Leet Code 12/09/2020

12 September 2020 16:15

?.

① Take middle linked-list:

- Count no of elements of ll:

si

- Go until meet the middle:

when $\left(\frac{si}{2}\right) + 1 == cnt$

② Jump until cant jump:

- Notice!

Suppose we can jump to position i and value at i is nums[i].

=> We can jump to all position from i to i + nums[i].

=> We only need to save the furthest position reachable at every position along the array. (max-jump)

- If at position i and max-jump < i then it means that i can't be reach.

(If i is reachable then $\forall i \in [1 \dots i]$. i is reachable)

Then ...

(If i is reachable then $\forall i \in [1 \dots i]$. i is reachable)

(3) Deconstruct regular expression:

Res = set answer.

- If string s has shape:

$s[0] = x$ for $x \in ['A' \dots 'Z']$.

then $R(s) = R(x) . R(s[1:])$.

- Else: then $s[0] = '('$

We iterate from 1 and keep cnt.

+ if $s[i] = '('$, cnt += 1

+ if $s[i] = ')'$

* if cnt != 0 then cnt -= 1

* Else if cnt = 0 it means this is the closing string of $s[0]$.

→ Add the part from last to i to res: res.update(list(creation(s[la:i]))

{a, b, c} → Then find creation $s[i+1:]$

And return set concatenate of that.

+ If $s[i] = ','$ and cnt = 0 then it is an union:

res.update(list(creation(s[la:i])).

res . update (list(create(s['a':'0'].

update last a = i + 1

→ return res .

Sort and we get the answer