Data Structure - LLDs -( 1 Week )

[List of data structures](https://en.wikipedia.org/wiki/List_of_data_structures)

Go through these topics if you're brand new to data structures. If you understand all the below, just skip to the key algorithms.

* Lists
  + [Design Linked List](https://leetcode.com/problems/design-linked-list/)
  + [Design Skiplist](https://leetcode.com/problems/design-skiplist/)
* Stacks
  + [Implement Stack using Queues](https://leetcode.com/problems/implement-stack-using-queues/)
  + [Design a Stack With Increment Operation](https://leetcode.com/problems/design-a-stack-with-increment-operation/)
  + [LRU Cache](https://leetcode.com/problems/lru-cache/)
  + [Min Stack](https://leetcode.com/problems/min-stack/)
  + [Max Stack](https://leetcode.com/problems/max-stack/)
  + [Dinner Plate Stacks](https://leetcode.com/problems/dinner-plate-stacks/)
  + [Implement Queue using Stacks](https://leetcode.com/problems/implement-queue-using-stacks/)
* Queue
  + [Design Circular Queue](https://leetcode.com/problems/design-circular-queue/)
* Hashtable
  + [Design HashMap](https://leetcode.com/problems/design-hashmap/)
  + [Design HashSet](https://leetcode.com/problems/design-hashset/)
* BST
  + [Binary Search Tree Iterator](https://leetcode.com/problems/binary-search-tree-iterator/)
  + [Serialize and Deserialize BST](https://leetcode.com/problems/serialize-and-deserialize-bst/)
* Red Black Tree
  + [Find Median from Data Stream](https://leetcode.com/problems/find-median-from-data-stream/)
  + [Count of Range Sum](https://leetcode.com/problems/count-of-range-sum/)
* Heaps
  + [Design Twitter](https://leetcode.com/problems/design-twitter/)
  + [Kth Largest Element in a Stream](https://leetcode.com/problems/kth-largest-element-in-a-stream/)
* Fibonacci Heaps
  + [Fibonacci Heaps](https://www.cs.princeton.edu/~wayne/teaching/fibonacci-heap.pdf)
* Disjoint Sets
  + [Review of two popular approaches, Disjoint Sets and DFS](https://leetcode.com/problems/most-stones-removed-with-same-row-or-column/discuss/393127/review-of-two-popular-approaches-disjoint-sets-and-dfs-including-optimizations-java-centric)
* Tries (PrefixTree, suffixTree)
  + [Implement Trie (Prefix Tree)](https://leetcode.com/problems/implement-trie-prefix-tree/)
  + [Add and Search Word - Data structure design](https://leetcode.com/problems/add-and-search-word-data-structure-design/)
* Interval Trees/Segment Tree
  + [Lazy Dynamic Segment Tree - A general template](https://leetcode.com/problems/my-calendar-iii/discuss/288928/Lazy-Dynamic-Segment-Tree-A-general-template)
  + [A Recursive approach to Segment Trees, Range Sum Queries & Lazy Propagation](https://leetcode.com/articles/a-recursive-approach-to-segment-trees-range-sum-queries-lazy-propagation/)
* Other Tree Data Structures(Graphs)
  + [Serialize and Deserialize N-ary Tree](https://leetcode.com/problems/serialize-and-deserialize-n-ary-tree/)
  + [Encode N-ary Tree to Binary Tree](https://leetcode.com/problems/encode-n-ary-tree-to-binary-tree/)

Algorithms - Analysis Time and Space - ( 3 Weeks )

* Sorting **- 2 Days**
  + Selection Sort - [Merge Sorted Array](https://leetcode.com/problems/merge-sorted-array/)
  + Bubble Sort - [Sort Colors](https://leetcode.com/problems/sort-colors/)
  + Insertion Sort - [Insertion Sort List](https://leetcode.com/problems/insertion-sort-list/)
  + Merge Sort - [Sort an Array](https://leetcode.com/problems/sort-an-array/)
  + Quick Sort
    - [Kth Largest Element in an Array](https://leetcode.com/problems/kth-largest-element-in-an-array/)
    - [K Closest Points to Origin](https://leetcode.com/problems/k-closest-points-to-origin/)
  + Counting Sort - [Relative Sort Array](https://leetcode.com/problems/relative-sort-array/)
  + Tree sort - [Convert Sorted List to Binary Search Tree](https://leetcode.com/problems/convert-sorted-list-to-binary-search-tree/)
  + Bucket Sort - [Top K Frequent Elements](https://leetcode.com/problems/top-k-frequent-elements/)
  + Radix Sort - [Maximum Gap](https://leetcode.com/problems/maximum-gap/)
  + Topological sort - Covered in Graphs
* Divide-and-Conquer **- 2 Days**
  + The maximum-subarray problem - [Maximum Subarray](https://leetcode.com/problems/maximum-subarray/)
  + Strassen’s algorithm for matrix multiplication - [Divide and Conquer | Set 5 (Strassen's Matrix Multiplication)](https://www.geeksforgeeks.org/strassens-matrix-multiplication/)
  + The substitution method for solving recurrences
  + The recursion-tree method for solving recurrences
  + The master method for solving recurrences
* Dynamic Programming **- 2 Days**
  + Rod cutting - [Integer Break](https://leetcode.com/problems/integer-break/)
  + [Dynamic Programming for the confused : Rod cutting problem](https://medium.com/@pratikone/dynamic-programming-for-the-confused-rod-cutting-problem-588892796840)
  + Matrix-chain multiplication - [Burst Balloons](https://leetcode.com/problems/burst-balloons/)
  + Elements of dynamic programming
  + Longest common subsequence - [Longest Common Subsequence](https://leetcode.com/problems/longest-common-subsequence/)
  + Optimal binary search trees
    - [Unique Binary Search Trees](https://leetcode.com/problems/unique-binary-search-trees/)
    - [Unique Binary Search Trees II](https://leetcode.com/problems/unique-binary-search-trees-ii/)
* Greedy Algorithms **- 2 Days**
  + An activity-selection problem - [Minimum Number of Arrows to Burst Balloons](https://leetcode.com/problems/minimum-number-of-arrows-to-burst-balloons/)
  + Elements of the greedy strategy
  + Huffman codes - [Construct Huffman Tree](https://leetcode.com/discuss/interview-question/125263/construct-huffman-tree/124125), [Google | Onsite | Software Engineer | Huffman Coding Algorithm](https://leetcode.com/discuss/interview-question/416316/google-onsite-software-engineer-huffman-coding-algorithm), [Minimum Cost Tree From Leaf Values](https://leetcode.com/problems/minimum-cost-tree-from-leaf-values/)
  + Matroids and greedy methods - [Matroid intersection in simple words](https://codeforces.com/blog/entry/69287)
  + A task-scheduling problem as a matroid - [Task Scheduler](https://leetcode.com/problems/task-scheduler/)
* Graph Algorithms **- 6 Days**

[Leetcode Pattern 1 | DFS + BFS == 25% of the problems](https://medium.com/leetcode-patterns/leetcode-pattern-2-dfs-bfs-25-of-the-problems-part-2-a5b269597f52)

For DFS and BFS, strongly recommend videos by MIT OCW. Watch the video foundationals first, and then specific algorithm examples. I found the videos by Eric Dermaine very useful

<https://www.youtube.com/watch?v=AfSk24UTFS8&t=503s>

Topological Sort Theory

* + [N-ary Tree Preorder Traversal](https://leetcode.com/problems/n-ary-tree-preorder-traversal/)
  + [N-ary Tree Postorder Traversal](https://leetcode.com/problems/n-ary-tree-postorder-traversal/)
  + [N-ary Tree Level Order Traversal](https://leetcode.com/problems/n-ary-tree-level-order-traversal/)
  + BFS
    - [Binary Tree Level Order Traversal](https://leetcode.com/problems/binary-tree-level-order-traversal/)
    - [Binary Tree Level Order Traversal II](https://leetcode.com/problems/binary-tree-level-order-traversal-ii/)
    - [Web Crawler Multithreaded](https://leetcode.com/problems/web-crawler-multithreaded/)
    - [Web Crawler](https://leetcode.com/problems/web-crawler/)
    - [Cut Off Trees for Golf Event](https://leetcode.com/problems/cut-off-trees-for-golf-event/)
    - [Course Schedule](https://leetcode.com/problems/course-schedule/)
  + DFS
    - [Binary Tree Postorder Traversal](https://leetcode.com/problems/binary-tree-postorder-traversal/)
    - [Binary Tree Preorder Traversal](https://leetcode.com/problems/binary-tree-preorder-traversal/)
    - [Binary Tree Inorder Traversal](https://leetcode.com/problems/binary-tree-inorder-traversal/)
    - [Is Graph Bipartite?](https://leetcode.com/problems/is-graph-bipartite/)
    - [Remove Invalid Parentheses](https://leetcode.com/problems/remove-invalid-parentheses/)
    - [Construct Binary Tree from Preorder and Inorder Traversal](https://leetcode.com/problems/construct-binary-tree-from-preorder-and-inorder-traversal/)
  + Topological Sort - [Topological Sort](https://leetcode.com/tag/topological-sort/)
    - Watch this video, and the intuition follows … <https://www.youtube.com/watch?v=eL-KzMXSXXI&t=329s>
  + Strongly Connected Components - SCC -[~~Course Schedule~~](https://leetcode.com/problems/course-schedule/), [Airbnb | Cover all vertices with the least number of vertices](https://leetcode.com/discuss/interview-question/algorithms/124861/airbnb-cover-all-vertices-with-the-least-number-of-vertices), [Critical Connections in a Network](https://leetcode.com/problems/critical-connections-in-a-network/)
    - no leetcode questions, but very similar to "most flights from unique start"
    - [Facebook | Minimum number of people to spread a message](https://leetcode.com/discuss/interview-question/124827/Find-minimum-number-of-people-to-reach-to-spread-a-message-across-all-people-in-twitter/),
  + Minimum spanning Tree - Prim's Algorithm
    - [Cheapest Flights Within K Stops](https://leetcode.com/problems/cheapest-flights-within-k-stops/)
    - [Minimum Height Trees](https://leetcode.com/problems/minimum-height-trees/)
    - [Number of Operations to Make Network Connected](https://leetcode.com/problems/number-of-operations-to-make-network-connected/)
    - [Connecting Cities With Minimum Cost](https://leetcode.com/problems/connecting-cities-with-minimum-cost/)
  + Shortest Path Algos -
    - Bellman-Ford - [Network Delay Time](https://leetcode.com/problems/network-delay-time/), <https://leetcode.com/problems/get-watched-videos-by-your-friends/>
    - Dijkstra’s algorithm
      * [Reachable Nodes In Subdivided Graph](https://leetcode.com/problems/reachable-nodes-in-subdivided-graph/)
      * [Shortest Path Visiting All Nodes](https://leetcode.com/problems/shortest-path-visiting-all-nodes/)
    - Floyd-Warshall
      * [Find the City With the Smallest Number of Neighbors at a Threshold Distance](https://leetcode.com/problems/find-the-city-with-the-smallest-number-of-neighbors-at-a-threshold-distance/)
      * [Evaluate Division](https://leetcode.com/problems/evaluate-division/)
    - Johnson’s algorithm
      * [All-pairs shortest paths - Johnson's algorithm for sparse graphs - GeeksforGeeks](https://massivealgorithms.blogspot.com/2014/09/johnsons-algorithm-for-all-pairs.html)
      * [Johnson's algorithm](https://en.wikipedia.org/wiki/Johnson%27s_algorithm)
    - The Ford-Fulkerson method
      * [Google | Onsite | Network flow for the matrix with given row and column sums](https://leetcode.com/discuss/interview-question/337344/Google-or-Onsite-or-Network-flow-for-the-matrix-with-given-row-and-column-sums)
      * [Ford-Fulkerson Algorithm for Maximum Flow Problem](https://www.geeksforgeeks.org/ford-fulkerson-algorithm-for-maximum-flow-problem/)
* Number-Theoretic Algorithms **- 2 Days**
  + The Chinese remainder theorem - [Check If It Is a Good Array](https://leetcode.com/problems/check-if-it-is-a-good-array/)
  + Greatest common divisor
    - [Greatest Common Divisor of Strings](https://leetcode.com/problems/greatest-common-divisor-of-strings/)
    - [X of a Kind in a Deck of Cards](https://leetcode.com/problems/x-of-a-kind-in-a-deck-of-cards/)
    - [Google | OA Summer Intern 2020 | Greatest Common Divisor](https://leetcode.com/discuss/interview-question/396996/google-oa-summer-intern-2020-greatest-common-divisor)
  + Powers of an element
    - [Pow(x, n)](https://leetcode.com/problems/powx-n/)
    - [Sort Integers by The Power Value](https://leetcode.com/problems/sort-integers-by-the-power-value/)
  + The RSA public-key cryptosystem
    - [Keys and Rooms](https://leetcode.com/problems/keys-and-rooms/)
    - [Shortest Path to Get All Keys](https://leetcode.com/problems/shortest-path-to-get-all-keys/)
  + Integer factorization
    - [Largest Component Size by Common Factor](https://leetcode.com/problems/largest-component-size-by-common-factor/)
    - [Minimum Factorization](https://leetcode.com/articles/minimum-factorization/)
    - [2 Keys Keyboard](https://leetcode.com/problems/2-keys-keyboard/)
    - [Bulb Switcher](https://leetcode.com/problems/bulb-switcher/)
* String Matching - **2 Day**
  + The Rabin-Karp algorithm
    - [Implement strStr()](https://leetcode.com/problems/implement-strstr/)
    - [Binary String With Substrings Representing 1 To N](https://leetcode.com/problems/binary-string-with-substrings-representing-1-to-n/)
    - [Shortest Palindrome](https://leetcode.com/problems/shortest-palindrome/)
    - [Find All Anagrams in a String](https://leetcode.com/problems/find-all-anagrams-in-a-string/)
  + String matching with finite automata
  + The Knuth-Morris-Pratt algorithm
    - [Shortest Palindrome](https://leetcode.com/problems/shortest-palindrome/)
    - [Rotate String](https://leetcode.com/problems/rotate-string/)
    - [KMP Algorithm for Pattern Searching](https://www.geeksforgeeks.org/kmp-algorithm-for-pattern-searching/)
* Approximation Algorithms **- 3 Days**
  + The vertex-cover problem
    - [Binary Tree Cameras](https://leetcode.com/problems/binary-tree-cameras/)
    - [Vertex Cover Problem-2](https://www.geeksforgeeks.org/vertex-cover-problem-set-2-dynamic-programming-solution-tree/)
    - [Vertex Cover Problem](https://www.geeksforgeeks.org/vertex-cover-problem-set-1-introduction-approximate-algorithm-2/)
  + The traveling-salesman problem [Find the Shortest Superstring](https://leetcode.com/problems/find-the-shortest-superstring/)
  + The set-covering problem
    - [Video Stitching](https://leetcode.com/problems/video-stitching/)
    - [Set Intersection Size At Least Two](https://leetcode.com/problems/set-intersection-size-at-least-two/)
    - [Non-overlapping Intervals](https://leetcode.com/problems/non-overlapping-intervals/)
  + Randomization and linear programming
  + The subset-sum problem
    - [Partition Equal Subset Sum](https://leetcode.com/problems/partition-equal-subset-sum/)
    - [Partition to K Equal Sum Subsets](https://leetcode.com/problems/partition-to-k-equal-sum-subsets/)
* Randomized Algorithms **- 1 Day**
  + Quick Sort
  + Min Cut [Palindrome Partitioning II](https://leetcode.com/problems/palindrome-partitioning-ii/)

Concepts Problems and Maths - ( 1 Week )

* Matrix Operations
* Linear Programming
* Polynomials - DFT, FFT
* Computational Geometry
  + Line-segment properties
  + Determining whether any pair of segments intersects
  + Finding the convex hull - [Erect the Fence](https://leetcode.com/problems/erect-the-fence/), [The Skyline Problem](https://leetcode.com/problems/the-skyline-problem/)
  + Finding the closest pair of points - [K Closest Points to Origin](https://leetcode.com/problems/k-closest-points-to-origin/)
* GCD and LCM
  + [X of a Kind in a Deck of Cards](https://leetcode.com/problems/x-of-a-kind-in-a-deck-of-cards/)
  + [Greatest Common Divisor of Strings](https://leetcode.com/problems/greatest-common-divisor-of-strings/)
  + [Nth Magical Number](https://leetcode.com/problems/nth-magical-number/)
  + [Ugly Number III](https://leetcode.com/problems/ugly-number-iii/)
* Prime Factorization and Divisors
  + [Largest Component Size by Common Factor](https://leetcode.com/problems/largest-component-size-by-common-factor/)
  + [2 Keys Keyboard](https://leetcode.com/problems/2-keys-keyboard/)
* Fibonacci Numbers
  + [Length of Longest Fibonacci Subsequence](https://leetcode.com/problems/length-of-longest-fibonacci-subsequence/)
  + [Split Array into Fibonacci Sequence](https://leetcode.com/problems/split-array-into-fibonacci-sequence/)
  + [Find the Minimum Number of Fibonacci Numbers Whose Sum Is K](https://leetcode.com/problems/find-the-minimum-number-of-fibonacci-numbers-whose-sum-is-k/)
* Catalan Numbers - [Unique Binary Search Trees](https://leetcode.com/problems/unique-binary-search-trees/)
* Modular Arithmetic
* Euler Totient Function
* nCr Computations
* Set Theory
* Factorial
  + [Last Substring in Lexicographical Order](https://leetcode.com/problems/last-substring-in-lexicographical-order/)
  + [Snakes and Ladders](https://leetcode.com/problems/snakes-and-ladders/)
  + [Factor Combinations](https://leetcode.com/problems/factor-combinations/)
  + [Path With Maximum Minimum Value](https://leetcode.com/problems/path-with-maximum-minimum-value/)
  + [Number of Closed Islands](https://leetcode.com/problems/number-of-closed-islands/)
* Prime numbers and Primality Tests
  + [Prime Arrangements](https://leetcode.com/problems/prime-arrangements/)
  + [K-th Smallest Prime Fraction](https://leetcode.com/problems/k-th-smallest-prime-fraction/)
* Sieve Algorithms
  + [Count Primes](https://leetcode.com/problems/count-primes/)
* Divisibility and Large Numbers
* Series
* Number Digit
* Triangles
  + [Triangle](https://leetcode.com/problems/triangle/)
  + [Valid Triangle Number](https://leetcode.com/problems/valid-triangle-number/)

Networks - ( 1 Week )

[Leetcode](https://leetcode.com/problemset/all/?search=network)

* Network Topology, OSI Architecture
* TCP/IP models
* TCP and UDP
* Firewall, DNS, Domains, workgroups
* Protocols i.e ICMP

OS - ( 1 week )

[Operating System Tutorial](https://www.guru99.com/operating-system-tutorial.html)  
[Shared Memory Systems](https://www.youtube.com/watch?v=uHtzOFwgD74)

* Cache
* Multithreading
* Producers-consumers problem
* Dining philosophers problem
* Cigarette smokers problem
* Readers–writers problem
* [Web Crawler Multithreaded](https://leetcode.com/problems/web-crawler-multithreaded/)
* Scheduling algorithms
* Deadlock
* Virtual Memory
* Mutex and semaphore
* Kernels
* Paging

Software Design Principles - ( 2 weeks )

[System Design Primer](https://github.com/donnemartin/system-design-primer)

[Start learning about Theory of Distributed Systems?](https://cs.stackexchange.com/questions/4793/start-learning-about-theory-of-distributed-systems)

[Challenges with distributed systems](https://aws.amazon.com/builders-library/challenges-with-distributed-systems/)

[**Microservices Design Guide 👨‍🏫 - Platform Engineer**](https://medium.com/platform-engineer/microservices-design-guide-eca0b799a7e8)

[Cloud design patterns - Azure Architecture Center](https://docs.microsoft.com/en-gb/azure/architecture/patterns/)

[Design patterns for microservices | Azure Blog and Updates](https://azure.microsoft.com/en-us/blog/design-patterns-for-microservices/)

**TO READ:**

**Domain Driven Design (DDD) | Bounded Context (BC) | Polyglot Persistence (PP)| Command and Query Responsibility Segregation (CQRS) | Command Query Separation (CQS) | Event-Sourcing (ES) | CAP Theorem | Eventual Consistency | Twelve-Factor App | SOLID Principles |**

**Just some things to focus on.**

* Load balancer
* API gateway
* Microservices - Scale Cube Concept, MVC - READ
* Database Sharding
* SQL vs NoSQL - Cassandra, Postgres, Hadoop, Data lake, other algorithms related to data lake, CAP Theorem

**Notes from Jason**

I also recommend keeping a log of all leetcode questions you've been solving, as well as how many attempts you have made. Here's mine for example

[**https://docs.google.com/spreadsheets/d/14-jhQIt5uXkt5DzX5aHdqnKpAnJmIO9e\_yJ-1Z94sKY/edit#gid=0**](https://docs.google.com/spreadsheets/d/14-jhQIt5uXkt5DzX5aHdqnKpAnJmIO9e_yJ-1Z94sKY/edit#gid=0)