Description: Are similar to trees DS, the difference is Tries can store up to 26 children in case we want to store words like a dictionary.

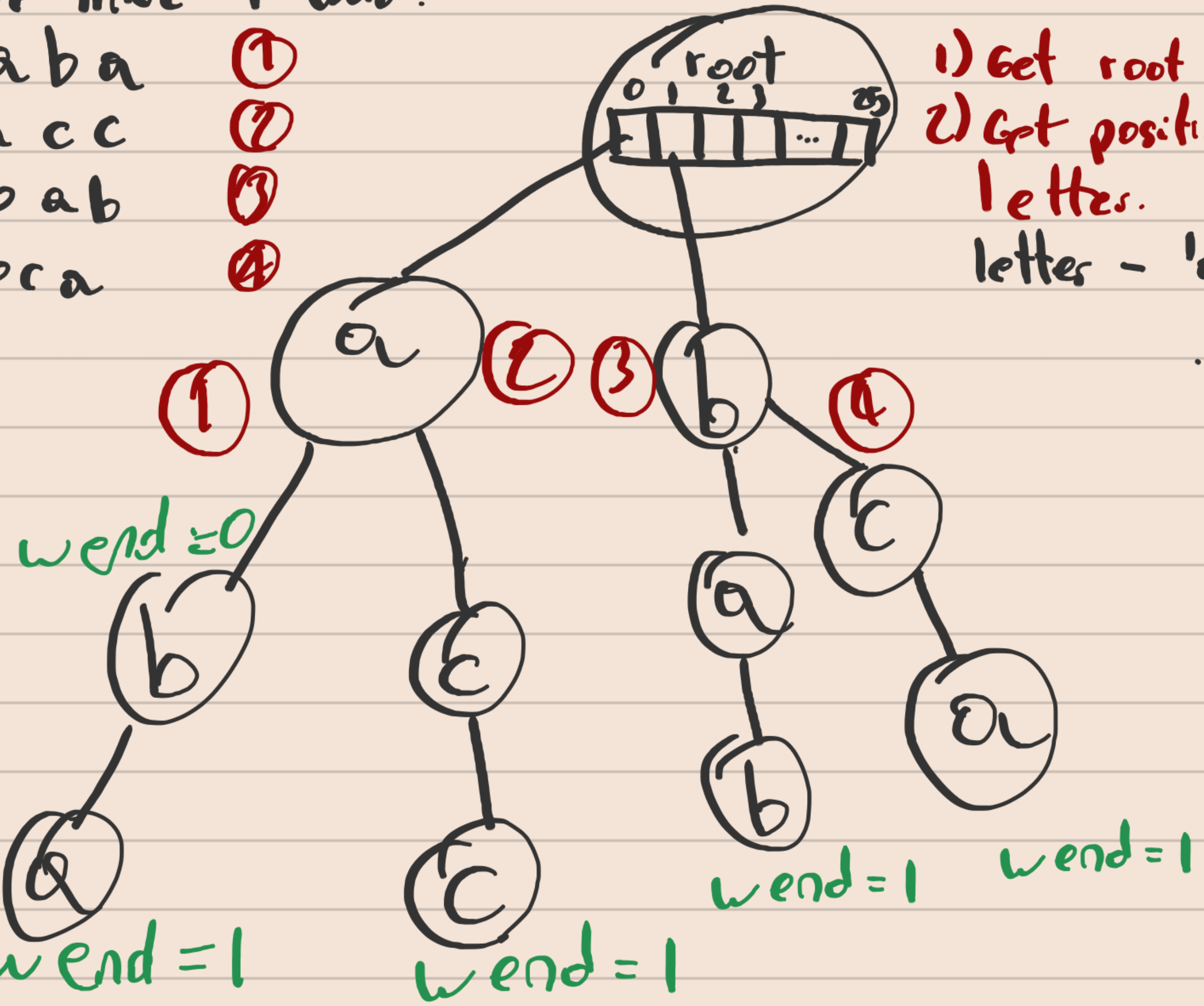
```
struct trienode
{
        char data;                    ①
        trienode *child[26]           ②
        int wordend                   ③
```

1) is the char stored
2) can create 26 children, total of alphabet's letters.
3) bool data type, if is the end of a word it's true.

ex:

we have 4 word:

aba ①
acc ②
bab ③
bca ④



1) Get root node.
2) Get position of letters.
letter - 'a'

Insertion:

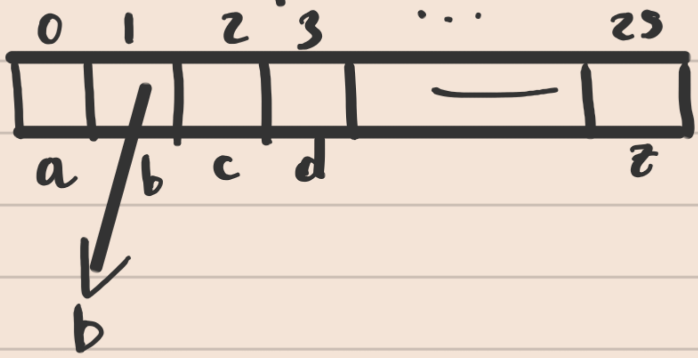
The function gets the string, then read it char by char calculating the relative position:

"bca"

↓

$98 - \text{'a'}(97) = \underline{\underline{1}}$

relative position = index in array's child.



when it reach the last char, the word end var. take a true (or 1 in int) value.