Backtracking Template: The Three Key Questions

1.When do we stop?

2.When do we store the result?

3.How do we choose and turn back?

- How many choices can we choose?
- How to go into the next recursion after choosing?
- How to revoke the choice (turn back)?

Question Type Summary

The five most common categories is:

- Subset problems
- Permutation problems
- Combination problems
- String segmentation problems
- Board problems

	SUBSET	PERMUTATION	COMBINATION	STRING SEGMENTATION	BOARD
Question Key Information	Find all valid subsets	Find all permutations of elements	Find all valid combinations of k elements summing to target	Given a string, find all valid ways to split it	Place elements or search paths on a 2D board/grid under constraints
Analysis	- Usually no strong constraints - Each element is either chosen or not - Order doesn't matter	- Each element used once - Order matters - All elements are selected eventually	- No repetition - Order doesn't matter - May include sum/length constraints	- Each substring must satisfy given rules (e.g., in dictionary / palindrome / valid IP) - Splits must be contiguous - May include constraints on count, length, or validity	- Positions must not conflict (e.g., queens/sudoku) - Paths may not revisit; limited directions
State definition	`path` + current `startIndex`	`path` + `visited[]` flag array	`path` + current `startIndex`	`path` + current starting index (`startIndex`)	Board state (`board`) + current level (e.g., row number or position)
When do we stop?	When `startIndex` reaches the end of array	When `path.length` equals input array length	When `path.length` satisfies given condition (e.g., equals k)	When `startIndex` reaches end of string	When all elements are placed or board is fully filled
When do we store?	Every step can potentially add a result (if valid)	Store when `path.length` equals input array length	Store when path meets conditions (e.g., length == k)	Store when we reach the end and all segments are valid	Store when a valid final state is reached (e.g., N queens placed)
How many choices can we choose?	Remaining elements from `startIndex` onward	All unused elements	Remaining elements from `startIndex` onward	All substrings starting from `startIndex` to the end	All valid positions or values at current state
How to go into	Add current	Add to path and	Add current	Add substring to path	Modify board state

the next recursion after choosing?	element, increment `startIndex + 1`	mark as used	element, `startIndex + 1`	and move `startIndex` to split end	(e.g., place queen), go to next row/position
How to revoke the choice (turn back)?	Remove last element from `path`	Remove last element + reset `visited[i]` to false	Remove last element from `path`	Remove last substring, restore `startIndex`	Remove element from board and restore board state