

Longest common prefix

To find the longest common prefix among a list of strings, here's a structured approach with detailed guidance:

Strategy: Vertical Learning (character by character).

This is one of the most intuitive and efficient methods.

Idea:

1. Pick the first string as a reference.

2. For each character position i in the reference string:

- Compare it with the character at position i in all other strings.

- If any string is too short, or a character doesn't match, stop.

3. Return the prefix built so far.

Example:

For ["flower", "flow", "flight"]

- compare 'f' at index 0 in all $\rightarrow \checkmark$

- compare 'l' at index 1 $\rightarrow \checkmark$

- compare 'o' at index 2 \rightarrow "flight" has 'i' $\rightarrow \times$ Stop

Return "fl"

Edge Cases:

- empty input \rightarrow return ""

- only one string \rightarrow return it

- one string is empty \rightarrow return ""

- no common prefix at all \rightarrow return ""

Pseudocode Outline:

if strs is empty:

return ""

prefix = strs[0]

for each string in strs[1:]:

while current string doesn't start with prefix:

remove last char from prefix

if prefix becomes empty:
return ""

return prefix

This method avoids full character-by-character comparison and uses the "start with" logic efficiently.