

Problem 1153: String Transforms Into Another String

Problem Information

Difficulty: Hard

Acceptance Rate: 0.00%

Paid Only: No

Problem Description

Given two strings

str1

and

str2

of the same length, determine whether you can transform

str1

into

str2

by doing

zero or more

conversions

.

In one conversion you can convert

all

occurrences of one character in

str1

to

any

other lowercase English character.

Return

true

if and only if you can transform

str1

into

str2

.

Example 1:

Input:

str1 = "aabcc", str2 = "ccdee"

Output:

true

Explanation:

Convert 'c' to 'e' then 'b' to 'd' then 'a' to 'c'. Note that the order of conversions matter.

Example 2:

Input:

```
str1 = "leetcode", str2 = "codeleet"
```

Output:

```
false
```

Explanation:

There is no way to transform str1 to str2.

Constraints:

```
1 <= str1.length == str2.length <= 10
```

4

str1

and

str2

contain only lowercase English letters.

Code Snippets

C++:

```
class Solution {
public:
    bool canConvert(string str1, string str2) {
```

```
}
```

```
};
```

Java:

```
class Solution {  
    public boolean canConvert(String str1, String str2) {  
  
    }  
}
```

Python3:

```
class Solution:  
    def canConvert(self, str1: str, str2: str) -> bool:
```

Python:

```
class Solution(object):  
    def canConvert(self, str1, str2):  
        """  
        :type str1: str  
        :type str2: str  
        :rtype: bool  
        """
```

JavaScript:

```
/**  
 * @param {string} str1  
 * @param {string} str2  
 * @return {boolean}  
 */  
var canConvert = function(str1, str2) {  
  
};
```

TypeScript:

```
function canConvert(str1: string, str2: string): boolean {  
  
};
```

C#:

```
public class Solution {  
    public bool CanConvert(string str1, string str2) {  
        }  
        }  
}
```

C:

```
bool canConvert(char * str1, char * str2){  
    }
```

Go:

```
func canConvert(str1 string, str2 string) bool {  
    }
```

Kotlin:

```
class Solution {  
    fun canConvert(str1: String, str2: String): Boolean {  
        }  
        }
```

Swift:

```
class Solution {  
    func canConvert(_ str1: String, _ str2: String) -> Bool {  
        }  
        }
```

Rust:

```
impl Solution {  
    pub fn can_convert(str1: String, str2: String) -> bool {
```

```
}
```

```
}
```

Ruby:

```
# @param {String} str1
# @param {String} str2
# @return {Boolean}
def can_convert(str1, str2)

end
```

PHP:

```
class Solution {

    /**
     * @param String $str1
     * @param String $str2
     * @return Boolean
     */
    function canConvert($str1, $str2) {

    }
}
```

Scala:

```
object Solution {
  def canConvert(str1: String, str2: String): Boolean = {
    }
}
```

Solutions

C++ Solution:

```
/*
 * Problem: String Transforms Into Another String
```

```

* Difficulty: Hard
* Tags: string, graph, 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 canConvert(string str1, string str2) {

}
};

```

Java Solution:

```

/**
* Problem: String Transforms Into Another String
* Difficulty: Hard
* Tags: string, graph, 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 canConvert(String str1, String str2) {

}
};

```

Python3 Solution:

```

"""
Problem: String Transforms Into Another String
Difficulty: Hard
Tags: string, graph, 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 canConvert(self, str1: str, str2: str) -> bool:
# TODO: Implement optimized solution
pass

```

Python Solution:

```

class Solution(object):

def canConvert(self, str1, str2):
"""

:type str1: str
:type str2: str
:rtype: bool

"""

```

JavaScript Solution:

```

/**
 * Problem: String Transforms Into Another String
 * Difficulty: Hard
 * Tags: string, graph, 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} str1
 * @param {string} str2
 * @return {boolean}
 */
var canConvert = function(str1, str2) {

};


```

TypeScript Solution:

```

/**
 * Problem: String Transforms Into Another String
 * Difficulty: Hard
 * Tags: string, graph, 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 canConvert(str1: string, str2: string): boolean {

};

```

C# Solution:

```

/*
 * Problem: String Transforms Into Another String
 * Difficulty: Hard
 * Tags: string, graph, 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 CanConvert(string str1, string str2) {

    }
}

```

C Solution:

```

/*
 * Problem: String Transforms Into Another String
 * Difficulty: Hard
 * Tags: string, graph, 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 canConvert(char * str1, char * str2){  
  
}  
  
}
```

Go Solution:

```
// Problem: String Transforms Into Another String  
// Difficulty: Hard  
// Tags: string, graph, 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 canConvert(str1 string, str2 string) bool {  
  
}
```

Kotlin Solution:

```
class Solution {  
    fun canConvert(str1: String, str2: String): Boolean {  
  
    }  
}
```

Swift Solution:

```
class Solution {  
    func canConvert(_ str1: String, _ str2: String) -> Bool {  
  
    }  
}
```

Rust Solution:

```

// Problem: String Transforms Into Another String
// Difficulty: Hard
// Tags: string, graph, 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 can_convert(str1: String, str2: String) -> bool {
        }

    }
}

```

Ruby Solution:

```

# @param {String} str1
# @param {String} str2
# @return {Boolean}
def can_convert(str1, str2)

end

```

PHP Solution:

```

class Solution {

    /**
     * @param String $str1
     * @param String $str2
     * @return Boolean
     */
    function canConvert($str1, $str2) {

    }
}

```

Scala Solution:

```

object Solution {
    def canConvert(str1: String, str2: String): Boolean = {

```

