Backtracking

Video - 15

Leetcode - 140

Hard → You will say, it was easy...

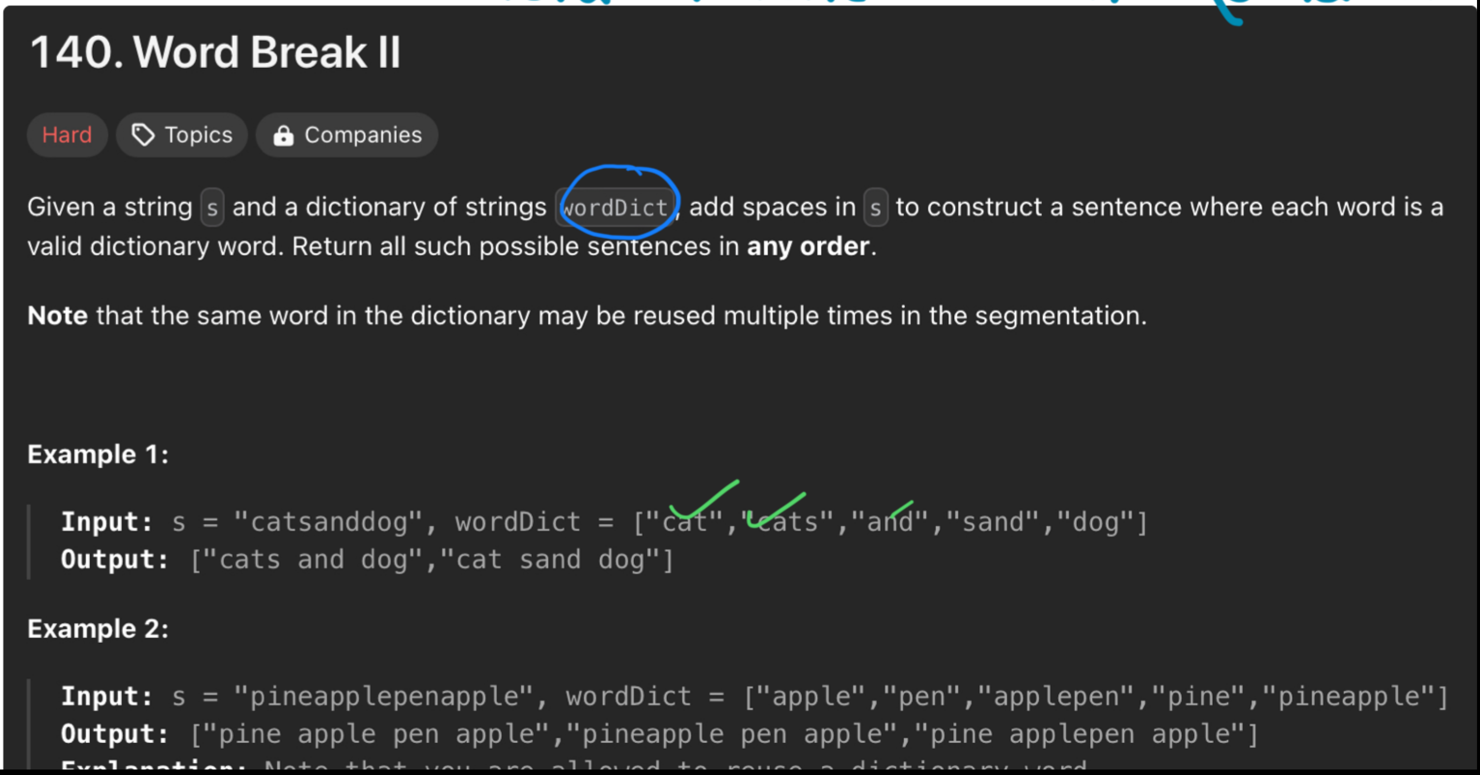
→ @codestorywithmiK

(Instagram, Facebook)

CSwithMIK → Twitter

→ codestorywithMIK

Meta Phone Screen Round ...



140. Word Break II

Hard | Topics | Companies

Given a string `s` and a dictionary of strings `wordDict`, add spaces in `s` to construct a sentence where each word is a valid dictionary word. Return all such possible sentences in **any order**.

Note that the same word in the dictionary may be reused multiple times in the segmentation.

Example 1:

```
Input: s = "catsanddog", wordDict = ["cat","cats","and","sand","dog"]
Output: ["cats and dog","cat sand dog"]
```

Example 2:

```
Input: s = "pineapplepenapple", wordDict = ["apple","pen","applepen","pine","pineapple"]
Output: ["pine apple pen apple","pineapple pen apple","pine applepen apple"]
Explanation: Note that you are allowed to reuse a dictionary word.
```

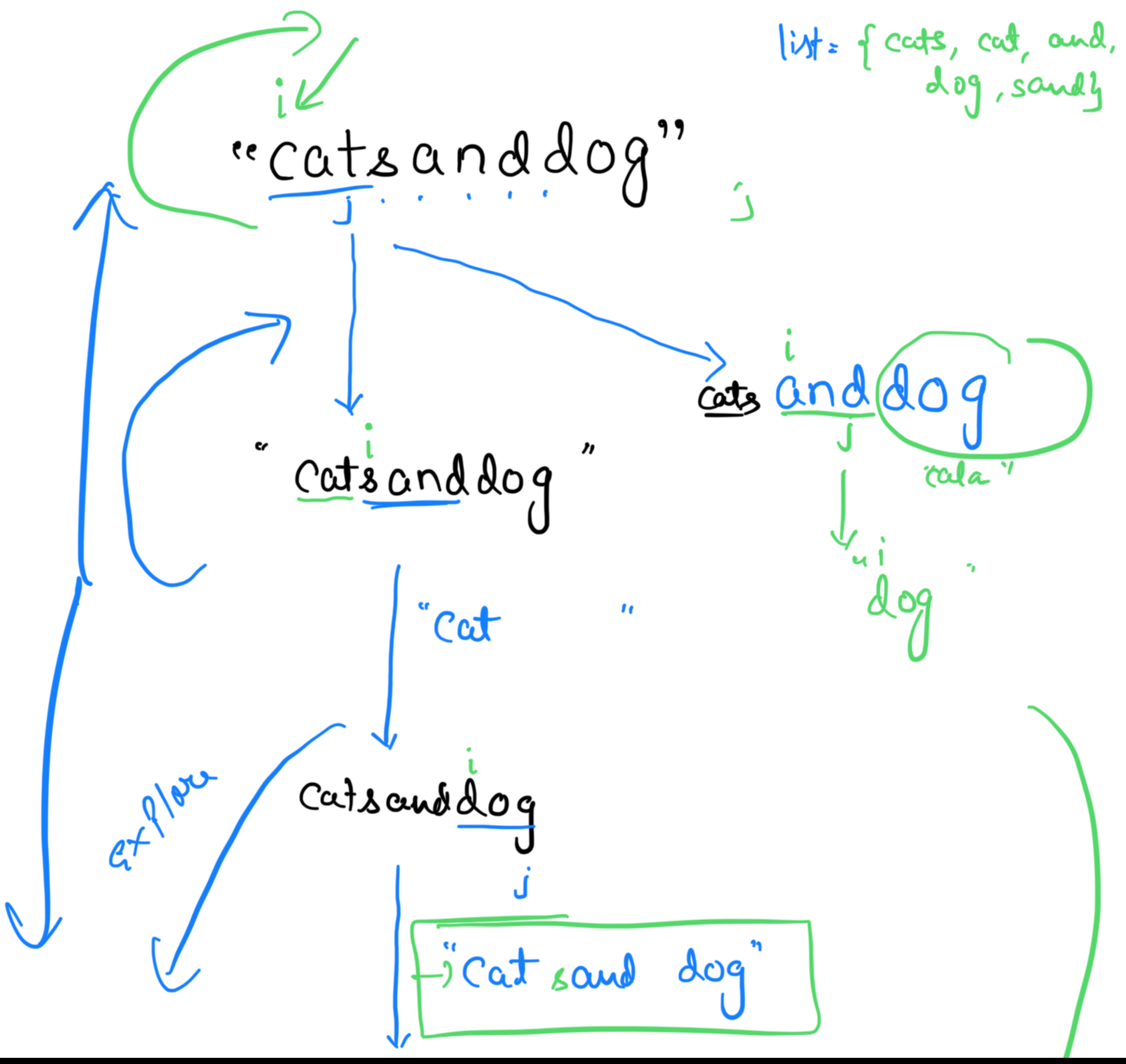
Explanation: Note that you are allowed to reuse a dictionary word.

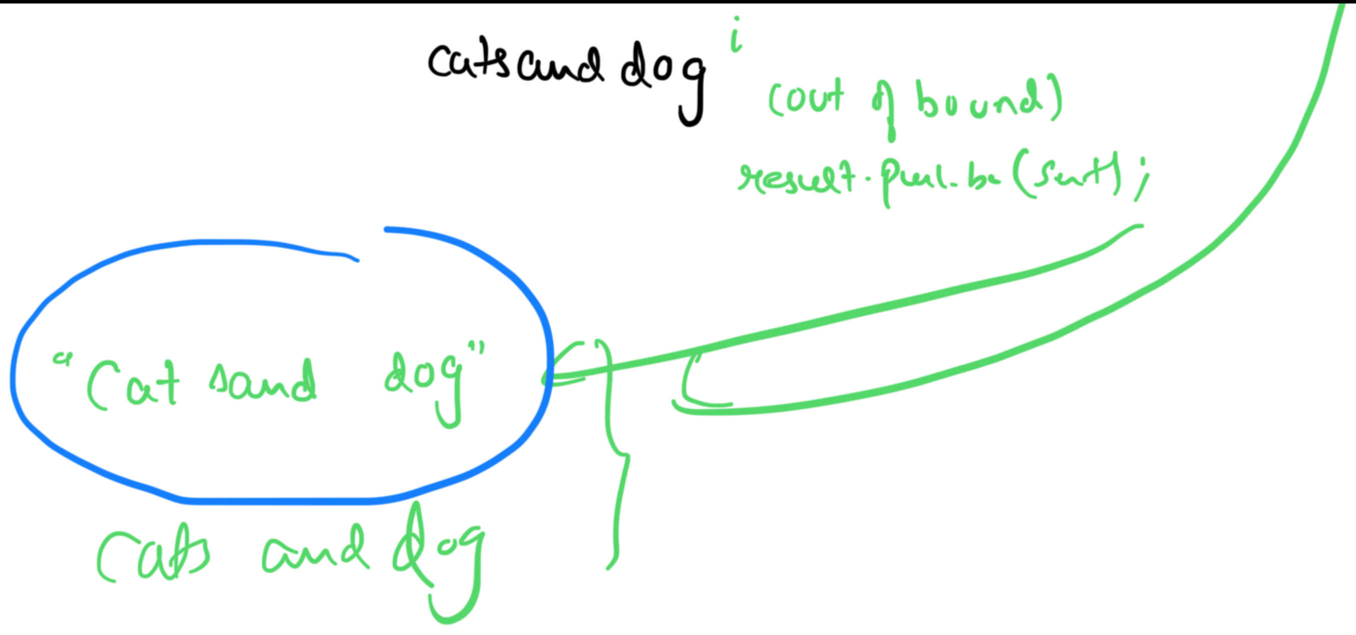
Example 3:

```
Input: s = "catsandog", wordDict = ["cats","dog","sand","and","cat"]
Output: []
```

Solve(0, s);

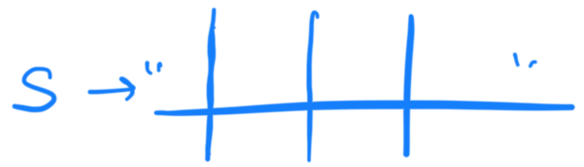
Thought Process





```
for(j = i;   j<n ;  j++){
    ~~word~~ = s.substr( i, j-i+1);//s[i:j]
    if(validword){
      => sentence += word
      => solve(j+1 , s     ) ; <=
    } => undo//.
}
```

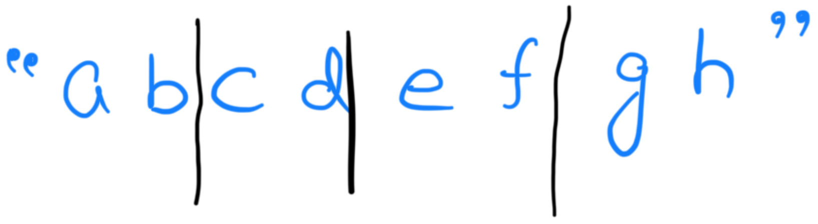
do
expl
und.



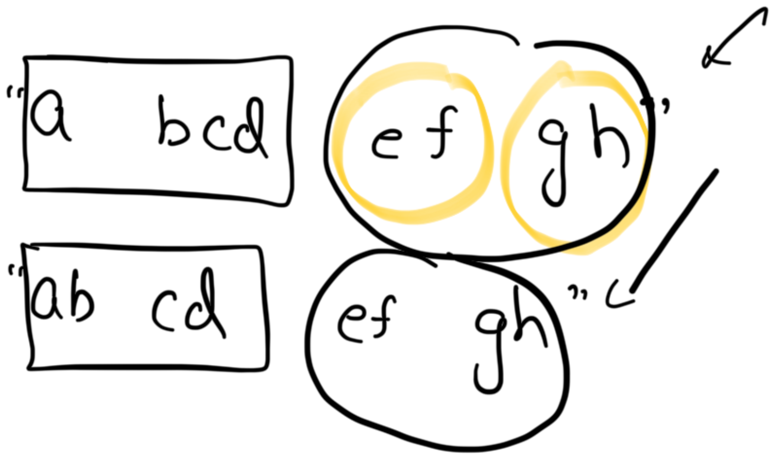
$T.C = O(2^n)$ Possibilities.

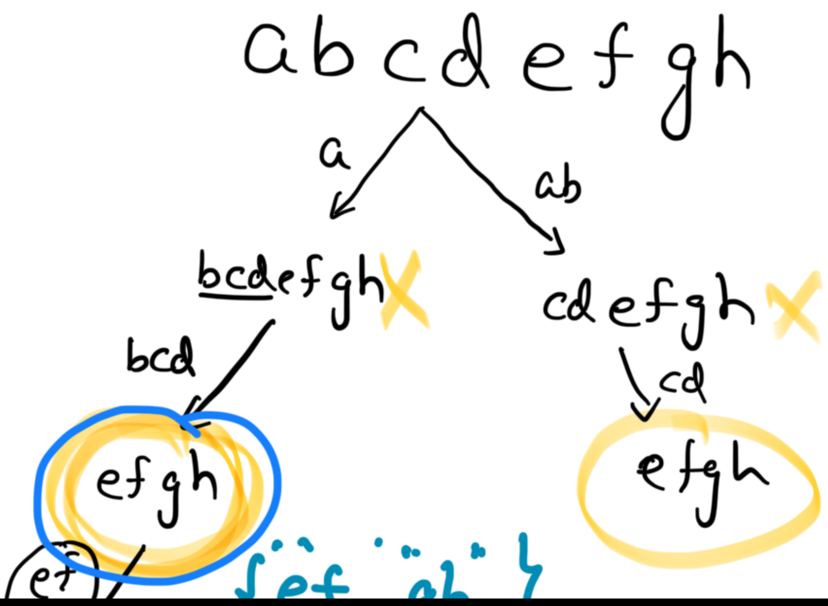
$S.C = O(n) + O(2^n).$

Can we try memoizing
Approach-1...



list = { "a" , "bcd" , "ef" , "gh" , "ab" , "cd" }

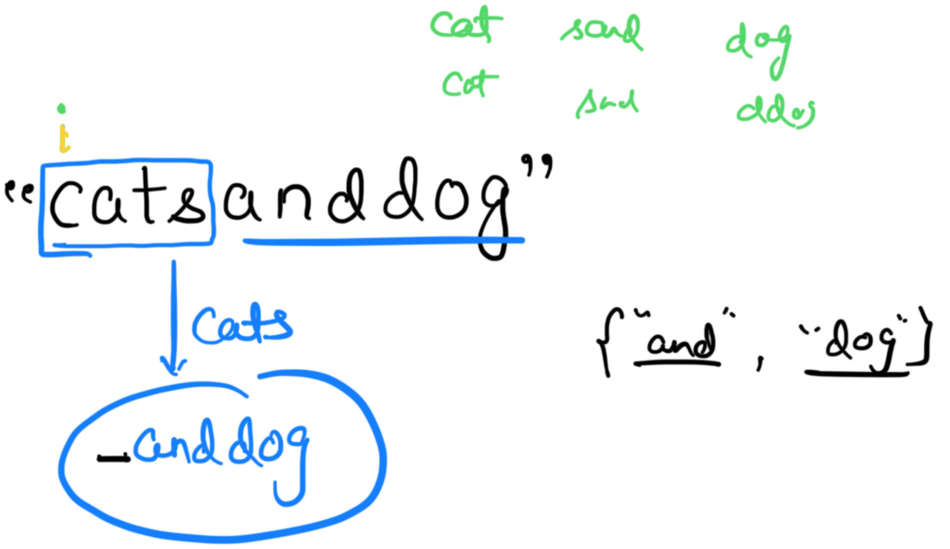




$$\text{Solve}(\overset{\text{int}}{i},\ \overset{\text{string}}{\text{currSentence}},\ s)$$

$(\text{int}, \text{string}) \longrightarrow$

Another Simple Approach



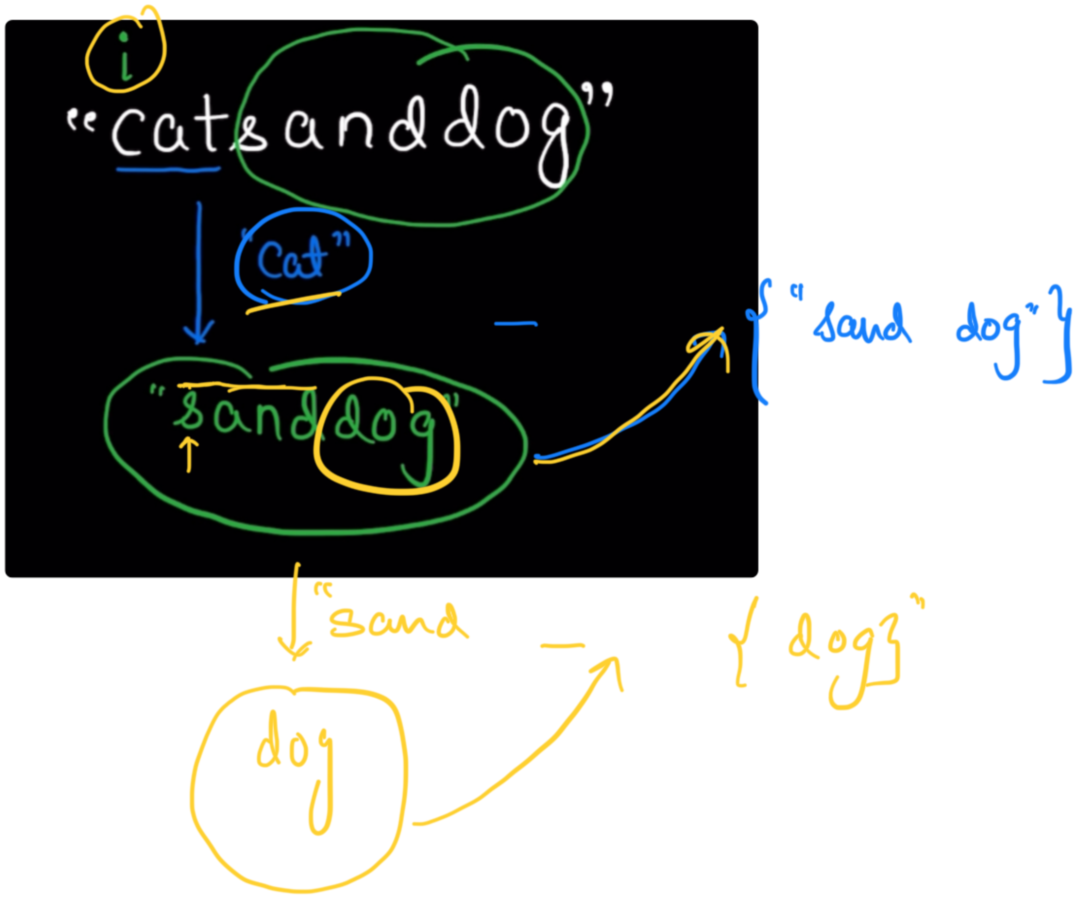
Recursi

Re

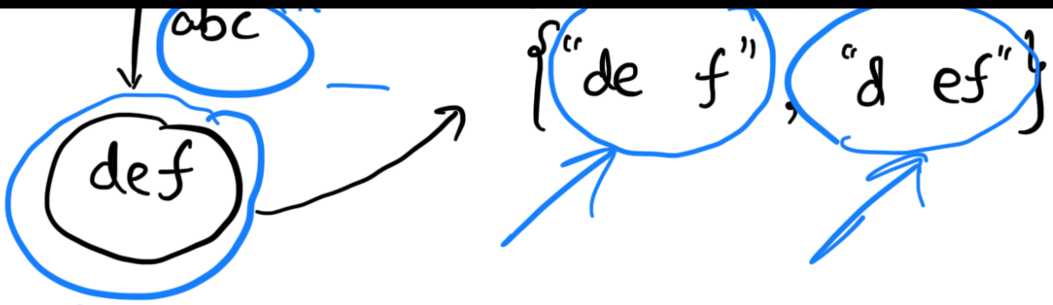
a leap of

R. ... faith

(·) Iterative ←
(·) Trie.



"abcdef"
abc,
de,
d,
ef



"abc de f"
"abc d ef"

$T \cdot C = O(2^n)$ poly

$S \cdot C = O(2^n)$.

(•) Iterative.

(•) Trie.