

Bloomberg Live Meets

Guest speakers

- Vidisha Shah:
 - fintech engineer
 - just joined in July
- Juan Uribe:
 - web solutions engineer
 - Princeton office

Intro video

Bloomberg Think Bigger

- Elevate public opinion
- Bringing transparency to public about financial data
- 85% of income goes to philanthropy

What should I expect in technical interviews?

What should I expect in my technical interviews?

The interview will include:

- “Hands on” working and talking through technical coding exercises with an engineer
- Questions about data structures/algorithms
- Constructing a solution to a problem
- Talking about previous projects and work experiences

Interview Preparation

Interview Preparation

- Review your resume
- Refresh your memory on projects and experiences
- Review your class notes
- Try practice questions using Leetcode or HackerRank
- Study with a friend, use a whiteboard, and talk through each step

For Data Structures:

implementations, runtime & storage complexity for operations, pros and cons

For Algorithms:

Searching and sorting; focus on strengths and weaknesses, make comparisons

- Be able to talk about previous experience

- Explain cost benefit analysis

- How those project bettered you as an engineer

Data

Data Structures & Algorithms

Data Structures

Be familiar with:

Arrays, Linked Lists, Binary Trees, Binary Search Trees, Graphs, Stacks, Sets, Queues, Heaps, Hash Tables

Don't overcomplicate! For most questions you usually just need an array or a hash table or both.

Make your code work before optimizing! It's better to have a working solution than struggling to make something fast.

Algorithms

Be familiar with:

Searching: Depth First Search, Breadth-First Search, Binary Search

Sorting: Insertion Sort, Selection Sort, Bubble Sort, Merge Sort, Quick Sort

Recursion

Things to know

- Most common are the *Arrays ... Graphs*
- Get familiar over the top 100 questions on LeetCode

How do we know which structures to use?

- Getting enough experience on problems
- Recognize those patterns
- Make a list of the data structures to run through when tackling a problem
 - Check off the ones that would help
- For a linked list questions...
 - ADD/ASK
- Ex: when to use a graph or BST over an array
- At times you may not know which structure to pick
 - Don't waste time on this and focus on problem at hand
- Max 2 questions may be asked !!!

Interviewing Steps

- Listen to the problem
- Ask clarifying questions
- Restate the problem
- Start with which solution comes to mind then work towards a complete design

7



- Spend up to 5 minutes understanding problem & ask questions
- Code in the language you're most comfortable with
- Not completely getting solution to run DOES NOT disqualify you

Interviewing Steps

- Note: for Bloomberg interviews, you may code in your preferred language
- Walk through your solution, thinking about edge cases
- Discuss runtime

8

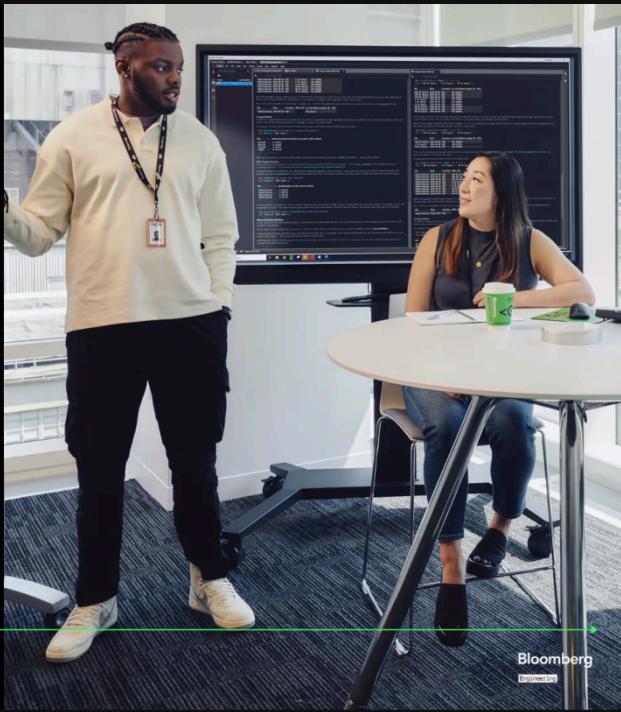


- It's ok to stop talking for a little bit to just code
- You can ask if you're on the right track

Communication Tips

1. Think out loud
2. It is helpful to start with small sample cases
3. Think about and vocalize on edge cases
4. Participate in the technical conversation with your interviewer // think out loud
5. Ask questions and respond to notes given
6. We want you to succeed and become our future teammates!
7. Take Hints!

9 Bloomberg Accelerator Summer School



How do we best communicate if we don't understand hints interviewer is giving?

- Ask clarifying questions to figure out what the hint means

Demo of interview!

```
1 // sky = [
2 //   [0, 1, 0, 1],
3 //   [1, 1, 1, 0],
4 //   [0, 1, 0, 1],
5 //   [1, 0, 1, 0]
6 // ]
7
8 // for each element
9 // check if cloud (value 1)
10 // recursively check hori/ vertically
11 // increment cloudcount
12
13 int
```

Prompt: Given a 2D matrix, where 1s mean clouds and 0s are the sky...

Goal: We want to find out how many clouds there are!

Interviewee —> Vidisha

Interviewer —> Juan

Vidisha's Assumptions

Can we assume that the diagonal means connected?

- No, only concerned with vertical and horizontal movement

Can we assume that there might be other values in the matrix besides 1s and 0s?

- No, there will only be 1s and 0s

Can we manipulate the original 2D list?

Vidisha's thoughts

- Apply a DFS solution could help
- Wrote down pseudocode

```
7
8 // for each element
9 // check if cloud (value 1)
10 // recursively check hori/ vertically
11 // increment cloudcount
12
```

Vidisha's early solution (pt. 1)

```
4
5 int cloudcount(vector<vector<int>> sky){
6     int count = 0;
7     for(int i = 0; i < sky.size(); i++){
8         for(int j=0; j < sky[0].size(); j++){
9             if(sky[i][j] == 1){
10                 dfs(sky, i, j);
11                 count++; // 5
12             }
13         }
14     }
15     return count;
16 }
```

- We want to traverse each element, every column for every row
- The DFS pseudocode is to show that we want to go deeper and explore if there are anymore 1s connected either horizontally or vertically to original position
- We update the count once we come back from the recursion call

Vidisha's early solution (pt. 2)

```

    ✓ void dfs(vector<vector<int>> sky, int i, int j){

        if(i < 0 || j < 0 || i > sky.size()-1 || j > sky[0].size()-1) return;
        if(sky[i][j] == 1)
        {
            sky[i][j] = 0;

            dfs(sky, i-1, j);
            dfs(sky, i+1, j);
            dfs(sky, i, j-1);
            dfs(sky, i, j+1);
        }
        return;
    }

```

- We check the boundary, return nothing if breaks it
- We set the original position to 0 to not consider it
- Then we recursively call for U, D, L, & R positions

Vidisha's checking edge cases

- She goes through example edge cases or inputs on the editor
- She walks through the solution, explaining to interviewer

Space & Time complexity

- Time: ORIGINAL ANSWER: $O(N \cdot M)$ - since we are going through elements on each N column for each M row
- Space:
 - ORIGINAL ANSWER: $O(1)$ - since we are manipulating the original matrix, BUT we are recursively calling
 - ANSWER AFTER HINT: $O(N \cdot M)$ - Maximum amount of times to recursively stack on frame is $O(N \cdot M)$

Edge cases include:

- The matrix is a “zero” matrix
- The boundary check is an edge case technically

Vidisha's Final solution

```

7
8 // sky = [
9 //     [0, 0, 0, 0],
10 //    [0, 0, 0, 0],
11 //    [0, 0, 0, 0],
12 //    [0, 0, 0, 0]
13 // ]
14
15
16 // for each element
17 // check if cloud (value 1)
18 // recursively check hori/ vertically
19 // increment cloudcount
20 ~void dfs(vector<vector<int>>& sky, int i, int j){
21
22     if(i < 0 || j < 0 || i > sky.size()-1 || j > sky[0].size()-1) return;
23     if(sky[i][j] == 1)
24     {
25         sky[i][j] = 0;
26
27         dfs(sky, i-1, j);
28         dfs(sky, i+1, j);
29         dfs(sky, i, j-1);
30         dfs(sky, i, j+1);
31     }
32     return;
33 }
34
35 ~int cloudcount(vector<vector<int>> sky){
36     int count = 0;
37     for(int i = 0; i < sky.size(); i++){      // O(n)
38         for(int j=0; j < sky[0].size(); j++){  // O(n)
39             if(sky[i][j] == 1){
40                 dfs(sky, i, j);
41                 count++; // 5
42             }
43         }
44     }
45     return count;
46 }
47
48 // Space: O(m*n)
49 // Time: O(m*n)

```

This is considered as a “medium” question in Bloomberg!

How they can make it harder:

- Adding premise for adding recursive stacks if sky is a very big matrix
- Adding diagonal movements
- Making it a 3D sky instead of 2D

Final advice

- Practice and remember patterns
 - The interviewers are there to help you
-