KAIST Technical Interview Workshop Part I

Yongwhan Lim
Senior Quantitative Software Engineer at Two Sigma
8pm KST, Wednesday, July 13, 2022

Yongwhan Lim

- Currently:
  - Senior Quantitative Software Engineer at Two Sigma
  - Lecturer in EECS at MIT
  - Research Mentor in SIMR at Harvard University
  - Associate in Computer Science at Columbia University
  - ICPC Head Coach at Columbia University
  - ICPC Judge for Greater New York Mid-Central Regionals in N.A.
- Previously:
  - Research Software Engineer at Google Research
- Education:
  - Stanford: Math & CS (BS '11) and 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 & Answers (Q & A)

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 Screens
  - first technical conversation with human.
- 4-7 Interviews in Onsite
  - similar to phone screening but more in-depth; you may get probed on your claimed expertise.
- 0-5 Fit Calls & Negotiation

Technical Interview Overview (Company Dependent)

- Recruiter Call
- **0-1 Online Coding Challenge**
  - automated screening with 2-3 questions.
- **2-3 Technical Phone Screens**
  - first technical conversation with human.
- **4-7 Interviews in Onsite**
  - similar to phone screening but more in-depth; you may get probed on your claimed expertise.
- 0-5 Fit Calls & 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 & Linked Lists
- Binary Trees
- Heaps
- Sorting

Important

- **Stacks & Queues**
- **Hash Tables**
- **Binary Search Trees**
- **Searching**
- **Recursion**

Real Differentiators

- **Strings**
- **Dynamic Programming**
- **Greedy Algorithms and Invariants**
- **Graphs**

Sample Interview Question #1 (Medium)
Data Structures and Algorithm

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

Sample Interview Question #1 (Medium)
Data Structures and Algorithm

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

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

Sample Interview Question #1 (Medium)
Data Structures and Algorithm

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

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

Any Ideas?

Sample Interview Question #1 (Medium) Data Structures and Algorithm

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

Now, do you see it?

Sample Interview Question #1 (Medium) Data Structures and Algorithm

- **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 (Hard)
Data Structures and Algorithm

- **Problem Statement (LeetCode #1201)**
  - 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 (Hard)
Data Structures and Algorithm

- **Problem Statement (LeetCode #1201)**
  - 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 (Hard)
Data Structures and Algorithm

- **Problem Statement (LeetCode #1201)**
  - 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**
  - $1 \le |s| \le 500$
  - *s* consists of lowercase English letters.

Any Ideas?

Sample Interview Question #2 (Hard)
Data Structures and Algorithm

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

Now, do you see it?

Sample Interview Question #2 (Hard)
Data Structures and Algorithm

- **Dynamic Programming (DP) 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
Data Structures and Algorithm

- Popular Websites
  - LeetCode
  - CodeForces
  - AtCoder, TopCoder, and CodeChef

Interview Preparation Resources
Data Structures and Algorithm

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

Interview Preparation Resources
Data Structures and Algorithm

- Popular Websites
  - **LeetCode**
  - **CodeForces**
  - AtCoder, TopCoder, and 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 1 at CodeForces
  - This will trivialize the technical interview.

Interview Preparation Resources
Data Structures and Algorithm

- References:
  - **Elements of Programming Interview (2nd edition)**
  - Overkill:
    - Competitive Programming 4
    - Guide to Competitive Programming

Part II: Q&A's

Q & A's

- How do you overcome nervousness?

Q & A's

- Could you provide a live solving of a technical question?

Q & A's

- Is interview process as an intern different from full-time technical interview?

Q & A's

- How do you get past the automatic filter?

Q & A's

- What are topics to prepare, the best way to prepare, and programming language expectations?

Q & A's

- Will interviewer evaluate applicants' technical knowledge other than coding skills?

Q & A's

- Are there any specific machine learning and artificial intelligence technical questions that frequently show up in interviews (and that we should prepare for)?

Q & A's

- If I do not have too much background on a position I am applying for, how do I leave a good impression to interviewer?

Contact Information

- yongwhan@mit.edu or yongwhan.lim@columbia.edu
- https://cs.columbia.edu/~yongwhan
- https://www.linkedin.com/in/yongwhan/