b'

Amazon Interview Experience | Set 189 (For SDE-1)

- Difficulty Level :\nMedium
- Last Updated :\n28 Jun, 2019

Recently, I was interviewed for Amazon SDE-1 Position. There were two telephonic rounds followed by 4 F2F rounds.

Telephonic Round 1 :
\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94
He seemed a bit unprepared for the interview. He started with some introduction and knowledge transfer about current work and then created a binary tree and asked me to write the level order, preorder, postorder and inorder traversals for that tree.
Then, the coding questions followed.

1) Flattening a multi-level linked list.
1st approach \xe2\x80\x93> Using a queue : T = O(n), S = O(n). Coded it.
2nd approach \xe2\x80\x93> Same as on GeeksForGeeks < Flattening a Linked List > : Didnt ask me to code. But this one requires changing the structure of data.

2) Flattening a multi-level linked list but the nodes in depth should be printed first. So, basically the 1st question resembles BFS and this one resembles DFS. Did this using recursion very easily.

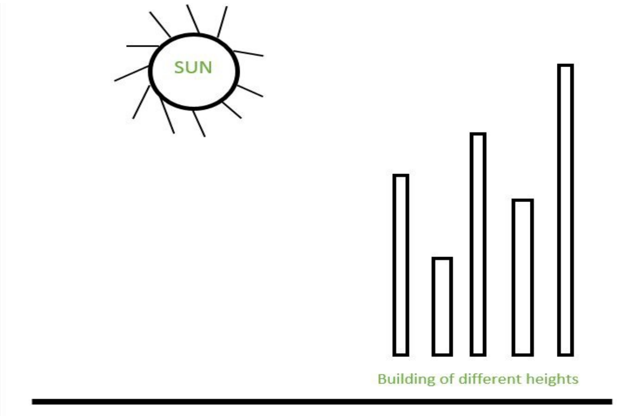
3) Process vs Threads. What happens when you type in a URL? High-level design. Handshaking protocol. HTTPS protocol, etc

\xc2\xa0
\xc2\xa0
Telephonic Round 2 :
\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94
I must say this guy who interviewed me was really smart. This round was a little more than 1 hour and there were 3 questions.

1) BST to Singly Linked List in place. A slight modification of this problem Coded it.

2) Finding Squares on the chessboard attackable by a rook
There is an NxN chessboard. Each square on the chessboard can be either empty or can have a rook. A rook as we know can attack either horizontally or vertically. Given a 2D matrix where 0 represents an empty square and 1 represents a rook, we have to fill in all the cells in the matrix with 1 which represent squares that can be attacked by any rook present on the chessboard. One can find a much simpler version of this problem here \xe2\x80\x94
Boolean Matrix Problem

3) An array of buildings is facing the sun. The heights of the building is given in an array. You have to tell which all buildings will see the sunset. This is pretty easy. The first building will definitely see the sunset and for the rest of the buildings, just maintain a variable max_height_seen_so_far and check with the height of the current building. However, he then twisted the question and asked what my approach would be if I was to scan the buildings from back to front rather than front to end. I used a stack and applied the logic similar to that used in Next Greater Element problem.



\xc2\xa0
F2F R1:
\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94\xe2\x80\x93
Started with introduction. A lot of questions about current company, current work, current project and then a design question.

How would you design the meeting invite feature of Microsoft Outlook? Considering each meeting invite as an object and that Web server is the storage space for the invites, design a data structure to receive and send invites to user in an efficient manner. The message objects must be received in a sorted manner based on the time of meeting. I suggested using a Binary Search Tree and justified the use of this DS. I gave an O(NlogN) solution. I was then asked to code it. I coded it in C#.

Followed with a lot of HR questions.

\xc2\xa0
F2F R2:
\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94\xe2\x80\x93
1) Reverse a sub array in an array. Pretty easy.

2) Rotate a subarray in an array where start and end indices of the sub array are provided and \xe2\x80\x98k\xe2\x80\x99 is provided which is the number of rotations to be done. The interviewer behaved really dumb in this question. All he wanted was a solution. He made me dry run the code again and again and he wasn\xe2\x80\x99t really bothered about the concept or the approach. I don\xe2\x80\x99t think he could relate to my solution which was O(n) in time and O(1) in space.

3) Find if a linked list has a loop. Old question. Take a fast and a slow pointer. But to get this solution wasn\xe2\x80\x99t really his motive. He asked why slow pointer should move by one node at a time and why the fast pointer should move at the speed of two nodes at a time. As led by the discussion, I was then asked to find the optimal speeds of slow and fast pointers for a given linked list. Again, led by the discussion, he asked if it\xe2\x80\x99s given that the linked list has a loop and the size of the loop is given, can I find the optimal speeds of the slow and fast pointers?

\xc2\xa0
F2F R3:
\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94\xe2\x80\x93
1) Same question as Q3 asked in telephonic round with the only difference that the heights of the buildings was provided in a linked list. Coded it in C. Then, the interviewer twisted the question by placing the sun after the last building(previously the sun was placed before the first building). Used a stack. However, this can simply be done by the reversing the heights array and using the same function written for the first part of the problem.

2) Design a data structure to represent the hierarchy of employees in an organization. Also, the design should be such that, you can retrieve the no. of reportees of a manager(not just the direct reportees but all the employees under him) very fast(O(1) was expected). Also, insertion of a new employee and removal of an employee should be fast too.

I suggested using a n-ary tree of hash tables. Also, used an additional hash table where key was employeeId and value was the address of the hash table (or the node) in the n-ary tree. My solution did give no. of reportees in O(1) and addition and removal of employees was in O(n) time, where n is the total number of employees. There wasn\xe2\x80\x99t enough time to code though.

\xc2\xa0
F2F R4:
\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94\xe2\x80\x94\xe2\x80\x93
This round had a lot of HR questions. Cultural info. Current project. He also asked coding questions but he wasn\xe2\x80\x99t really bothered about the optimality of the solution.

1) Two nodes in a BST are swapped. Find them. Told my approach. Didn\xe2\x80\x99t ask me to code it.

2) Print all the permutations of a string in lexicographic order. Coded it. Took me a lot of dry runs to make him understand that the code is correct \xf0\x9f\x98\x80

'