TWO SIGMA

5pm ET, Friday, October 14, 2022

Technical Interview Workshop @ Harvard

Yongwhan Lim
Senior Quantitative Software Engineer

Important Legal Information

This document is being distributed for informational and educational purposes only and is not an offer to sell or the solicitation of an offer to buy any securities or other instruments. The information contained herein is not intended to provide, and should not be relied upon for, investment advice. The views expressed herein are not necessarily the views of Two Sigma Investments, LP or any of its affiliates (collectively, "Two Sigma"). Such views reflect the assumptions of the author(s) of the document and are subject to change without notice. The document may employ data derived from third-party sources. No representation is made by Two Sigma as to the accuracy of such information and the use of such information in no way implies an endorsement of the source of such information or its validity.

Disclaimer

The copyrights and/or trademarks in some of the images, logos or other material used herein may be owned by entities other than Two Sigma. If so, such copyrights and/or trademarks are most likely owned by the entity that created the material and are used purely for identification and comment as fair use under international copyright and/or trademark laws. Use of such image, copyright or trademark does not imply any association with such organization (or endorsement of such organization) by Two Sigma, nor vice versa.

Yongwhan Lim

Senior Quantitative Software Engineer at Two Sigma

- Currently:
  - Senior Quantitative Software Engineer at Two Sigma
  - Lecturer in EECS at MIT
  - Associate in Computer Science at Columbia University
  - Visiting Instructor at Cornell-Tech
  - ICPC Head Coach at Columbia University
  - ICPC Judge for Greater New York and Mid-Central Regionals in North America
  - ICPC Judge for North America Qualifier
- Previously:
  - Research Software Engineer at Google Research
- Education:
  - Stanford
    - Math (BS ‘11) and CS (BS ‘11)
    - CS (MS ‘13)
  - MIT
    - Operations Research (PhD, started in 2013 but on an extended leave-of-absence since 2016)

https://www.cs.columbia.edu/~yongwhan/

Overview

- Part I
  - Interview Types
  - Technical Interview
  - Interview Topics
  - 2 Sample Interview Questions
  - Interview Preparation Resources

- Part II: Questions and Answers (Q&A's)

Part I

Interview Types

- Technical Interview
  - tests technical skill-sets required for a job.

- Behavioral Interview
  - tests soft skills (e.g., effective communication, conflict resolution, etc.)

Interview Types

- **Technical Interview**
  - tests technical skill-sets required for a job.

- Behavioral Interview
  - tests soft skills (e.g., effective communication, conflict resolution, etc.)

Technical Interview Overview (Company Dependent)

- Recruiter Call

- 0-1 Online Coding Challenge
  - automated screening with 2-3 questions.

- 2-3 Technical Phone Screenings
  - first technical conversation with human.

- 4-7 Interviews Onsite
  - similar to phone screening but more in-depth.
  - you may get probed on your claimed expertise.

- 0-5 Fit Calls and Negotiation

Technical Interview Overview (Company Dependent)

- Recruiter Call

- **0-1 Online Coding Challenge**
  - automated screening with 2-3 questions.

- **2-3 Technical Phone Screenings**
  - first technical conversation with human.

- **4-7 Interviews Onsite**
  - similar to phone screening but more in-depth.
  - you may get probed on your claimed expertise.

- 0-5 Fit Calls and Negotiation

Interview Topics Overview

- Data Structures and Algorithms
- (> entry level) System Design Problems

Interview Topics Overview

- **Data Structures and Algorithms**

- (> entry level) System Design Problems

Interview Topics Overview

- **Fundamentals**
  - Primitive Types
  - Arrays and Linked Lists
  - Binary Trees
  - Heaps
  - Sorting
- **Important**
  - Stacks and Queues
  - Hash Tables
  - Binary Search Trees
  - Searching
  - Recursion
- **Real Differentiators**
  - Strings
  - Dynamic Programming
  - Greedy Algorithms and Invariants
  - Graphs

Interview Topics Overview

- **Fundamentals**
  - Primitive Types
  - Arrays and Linked Lists
  - Binary Trees
  - Heaps
  - Sorting
- **Important**
  - Stacks and Queues
  - Hash Tables
  - Binary Search Trees
  - Searching
  - Recursion
- **Real Differentiators**
  - Strings
  - Dynamic Programming
  - Greedy Algorithms and Invariants
  - Graphs

Sample Interview Question #1

- **Problem Statement** (LeetCode #1201: Medium)
  - An **ugly number** is a positive integer that is divisible by a, b, or c.
  - Given four integers n, a, b, and c, return n^{th} **ugly number**.

Sample Interview Question #1

- **Problem Statement** (LeetCode #1201: Medium)
  - An **ugly number** is a positive integer that is divisible by a, b, or c.
  - Given four integers n, a, b, and c, return n^{th} **ugly number**.

- **Constraints**
  - n, a, b, c ≤ 1,000,000,000

Sample Interview Question #1

- **Problem Statement** (LeetCode #1201: Medium)
  - An **ugly number** is a positive integer that is divisible by a, b, or c.
  - Given four integers n, a, b, and c, return n^{th} **ugly number**.

- **Constraints**
  - n, a, b, c ≤ 1,000,000,000

Ideas?

Sample Interview Question #1

- **Binary Search Solution (Logarithmic):**

Now, do you see it?

Sample Interview Question #1

- **Binary Search Solution (Logarithmic):**

```
#include<bits/stdc++.h>
using namespace std;

int nthUglyNumber(int n, int a, int b, int c) {
        int low = 1, high = INT_MAX;
        while(low < high) {
                int mid = low + ((high - low) >> 1);
                if(eval(mid, a, b, c) >= n) {
                        high = mid;
                } else {
                        low = mid + 1;
                }
        }
        return low;
}
```

```
typedef long long ll;

ll lcm(ll a, ll b) {
        return a/__gcd(a,b)*b;
}

ll eval(ll x, ll a, ll b, ll c) {
        return x/a + x/b + x/c - x/lcm(a,b) - x/lcm(a,c) - x/lcm(b,c) + x/lcm(a,lcm(b,c));
}
```

Sample Interview Question #2

- **Problem Statement** (LeetCode #1312: Hard)
  - Given a string s. In one step you can insert any character at any index of the string.
  - Return *the minimum number of steps* to make s palindrome.
  - A **Palindrome String** is one that reads the same backward as well as forward.

Sample Interview Question #2

- **Problem Statement** (LeetCode #1312: Hard)
  - Given a string s. In one step you can insert any character at any index of the string.
  - Return *the minimum number of steps* to make s palindrome.
  - A **Palindrome String** is one that reads the same backward as well as forward.

- **Constraints**
  - $1 \leq |s| \leq 500$
  - s consists of lowercase English letters.

Sample Interview Question #2

- **Problem Statement** (LeetCode #1312: Hard)
  - Given a string s. In one step you can insert any character at any index of the string.
  - Return *the minimum number of steps* to make s palindrome.
  - A **Palindrome String** is one that reads the same backward as well as forward.

- **Constraints**
  - $1 \leq |s| \leq 500$
  - s consists of lowercase English letters.

Ideas?

Sample Interview Question #2

- **Dynamic Programming Solution (Quadratic):**

Now, do you see it?

Sample Interview Question #2

- **Dynamic Programming Solution (Quadratic):**

```
#include<bits/stdc++.h>
using namespace std;

int minInsertions(string &s) {
        int n = s.size();
        vector<vector<int>> dp(n, vector<int>(n,0));
        for (int i = 1; i < n; i++)
                for (int j = 0, k = i; k < n; j++, k++)
                        dp[j][k] = (s[j]==s[k]) ? dp[j+1][k-1] : min(dp[j][k-1],dp[j+1][k])+1;
        return dp[0][n-1];
}
```

Interview Preparation Resources

- **Popular Websites**
  - LeetCode
  - CodeForces
  - AtCoder
  - TopCoder
  - CodeChef

Interview Preparation Resources

- **Popular Websites**
  - **LeetCode**
  - CodeForces
  - AtCoder
  - TopCoder
  - CodeChef

Interview Preparation Resources

- **Popular Websites**
  - **LeetCode**
  - CodeForces
  - AtCoder
  - TopCoder
  - CodeChef

- Try to solve all problems from biweekly/weekly LeetCode contest **fast**.
  - Here, fast means under 1 hour for all four questions!

- Aim to be on **division I** at CodeForces
  - This will trivialize most of the technical interview.

Interview Preparation Resources

- Standard
  - *Elements of Programming Interview* by Adnan Aziz, et. al.
  - *Cracking the Coding Interview* by Gayle Laakmann McDowell

- Overkill
  - *Competitive Programming 4* by Steven Halim, et. al.
  - *Guide to Competitive Programming* by Antti Laaksonen

Interview Preparation Resources

- Standard
  - ***Elements of Programming Interview*** by Adnan Aziz, et. al.
  - ***Cracking the Coding Interview*** by Gayle Laakmann McDowell

- Overkill
  - *Competitive Programming 4* by Steven Halim, et. al.
  - *Guide to Competitive Programming* by Antti Laaksonen

Part II
Q & A's

Contact Information

- Emails:
  - yongwhan.lim@twosigma.com
  - yongwhan@mit.edu
  - yongwhan.lim@columbia.edu

- Personal Website: https://cs.columbia.edu/~yongwhan

- LinkedIn Profile: https://www.linkedin.com/in/yongwhan/
  - Feel free to send me a connection request.
  - Always happy to make connections with promising students!