General Pattern

There are many patterns that can be recognized when solving data structures and algorithms (DSA) questions. Identifying these patterns by reading can help you solve problems more efficiently within your limited time. Over time and with practice, people start recognizing these patterns.

So, this time before diving into the answer, let's understand a few general patterns that you can use in your future journey:

Sorted Input:

- Apply binary search for efficient element lookup.
- Use the two-pointer technique for problems involving pairs or segments.

Unsorted Input:

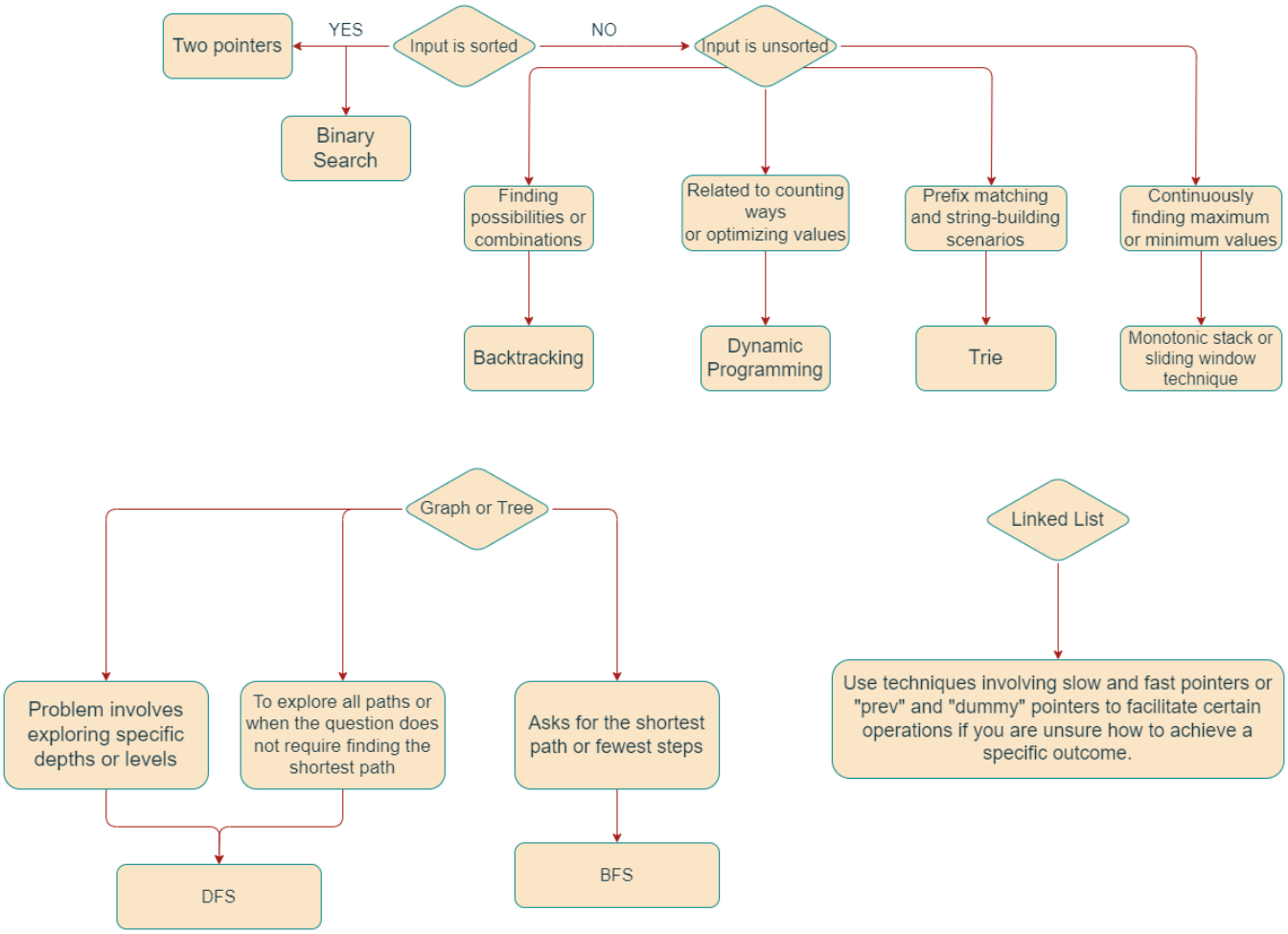
- Apply dynamic programming for questions related to counting ways or optimizing values.
- Use backtracking for problems that ask for all possibilities or combinations (this is also a suitable fallback if dynamic programming isn't going to work).
- Use a Trie for prefix matching and string-building scenarios.
- Use a hash map or set to find specific elements quickly.
- Implement a monotonic stack or sliding window technique for managing elements while continuously finding maximum or minimum values.

Input is a Graph or Tree:

- Use DFS to explore all paths or when the question does not require finding the shortest path.
- Use BFS when the question asks for the shortest path or fewest steps.
- For binary trees, use DFS if the problem involves exploring specific depths or levels.

Linked List Input:

- Use techniques involving slow and fast pointers or "prev" and "dummy" pointers to facilitate certain operations if you are unsure how to achieve a specific outcome.


Input is sorted
YES
Two pointers
Binary Search
NO
Input is unsorted
Finding possibilities or combinations
Backtracking
Related to counting ways or optimizing values
Dynamic Programming
Prefix matching and string-building scenarios
Trie
Continuously finding maximum or minimum values
Monotonic stack or sliding window technique
Graph or Tree
Problem involves exploring specific depths or levels
To explore all paths or when the question does not require finding the shortest path
DFS
Asks for the shortest path or fewest steps
BFS
Linked List
Use techniques involving slow and fast pointers or "prev" and "dummy" pointers to facilitate certain operations if you are unsure how to achieve a specific outcome.