Arranged List of DSA Topics and Questions (Beginner to Advanced)

Beginner Level

These topics introduce fundamental concepts of data structures, algorithms, and programming basics, requiring minimal prior knowledge.

1. **Concepts of Data Structures**
    - Data structure
    - Types of data structures
    - Linear vs non-linear data structures
2. **Basic Programming Concepts**
    - Algorithm
    - Why need algorithm
    - Primitive vs non-primitive data types
    - Dynamic typed language
    - Statically typed vs dynamically typed
3. **Strings: Basic Operations**
    - Concepts of string
    - Strings are immutable in Python
    - Is string mutable or immutable in JavaScript?
    - Reverse a string
    - Title Case words in a string
    - Function to make sure sentence begins with uppercase and ends with period
    - Hello World to olleH dlroW convert string problem
4. **Arrays: Basic Concepts**
    - Concepts of array
    - Array operation complexity
    - Time complexities for common array operations (accessing, inserting, deleting)
    - Reverse an array
    - Reverse an array with O(n) complexity
    - Sum of elements in an array using recursion
    - Create a function to find the average of even numbers in an array
5. **Memory Basics**

- Memory allocation
- Types of memory allocation
- Static vs dynamic memory allocation
- Advantages of static memory allocation
- Disadvantages of dynamic memory allocation
- Stack vs heap memory

6. **Character Encoding and Control Characters**
    - Character encoding
    - Control characters
    - Escape sequences
    - ASCII
    - UTF-8
    - \t
7. **Asymptotic Analysis: Introduction**
    - Complexity analysis
    - Asymptotic analysis
    - Big-O notation
    - Common notation used to represent time complexity
    - What is O(log n) time complexity?
    - Best case in asymptotic analysis
8. **Recursion: Basics**
    - Recursion
    - Base case in recursion
    - Recursion vs loop
    - Advantages of recursion
    - Disadvantages of recursion
    - Applications of recursion

Intermediate Level

These topics build on beginner concepts, introducing more complex data structures, algorithms, and problem-solving techniques.

1. **Arrays: Intermediate Operations**

- Multidimensional array
- Jagged array
- Two-dimensional array
- Sparse array
- Flatten a multidimensional array
- Flatten nested list
- Find the second largest element in an array
- Fix logic for finding the second largest element to handle negative numbers
- Second smallest element in an array
- Find the third largest element in an array without sorting
- Find kth largest in an array
- Find the frequency of occurrence of each number in an array
- How to delete a specific element from an array
- What is a subarray?
- Subarray with max elements in continuously increasing order
- Convert an array [1, 2, 3, 4, 5] to a linked list
- Product except itself

2. **Strings: Intermediate Operations**
    - Check if two strings are anagrams
    - Find the first non-repeating character in a string
    - Longest word in a string
    - Find shortest word in a string
    - Reverse each word in a string
    - Reverse each word in a string without using built-in methods
    - Recursively remove a character from a string
    - Remove all occurrences of a specific character from a string
    - Hide "l" from hello using recursion
    - Extract numbers from strings
    - Extract digits from string (needs improvement)
    - Find largest substring without vowels from a string (not completed)
    - Longest substring without repeating characters

- String permutations
- Valid parentheses problem

3. **Linked Lists: Basic Operations**
    - Concepts of linked list
    - Linked list implementation
    - Singly linked list
    - Sorted single linked list
    - SLL implementation
    - SLL with tail (tail not utilized)
    - Applications of linked list
    - Advantages and disadvantages of linked list
    - Memory allocation in linked list
    - Print all elements of a linked list in order and reverse order
    - Time complexity of accessing element from linked list when index is known
    - Time complexity to insert element in linked list
    - Append & prepend in linked list
4. **Search Algorithms**
    - Linear search
    - Binary search
    - Binary search implementation
    - Binary search using recursion
    - Complexity of binary search
    - Time complexity of binary search
    - Worst-case time complexity of binary search
5. **Recursion: Intermediate**
    - Tail vs head recursion
    - Direct vs indirect recursion
    - Binary recursion
    - Tail recursion
    - Fibonacci series using recursion
    - Print first 10 elements of Fibonacci series

- Factorial using recursion
- Space complexity of recursion

6. **Asymptotic Analysis: Intermediate**
    - Big-Theta notation
    - Big-Omega notation
    - Theta vs Omega vs Big-O
    - Asymptotic notations for quadratic equations
    - Logarithmic values
    - n log n vs log n differences
    - O(n) vs O(log n)
7. **Memory Management**
    - Virtual memory
    - Memory leak
    - How to prevent memory leakage
    - Garbage collector
    - Garbage collection & working
8. **Linked Lists: Intermediate Operations**
    - Doubly linked list
    - Circular linked list
    - Circular doubly linked list
    - Difference between singly linked lists and doubly linked lists
    - How does a doubly linked list differ in traversal compared to a singly linked list
    - Applications of doubly linked list
    - Insert a node after a node (check all conditions)
    - Insert to a doubly linked list (incomplete)
    - Traverse a doubly linked list
    - Delete front and back node from where data == data in doubly linked list

Advanced Level

These topics involve advanced data structures, algorithms, and problem-solving techniques, requiring a strong foundation in DSA.

1. **Linked Lists: Advanced Operations**

- Reverse a singly linked list
- Reverse a doubly linked list
- Reverse nodes in a linked list
- Sort nodes in a linked list
- Remove duplicates from a linked list
- Remove the middle element from a linked list
- Delete middle element from linked list in O(n) time
- Delete an element from a specific position in a singly linked list
- Delete function for doubly linked list
- Remove last instance of a value from SLL
- Remove odd element nodes from linked list (not completed)
- Find middle of linked list in one iteration (fast and slow pointer)
- Get middle element in a linked list
- Delete nth node from the end of a singly linked list (two pointer)
- Detect cycle/loop in a singly linked list (Floyd's algorithm)
- Merge two sorted linked lists

2. **Advanced Search and Sorting**
    - Min from rotated sorted array (binary search)
    - Binary search tree (BST) implementation
3. **Advanced Algorithms and Techniques**
    - Two-pointer technique
    - Fast and slow pointer approach
    - Sliding window approach
    - Buy and sell stock (sliding window approach)
    - Two sum LeetCode problem
    - Palindrome code
4. **Advanced Asymptotic Analysis**
    - Time and space complexity calculations
    - Space complexity calculation
    - Time complexity in recursion
    - Space complexity

- Identifying time complexity of programs
- Determine how to find complexity of algorithms
- What is an algorithm’s best-case time complexity
- What is the time complexity of deleting an element from the middle of a linked list
- Find combination of two numbers given sum as 4

5. **Advanced Memory Concepts**
    - Memory pool
    - Stack overflow
    - Circular referencing
6. **Advanced Recursion**
    - Generator function to yield numbers from a 2D list
    - Recursion that recurses only 5 times
7. **Miscellaneous Advanced Topics**
    - Don’t overwrite built-in names
    - Hierarchical data structures
    - Multilinked list

Separate Section: Comparison ("vs") and Non-DSA Topics

These topics involve comparisons or are not directly tied to core DSA concepts but are important for your review.

Comparison Topics ("vs")

- Stack memory vs heap memory
- Nibble vs byte
- Kilobyte vs kibibyte
- Homogenous vs heterogeneous array
- Linear search vs binary search
- Linked list vs arrays
- Advantage of array over linked list
- Traditional arrays vs JavaScript arrays
- Theta notation vs Omega notation
- Big-Oh vs Big-Omega vs Big-Theta

Non-DSA Topics

- Typed arrays in JavaScript
- Map
- Derived data types
- Contiguous vs non-contiguous
- Mutable vs immutable
- Virtual memory relation with data structure