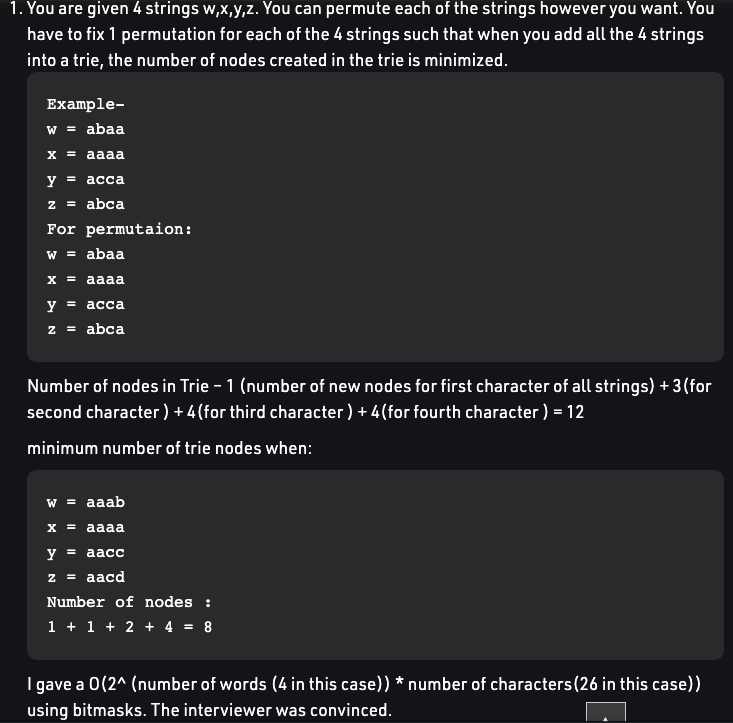
**Problem :**

****

**Approach :**

-> Initially we take what we can take common from all 4 strings.Then one of the strings has to split and go into a new path. We have to try all possibilities here.So now all the remaining nodes in the split string can’t be merged with others.

-> Same way now 3 strings will be left. So take whatever common we can take here and now one of the strings has to split and go to a new path and so on…

-> **So in 1st iteration :(bitmask 1111)**

No of nodes in trie added to answer at this instant =

No of nodes in selected split string, which are not common in all 4.

**No need to worry about nodes which are common as they will be definitely added later,as they are common on all strings.**

**-> 2nd iteration : (one of possible bitmask 0111) [1st string split]**

No of nodes in trie added to answer at this instant =

No of nodes in selected split string, which are not common in all remaining 3.

**-> 3nd iteration : (one of possible bitmask 0011) [1st string split]**

**[base case when only 2 bits left]**

No of nodes in trie added to answer at this instant =

All non common letters in both

+all common letters in both (added once)

**-> If we reach 0011 through path 1011 or 0111 , it doesn’t make any difference, the answer for 0011 subproblem is the same.**

**So for 2^4 unique problems , we need to check freq of all 26 letters.**

**So time = O(2^4 \* 26).**