw some problem and took the initiative to correct it rather than waiting for someone else to do it.
- Give me an example of a time you were able to be creative with your work. What was exciting or difficult about it?
- Tell me about a time you were dissatisfied in your work. What could have been done to make it better?

5. Teamwork

- Describe a time when you worked closely with someone who had a very different personality than you.
- Tell me about a time you faced a conflict while working on a team. How did you handle the conflict?
- Describe a time when you struggled to build a relationship with someone important.
- Tell me about a time you needed to get information from someone who wasn't very responsive. What did you do?

6. Ability to Adapt

- Tell me about a time you were under a lot of pressure. What was the situation and how did you get through it?
- Describe a time when your team or company was undergoing change. How did it impact you, and how did you adapt?
- Tell me about your very first job. What did you do to learn the ropes?
- Tell me about a time you failed. How did you deal with this situation?

7. Time Management Skills

- Tell me about a long-term project that you managed. How did you keep organized and make sure everything was moving along as planned?
- Tell me about a time you set a goal for yourself. How did you ensure that you would meet your objective?
- Give me an example of a time you managed multiple responsibilities. How did you handle it?

8. Communication Skills

- Tell me about a time you successfully persuaded someone to understand your perspective at work.
- Describe a time when you were the primary “expert”. How did you ensure that everyone understood you?
- Describe a time when you could only use written communication to get your ideas across to your team.

Specifics

```
  Why you want to join the company
  Why are you looking for a change
  What pisses you off
  What's your greatest strength?
  When have you worked the hardest?
  why you're the right person to hire?
  Tell me about a two improvements you have made in the last six months
  I'm interested in how you recharge when you're not working. What do you do
with your downtime?
  Major success, how you got it/experience.
  Tell me about a major mistake you made, and what you did to
correct/deal/learn it
  Describe what you do to control mistakes in your work
  Tell me about a time you heard difficult feedback and how you handled it
```

Customer Obsession

```
  How do you show your customer Obsession?
  How do you wow your customers?
  How do you develop client relationships?
  How do you understand your customer's needs?
  Describe your style in dealing with irate customers  Tell me about a time
when you made sure a customer was pleased with your service
  Tell me about the last time a customer or coworker got upset with you
  When you're working with a large number of customers, it's tricky to
deliver excellent service to them all. How do you go about prioritizing your
customers' needs?  We all deal with difficult customers from time to time.
Tell me about a challenging client-facing situation and how you handled it.
```

Tell me about the longest time it took you to conclude a deal with a customer
Tell me about a time you made a mistake that affected a client adversely and how you coped with it
Describe a time when it was especially important to make a good impression on a client. How did you go about doing so?
Give me an example of a time when you did not meet a client's expectation. What happened, and how did you attempt to rectify the situation?

Ownership

A time when you were 75% through a project, and you had to pivot strategy
Provide an example of when you had personally demonstrated ownership
When you took on something outside your area of responsibility. Outcome?
Describe a problem you solved. What was the root cause of the problem?
Tell me about a time when you had to juggle multiple important projects
Describe a time when you saw some problem and took the initiative to correct it rather than waiting for someone else to do it

Leadership

Describe a time when you led by example
When have you delegated efficiently?
Tell me about a time you stepped up into a leadership role
Tell me about a time when you demonstrated leadership skills
Give me an example of when you showed initiative and took the lead
Tell me about a time when you took the lead on a difficult project
Tell me about a time when you had to manage multiple responsibilities. How did you handle it?

Hire and Develop the Best

Share an example of how you were able to motivate employees or co-workers
What did you do when you needed to motivate a group of individuals?

If you were the resident technical expert. What did you do to make sure everyone was able to understand you?
Tell me about a successful presentation you gave and why you think it was a hit

Who have you coached or mentored to achieve success?
Tell me about a time when you had to coach an employee

An example of when you saw a peer struggling and decided to step in and help

Are you easy to get along with?
Do you collaborate well?
Tell me about a time where you had to delegate tasks during a project
Describe a time when you had to bring two departments together to work more efficiently with each other
Describe a situation when you were able to use persuasion to successfully convince someone to see things your way

We've all had to work with people that don't like us. How do you deal with someone that doesn't like you?

Team skills

The last time you had to apologize to someone

Tell me about a time when you worked with a difficult team member
How have you reacted to a colleague who regularly lets the team down?
A time when a team member didn't meet your expectations on a project
Describe a time when you found it difficult to work with someone from a different background
Tell me about a time you worked on a team with individuals from different cultural backgrounds

Tell me about a team project you worked on
Tell me about a time when you had to rely on written communication to get your ideas across to your team.
Describe a time when your team or company was undergoing some change. How did that impact you, and how did you adapt?
What is the most significant impact of your work inside a team?
Describe a time that you demonstrated the ability to be an effective team member
Share a rewarding team experience(neer praising)
Give me an example of a team project that failed(oyo project)
Give me an example of a time you faced a conflict while working on a team. How did you handle that?
Describe a time when you had to convince the leaders of your team despite their disagreement. What did you do to convince them
Describe a situation when you negotiated with others in your organization to reach agreement
Let's say you need something important from a coworker and that person isn't responding. How would you deal with this?
Tell me about a time when you were asked to take sides regarding another employee and you remained neutral
Describe a time when you struggled to build a relationship with someone important. How did you eventually overcome that?(very senior -> frozen -> eat lunch-> everybody doesn't have time- doesn't mean rude)
Tell me about a time you needed to get information from someone who wasn't very responsive. What did you do?(ping, ping, then include someone senior in the loop)
Tell me about a time you had to raise an uncomfortable issue with your boss
Give an example of a situation where you had to take into account the sensitiveness of different parties
Tell me about a time you observed culturally insensitive behavior on the job
What experiences have you had with recruiting, hiring, training, and/or supervising a diverse workforce?

Invent and Simplify

How have you leveraged data to develop a strategy? (Onboarding hiring, UX-jam splitting)

Tell me about a time when you helped improve an internal work process. How you did it and what was the result?
What is the best invention or idea you had in the past two years?(CI-CD pipeline and slack)

Are Right, A Lot

Give me an example where you strongly held an opinion and you were the outlier
Tell me about a time you led an important meeting
Describe a difficult decision you had to make in your business life and how you went about doing it
Describe a decision you made that wasn't popular and how you handled implementing it
What is the most difficult decision you ever took in software?
Tell me about a time you made a decision without consulting your manager
Tell me about the toughest decision you've had to make in the past six months
Tell me about a time you had to make decision under pressure to meet a deadline

Learn and Be Curious

Tell me about a time when you worked under close supervision or extremely loose supervision. How did you handle that?
Tell me about a time you had to deal with ambiguity
Give an example of a situation in which you took specific steps to further your career
Tell me about a time when you were curious; what were your expectations and what did you learn from them?

Insist on the Highest Standards

Talk about a time when you had to work closely with someone whose personality was very different from yours
Has your manager/supervisor ever asked you to do something that you were uncomfortable with? How did you handle this?
Tell me about a time you witnessed a fellow employee do something you didn't think was appropriate

Think Big

Describe a time when you made a judgement call with limited information
Describe a time when you sacrificed short term goals for long term success
Tell me about a time when you had to choose between technologies for a project
Tell me about a time when you had to analyze information and make a recommendation
Tell me about a time you were dissatisfied in your work. What could have been done to make it better?

Bias for Action & Speed

Give me an example of a time you were able to be creative with your work. What was exciting or difficult about it?
Tell me about a time you went above and beyond
Describe a project or idea (not necessarily your own) that was implemented primarily because of your efforts
Tell me about a time you had to be very strategic in order to meet all your top priorities
Describe a long-term project that you managed. How did you keep everything along in a timely manner?
Tell me about a time you set a goal for yourself. How did you go about ensuring that you would meet your objective?
Tell me about the last time your workday ended before you were able to get everything done
Tell me about a time when you came up with a new approach to a problems
What's the most innovative new idea that you have implemented?
Describe a situation in which you found a creative way to overcome an obstacle
Describe a project that required input from people at different levels in the organization
Tell me about a time you knew you were right, but still had to follow directions or guidelines

Tell me about a time you had to work at a fast pace for an extended period of time. How did you maintain your work pace?
Describe a situation when you had to overcome a number of obstacles to achieve an objective
Describe a situation in which you had to schedule your activities to meet a work objective

Tell me about a time you had to quickly adjust your work priorities to meet changing demands
Tell me about a time when you overcame an obstacle and delivered results
Tell me about a time you exceeded your expectations
How would you handle a project that is expected to be behind schedule

Tell me about a time when you were not able to meet a time commitment. What prevented you from meeting it? What was the outcome and what did you learn from it?
Describe an example of when you took risk and failed
Describe a time when you were faced with a stressful situation and used your coping skills

Frugality

Describe a time when you improved a process with limited budget
Describe a time when you went beyond your job description to save your company time and money

Earn Trust

Tell me about your proudest professional accomplishment

Describe a new idea or suggestion that you made to your supervisor recently
Describe a project that you are particularly proud of. How did it impact your company? What challenges did you encounter and how did you solve them?

Dive Deep

Tell me about a time you recovered from a difficult situation
Describe a time when you anticipated potential problems and developed preventive measures
Let's say you're working on a major project and you're in the weeds. How do you find your way out?
We have an situation that may require everyone to work extra hours. How would you handle that?
Talk about a time when you were new on the job and had a lot to learn. How did you manage?

Have Backbone; Disagree and Commit/Conflicts

Tell me about a goal you failed to achieve
Tell me about a decision that you regret
What's your greatest professional regret?
A time when you faced a problem that had multiple solutions
When did you take a risk, make a mistake or fail? How did you respond?
Tell me about a time you failed and what you learned from it
Describe a situation where you disagreed with a supervisor?
Tell me about a time when you disagreed with your hiring manager. What did you do?
Tell me about a time when you disagreed with a rule or approach
Tell me about a time when you missed an obvious solution to a problem
An example of when you had to push back to HQ or challenged a decision
Tell me about a time when you disagreed with your boss
Tell me about a time you had a conflict at work
Describe a time when you received criticism and how you handled it
Tell me about a time when you received negative feedback from your manager. How did you respond?

Deliver Results

Tell me about a time you recovered from a difficult situation
Tell me about a situation where you had to solve a difficult problem
Do you feel you work well under pressure? If so, describe a time when you have done so...
Explain a situation where you faced a problem and how did you deal with it?
Tell me about a time your responsibilities got a bit overwhelming. What did you do? What things you left/depriortise
Tell me about a time you handled a crisis
Give me an example of a time when you set a goal and were able to meet or achieve it
Tell me about a time when you were under a lot of pressure. What was going on, and how did you get through it?

https://leetcode.com/discuss/general-discussion/786126/Python-Powerful-Ultimate-Binary-Search-Template.-Solved-many-problems

Powerful Ultimate Binary Search Template. Solved many problems

>> Intro

Binary Search is quite easy to understand conceptually. Basically, it splits the search space into two halves and only keep the half that probably has the search target and throw away the other half that would not possibly have the answer. In this manner, we reduce the search space to half the size at every step, until we find the target. Binary Search helps us reduce the search time from linear O(n) to logarithmic O(log n). **But when it comes to implementation, it's rather difficult to write a bug-free code in just a few minutes**. Some of the most common problems include:

- When to exit the loop? Should we use left < right or left <= right as the while loop condition?
- How to initialize the boundary variable left and right?
- How to update the boundary? How to choose the appropriate combination from left = mid, left = mid + 1 and right = mid, right = mid - 1?

A rather common misunderstanding of binary search is that people often think this technique could only be used in simple scenario like "Given a sorted array, find a specific value in it". As a matter of fact, it can be applied to much more complicated situations.

After a lot of practice in LeetCode, I've made a powerful binary search template and solved many Hard problems by just slightly twisting this template. I'll share the template with you guys in this post. **I don't want to just show off the code and leave. Most importantly, I want to share the logical thinking: how to apply this general template to all sorts of problems**. Hopefully, after reading this post, people wouldn't be pissed off any more when LeetCoding, "This problem could be solved with binary search! Why didn't I think of that before!"

>> Most Generalized Binary Search

Suppose we have a **search space**. It could be an array, a range, etc. Usually it's sorted in ascending order. For most tasks, we can transform the requirement into the following generalized form:

Minimize k , s.t. condition(k) is True

The following code is the most generalized binary search template:

```
def binary_search(array) -> int:
    def condition(value) -> bool:
        pass


    left, right = min(search_space), max(search_space) # could be [0, n], [1, n] etc. Depends on problem
    while left < right:
        mid = left + (right - left) // 2
        if condition(mid):
            right = mid
        else:
            left = mid + 1
    return left
```

What's really nice of this template is that, for most of the binary search problems, **we only need to modify three parts after copy-pasting this template, and never need to worry about corner cases and bugs in code any more**:

- Correctly initialize the boundary variables left and right to specify search space. Only one rule: set up the boundary to **include all possible elements**;
- Decide return value. Is it return left or return left - 1? Remember this: **after exiting the while loop, left is the minimal k satisfying the condition function**;
- Design the condition function. This is the most difficult and most beautiful part. Needs lots of practice.

Below I'll show you guys how to apply this powerful template to many LeetCode problems.

>> Basic Application

278. First Bad Version [Easy]

You are a product manager and currently leading a team to develop a new product. Since each version is developed based on the previous version, all the versions after a bad version are also bad. Suppose you have n versions [1, 2, ..., n] and you want to find out the first bad one, which causes all the following ones to be bad. You are given an API bool isBadVersion(version) which will return whether version is bad.

Example:

Given n = 5, and version = 4 is the first bad version.

call isBadVersion(3) -> false

call isBadVersion(5) -> true

call isBadVersion(4) -> true

Then 4 is the first bad version.

First, we initialize left = 1 and right = n to include all possible values. Then we notice that we don't even need to design the condition function. It's already given by the isBadVersion API. Finding the first bad version is equivalent to finding the minimal k satisfying isBadVersion(k) is True. Our template can fit in very nicely:

```python
class Solution:
  def firstBadVersion(self, n) -> int:
    left, right = 1, n
    while left < right:
      mid = left + (right - left) // 2
      if isBadVersion(mid):
        right = mid
      else:
        left = mid + 1
    return left
```

69. Sqrt(x) [Easy]

Implement int sqrt(int x). Compute and return the square root of *x*, where *x* is guaranteed to be a non-negative integer. Since the return type is an integer, the decimal digits are truncated and only the integer part of the result is returned.

Example:

Input: 4

Output: 2

Input: 8

Output: 2

Easy one. First we need to search for minimal k satisfying condition k^2 > x, then k - 1 is the answer to the question. We can easily come up with the solution. Notice that I set right = x + 1 instead of right = x to deal with special input cases like x = 0 and x = 1.

```
def mySqrt(x: int) -> int:
  left, right = 0, x + 1
  while left < right:
    mid = left + (right - left) // 2
    if mid * mid > x:
      right = mid
    else:
      left = mid + 1
  return left - 1  # `left` is the minimum k value, `k - 1` is the answer
```

35. Search Insert Position [Easy]

Given a sorted array and a target value, return the index if the target is found. If not, return the index where it would be if it were inserted in order. You may assume no duplicates in the array.

Example:

Input: [1,3,5,6], 5

Output: 2

Input: [1,3,5,6], 2

Output: 1

Very classic application of binary search. We are looking for the minimal k value satisfying nums[k] >= target, and we can just copy-paste our template. Notice that our solution is correct regardless of whether the input array nums has duplicates. Also notice that the input target might be larger than all elements in nums and therefore needs to placed at the end of the array. That's why we should initialize right = len(nums) instead of right = len(nums) - 1.

```
class Solution:
  def searchInsert(self, nums: List[int], target: int) -> int:
    left, right = 0, len(nums)
    while left < right:
      mid = left + (right - left) // 2
      if nums[mid] >= target:
        right = mid
      else:
        left = mid + 1
    return left
```

>> Advanced Application

The above problems are quite easy to solve, because they already give us the array to be searched. We'd know that we should use binary search to solve them at first glance. However, **more often are the situations where the search space and search target are not so readily available**. Sometimes we won't even realize that the problem should be solved with binary search -- we might just turn to dynamic programming or DFS and get stuck for a very long time.

As for the question "When can we use binary search?", my answer is that, **If we can discover some kind of monotonicity, for example, if condition(k) is True then condition(k + 1) is True, then we can consider binary search**.

1011. Capacity To Ship Packages Within D Days [Medium]

A conveyor belt has packages that must be shipped from one port to another within D days. The i-th package on the conveyor belt has a weight of weights[i]. Each day, we load the ship with packages on the conveyor belt (in the order given by weights). We may not load more weight than the maximum weight capacity of the ship.

Return the least weight capacity of the ship that will result in all the packages on the conveyor belt being shipped within D days.

Example :

Input: weights = [1,2,3,4,5,6,7,8,9,10], D = 5

Output: 15

Explanation:

A ship capacity of 15 is the minimum to ship all the packages in 5 days like this:

1st day: 1, 2, 3, 4, 5

2nd day: 6, 7

3rd day: 8

4th day: 9

5th day: 10

Note that the cargo must be shipped in the order given, so using a ship of capacity 14 and splitting the packages into parts like (2, 3, 4, 5), (1, 6, 7), (8), (9), (10) is not allowed.

Binary search probably would not come to our mind when we first meet this problem. We might automatically treat weights as search space and then realize we've entered a dead end after wasting lots of time. In fact, we are looking for the minimal one among all feasible capacities. We dig out the monotonicity of this problem: if we can successfully ship all packages within D days with capacity m, then we can definitely ship them all with any capacity larger than m. Now we can design a condition function, let's call it feasible, given an input capacity, it returns whether it's possible to ship all packages within D days. This can run in a greedy way: if there's still room for the current package, we put this package onto the conveyor belt, otherwise we wait for the next day to place this package. If the total days needed exceeds D, we return False, otherwise we return True.

Next, we need to initialize our boundary correctly. Obviously capacity should be at least max(weights), otherwise the conveyor belt couldn't ship the heaviest package. On the other hand, capacity need not be more thansum(weights), because then we can ship all packages in just one day.

Now we've got all we need to apply our binary search template:

```
def shipWithinDays(weights: List[int], D: int) -> int:
    def feasible(capacity) -> bool:
        days = 1
        total = 0
        for weight in weights:
            total += weight
            if total > capacity:  # too heavy, wait for the next day
                total = weight
                days += 1
                if days > D:  # cannot ship within D days
                    return False
        return True

    left, right = max(weights), sum(weights)
    while left < right:
        mid = left + (right - left) // 2
        if feasible(mid):
            right = mid
        else:
            left = mid + 1
    return left
```

410. Split Array Largest Sum [Hard]

Given an array which consists of non-negative integers and an integer *m*, you can split the array into *m* non-empty continuous subarrays. Write an algorithm to minimize the largest sum among these *m* subarrays.

Example:

Input:

nums = [7,2,5,10,8]

m = 2

Output:

18

Explanation:

There are four ways to split nums into two subarrays. The best way is to split it into [7,2,5] and [10,8], where the largest sum among the two subarrays is only 18.

If you take a close look, you would probably see how similar this problem is with LC 1011 above. Similarly, we can design a feasible function: given an input threshold, then decide if we can split the array into several subarrays such that every subarray-sum is less than or equal to threshold. In this way, we discover the monotonicity of the problem:
if feasible(m) is True, then all inputs larger than m can satisfy feasible function. You can see that the solution code is exactly the same as LC 1011.

```
def splitArray(nums: List[int], m: int) -> int:
  def feasible(threshold) -> bool:
    count = 1
    total = 0
    for num in nums:
      total += num
      if total > threshold:
        total = num
        count += 1
```

```
            if count > m:
                return False
        return True

    left, right = max(nums), sum(nums)
    while left < right:
        mid = left + (right - left) // 2
        if feasible(mid):
            right = mid
        else:
            left = mid + 1
    return left
```

But we probably would have doubts: It's true that left returned by our solution is the minimal value satisfying feasible, but how can we know that we can split the original array to **actually get this subarray-sum**? For example, let's say nums = [7,2,5,10,8] and m = 2. We have 4 different ways to split the array to get 4 different largest subarray-sum correspondingly: 25:[[7], [2,5,10,8]], 23:[[7,2], [5,10,8]], 18:[[7,2,5], [10,8]], 24:[[7,2,5,10], [8]]. Only 4 values. But our search space [max(nums), sum(nums)] = [10, 32] has much more that just 4 values. That is, no matter how we split the input array, we cannot get most of the values in our search space.

Let's say k is the minimal value satisfying feasible function. We can prove the correctness of our solution with **proof by contradiction**. Assume that no subarray's sum is equal to k, that is, every subarray sum is less than k. The variable total inside feasible function keeps track of the total weights of current load. If our assumption is correct, then total would always be less than k. As a result, feasible(k - 1) must be True, because total would at most be equal to k - 1 and would never trigger the if-clause if total > threshold, **therefore feasible(k - 1) must have the same output as feasible(k), which is True**. But we already know that k is the minimal value satisfying feasible function, **so feasible(k - 1) has to be False, which is a contradiction**. So our assumption is incorrect. Now we've proved that our algorithm is correct.

[875. Koko Eating Bananas [Medium]]

Koko loves to eat bananas. There are N piles of bananas, the i-th pile has piles[i] bananas. The guards have gone and will come back in H hours. Koko can decide her bananas-per-hour eating speed of K. Each hour, she chooses some pile of bananas, and eats K bananas from that pile. If the pile has less than K bananas, she eats all of them instead, and won't eat any more bananas during this hour.

Koko likes to eat slowly, but still wants to finish eating all the bananas before the guards come back. **Return the minimum integer K such that she can eat all the bananas within H hours**.

Example :

Input: piles = [3,6,7,11], H = 8

Output: 4

Input: piles = [30,11,23,4,20], H = 5

Output: 30

Input: piles = [30,11,23,4,20], H = 6

Output: 23

Very similar to LC 1011 and LC 410 mentioned above. Let's design a feasible function, given an input speed, determine whether Koko can finish all bananas within H hours with hourly eating speed speed. Obviously, the lower bound of the search space is 1, and upper bound is max(piles), because Koko can only choose one pile of bananas to eat every hour.

```
def minEatingSpeed(piles: List[int], H: int) -> int:
    def feasible(speed) -> bool:
        # return sum(math.ceil(pile / speed) for pile in piles) <= H  # slower
        return sum((pile - 1) // speed + 1 for pile in piles) <= H  # faster


    left, right = 1, max(piles)
    while left < right:
        mid = left  + (right - left) // 2
        if feasible(mid):
            right = mid
        else:
            left = mid + 1
    return left
```

1482. Minimum Number of Days to Make m Bouquets [Medium]

Given an integer array bloomDay, an integer m and an integer k. We need to make m bouquets. To make a bouquet, you need to use k **adjacent flowers** from the garden. The garden consists of n flowers, the ith flower will bloom in the bloomDay[i] and then can be used in **exactly one** bouquet. Return *the minimum number of days* you need to wait to be able to make m bouquets from the garden. If it is impossible to make m bouquets return **-1**.

Examples:

Input: bloomDay = [1,10,3,10,2], m = 3, k = 1

Output: 3

Explanation: Let's see what happened in the first three days. x means flower bloomed and _ means flower didn't bloom in the garden.

We need 3 bouquets each should contain 1 flower.

After day 1: [x, _, _, _, _]   // we can only make one bouquet.

After day 2: [x, _, _, _, x]   // we can only make two bouquets.

After day 3: [x, _, x, _, x]   // we can make 3 bouquets. The answer is 3.

Input: bloomDay = [1,10,3,10,2], m = 3, k = 2

Output: -1

Explanation: We need 3 bouquets each has 2 flowers, that means we need 6 flowers. We only have 5 flowers so it is impossible to get the needed bouquets and we return -1.

Now that we've solved three advanced problems above, this one should be pretty easy to do. The monotonicity of this problem is very clear: if we can make m bouquets after waiting for d days, then we can definitely finish that as well if we wait for more than d days.

```
def minDays(bloomDay: List[int], m: int, k: int) -> int:
    def feasible(days) -> bool:
        bonquets, flowers = 0, 0
        for bloom in bloomDay:
            if bloom > days:
                flowers = 0
            else:
                bonquets += (flowers + 1) // k
                flowers = (flowers + 1) % k
        return bonquets >= m

    if len(bloomDay) < m * k:
        return -1
    left, right = 1, max(bloomDay)
    while left < right:
        mid = left + (right - left) // 2
        if feasible(mid):
            right = mid
        else:
            left = mid + 1
    return left
```

668. Kth Smallest Number in Multiplication Table [Hard]

Nearly every one have used the Multiplication Table. But could you find out the k-th smallest number quickly from the multiplication table? Given the height m and the length n of a m * n Multiplication Table, and a positive integer k, you need to return the k-th smallest number in this table.

Example :

Input: m = 3, n = 3, k = 5

Output: 3

Explanation:

The Multiplication Table:

1	2	3
2	4	6
3	6	9

The 5-th smallest number is 3 (1, 2, 2, 3, 3).

For Kth-Smallest problems like this, what comes to our mind first is Heap. Usually we can maintain a Min-Heap and just pop the top of the Heap for k times. However, that doesn't work out in this problem. We don't have every single number in the entire Multiplication Table, instead, we only have the height and the length of the table. If we are to apply Heap method, we need to explicitly calculate these m * n values and save them to a heap. The time complexity and space complexity of this process are both O(mn), which is quite inefficient. This is when binary search comes in. Remember we say that designing condition function is the most difficult part? In order to find the k-th smallest value in the table, we can design an enough function, given an input num, determine whether there're at least k values less than or equal to num. **The minimal num satisfying enough function is the answer we're looking for**. Recall that the key to binary search is discovering monotonicity. In this problem, if num satisfies enough, then of course any value larger than num can satisfy. This monotonicity is the fundament of our binary search algorithm.

Let's consider search space. Obviously the lower bound should be 1, and the upper bound should be the largest value in the Multiplication Table, which is m * n, then we have search space [1, m * n]. The overwhelming advantage of binary search solution to heap solution is that it doesn't need to explicitly calculate all numbers in that table, all it needs is just picking up one value out of the search space and apply enough function to this value, to determine should we keep the left half or the right half of the search space. In this way, binary search solution only requires constant space complexity, much better than heap solution.

Next let's consider how to implement enough function. It can be observed that every row in the Multiplication Table is just multiples of its index. For example, all numbers in 3rd row [3,6,9,12,15...] are multiples of 3. Therefore, we can just go row by row to count the total number of entries less than or equal to input num. Following is the complete solution.

```
def findKthNumber(m: int, n: int, k: int) -> int:
  def enough(num) -> bool:
    count = 0
    for val in range(1, m + 1):  # count row by row
      add = min(num // val, n)
      if add == 0:  # early exit
        break
      count += add
    return count >= k

  left, right = 1, n * m
  while left < right:
    mid = left + (right - left) // 2
    if enough(mid):
      right = mid
    else:
      left = mid + 1
  return left
```

In LC 410 above, we have doubt "Is the result from binary search actually a subarray sum?". Here we have a similar doubt: "**Is the result from binary search actually in the Multiplication Table?**". The answer is yes, and we also can apply proof by contradiction. Denote num as the minimal input that satisfies enough function. Let's assume that num is not in the table, which means that num is not divisible by any val in [1, m], that is, num % val > 0. Therefore, changing the input from num to num - 1 doesn't have any effect on the expression add = min(num // val, n). So enough(num - 1) would also return True, same as enough(num). But we already know num is the minimal input satisfying enough function, so enough(num - 1) has to be False. Contradiction! The opposite of our original assumption is true: num is actually in the table.

719. Find K-th Smallest Pair Distance [Hard]

Given an integer array, return the k-th smallest **distance** among all the pairs. The distance of a pair (A, B) is defined as the absolute difference between A and B.

Example :

Input:

nums = [1,3,1]

k = 1

Output: 0

Explanation:

Following are all the pairs. The 1st smallest distance pair is (1,1), and its distance is 0.

(1,3) -> 2

(1,1) -> 0

(3,1) -> 2

Very similar to LC 668 above, both are about finding Kth-Smallest. Just like LC 668, We can design an enough function, given an input distance, determine whether there're at least k pairs whose distances are less than or equal to distance. We can sort the input array and use two pointers (fast pointer and slow pointer, pointed at a pair) to scan it. Both pointers go from leftmost end. If the current pair pointed at has a distance less than or equal to distance, all pairs between these pointers are valid (since the array is already sorted), we move forward the fast pointer. Otherwise, we move forward the slow pointer. By the time both pointers reach the rightmost end, we finish our scan and see if total counts exceed k. Here is the implementation:

```
def enough(distance) -> bool:  # two pointers
    count, i, j = 0, 0, 0
    while i < n or j < n:
        while j < n and nums[j] - nums[i] <= distance:  # move fast pointer
            j += 1
        count += j - i - 1  # count pairs
        i += 1  # move slow pointer
    return count >= k
```

Obviously, our search space should be [0, max(nums) - min(nums)]. Now we are ready to copy-paste our template:

```
def smallestDistancePair(nums: List[int], k: int) -> int:
    nums.sort()
    n = len(nums)
    left, right = 0, nums[-1] - nums[0]
    while left < right:
        mid = left + (right - left) // 2
        if enough(mid):
            right = mid
        else:
            left = mid + 1
    return left
```

1201. Ugly Number III [Medium]

Write a program to find the n-th ugly number. Ugly numbers are **positive integers** which are divisible by a **or** b **or** c.

Example :

Input: n = 3, a = 2, b = 3, c = 5

Output: 4

Explanation: The ugly numbers are 2, 3, 4, 5, 6, 8, 9, 10... The 3rd is 4.

Input: n = 4, a = 2, b = 3, c = 4

Output: 6

Explanation: The ugly numbers are 2, 3, 4, 6, 8, 9, 10, 12... The 4th is 6.

Nothing special. Still finding the Kth-Smallest. We need to design an enough function, given an input num, determine whether there are at least n ugly numbers less than or equal to num. Since a might be a multiple of b or c, or the other way round, we need the help of greatest common divisor to avoid counting duplicate numbers.

```
def nthUglyNumber(n: int, a: int, b: int, c: int) -> int:
    def enough(num) -> bool:
        total = num//a + num//b + num//c - num//ab - num//ac - num//bc + num//abc
        return total >= n

    ab = a * b // math.gcd(a, b)
    ac = a * c // math.gcd(a, c)
    bc = b * c // math.gcd(b, c)
    abc = a * bc // math.gcd(a, bc)
    left, right = 1, 10 ** 10
    while left < right:
        mid = left + (right - left) // 2
        if enough(mid):
            right = mid
        else:
            left = mid + 1
    return left
```

1283. Find the Smallest Divisor Given a Threshold [Medium]

Given an array of integers nums and an integer threshold, we will choose a positive integer divisor and divide all the array by it and sum the result of the division. Find the **smallest** divisor such that the result mentioned above is less than or equal to threshold.

Each result of division is rounded to the nearest integer greater than or equal to that element. (For example: 7/3 = 3 and 10/2 = 5). It is guaranteed that there will be an answer.

Example :

Input: nums = [1,2,5,9], threshold = 6

Output: 5

Explanation: We can get a sum to 17 (1+2+5+9) if the divisor is 1.

If the divisor is 4 we can get a sum to 7 (1+1+2+3) and if the divisor is 5 the sum will be 5 (1+1+1+2).

After so many problems introduced above, this one should be a piece of cake. We don't even need to bother to design a condition function, because the problem has already told us explicitly what condition we need to satisfy.

```python
def smallestDivisor(nums: List[int], threshold: int) -> int:
    def condition(divisor) -> bool:
        return sum((num - 1) // divisor + 1 for num in nums) <= threshold

    left, right = 1, max(nums)
    while left < right:
        mid = left + (right - left) // 2
        if condition(mid):
            right = mid
        else:
            left = mid + 1
    return left
```

End

Wow, thank you so much for making it to the end! Really appreciate that. As you can see from the python codes above, they all look very similar to each other. That's because I copy-pasted my own template all the time. No exception. This is the strong proof of my template's powerfulness and adaptability. I believe everyone can acquire this binary search template to solve many problems. All we need is just more practice to build up our ability to discover the monotonicity of the problem and to design a beautiful condition function.

Hope this helps.

Reference

- [C++ / Fast / Very clear explanation / Clean Code] Solution with Greedy Algorithm and Binary Search
- Approach the problem using the "trial and error" algorithm
- Binary Search 101 The-Ultimate-Binary-Search-Handbook - LeetCode
- ugly-number-iii Binary Search with picture & Binary Search Template - LeetCode