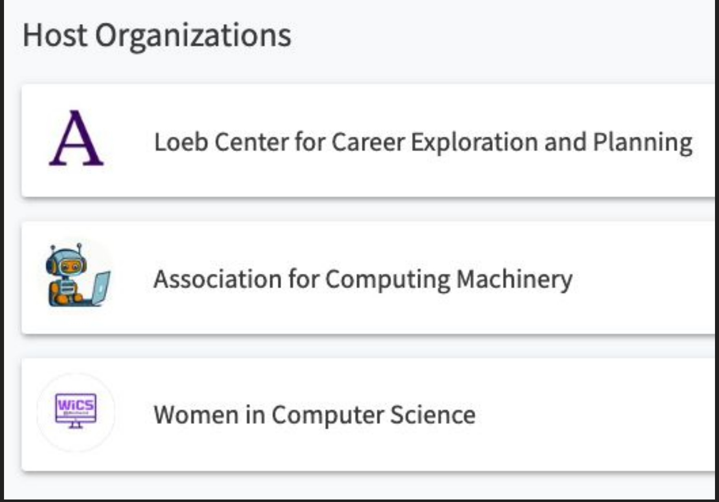
2024-25 Preparing for the Coding Interview:

Insights & Advice for Entering the Tech World

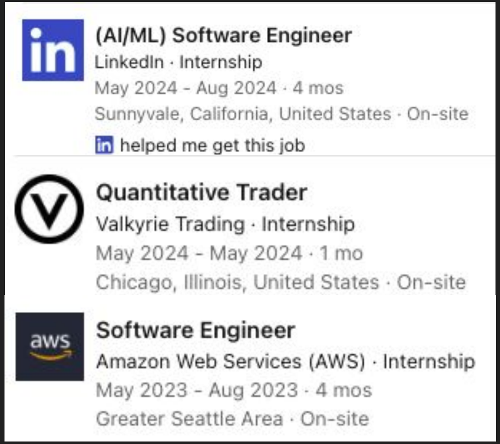


Host Organizations

- Loeb Center for Career Exploration and Planning
- Association for Computing Machinery
- Women in Computer Science

Presenter's Biographical Information:

Hello World! I'm Dhyey Mavani.



(AI/ML) Software Engineer
LinkedIn · Internship
May 2024 - Aug 2024 · 4 mos
Sunnyvale, California, United States · On-site
in helped me get this job

Quantitative Trader
Valkyrie Trading · Internship
May 2024 - May 2024 · 1 mo
Chicago, Illinois, United States · On-site

Software Engineer
Amazon Web Services (AWS) · Internship
May 2023 - Aug 2023 · 4 mos
Greater Seattle Area · On-site

I am also a Peer Career Advisor at the Loeb Center

Overview

1. Bird’s-Eye View at Tech-Industry Interview Process
2. What are technical interviews?
3. UMPIRES Method & What the interviewer ranks you on?
4. Coding (Leetcode-style) Question Walkthrough
5. BLIND 75 + Neetcode.io + Resources
6. Meet & Greet with Alumni (Ryan Cunningham & Alex Lee)

1. Bird’s-Eye View at Tech-Industry Interview Process

1. Resume Screening: *Tailor your resume with relevant keywords to pass the ATS.*

2. HireVue Video Interview: *Practice clear, confident responses to common behavioral questions.*

3. Behavioral Interview: *Research company & prepare answers about your motivations & experience.*

4. Online Coding Challenge: *Practice coding problems on platforms like LeetCode and HackerRank.*

5. Technical Phone Screen: *Use collaborative coding tools and explain your thought process clearly.*

6. On-Site Interviews (2-3): *Be ready for design discussions and whiteboard coding challenges.*

7. Final Decision: *Send thank-you notes and prepare for potential offer negotiations.*

2. What are Technical Interviews

- Company’s opportunity to get a sense of how would it be to work with you on interesting problems

- Designed to test your problem solving skills + receptiveness to feedback

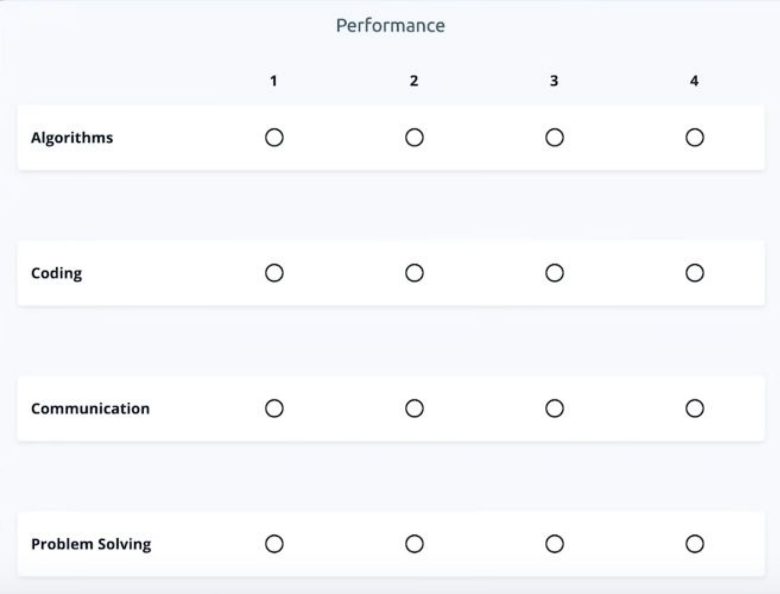
- Gain a better understanding of your coding style / design clarity

- They might contain small problems (implement a function) or problems where you will have to lay the groundwork (class structure) with an objective

3.1 UMPIRES Method

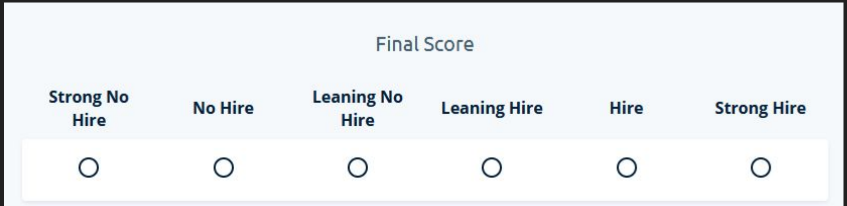
Understand	(ask clarifying questions about edge-cases for instance)
Match	(identify what “kind” of problem you are trying to solve)
Plan	(think about paradigm & data structures you would like to invoke)
Implement	(translate into code, this should be easy if the Plan was descriptive)
Review	(give run-through of your code & test to detect errors / edge-cases)
Evaluate	(state run-time space & time complexity, discuss pros/cons)
Supplement	(brainstorm & if time permits, implement improvements)

3.2 What Interviewers Rank you on?



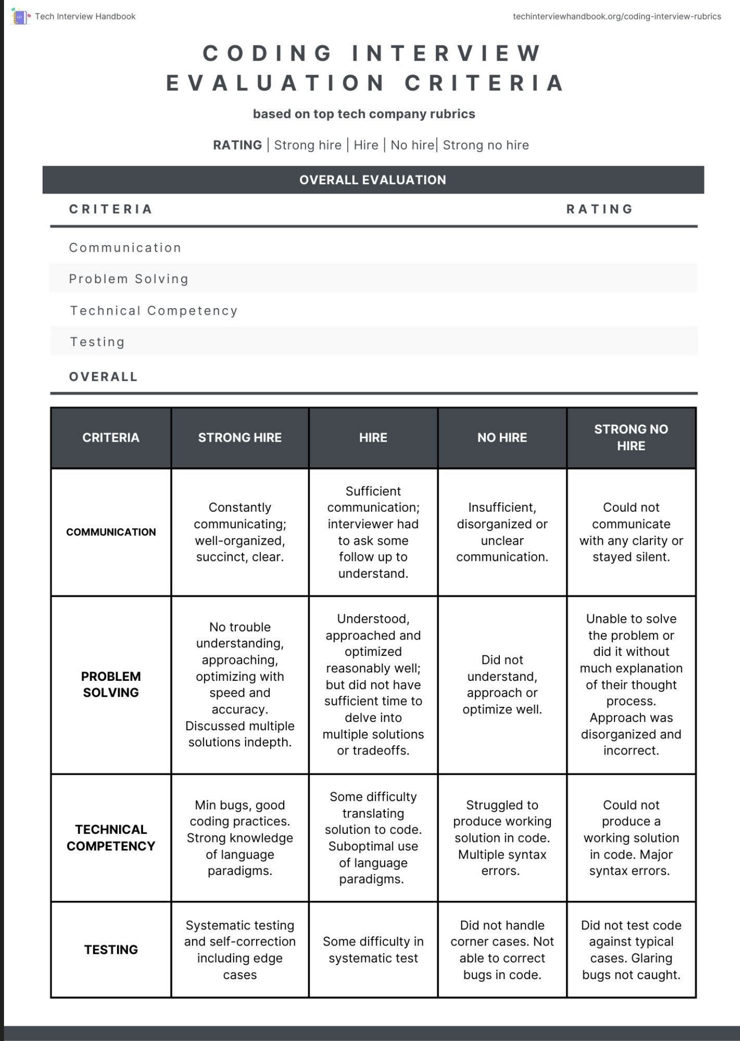
Performance

	1	2	3	4
Algorithms	○	○	○	○
Coding	○	○	○	○
Communication	○	○	○	○
Problem Solving	○	○	○	○



Final Score

Strong No Hire	No Hire	Leaning No Hire	Leaning Hire	Hire	Strong Hire
○	○	○	○	○	○



Tech Interview Handbook techinterviewhandbook.org/coding-interview-rubrics

CODING INTERVIEW EVALUATION CRITERIA

based on top tech company rubrics

RATING | Strong hire | Hire | No hire| Strong no hire

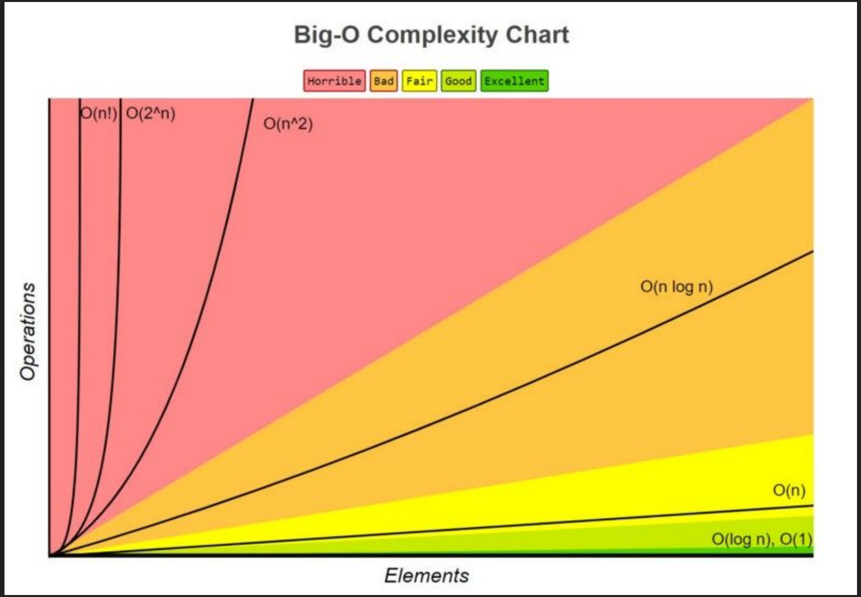
OVERALL EVALUATION

CRITERIA	RATING
Communication	
Problem Solving	
Technical Competency	
Testing	
OVERALL	

CRITERIA	STRONG HIRE	HIRE	NO HIRE	STRONG NO HIRE
COMMUNICATION	Constantly communicating; well-organized, succinct, clear.	Sufficient communication; interviewer had to ask some follow up to understand.	Insufficient, disorganized or unclear communication.	Could not communicate with any clarity or stayed silent.
PROBLEM SOLVING	No trouble understanding, approaching, optimizing with speed and accuracy. Discussed multiple solutions indepth.	Understood, approached and optimized reasonably well; but did not have sufficient time to delve into multiple solutions or tradeoffs.	Did not understand, approach or optimize well.	Unable to solve the problem or did it without much explanation of their thought process. Approach was disorganized and incorrect.
TECHNICAL COMPETENCY	Min bugs, good coding practices. Strong knowledge of language paradigms.	Some difficulty translating solution to code. Suboptimal use of language paradigms.	Struggled to produce working solution in code. Multiple syntax errors.	Could not produce a working solution in code. Major syntax errors.
TESTING	Systematic testing and self-correction including edge cases	Some difficulty in systematic test	Did not handle corner cases. Not able to correct bugs in code.	Did not test code against typical cases. Glaring bugs not caught.

3.3 Wait … What's time/space Complexity? (The Big-O)

- Big-O Notation is a way to measure how much time/storage required by an algorithm grows, as the amount of data it processes increases.



- Let's Think through some examples …

- What does this mean in the context of space?

4.1 Coding (Leetcode) Question Walkthrough # 1

Objective: Find duplicate items! *(Intentionally vague for demonstration)*

4.1 Coding (Leetcode) Question Walkthrough # 1

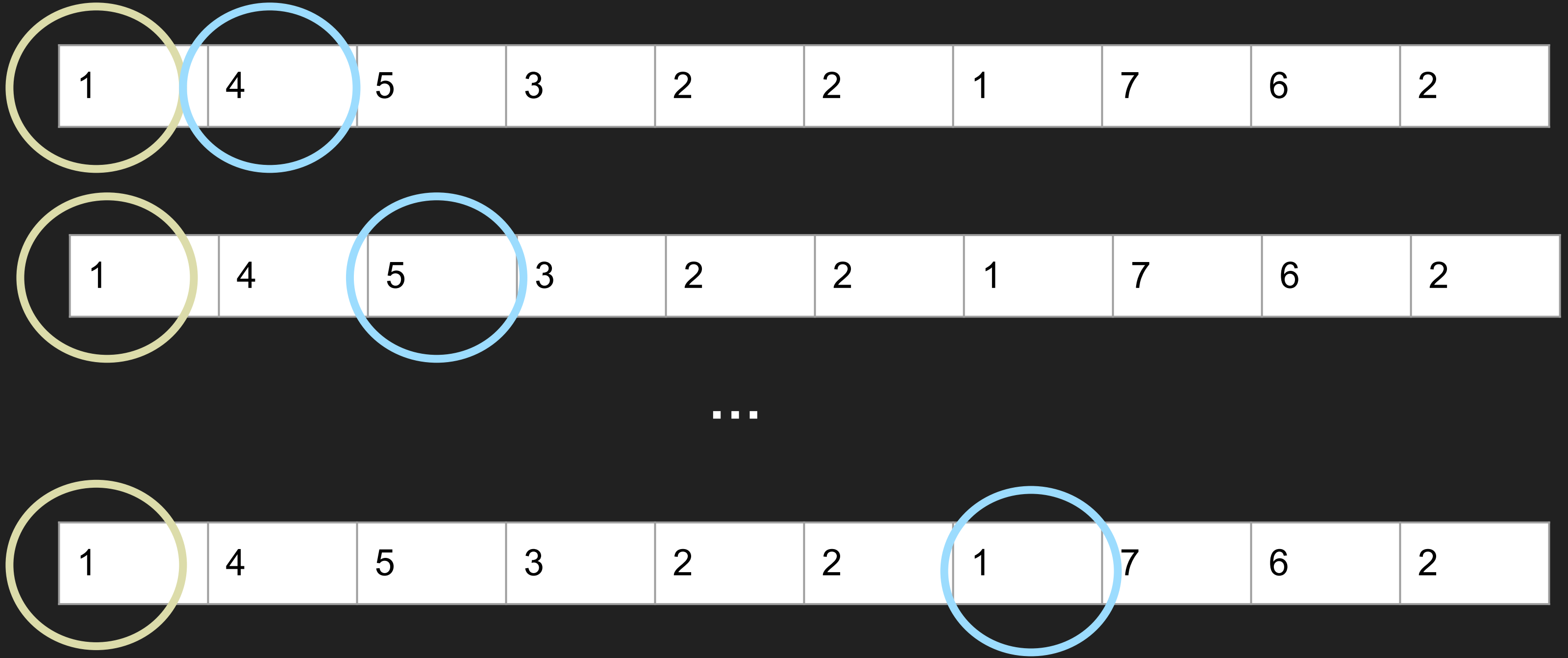
Objective: Find duplicate items! *(Intentionally vague for demonstration)*

You may assume that you have an array of integers.

1	4	5	3	2	2	1	7	6	2

Specifically, for now, our goal is to return True / False indicating presence or absence of a duplicate in the array.

3. Solving an “Easy” Leetcode question



1	4	5	3	2	2	1	7	6	2

1	4	5	3	2	2	1	7	6	2

…

1	4	5	3	2	2	1	7	6	2

false!

4.1 Coding (Leetcode) Question Walkthrough # 1

```java
// The Brute Force Approach:

public boolean containsDuplicate(int[] nums) {
    for(int left = 0; left < nums.length; left++)
    {
        for(int right = left + 1; right < nums.length; right++)
        {
            if(nums[left] == nums[right]) return true;
        }
    }
    return false;
}
```

4.1 Coding (Leetcode) Question Walkthrough # 1

Objective: Find duplicate items! *(Intentionally vague for demonstration)*

You may assume that you have an array of integers & that it is **Pre-SORTED**

1	1	2	2	2	3	4	5	6	7

Specifically, for now, our goal is to return True / False indicating presence or absence of a duplicate in the array.

4.1 Coding (Leetcode) Question Walkthrough # 1

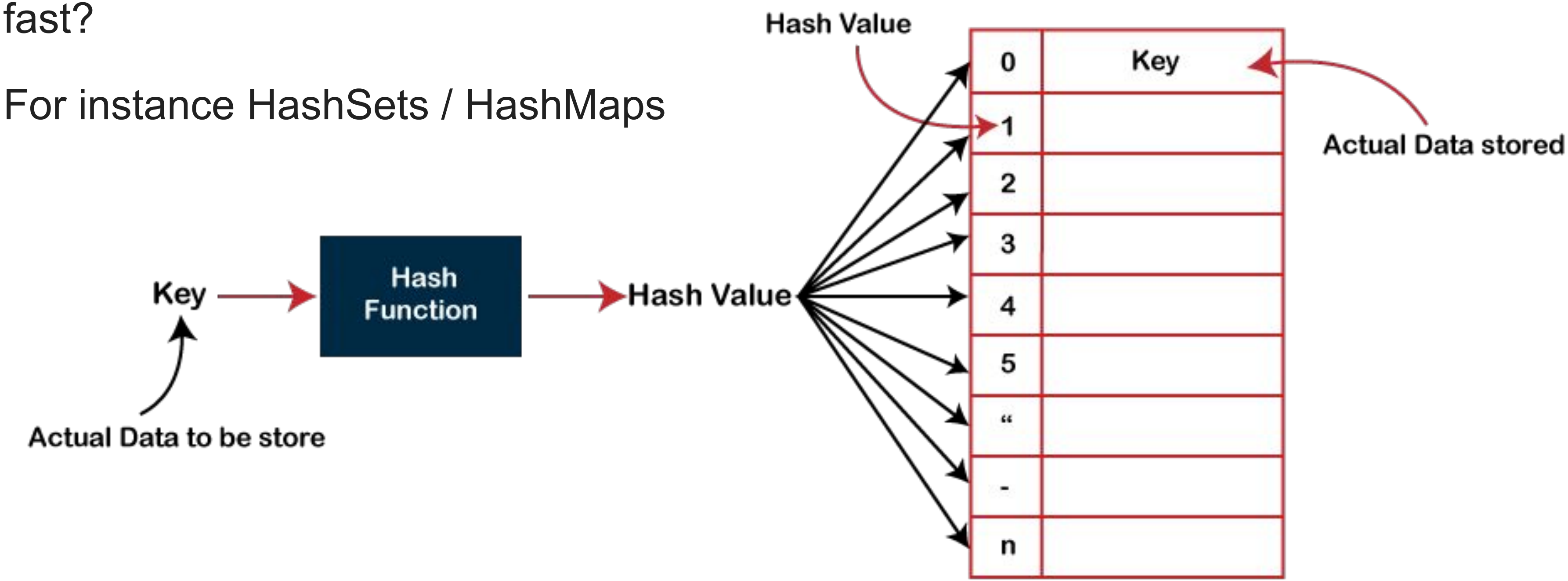
```
// Approach for sorted array:

public boolean containsDuplicate(int[] nums) {
     for(int left = 0; left < nums.length - 1; left++) {
         if (nums[left] == nums[left + 1]) return true;
     }
     return false;
 }
```

4.1 Coding (Leetcode) Question Walkthrough # 1

What if we have a data structure in which we can find a particular element really fast?

For instance HashSets / HashMaps



4.1 Coding (Leetcode) Question Walkthrough # 1

```
// Approach for HashSets:
public boolean containsDuplicate(int[] nums) {
    HashSet<Integer> myNumbers = new HashSet<Integer>();
    for(int index = 0; index < nums.length; index++) {
        if(myNumbers.contains(nums[index])) return true;
        myNumbers.add(nums[index]);
    }
    return false;
}
```

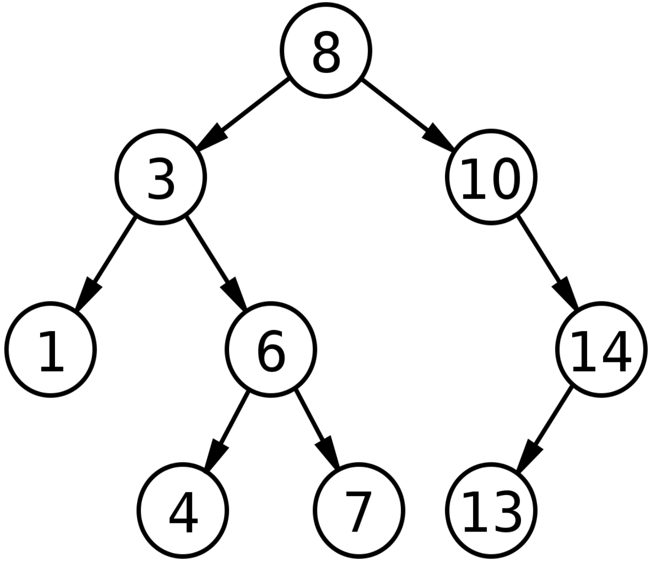
4.2 Coding (Leetcode) Question Walkthrough # 2

Given the root of a Binary Search Tree determine if it is a valid BST or not.

A valid BST is defined as follows:

- Left subtree of a node contains only nodes with val less than that of node
- Right subtree of a node contains only nodes with val greater than that of node
- Both the left and right subtrees must also be binary search trees.

4.2 Coding (Leetcode) Question Walkthrough # 2



```
# Python Definition for a binary tree node.
class TreeNode:
    def __init__(self, val=0, left=None, right=None):
            self.val = val
            self.left = left
            self.right = right
class Solution:
    def isValidBST(self, root: TreeNode) -> bool:
            return valid(root, float("-inf"), float("inf"))

    def valid(node, left, right):
            if not node:
                return True
            if not (left < node.val < right):
                return False

            return valid(node.left, left, node.val) and
                     valid(node.right, node.val, right)
```

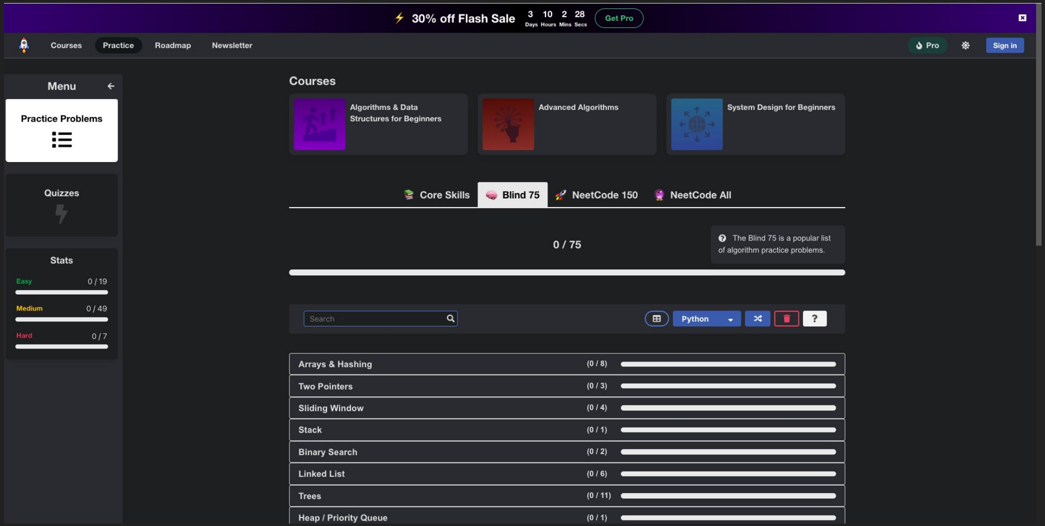
5. BLIND 75 + Neetcode.io + Resources

LeetCode

- Blind-75: A list of 75 most frequently used Leetcode questions.

- In your preparation, it is recommended that you start with these questions.

- Go to neetcode.io/practice for a list of these questions & some great Python explanations & walkthroughs.



Stats

Easy	0 / 19
Medium	0 / 49
Hard	0 / 7

Courses

Algorithms & Data Structures for Beginners
Advanced Algorithms
System Design for Beginners
Core Skills
Blind 75
NeetCode 150
NeetCode All

0 / 75

The Blind 75 is a popular list of algorithm practice problems.

Arrays & Hashing	(0 / 8)
Two Pointers	(0 / 3)
Sliding Window	(0 / 4)
Stack	(0 / 1)
Binary Search	(0 / 2)
Linked List	(0 / 6)
Trees	(0 / 11)
Heap / Priority Queue	(0 / 1)

HackerRank

CodeSignal

5.1 Additional Resources

- Simplify.jobs: web-extension to auto-fill mundane details of the applications

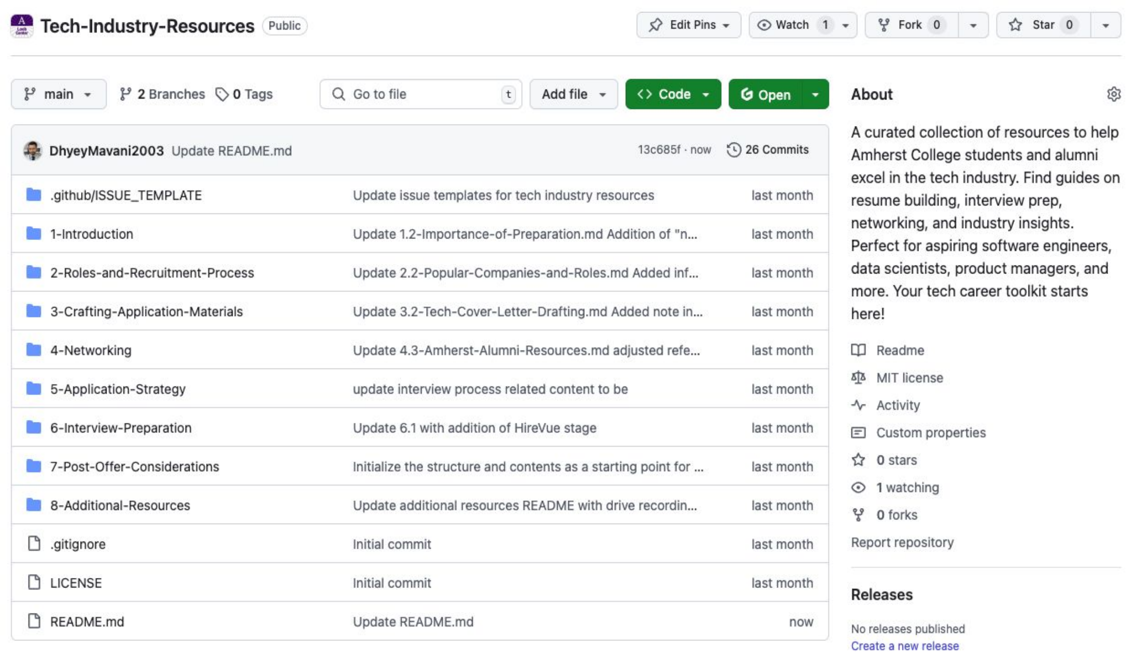
- Job Trackers (GitHub Repos):
  - https://github.com/SimplifyJobs/Summer2025-Internships
  - https://github.com/SimplifyJobs/New-Grad-Positions

- Levels.fyi: a salary-transparency website to learn more about median compensation packages of your target company & have job list by pay order

- LeetHub: a chrome extension that automatically pushes your code, comments & relevant metadata to your GitHub repository when you solve a Leetcode Q

BONUS: Loeb Center’s Central Tech Resources Page

- https://github.com/Amherst-College-Loeb-Center/Tech-Industry-Resources/tree/main



DhyeyMavani2003 Update README.md 13c685f · now 26 Commits

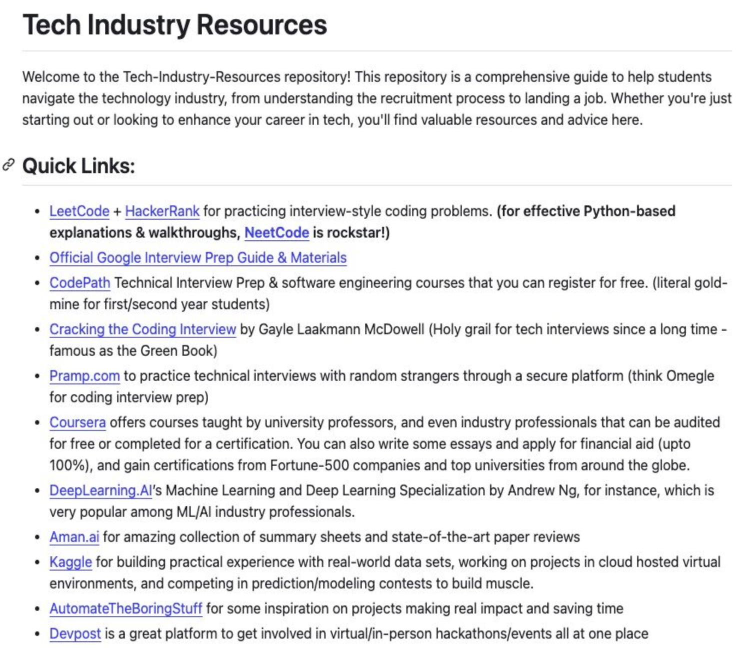
.github/ISSUE_TEMPLATE	Update issue templates for tech industry resources	last month
1-Introduction	Update 1.2-Importance-of-Preparation.md Addition of "n...	last month
2-Roles-and-Recruitment-Process	Update 2.2-Popular-Companies-and-Roles.md Added inf...	last month
3-Crafting-Application-Materials	Update 3.2-Tech-Cover-Letter-Drafting.md Added note in...	last month
4-Networking	Update 4.3-Amherst-Alumni-Resources.md adjusted refe...	last month
5-Application-Strategy	update interview process related content to be	last month
6-Interview-Preparation	Update 6.1 with addition of HireVue stage	last month
7-Post-Offer-Considerations	Initialize the structure and contents as a starting point for ...	last month
8-Additional-Resources	Update additional resources README with drive recordin...	last month
.gitignore	Initial commit	last month
LICENSE	Initial commit	last month
README.md	Update README.md	now

About

A curated collection of resources to help Amherst College students and alumni excel in the tech industry. Find guides on resume building, interview prep, networking, and industry insights. Perfect for aspiring software engineers, data scientists, product managers, and more. Your tech career toolkit starts here!

Releases

No releases published
Create a new release



Tech Industry Resources

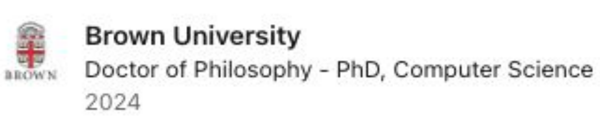
Welcome to the Tech-Industry-Resources repository! This repository is a comprehensive guide to help students navigate the technology industry, from understanding the recruitment process to landing a job. Whether you're just starting out or looking to enhance your career in tech, you'll find valuable resources and advice here.

Quick Links:

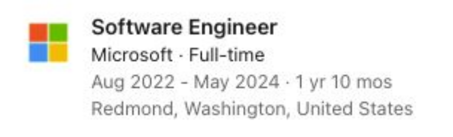
- LeetCode + HackerRank for practicing interview-style coding problems. **(for effective Python-based explanations & walkthroughs, NeetCode is rockstar!)**
- Official Google Interview Prep Guide & Materials
- CodePath Technical Interview Prep & software engineering courses that you can register for free. (literal gold-mine for first/second year students)
- Cracking the Coding Interview by Gayle Laakmann McDowell (Holy grail for tech interviews since a long time - famous as the Green Book)
- Pramp.com to practice technical interviews with random strangers through a secure platform (think Omegle for coding interview prep)
- Coursera offers courses taught by university professors, and even industry professionals that can be audited for free or completed for a certification. You can also write some essays and apply for financial aid (upto 100%), and gain certifications from Fortune-500 companies and top universities from around the globe.
- DeepLearning.AI’s Machine Learning and Deep Learning Specialization by Andrew Ng, for instance, which is very popular among ML/AI industry professionals.
- Aman.ai for amazing collection of summary sheets and state-of-the-art paper reviews
- Kaggle for building practical experience with real-world data sets, working on projects in cloud hosted virtual environments, and competing in prediction/modeling contests to build muscle.
- AutomateTheBoringStuff for some inspiration on projects making real impact and saving time
- Devpost is a great platform to get involved in virtual/in-person hackathons/events all at one place

6. Meet & Greet with Alumni

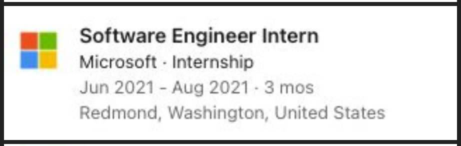
Alexander Lee ‘22



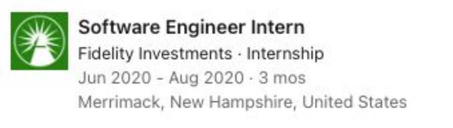
Brown University
Doctor of Philosophy - PhD, Computer Science
2024



Software Engineer
Microsoft · Full-time
Aug 2022 - May 2024 · 1 yr 10 mos
Redmond, Washington, United States

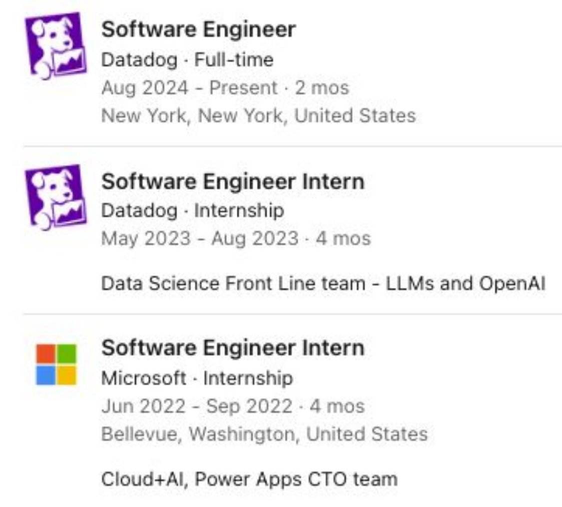


Software Engineer Intern
Microsoft · Internship
Jun 2021 - Aug 2021 · 3 mos
Redmond, Washington, United States



Software Engineer Intern
Fidelity Investments · Internship
Jun 2020 - Aug 2020 · 3 mos
Merrimack, New Hampshire, United States

Ryan Cunningham ‘24



Software Engineer
Datadog · Full-time
Aug 2024 - Present · 2 mos
New York, New York, United States

Software Engineer Intern
Datadog · Internship
May 2023 - Aug 2023 · 4 mos

Data Science Front Line team - LLMs and OpenAI

Software Engineer Intern
Microsoft · Internship
Jun 2022 - Sep 2022 · 4 mos
Bellevue, Washington, United States

Cloud+AI, Power Apps CTO team