spelling

word → valid words

$O(n * l) * q$

Autocomplete

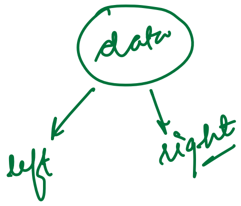
Trie → N-ary Tree

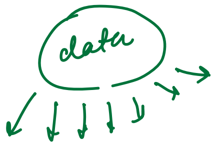
└ prefix-tree

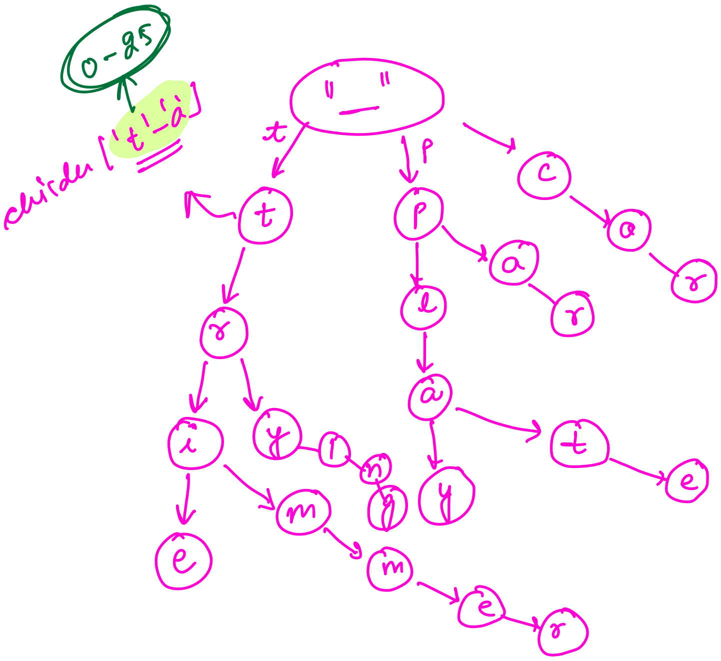
→ information retrieval

Tree like data-structure which stores the data from top to bottom

Trie	Try	Trim	play
plate	car	par	trimmer
trying	pla		







```
class Node {
    char data;
    Node children[26];
}
```

bool isEnd

false;

null

'a' — 0 — 'a'-'a'

'b' — 1 ← 'b'-'a'

'c' — 2 — 'c'-'a'

'd' — 3

search(teap) : Traverse your string & for every character traverse your trie

(tri)

(pla)

you don't find any one of the character ↓ stop Not found

find all characters

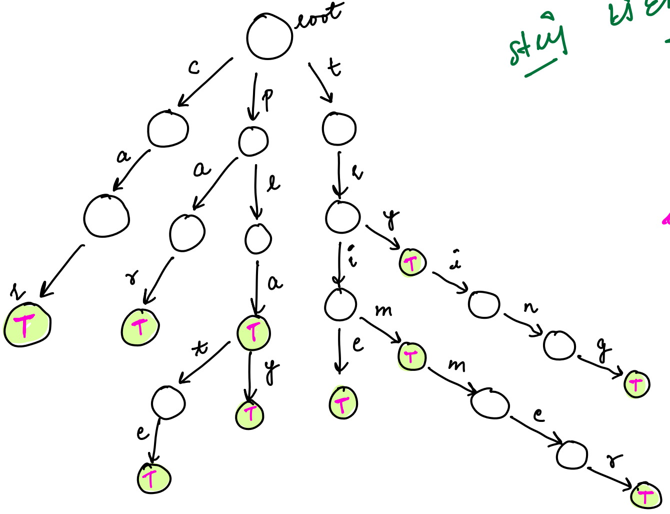
the last characters of the string is isEnd is false

isEnd=true

tr

⇑

autocomplete



spell

search ⇒ T.C: O(l)

S.C: n*l*26

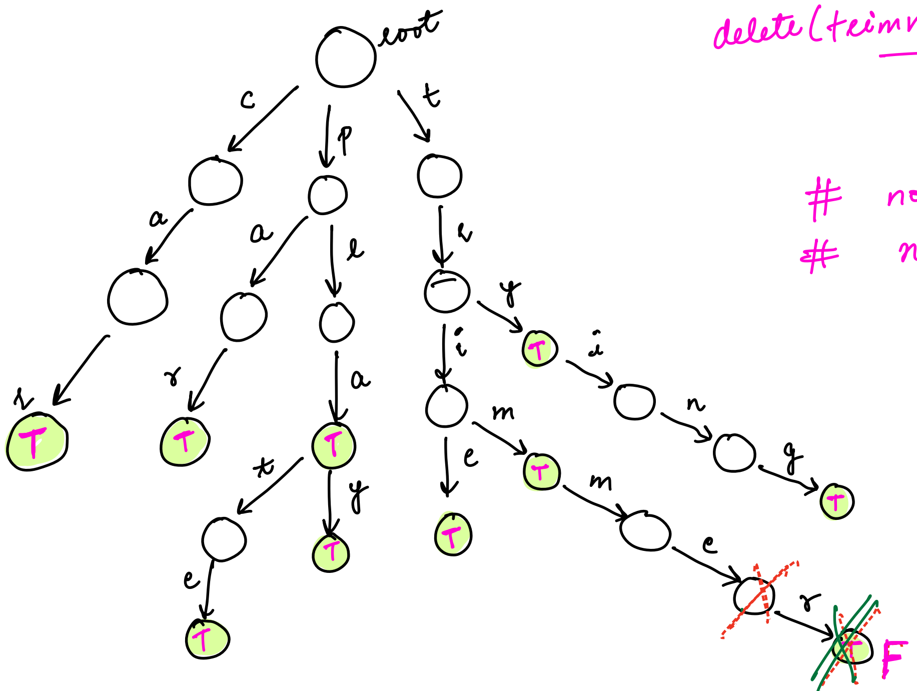
word is present multiple times ?

```
                    class Node{
                        char data;
                        bool isEnd;     Int freq;
                        Node childen[26];
                    }

        void insert( root, word)
        {
                curr = root;
                int l = word.length();
                for( int i=0; i<l; i++)
                {
                    int idx= word[i]-'a';
                    if( curr.childen[idx] ==null)
                    {
                    curr.childen[idx]  = new Node( word[i]);
                    }
                    curr= curr.childen[idx];
                }
                curr.freq++;        //  curr.isEnd = true;
        }
```

```
bool search( root, word)
{
        curr = root;
        int l = word.length();
        for( int i=0; i<l; i++)
        {
            int idx= word[i]-'a';
            if( curr.children[idx] ==null)
                  return false;
            curr= curr.children[idx];
        }
        if( curr.isEnd) return true; ⟸ curr.freq>0
           return false;
}
```

deletion



e.children['r'-'a']=null

delete(trimmer)

node completes a word

nodes has more children

stack r

e

m

m

i

r

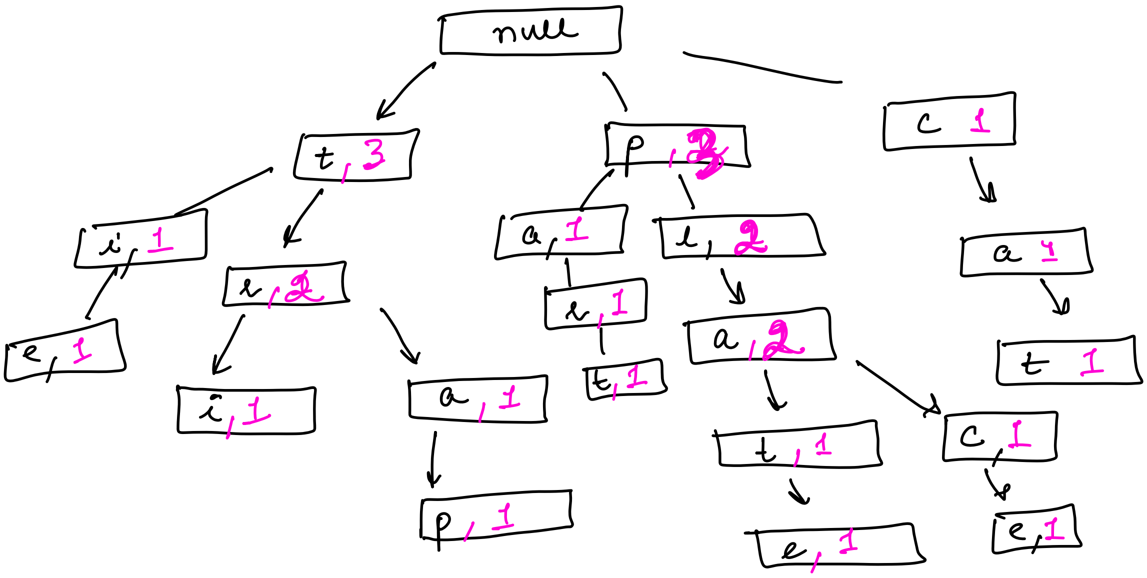
t

only store that node which is definitely not going to be deleted.

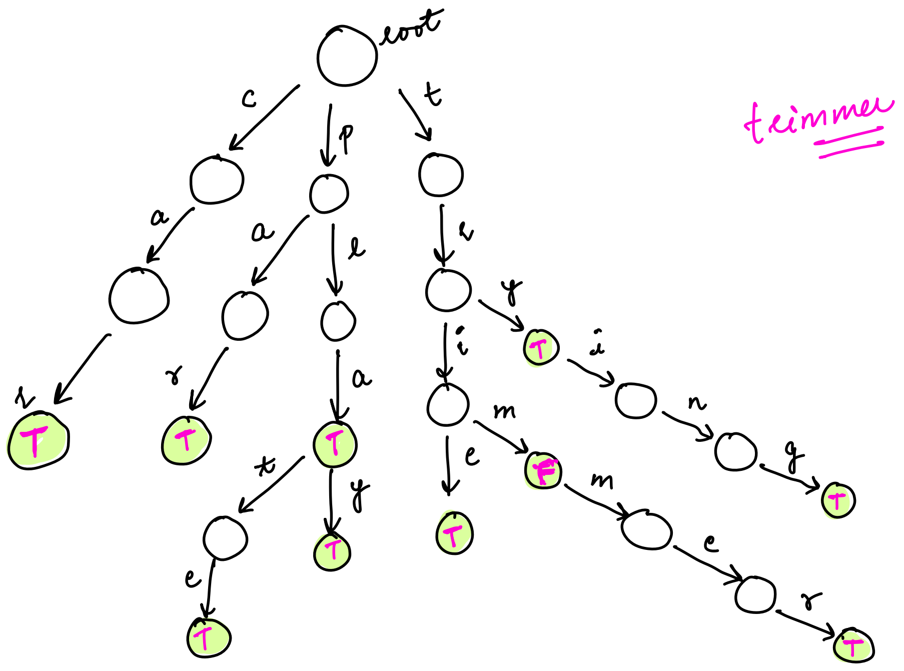
Q Find shortest unique prefix to represent each word.

Note: Assume that no word is prefix of another
In other words, the representation is always possible.

tri	trap	plate	cat	part	place	tie
⇓	⇓	⇓	⇓	⇓	⇓	⇓
tri	tra	plat	c	pa	plac	ti



# of a string is prefix in how many differ words


root
c
p
t
a
a
l
r
r
r
a
y
i
i
m
n
g
e
t
y
e
m
e
r
T
F
trimmer