TODO

1. Quick sort
2. Binary Search
3. Merge sort
4. Heap sort
5. Radix sort
6. Balanced binary Tree
7. Implement Heap
8. BST delete

Sorting and Searching

1. Binary Search
2. Selection Sort
3. Bubble Sort
4. Insert Sort
5. Merge Sort
6. Heap Sort
7. Quick Sort/Quick select
    - Nuts and bolts problem
    - Median finding
8. Radix Sort
    - Given array of integers, take each integer mod 10 divide 1. Put into linked list.
    - Then consider linked lists in order.
    - Then repeat for mod 100 divide 10.
    - Repeat his process util divide by biggest power of 10 in array.
    - Sort n numbers from range 0 to 2^n-1 in linear time
9. Counting Sort
10. Bucket Sort
11. Shell Sort

Greedy Algorithms

1. Activity Selection: You are given n activities with their start and finish times. Select the maximum number of activities that can be performed by a single person, assuming that a person can only work on a single activity at a time.
2. Kruskal's Minimum Spanning Tree Algorithm:
    1. Sort all edges in increasing order of weight

    2. Pick smallest edge, check if it forms a cycle using the Union-Find algorithm. If not, add edge.
    3. Repeat until there are v-1 edges
3. Huffman Coding: TODO
4. Prims Algorithm for MST
5. Dijkstra's Shortest Path
6. Job Sequencing Problem: Given an array of jobs where every job has a deadline and associated profit if the job is finished before the deadline
    - Sort jobs by decreasing profit and pick jobs that can fit

Dynamic Programming

1. Longest Increasing Subsequence
    - O(n^2) time complexity
    - Store longest increasing subsequence so far
2. Edit distance
    - TODO
3. Longest increasing subsequence
    - L(A, B) = Max(L(A[:n-1], B[n-1])) + 1 if A[n] == B[n]
    - Max(L(A[:n-1], B[:n]), L(A[:n], B[:n-1]))

Pattern Search

1. Naive: consider each point as the starting point
2. KMP Algorithm:
    - Build longestPrefixSuffix Array by setting i = 0, j = 1. If i == j, store i+1 in LPS and increment i and j. If i != j, increment j, store 0 at position j in LPS Array or make i = LPS[i-1], and check again.
    - Iterate through array (j) and pattern array (i). If mismatch, then LPS[i-1] is the next point of comparison in pattern array.
3. Rabin-Karp
4. Suffix Array
5. Anagram substring: keep counts of each letter in hash, keep running window count. If same, anagram found.
6. Longest Palindrome Substring: Manacher's algorithm

Backtracking

1. Print all permutations
2. Knights Tour
3. Rat in the Maze
4. N Queens
5. Subset sum
6. M Coloring Problem
7. Sudoku
8. Generate two subsets of equal size with minimum difference

Divide and Conquer

1. Find median of two sorted arrays.
    - Take median of both arrays: med1, med2.
    - If med1 < med2, look for median in right half of arr1 and left half of arr2.
    - Repeat til only 4 elements left.
2. Counting Inversions: merge sort with counter
3. Closest pair of points
    - Sort by x coordinate.
    - Split plane into two halves
    - Find min = min(min_left, min_right)
    - Consider all points distance min from split point
    - min = min(min, min_split)

Bit stuff

Graphs

1. Topological Sort: Recursive call to sort for all children before processing this element
    - TODO
2. Bipartiate Graph: BFS
3. Dijkstra's Shortest Path
    - At each step, add vertex from unincluded set that has minimum distance
4. Minimum Spanning Tree
5. Given an array of strings, can the strings chain to form a circle.
    - For each string, take first and last character and add a edge

- If eulerian circuit exists, then true.
- Eulerian circuit if single strongly connected component and in degree = out degree.

6. Given sorted order of words in an alien language, figure out order of letters in the language.
    - Make a graph where edges are when first character mismatch.
    - Find topological sorting
7. Shortest chain of words to reach a target word: BFS
8. Find same contact in a list of contacts: find connected components.

Union Find