Valid Word Square

Given a sequence of words, check whether it forms a valid word square.

A sequence of words forms a valid word square if thek^{th} row and column read the exact same string, where $0 \le k < \max(\text{numRows}, \text{numColumns})$.

Note:

1. The number of words given is at least 1 and does not exceed 500.
2. Word length will be at least 1 and does not exceed 500.
3. Each word contains only lowercase English alphabet `a-z`.

Example 1:

```
Input:
[
  "abcd",
  "bnrt",
  "crmy",
  "dtye"
]

Output:
true

Explanation:
The first row and first column both read "abcd".
The second row and second column both read "bnrt".
The third row and third column both read "crmy".
The fourth row and fourth column both read "dtye".

Therefore, it is a valid word square.
```

Example 2:

```
Input:
[
  "abcd",
  "bnrt",
  "crm",
  "dt"
]

Output:
true

Explanation:
The first row and first column both read "abcd".
The second row and second column both read "bnrt".
The third row and third column both read "crm".
The fourth row and fourth column both read "dt".

Therefore, it is a valid word square.
```

Example 3:

Input:
```
[
  "ball",
  "area",
  "read",
  "lady"
]
```

Output:
```
false
```

Explanation:
The third row reads "read" while the third column reads "lead".

Therefore, it is **NOT** a valid word square.

Solution 1

```
def validWordSquare(self, words):
    """
    :type words: List[str]
    :rtype: bool
    """
    lenw = len(words)
    try:
        for i in range(lenw):
            lenj = len(words[i])
            for j in range(lenj):
                if words[i][j] != words[j][i]:
                    return False
        return True
    except IndexError:
        return False
```

written by leakey original link here

Solution 2

Solution:

```
def validWordSquare(self, words):
    return map(None, *words) == map(None, *map(None, *words))
```

Or saving some work but taking two lines:

```
def validWordSquare(self, words):
    t = map(None, *words)
    return t == map(None, *t)
```

Explanation:

The `map(None, ...)` transposes the "matrix", filling missing spots with `None`. For example:

```
["abc",           [('a', 'd', 'f'),
 "de",     =>      ('b', 'e', None),
 "f"]              ('c', None, None)]
```

And then I just need to check whether transposing it once more changes it.

written by StefanPochmann original link here

Solution 3

```
  bool validWordSquare(vector<string>& words) {
       for(int i = 0; i < words.size(); ++i) {
           for(int j = 0; j < words[i].size(); ++j)              {
               if(j >= words.size() || words[j].size() <= i || words[j][i] != wo
rds[i][j])
                  return false;
           }
       }
       return true;
  }
```

written by i9r0k original link here