|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Leetcode Workshop** | | | | | |
| Week | Title | Objective | Outline | | Resources / Links |
| *Content* | *Duration* |
| 3 | A Brief Introduction to Leetcode & Hash Tables  **"You seriously don't have a life, do you?"** | Welcome | 1. Introduction to the training programme 2. Objectives and Agenda 3. Provided Resources and utilities | 5 min | - |
| Acing your Coding Interview | 1. Class Pedagogy (Pair Programming) 2. How to approach a coding interview | 10 min |  |
| A tour of the Leetcode IDE | 1. Why we use LeetCode 2. Introducing the LeetCode IDE 3. Default programming language (Python) | 15 min | [Leetcode](https://support.leetcode.com/hc/en-us/articles/360012016874-Start-your-Coding-Practice) IDE |
| Programming Warmup - Single Number | 1. Problem Statement 2. The point of various solutions (Complexity, Creativity) | 20 min | [Single Number](https://leetcode.com/problems/single-number/) |
| Introducing Hash Tables | 1. Runtime Analysis of Basic operations 2. Advantages and Disadvantages | 10 min | [Python Dictionaries](https://www.digitalvidya.com/blog/python-dictionary/) |
| Case Study: Contains Duplicate | 1. Naïve Implementation (Quadratic time Complexity) 2. Hash Table Implementation (Linear Time complexity, Some space tradeoff) | 10 min | [Contains Duplicate I](https://leetcode.com/problems/contains-duplicate/) |
| Easy Practice: Two Sum | 1. Hash-map Implementation 2. Short discussion on applying Duplicate to Two Sum | 20 min | [Two Sum](https://leetcode.com/problems/two-sum/) |
| Possible Extension: 3 Sum | 1. Applying ideas from 2-Sum to solve 3-Sum | 30 min | [3 Sum](https://leetcode.com/problems/3sum/) |
| Bonus: Longest Substring without repeating characters | 1. Short introduction to Pointer Arithmetic 2. Why is a hash-table so powerful here? | - | [Longest Substring without repeating Characters](https://leetcode.com/problems/longest-substring-without-repeating-characters/) |
| 4 | Topic 2 - Arrays  **"Run of the mill."** | Agenda | 1. Objectives and Agenda | 10 min | - |
| Introducing Arrays | 1. Runtimes Analysis of Basic operations 2. Advantages and Disadvantages 3. Relevance in Coding Interviews (Staple Data Structure) 4. Scope of testing - Pointers, DP, Everything. | 10 min | [Python Arrays](https://medium.com/@blobbyblobfish/arrays-and-linked-lists-pros-cons-b763b383955b) |
| Easy Practice: Remove Duplicates from Sorted Array | 1. Basic Introduction to problem 2. Introducing Set Syntax for Python | 20 min | [Remove Duplicates](https://leetcode.com/problems/remove-duplicates-from-sorted-array/) |
| Easy Practice: Move Zeros | 1. Space Optimal Solution - Introducing In-place moving | 20 min | [Move Zeros](https://leetcode.com/problems/move-zeroes/solution/) |
| Medium Practice: Sort Colours | 1. Basic Introduction to problem 2. Short discussion on flexibility of problem solutions | 30 min | [Sort Colours](https://leetcode.com/problems/coin-change/) |
| Possible Extension: Number of Islands | 1. Scope of the Problem 2. A possible issue: Out of bounds 3. My approach (Recursion) | 25 min | [Number of Islands](https://leetcode.com/problems/number-of-islands/) |
| Bonus: Container with Most Water | 1. Some intuition 2. Hints and debrief | 20 min | [Container with Most Water](https://leetcode.com/problems/container-with-most-water/) |
| 5 | Topic 3 - Dynamic Programming  **"Those who cannot remember the past are condemned to repeat it"** | Agenda | 1. Objectives and Agenda | 10 min | - |
| What is Dynamic Programming? | 1. Definition of Dynamic Programming (Layman) 2. How can a problem be classified under DP? 3. How can interviewers make your life difficult? | 10 min | [Dynamic Programming](https://www.javatpoint.com/dynamic-programming-introduction) |
| A classic case study: The Fibonacci Sequence | 1. Naïve Approach to calculate n-th Fibonacci number. 2. DP - Array 3. DP - Memorization 4. Ground-up and Top-down | 15 min | - |
| Easy Practice: Unique Paths | 1. DP Problems can come in the form of matrices. 2. Alternative Method using Combinatorics | 25 min | [Unique Paths](https://leetcode.com/problems/unique-paths/) |
| Medium Practice: Unique Paths II | 1. Updates in Condition 2. Recap about how problem can be twisted | 30 min | [Unique Paths II](https://leetcode.com/problems/unique-paths-ii/) |
| Possible Extension: Coin Change | 1. Basic Introduction to problem 2. Dynamic Programming based on Comprehension | 30 min | [Coin Change](https://leetcode.com/problems/coin-change/) |
| Bonus (Hard): Longest Palindromic Substring | 1. Intuition to the Problem with some hints | - | [Longest Palindromic Substring](https://leetcode.com/problems/longest-palindromic-substring/) |
| Bonus: Longest Valid Parentheses | 1. Intuition to the Problem with some hints | - | [Longest Valid Parentheses](https://leetcode.com/problems/longest-valid-parentheses/) |
| 6 | Topic 4 - Strings  **"Most prone to typo errors"** | Agenda | 1. Objectives and Agenda | 10 min | - |
| Recap on Strings | 1. Immutability and expensive copying operation 2. Pros and Cons and considerations 3. Vulnerability to Naïve Implementations | 10 min | - |
| Easy Practice: Valid Parentheses | 1. Quick explanation on Valid Parentheses 2. Recursion method 3. Stack method | 20 min | [Valid Parentheses](https://leetcode.com/problems/valid-parentheses/) |
| Medium Practice: Generate Parentheses | 1. Introducing Notions of Backtracking | 40 min | [Generate Parentheses](https://leetcode.com/problems/generate-parentheses/) |
| Medium Practice: Palindromic Substrings | 1. Brute Force Approach 2. Dynamic Programming Approach 3. Pointers Approach | 40 min | [Palindromic Substrings](https://leetcode.com/problems/palindromic-substrings/) |
| Bonus: Substring with Concatenation of All Words | 1. Sliding Window Approach 2. Sensitivity to Question Constraints | - | [Substring with Concatenation of All Words](https://leetcode.com/problems/substring-with-concatenation-of-all-words/) |
| 7 | Topic 5 - Depth-First Traversal  **"Da\_\_y's Favourite."** | Agenda | 1. Objectives and Agenda | 10 min |  |
| Binary Trees | 1. Introduction to Binary Trees 2. Template Dfs Traversal (L-R, R-L) | 20 min | [Reading](https://www.educative.io/edpresso/what-is-a-binary-tree) |
| Quick Practice: Same Tree | 1. Basic Introduction - Solve using DFS | 20 min | [Same Tree](https://leetcode.com/problems/same-tree/) |
| Depth-first Search | 1. Mechanism 2. Relation to Stacks 3. Discussion of pros and cons | 20 min | [Reading](https://medium.com/basecs/demystifying-depth-first-search-a7c14cccf056) |
| Easy Practice: Maximum Depth of Binary Tree | 1. Basic practice for Binary trees using a DFS approach | 15 min | [Maximum Depth of Binary Tree](mask_modelhttps:/leetcode.com/problems/maximum-depth-of-binary-tree/) |
| Medium Practice: Validate BST | 1. Define what is a Binary Search Tree 2. Emphasize how values can be traversed | 35 min | [Validate BST](https://leetcode.com/problems/validate-binary-search-tree/) |
| Bonus: Path Sum II | 1. Reiterate how accumulators can be used | - | [Path Sum II](https://leetcode.com/problems/path-sum-ii/) |
| 11 | Topic 6 - Breadth-First Traversal  **"Most Preferred Method of Tinder Dating."** | Agenda | 1. Objectives and Agenda | 10 min |  |
| Breadth-First Search | 1. Illustration of Mechanism 2. Why is a Queue Necessary for BFS 3. Some templating code for BFS | 20 min | [Reading](https://www.interviewcake.com/concept/java/bfs) |
| Revisit: Number of Islands | 1. Re-explaining the Problem 2. Looking at Queue implementation | 30 min | [Number of Islands - BFS](https://leetcode.com/problems/number-of-islands/) |
| Revisit: Maximum Depth of Binary Tree | 1. Basic practice for Binary trees using a DFS approach | 30 min | [Maximum Depth of Binary Tree](mask_modelhttps:/leetcode.com/problems/maximum-depth-of-binary-tree/) |
| Medium Practice: Level Order Traversal | 1. Explaining the Problem 2. Dealing with "Levels" | 30 min | [Binary Tree Level Order Traversal](https://leetcode.com/problems/binary-tree-level-order-traversal/) |
| 12 | Topic 7 - (Singly) Linked List  **"You can only see what's ahead of you."** | Agenda | 1. Objectives and Agenda | 10 min | - |
| Introducing Linked Lists | 1. Basic Operations of a Linked List 2. Advantages and Disadvantages 3. Some intuition to get by | 20 min | [Linked Lists](https://www.geeksforgeeks.org/advantages-and-disadvantages-of-linked-list/) |
| Easy Practice: Remove Duplicates from Sorted List | 1. Basic Question to get started on basic Linked List Operations | 20 min | [Remove Duplicates from Sorted List](https://leetcode.com/problems/remove-duplicates-from-sorted-list/) |
| Easy Practice: Merge 2 Sorted Lists | 1. Same as above | 20 min | [Merge 2 Sorted Lists](https://leetcode.com/problems/merge-two-sorted-lists/) |
| Medium Practice: Reverse a Linked List | 1. The (stupid) base approach 2. How to use Linked List properties to achieve O(n) | 25 min | [Reverse a Linked List](https://leetcode.com/problems/reverse-linked-list/) |
| Medium Practice: Add Two Numbers | 1. Same as above, except with a little arithmetic twist. | 25 min | [Add Two Numbers](https://leetcode.com/problems/add-two-numbers/) |
| Bonus: Reverse Linked List II | 1. Combines all the learning achieved about Linked List so far | - | [Reverse Linked List II](https://leetcode.com/problems/reverse-linked-list-ii/) |