

Problem 1554: Strings Differ by One Character

Problem Information

Difficulty: Medium

Acceptance Rate: 0.00%

Paid Only: No

Problem Description

Given a list of strings

dict

where all the strings are of the same length.

Return

true

if there are 2 strings that only differ by 1 character in the same index, otherwise return

false

.

Example 1:

Input:

```
dict = ["abcd", "acbd", "aacd"]
```

Output:

true

Explanation:

Strings "a

b

cd" and "a

a

cd" differ only by one character in the index 1.

Example 2:

Input:

```
dict = ["ab", "cd", "yz"]
```

Output:

false

Example 3:

Input:

```
dict = ["abcd", "cccc", "abyd", "abab"]
```

Output:

true

Constraints:

The number of characters in

dict \leq 10

5

dict[i].length == dict[j].length

dict[i]

should be unique.

dict[i]

contains only lowercase English letters.

Code Snippets

C++:

```
class Solution {
public:
    bool differByOne(vector<string>& dict) {

    }
};
```

Java:

```
class Solution {
    public boolean differByOne(String[] dict) {

    }
}
```

Python3:

```
class Solution:
    def differByOne(self, dict: List[str]) -> bool:
```

Python:

```
class Solution(object):
    def differByOne(self, dict):
        """
        :type dict: List[str]
```

```
:rtype: bool
"""
```

JavaScript:

```
/**
 * @param {string[]} dict
 * @return {boolean}
 */
var differByOne = function(dict) {

};
```

TypeScript:

```
function differByOne(dict: string[]): boolean {

};
```

C#:

```
public class Solution {
    public bool DifferByOne(string[] dict) {

    }
}
```

C:

```
bool differByOne(char** dict, int dictSize) {

}
```

Go:

```
func differByOne(dict []string) bool {

}
```

Kotlin:

```

class Solution {
    fun differByOne(dict: Array<String>): Boolean {

    }
}

```

Swift:

```

class Solution {
    func differByOne(_ dict: [String]) -> Bool {

    }
}

```

Rust:

```

impl Solution {
    pub fn differ_by_one(dict: Vec<String>) -> bool {

    }
}

```

Ruby:

```

# @param {String[]} dict
# @return {Boolean}
def differ_by_one(dict)

end

```

PHP:

```

class Solution {

    /**
     * @param String[] $dict
     * @return Boolean
     */
    function differByOne($dict) {

    }
}

```

Dart:

```
class Solution {  
  bool differByOne(List<String> dict) {  
  
  }  
}
```

Scala:

```
object Solution {  
  def differByOne(dict: Array[String]): Boolean = {  
  
  }  
}
```

Elixir:

```
defmodule Solution do  
  @spec differ_by_one(dict :: [String.t]) :: boolean  
  def differ_by_one(dict) do  
  
  end  
end
```

Erlang:

```
-spec differ_by_one(Dict :: [unicode:unicode_binary()]) -> boolean().  
differ_by_one(Dict) ->  
.
```

Racket:

```
(define/contract (differ-by-one dict)  
  (-> (listof string?) boolean?)  
)
```

Solutions

C++ Solution:

```

/*
 * Problem: Strings Differ by One Character
 * Difficulty: Medium
 * Tags: string, hash
 *
 * Approach: String manipulation with hash map or two pointers
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) for hash map
 */

class Solution {
public:
    bool differByOne(vector<string>& dict) {

    }
};

```

Java Solution:

```

/**
 * Problem: Strings Differ by One Character
 * Difficulty: Medium
 * Tags: string, hash
 *
 * Approach: String manipulation with hash map or two pointers
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) for hash map
 */

class Solution {
    public boolean differByOne(String[] dict) {

    }
}

```

Python3 Solution:

```

"""
Problem: Strings Differ by One Character
Difficulty: Medium
Tags: string, hash

```

```

Approach: String manipulation with hash map or two pointers
Time Complexity: O(n) or O(n log n)
Space Complexity: O(n) for hash map
"""

class Solution:
    def differByOne(self, dict: List[str]) -> bool:
        # TODO: Implement optimized solution
        pass

```

Python Solution:

```

class Solution(object):
    def differByOne(self, dict):
        """
        :type dict: List[str]
        :rtype: bool
        """

```

JavaScript Solution:

```

/**
 * Problem: Strings Differ by One Character
 * Difficulty: Medium
 * Tags: string, hash
 *
 * Approach: String manipulation with hash map or two pointers
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) for hash map
 */

/**
 * @param {string[]} dict
 * @return {boolean}
 */
var differByOne = function(dict) {

};

```

TypeScript Solution:


```

/**
 * Problem: Strings Differ by One Character
 * Difficulty: Medium
 * Tags: string, hash
 *
 * Approach: String manipulation with hash map or two pointers
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) for hash map
 */

function differByOne(dict: string[]): boolean {

};

```

C# Solution:

```

/*
 * Problem: Strings Differ by One Character
 * Difficulty: Medium
 * Tags: string, hash
 *
 * Approach: String manipulation with hash map or two pointers
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) for hash map
 */

public class Solution {
    public bool DifferByOne(string[] dict) {

    }
}

```

C Solution:

```

/*
 * Problem: Strings Differ by One Character
 * Difficulty: Medium
 * Tags: string, hash
 *
 * Approach: String manipulation with hash map or two pointers
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) for hash map

```

```

*/

bool differByOne(char** dict, int dictSize) {

}

```

Go Solution:

```

// Problem: Strings Differ by One Character
// Difficulty: Medium
// Tags: string, hash
//
// Approach: String manipulation with hash map or two pointers
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(n) for hash map

func differByOne(dict []string) bool {

}

```

Kotlin Solution:

```

class Solution {
    fun differByOne(dict: Array<String>): Boolean {

    }
}

```

Swift Solution:

```

class Solution {
    func differByOne(_ dict: [String]) -> Bool {

    }
}

```

Rust Solution:

```

// Problem: Strings Differ by One Character
// Difficulty: Medium
// Tags: string, hash

```

```
//
// Approach: String manipulation with hash map or two pointers
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(n) for hash map

impl Solution {
    pub fn differ_by_one(dict: Vec<String>) -> bool {

    }
}
```

Ruby Solution:

```
# @param {String[]} dict
# @return {Boolean}
def differ_by_one(dict)

end
```

PHP Solution:

```
class Solution {

    /**
     * @param String[] $dict
     * @return Boolean
     */
    function differByOne($dict) {

    }
}
```

Dart Solution:

```
class Solution {
    bool differByOne(List<String> dict) {

    }
}
```

Scala Solution:

```
object Solution {  
  def differByOne(dict: Array[String]): Boolean = {  
  
  }  
}
```

Elixir Solution:

```
defmodule Solution do  
  @spec differ_by_one(dict :: [String.t]) :: boolean  
  def differ_by_one(dict) do  
  
  end  
end
```

Erlang Solution:

```
-spec differ_by_one(Dict :: [unicode:unicode_binary()]) -> boolean().  
differ_by_one(Dict) ->  
.
```

Racket Solution:

```
(define/contract (differ-by-one dict)  
  (-> (listof string?) boolean?)  
)
```