→ @codestorywithmiK
(Instagram, Facebook)
cswithMIK → Twitter
codestorywithMIK → whatsapp

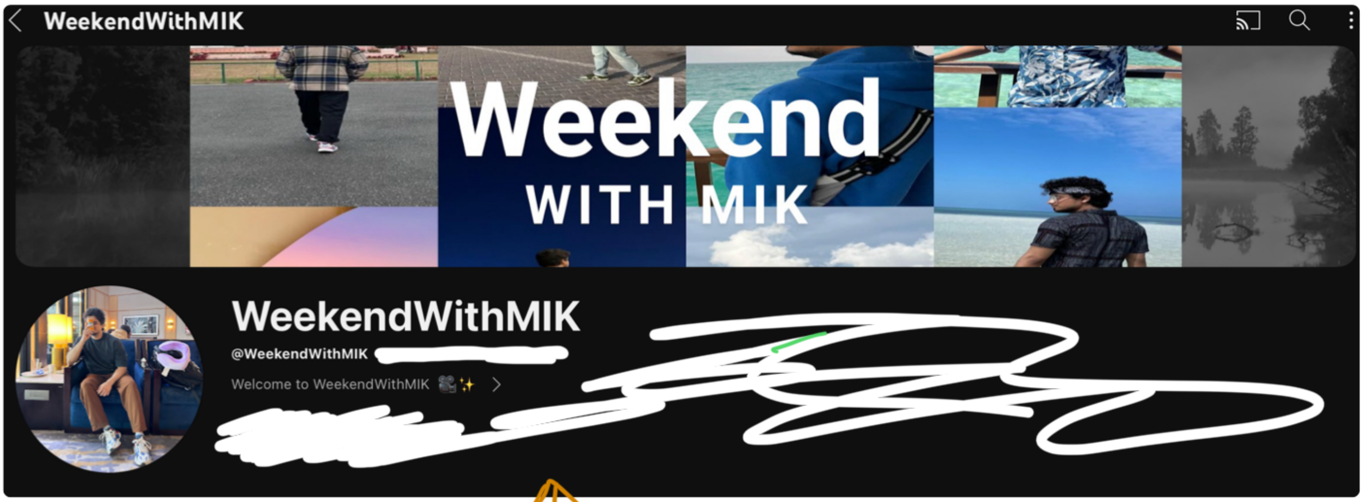
video-61

STRINGS...

Leetcode
-3403

Medium

My Dad is not well. will not be active over comments today.



WeekendWithMIK

@WeekendWithMIK [redacted]

Welcome to WeekendWithMIK 🎥✨ >

[redacted]

↑

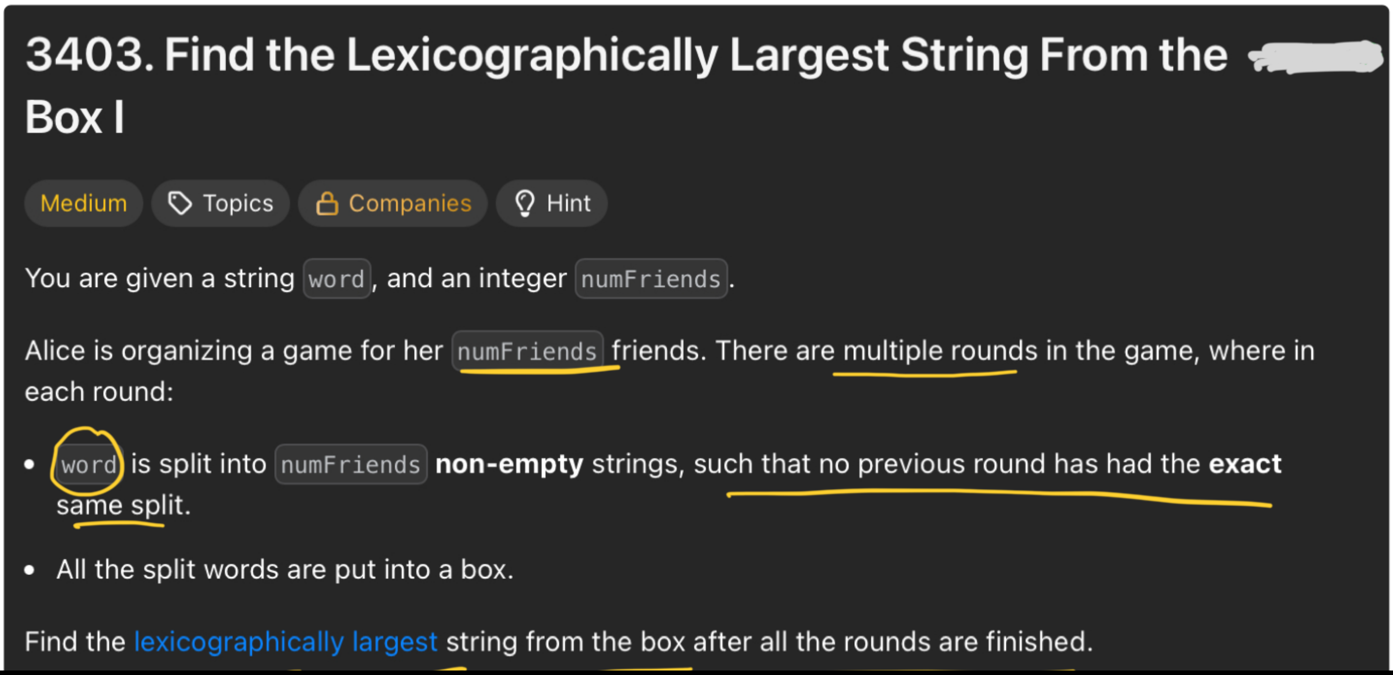
Try this channel to see Life Behind The Scenes ...
+ Tech News

Motivation Of the Day :-

Successful people are not gifted ;
they just work hard , then succeed
on purpose.

Instead of dreaming of it , work for it ...

MIK...



3403. Find the Lexicographically Largest String From the [redacted] Box I

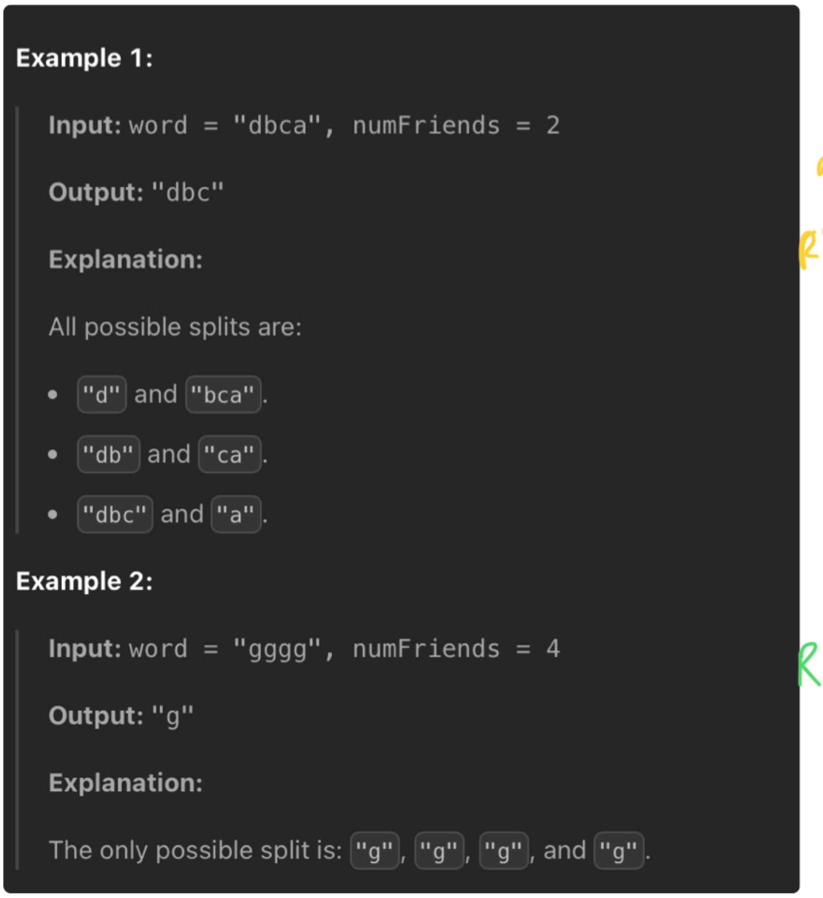
Medium | Topics | Companies | Hint

You are given a string `word`, and an integer `numFriends`.

Alice is organizing a game for her `numFriends` friends. There are multiple rounds in the game, where in each round:

- `word` is split into `numFriends` **non-empty** strings, such that no previous round has had the **exact** same split.
- All the split words are put into a box.

Find the lexicographically largest string from the box after all the rounds are finished.



Example 1:

Input: `word = "dbca", numFriends = 2`

Output: `"dbc"`

Explanation:

All possible splits are:

- `"d"` and `"bca"`.
- `"db"` and `"ca"`.
- `"dbc"` and `"a"`.

Example 2:

Input: `word = "gggg", numFriends = 4`

Output: `"g"`

Explanation:

The only possible split is: `"g"`, `"g"`, `"g"`, and `"g"`.

R1	d	bca
R1	db	ca
R3	dbc	a

R= g g gg

Thought Process

lexicographically largest → which appears later in the dictionary.

"ebcda" → "e"

cbcde e

word = "dbca" , numFriends = 2

look for largest character. → 'd'

Friend1	Friend2
d	bca
db	ca
dbc	a

"dbca"

Fr1	Fr 2	Fr 3
db	c	a

numFriends

(numFrinds -1)

← word.length() – (numFr -1) ;

"eabcde" , numFrie = 3

F1	F2	F3
4 length	1ch	1 ch
eabc	d	e

0 1 2 3 4

" a b c e d " , Fr = 3

i

F1	F2	F3
'e' → 3ch	1ch	1ch
ed	ab	c

e → idx = 3

```
longestPossible = n - (numFr - 1) ;
take possible length = min(longestPossible, n-i);
Substring (i , take possible length);
```

$$\begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \\ e & a & b & c & e & d \end{array}$$

"eabced" , Friends = 3

i

longestPossible = n - (Fr - 1)
= 6 - (3-1)
= 4

"eabc"
"abce"
"bced"
"ced"
"ed" * ⇐
"d"

⇒ 'e'
⇒ {0, 4}

eabc < ed

ed.

```
longestPossible = n - (numFr - 1) ;

for( i=0 ; i<n ; i++) {  → O(n)

    cantake length = min( longestPossible , n-i) ;

    result = max (result, word.substr ( i , cantake length));
                          └ O(n)
}
```

ret result;

$$\left\{\begin{array}{l} T \cdot C = O(n^2) \\ S \cdot C = O(1) \end{array}\right\}$$

$\underline{\underline{O(n)}}$. ⟸

[redacted]