216. Combination Sum III

recursive + backtracking approach

dfs (start, targetsum) → function

check for → current combination is valid

and
- k elements are present
- targetSum = 0 (no more number needed)

True → add combination → return

check again (independendly)

or
- number of items ≥ k (can't add more)
- start > 9 (can't use greater than 9)
- start > targetsum (current num exceeds the sum)

True → return

if none of the returns triggered

$\boxed{1.}$ add start to current combination

$\boxed{2.}$ recall dfs$(\underbrace{\text{start}+1}_{\text{move to next sum}}, \underbrace{\text{targetsum}-\text{start}}_{\text{look for remaining sum}})$

$\boxed{3.}$ remove the last element from combination.

↳ backtracking starts, if one of the conditions do not match to proceed, returns will be triggered and 3rd line will be executed after 2nd.

$\boxed{4.}$ recall the dfs$(\underbrace{\text{start}+1}_{\text{same number}}, \underbrace{\text{targetsum}}_{\text{removed the last number}})$

↘ ↙

looking for a new combination

Example Walkthrough

$\boxed{n = 7,\ k = 3}$

$$\begin{tikzcd}
\text{dfs}(1,7) \arrow[r] \arrow[d, "②"] & \text{comb} = [1] & \\
\text{dfs}(2,6) \arrow[r] \arrow[d, "②"] & \text{comb} = [1,2] & \\
\text{dfs}(3,4) \arrow[r] \arrow[d, "②"] & \text{comb} = [1,2,3] & \\
\text{dfs}(4,1) \arrow[r, bend left, "③"] \arrow[r, phantom, "\text{second if returned}"] & \text{comb} = [1,2] \arrow[dl, "④"'] & \\
\text{dfs}(4,4) \arrow[r] \arrow[d] \arrow[r, phantom, "1+3 \text{ added back}"', near start] & \text{comb} = [1,2,4] & \\
\text{dfs}(5,\overset{*}{0}) \arrow[r] & \text{first if returned} \arrow[r] & \text{combination added to result}
\end{tikzcd}$$