



Intro to Data Structures and Algorithms
and Study Strategies



Let's set expectations

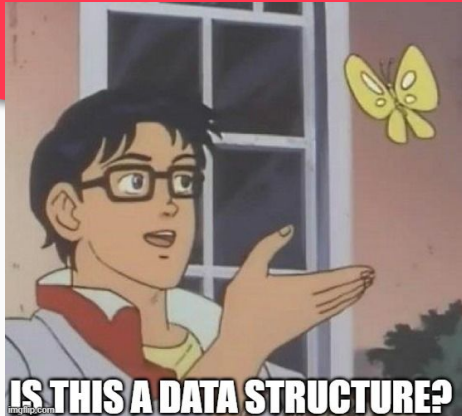
This workshop is meant to give you a solid foundation and framework to build upon. We will cover the most common patterns, but we cannot expect to be able to quickly recognize every pattern in different forms within a week's time. The content we cover will be extremely dense, and we will be moving very quickly.

Being able to recognize patterns quickly takes a LOT of practice and time, and there are **NO SHORTCUTS**.

By the end of this week, **our goal is to understand what all our common tools are**. To recognize when to use these tools will take time.

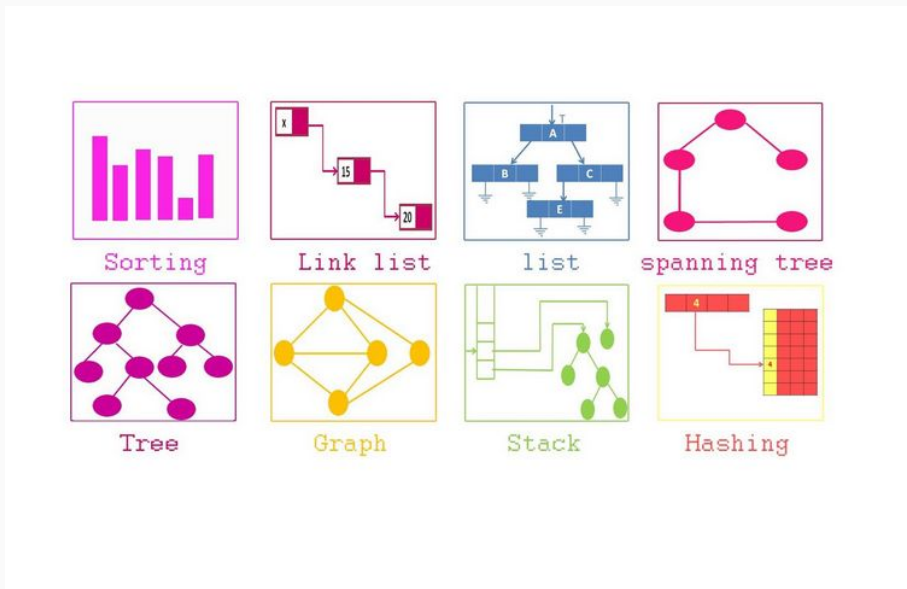


What are data structures and algorithms anyways...?



Data Structure and Algorithms

- A data structure is simply a way to to STORE data
 - examples:
 - Arrays
 - Hash tables
 - Hash sets
 - Linked Lists
 - Trees
 - Graphs
- Algorithms are ways to INTERACT with our data
 - The goal is to create, read, edit, or delete data in a way that is useful.
- We will look at this in the context of coding interviews
 - Design
 - Implementation
 - Trade offs



Why do I need to know this...?

- Helps us understand how efficient our solutions are in a measurable way.
- Use of good DS&A practices make your code more SCALABLE.
- Most importantly: **IT'S ON THE INTERVIEW**
 - Whether we agree or disagree with the efficacy of the process in finding good candidates is irrelevant. These are the rules of the game, and we have to play regardless.



Is this in? According to FIFA rulings, yes.

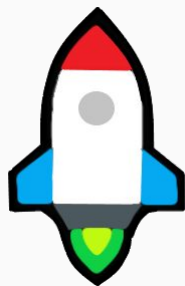


What is the best way to study DS&A?



History Lesson

- [Blind 75](#)
 - [Leetcode Discussion](#)
 - created by someone who did hundreds of Leetcode problems and found these 75 to be the most useful to him for reviewing fundamentals
 - some of these problems are VERY difficult for someone just starting though
- [NeetCode](#)
 - started by an unemployed aspiring software developer
 - slowly practiced over the course of a year and was able to get into Google
 - ordered things in a way that made sense (actually very similar to how Structy is formatted)



Three Different Ways to Study

1. Targeted Study

- Choose a specific topic(s) and study problems from those topic(s)

2. Random Study

- Choose randomly selected problems to solve

3. Mock Interviews

- Solve a random problem under a time constraint in front of someone who asks you follow-up questions



Pros and Cons for each Study Strategy

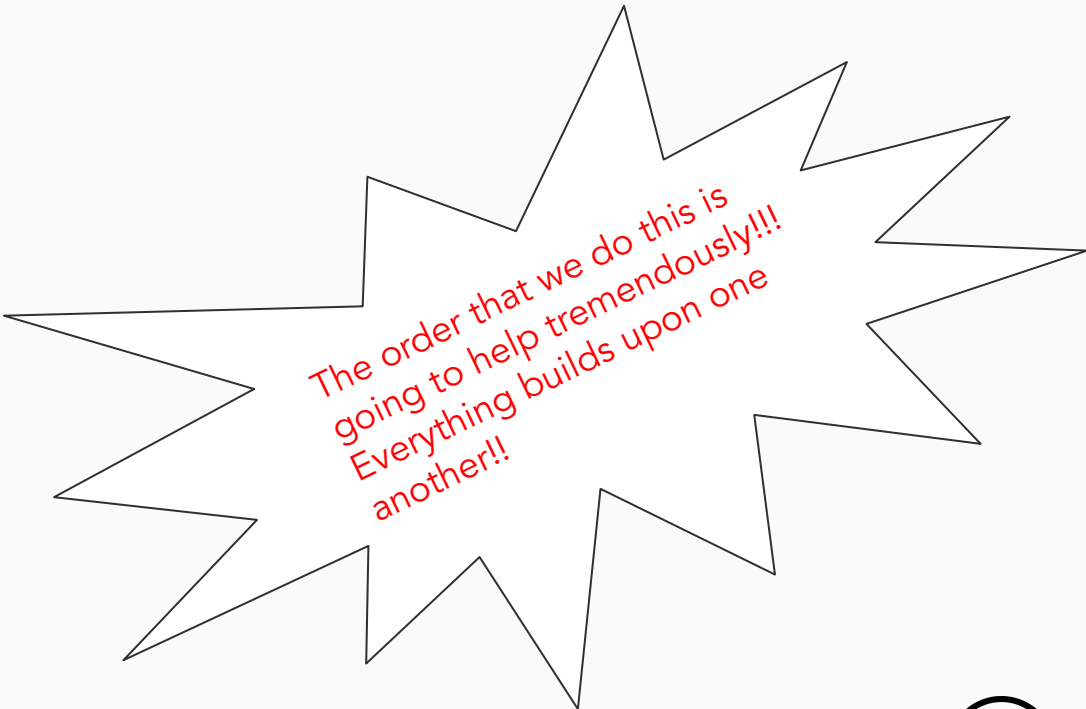
	PROS	CONS
TARGETED STUDY	Allows you to repeat similar problems together to more easily see patterns that would indicate when to use a particular strategy.	You already have an notion of what kind of strategy to use, which isn't accurate to how a real-life interview works.
RANDOM STUDY	Forces you to read the prompt more carefully in order to recognize what kind of strategy to use.	Many find it very difficult to see patterns for a particular topic when constantly switching around, especially when first starting out.
MOCK INTERVIEWS	Forces you to practice communicating your thoughts and manage your time, which are additional skills needed to succeed in an interview.	Can be time consuming and not time efficient when it comes to learning material.

You need to do ALL these consistently. How much time you dedicate to each one depends on what your current objectives are.



So where do we start?

- Arrays
- Hashing
- Linked Lists
- Recursion
- Sorting Algorithms
- Binary Search
- Trees
- Heaps
- Graphs
- Dynamic Programming



The order that we do this is
going to help tremendously!!!
Everything builds upon one
another!!



Is it enough to solve the problem?

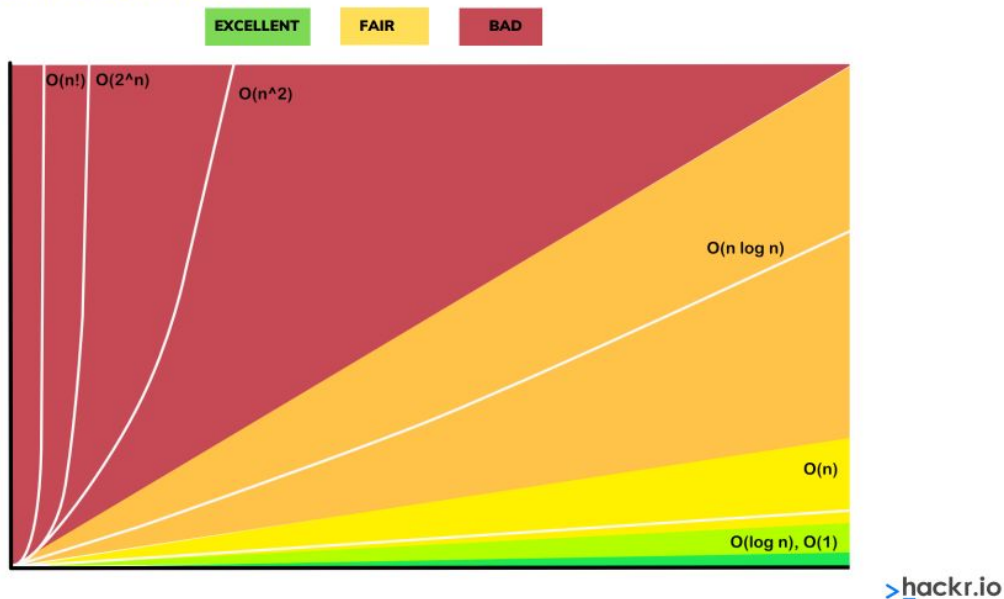
- We must UNDERSTAND and ANALYZE our solutions
- We must ensure that we can confidently communicate our ideas
- Track your progress! Memorize ideas and concepts, NOT solutions

NO!



Big O

BIG O COMPLEXITY CHART



- Big O is how we analyze our code
- Usually, we are concerned with the **worst case scenario**, though there are exceptions
- Our Big O analysis is based upon **how our algorithm scales according to the input**



Questions?

