APT: Word Ladder I

Problem Statement

A *word ladder* is a sequence of words in which each word can be transformed into the next word by changing one letter. For example, the word ladder below changes 'lot' to 'log'.

```
lot dot dog log
```

Class

```
public class WordLadder {
  public String ladderExists(String[] words,
                                 String from, String to) {
      // fill in code here
  }
}
```

This is not the shortest word-ladder between 'lot' and 'log' since the former can be immediately changed to the latter yielding a word ladder of length two:

```
lot log
```

The first and last words in a word ladder are the *anchor rungs* of the ladder. Any other words are *interior rungs*. For example, there are three interior rungs in the ladder below between 'smile' and 'evote'.

```
smile smite smote emote evote
```

In this problem you'll write a method that has parameters representing potential interior rungs: an array of strings (these may be nonsense or English words), and the anchor rungs --- two strings. Your code must determine whether there exists any ladder between the exterior rungs that uses at least one interior rung. If there is any ladder the method returns `"ladder"`, otherwise it should return `"none"`.

Notes and Constraints

- The parameters `from` and `to` are the anchor rungs, they must be connected by at least one interior rung from `words` or there are no valid word ladders.

- `words` contains at most 50 words.

- All strings contain only lowercase, alphabetic characters.

- All strings in `word` are the same length and are the same length as `from` and `to`.

Examples

1.
```
words = [hot, dot, dog]
from = hit
to = cog

Returns "ladder"
```

The only ladder is `hit hot dot dog cog` which has length five.

2.
```
words = [hot, dot, dog, lot, log]
from = hit
to = cog

Returns "ladder"
```

Now there are two length-five ladders:

```
  hit hot dot dog cog
  hit hot lot log cog
```

3.
```
words = [rain, ruin, gain, grin, grit, main, pain, pair, pail, mail]
from = sail
to = ruip

Returns "ladder"
```

There are two ladders of length six and no shorter ladders.

```
  sail mail main rain ruin ruip
  sail pail pain rain ruin ruip
```

4.
```
words = [most, mist, fist, fish,]
from = lost
to = cost

Returns: "ladder"
```

Although `lost` is directly connected to `cost`, a valid word ladder must contain an interior rung so the shortest ladder is

```
  lost most cost
```

5.  ```
    words = [mist, fist, fish,]
    from = lost
    to = cost

    Returns: "none"
    ```

    Although `lost` is directly connected to `cost`, a valid word ladder must contain an interior rung so there is no valid ladder.