Pending's Bucket: Data Structures and Algorithms

Theory

This section covers theoretical concepts related to data structures and algorithms that need to be studied or revisited for better understanding.

- **Data Structures**
  - Types of data structures: Linear vs. non-linear data structures
  - Arrays: Homogeneous vs. heterogeneous arrays, jagged arrays, sparse arrays, multidimensional arrays
  - Linked Lists: Singly linked list, doubly linked list, circular linked list, applications, advantages, and disadvantages
  - Memory Management: Stack vs. heap memory, static vs. dynamic memory allocation, virtual memory, memory pool, memory leaks, garbage collection
  - Hierarchical data structures
  - Contiguous vs. non-contiguous data structures
- **Algorithms**
  - Binary search vs. linear search: Concepts, differences, and limitations
  - Recursion: Direct vs. indirect recursion, tail vs. head recursion, binary recursion, applications, advantages, and disadvantages
  - Complexity Analysis: Big-O, Big-Theta, Big-Omega notations, asymptotic analysis, time and space complexity calculations
  - Logarithmic functions: Understanding log n, n log n, and their differences
  - Algorithm design: Brute force, sliding window, two-pointer, fast and slow pointer techniques
- **Strings**
  - Character encoding: ASCII, UTF-8, control characters
  - String properties: Mutable vs. immutable strings in JavaScript and Python
  - String operations and their applications
- **Miscellaneous**

  - Memory allocation in arrays and linked lists
  - Stack overflow and heap overflow
  - Dynamic typed languages
  - Guidelines for writing algorithms
  - Git concepts: Clone vs. fork, fetch vs. pull, checkout, log, stashing, conflicts, pull requests, cherry-pick, revert, rebase
  - Networking: Load balancing, DNS, SSL, TLS, HTTP vs. HTTPS, Nginx architecture

Practicals

This section lists practical exercises and coding tasks to be implemented or practiced, focusing on data structures and algorithms.

- **Array Operations**
  - Find the second largest element in an array (handle negative numbers)
  - Find the third largest element in an array without sorting
  - Find the kth largest element in an array
  - Reverse each word in a string (e.g., "HELLO WORLD" to "OLLEH DLROW")
  - Find the frequency of each number in an array
  - Find the average of even numbers in an array
  - Find the last occurrence of an element in a sorted array with duplicates
  - Find the minimum in a sorted rotated array (binary search)
  - Find combinations of two numbers summing to a target (e.g., 4 or 5)
  - Flatten a multidimensional array
  - Find the subarray with the maximum number of elements in increasing order
  - Remove a specific element from an array
  - Two sum problem (LeetCode)
  - Product except self
- **Linked List Operations**
  - Implement singly, doubly, and circular linked lists
  - Reverse a singly linked list
  - Reverse a doubly linked list
  - Delete a node from a specific position in a singly or doubly linked list
  - Delete the nth node from the end of a linked list (two-pointer)

  - Find the middle element of a linked list in one iteration (fast and slow pointer)
  - Detect a cycle in a linked list (Floyds algorithm)
  - Remove duplicates from a linked list without extra data structures
  - Convert an array to a linked list (e.g., [1, 2, 3, 4, 5])
  - Merge two sorted linked lists
  - Add a node after a specific data value in a doubly linked list
  - Remove all nodes with a specific value from a singly linked list
  - Convert a singly linked list to a doubly linked list
  - Convert a linked list to a circular linked list and validate

- **String Operations**
  - Check if two strings are anagrams
  - Check if parentheses are balanced (e.g., "" true)
  - Reverse a string without using built-in methods
  - Find the first non-repeating character in a string
  - Find the longest substring palindrome
  - Find the longest substring without repeating characters
  - Find the longest substring without vowels
  - Recursively remove a character from a string (e.g., hide "l" from "hello")
  - Implement string permutations
  - Ensure a string ends with a period
  - Convert PascalCase to snake$_{c}aseFindtheshortestwordinastring$

- Count words in a sentence without built-in functions

- **Recursion**
  - Implement Fibonacci series (first 10 elements) using recursion
  - Calculate factorial using recursion
  - Sum of array elements using recursion
  - Binary search using recursion
  - Reverse a string using recursion
  - Remove even numbers from an array using recursion
  - Recursion that recurses only 5 times
  - Sum of even numbers using recursion

- **Binary Search**

    - Implement binary search (iterative and recursive)
    - Replace a number with 0 using binary search
    - Find the minimum in a sorted rotated array
- **LeetCode Blind 75 Practice**
    - Two Sum
    - Buy and Sell Stock (sliding window)
    - Remove Nth Node from End of List (two-pointer)
    - Find Minimum in Rotated Sorted Array (binary search)
    - Longest Substring Without Repeating Characters
    - Merge Two Sorted Lists
    - Valid Parentheses
    - Detect Cycle in a Linked List (fast and slow pointer)

Results

The following summarizes the status of the pending topics and tasks:

- **Completed Topics**: Basic recursion, factorial using recursion, and some linked list operations (e.g., singly linked list creation) have been covered.
- **Partially Completed**:
    - Delete kth element from the end of a linked list
    - Find the longest substring without vowels
    - Remove odd element nodes from a linked list
    - Insert to a doubly linked list
- **Pending Tasks**:
    - Complete implementation of doubly linked list delete function
    - Deepen understanding of asymptotic analysis (Big-Theta, Big-Omega)
    - Practice more problems from Blind 75 LeetCode with brute force and optimal solutions
    - Revise memory management concepts (virtual memory, memory pool, memory leaks)
    - Explore applications of doubly linked lists and recursion
    - Implement and validate circular linked list conversions
    - Solve string problems without built-in methods
- **Recommendations**:

- Refer to resources like NeetCode.io, FreeCodeCamp YouTube, and LeetCode for practical exercises.
- Focus on mastering two-pointer, fast-slow pointer, and sliding window techniques.
- Regularly revisit theoretical concepts to ensure clarity.