Question statement (4):

Write a function to return a copy of a list of strings with duplicates removed. Preserve order in the original list as much as possible (keep first occurrence).

If the candidate implements this quickly, then ask to reimplement to keep nth occurrence (where n is specified as a parameter).

Example

["foo", "bar", "baz", "foo", "bar"] -> ["foo", "bar", "baz"]

Question (4)

I-dentify the problem

There are duplicate strings in a list

Define the goal

Remove duplicates by returning a list with only the 1st occurence of a word

-) if a word shows up more than once, delete the copies - leaving only the ~~1st~~ original ~~[illegible]~~

Explore Anticipate / Act through Duke's Approach

1) Examples by Hand

-) ["foo", "bar", "baz", "foo", "bar"]

->

["foo", "bar", "baz"]

→ ["dack", "duck", "duck", "daffy"]

→

["dack", "duck", "daffy"]

→ ["a", "a", "b", "b", "a", "a"]

→

["a", "b"]

→ ["a", "if", "off", "a", "af", "if",

"off"]

→

["a", "if", "off", "af",]

2) write what I did

The fist word in I don't need to check for duplicates

```
repeat
  move to next word,
    check if it's already in the ~~"ans"~~ new list
      - if no, add to new list
      - if yes, ~~move to~~ nothing
return new list
```

3) find patterns

- no real need for sets but sets would reduce big O
- checking every word in old list against words in new list

4) checking by hand

5) translate to code

built into Leet code

6) Run Test cases

7) Debug faild tet cases