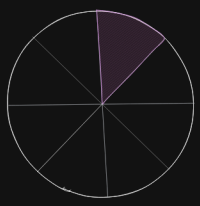
How to answer any DSA question in an interview

Time: 45-60 minutes

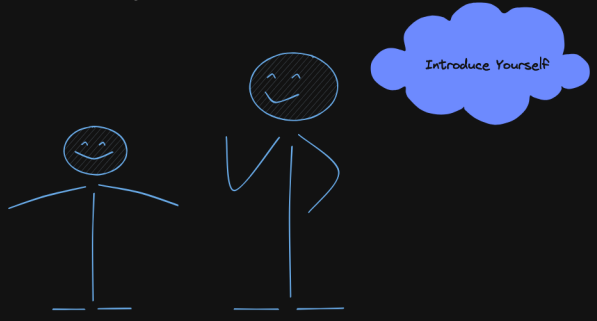
There can be multiple rounds (1-4)
It really depends upon the companies that you are interviewing for

Let's talk about 1 round
1 round can have many stages



Each stage is important. And in each stage, there are multiple things that you can do to maximise your chances of success

1. Introductions



An interviewer will introduce himself and his role at the company
Then he will ask you to introduce yourself

- Summarise your education, work experience and interests in 30-60

seconds

- Prepare and rehearse
- Smile but not too much

- Speak with confidence

- When interviewer talks about his work, pay attention (you can ask questions later)

- Connect with the interviewer. If he mentions something that you like too, either work or hobby, say it

2. Problem Statement

- Now the interviewer will give you a problem statement. Probably in shared text editor or online

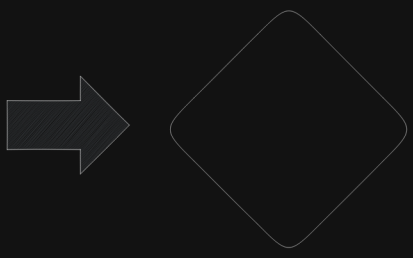
- The interviewer will paste the problem description and also a test case and then read the question to you

- Understand the question fully. Confirm it with the interviewer

- Ask clarifying questions regarding the input

1. Only integers? Or can it have other types?
2. Sorted or unsorted?
3. Empty or non-empty input?
4. What to do what it has invalid inputs

- ask about the expected input size (number of elements)



1. n is very small (backtracking)
2. n is 100-1000 (n^2)
3. n is very large, need to better around O(n) or O(n.logn) at worst

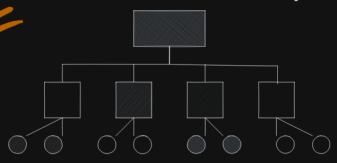
- interviwer has given you one or two example test cases. Quickly run it through to confirm you understand the problem

Ask clarifying questions
Understand the problem better
Shows that you are attentive to details
Be aware of edge cases

3. Brainstorming Data Structures and Algorithms

- Figure out what data structure or algo can be used

- Break the problem down and try to find patterns that you know



- Figure out what the problem needs you to do and then also think about the data structure or algorithm that can solve it with optimal time and space complexity

- Think out loud and talk with your interviwer. It will show that you can think of multiple data structures and their tradeoffs at the same time

- For example, we did one problem (Maximum Consecutive Ones ii) using sliding window. And if you know be vocal about it.
    1. You are considering a sliding window because every window

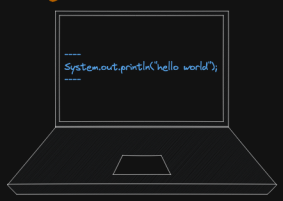
can represent the segment of max 1s until entered in a invalid state
2. Even wrong, the interviewer will like your thought process
3. How can you get this skill? Practice Leetcode problems

There is another unsaid advantage to this
1. Interviewer will know your thought process
2. Can give you hints and guide you in the right direction

- Decided on the data structure/algorithms to employ, now try to think of rough steps for your algo before coding. Explain it to interviewer, and make sure he sees it through and gives you a thumbs up or may be gives you a hint if you are entirely wrong

Always welcome the interviewer's suggestions
He knows the optimal answer to the solution
If he is giving you a hint, it is because they want you to do well
Be flexible and work with the ideas that interviewer shares with you

4. Implementation



- It's time to write code

1. Using a library like java's collections or sort, make sure the interviewer is fine with it

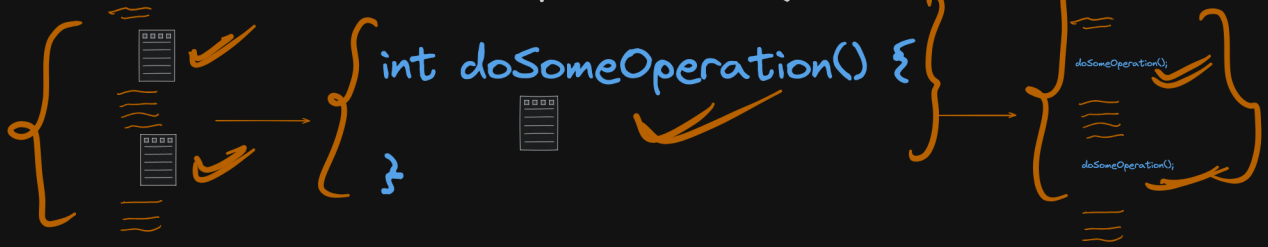
2. As you code, explain your decision making. For example, if you are solving a problem like find max length of substring without repeating characters and you are declaring a map (characterCounts), explain it is to store the character and the count to maintain the sliding window with characters only once

3. Write clean code. Depends upon the programming language. Online docs are available for different styleguides

(https://google.github.io/styleguide/)

- case conventions
- indentations
- variable naming
- function params
- etc, etc

4. Avoid duplicated code
    - Create a function if you are doing few steps multiple times



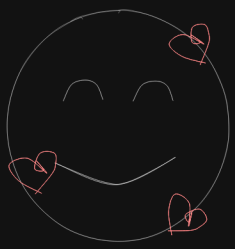
5. Use helper functions if needed. They make code modular and easy to read and a very important in real software engineering

```
createCharacterCountMap(String str) -> Map<Character, Integer>
```

One important that can come out of this is follow ups become much easier

Don't panic if you get in place where you realise that your solution might not work
COMMUNICATE with your interviewer about the concerns
It's a whole different world of hardships if you are struggling in silence (One of the worst thing in Software Engineering World)

Strategy
- Implement a brute force solution and tell that you know it is not an optimal solution (shows that you can code)
- then analyse each part of the algorithm, figure out which steps are slow and think what can be sped up
- convey it to your interviewer and include him in the discussion - they are there to help



5. Testing & Debugging

- Done with your implementation. Now what? The interviewer will not say it explicity sometimes but he would want you to test your code. It really depends how you will test it

1. Built-in test cases. Just run the code
   - Hackerrank, Hackerearth, etc. Such platforms are similar to leetcode with lots of test cases and has even edge cases too
   - Don't have to create test cases on your own. Less tension
   - Will test your solution for all the test cases. More tension

2. Write your own test cases (Shared code editor). Code is run
   - Interviewer and you have a shared code editor
   - He will want you to write your own test cases
   - From the place where the code will run first, call your code with test cases you wrote and maybe prints it to the console
   - Cover a range of test cases (Edge cases, Valid and good inputs, and invalid inputs

3. Write your own test cases (Shared doc)
   - Shared doc or text editor between you and the interviewer. Write your own test cases and walk through them manually
   - Have to manually go through algorithm with each test case
   - If you have come up with a suboptimal solution that uses two loops, don't walk through the entirety of it.

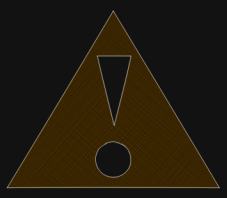
> If you have built a map to store the character counts just run it for first 4-5 characters and tell that after running the loop the map will look like this

   - As you walk through the code,
update variables and its value by writing them just right next to

the code

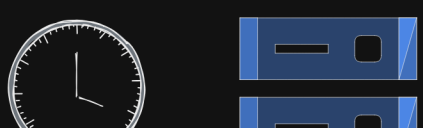
If you see that your code has error, that's ok.
- If you can run your code, put print statements for a small test case and see what the
expected values should be and what the actual values are. Identify the issue and be vocal about to your interviewer. Only then he will be able to help you
- if you cannot run your code, walk through a small test case manually. Again be vocal about the expected values and the actual values.
REMEMBER: Interviewer wants to help you but he needs to know your thought process

6. Explanation and follow-ups

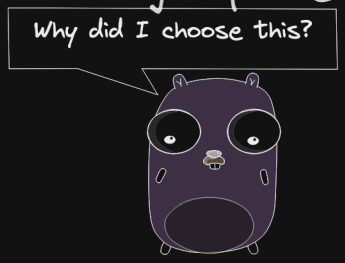


- Got the DSA, Coded it, tried all the test cases. Now what?
- Now the interviewer may ask questions around your algorithm

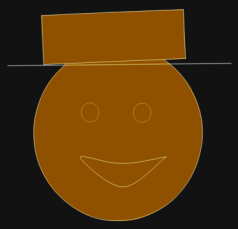
Q - What is the space and time complexity of the algorithm?
Talk about the worst case scenario. Talk about average case only if worst case is rare and average case has better runtime



Q - Why did you choose a _____ here?
Choice of Data structure, choice of algo, choice of loop, etc.
Explain your thought process properly

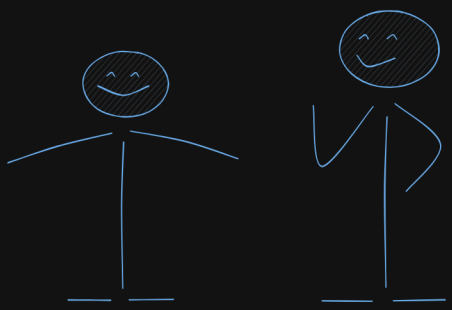


Q - Do you think that the algo can be done better in terms of time or space complexity?
You need to practice a lot of problems and data structures to really understand this
The answer is generally YES if he has asked you this. Don't be too strong about your algorithm. It's ok to be wrong, but it's not okay to not welcome other's suggestions and improve upon it.
That's a very important aspect in software engineering



if T << 45 minutes
Interviewer might ask another question
Start from step 2 (Problem statement)
Or you may be asked more follow-ups to the question asked before. New contraints, or may be imporve the space complexity, could be anything

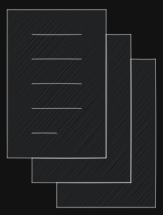
7. Outro



- Done with everything, now what? The interviewer will generally end the interview 5 minutes before the scheduled time or keep a 5 minute buffer to take questions for them or around the company

- At this time, you should try to get to know the company better and see if you would like to work there. Some sample questions you can ask are:

Q - What does an average day look like?
Q - What do you like about this company?
Q - What is your favorite thing about the job? And least favorite thing about it?
Q - What kinda of work can I expect to start with?
Q - What does the onboarding experience look like?



- Companies generally have blogs. Go through their website before and prepare a list of questions beforehand. It shows you are genuinely interested in the company and why the company functions in a certain way

Show interest at all times, keep smiling, listen well, ask follow-up questions to show that you understand their answers
It's not good if you appear bored or uninterested. It could give a bad signal to the interviewer
To sum it up, this round will hardly improve what you did in your technical round, but it can make it worse. Because if he doesn't like you in the end, it's game over

Depending on how well you did in above stages of a round in an interview, you will get a
Strong Yes
Soft Yes
Soft No
Hard NO

Yes

No

Thank you :)

LIKE, SHARE SUBSCRIBE