Problem 2515: Shortest Distance to Target String in a Circular Array

Problem Information

Difficulty: Easy
Acceptance Rate: 50.43%
Paid Only: No
Tags: Array, String

Problem Description

You are given a **0-indexed** **circular** string array `words` and a string `target`. A **circular array** means that the array's end connects to the array's beginning.

* Formally, the next element of `words[i]` is `words[(i + 1) % n]` and the previous element of `words[i]` is `words[(i - 1 + n) % n]`, where `n` is the length of `words`.

Starting from `startIndex`, you can move to either the next word or the previous word with `1` step at a time.

Return _the**shortest** distance needed to reach the string_ `target`. If the string `target` does not exist in `words`, return `-1`.

Example 1:

Input: words = ["hello","i","am","leetcode","hello"], target = "hello", startIndex = 1 **Output:** 1 **Explanation:** We start from index 1 and can reach "hello" by - moving 3 units to the right to reach index 4. - moving 2 units to the left to reach index 4. - moving 4 units to the right to reach index 0. - moving 1 unit to the left to reach index 0. The shortest distance to reach "hello" is 1.

Example 2:

Input: words = ["a","b","leetcode"], target = "leetcode", startIndex = 0 **Output:** 1 **Explanation:** We start from index 0 and can reach "leetcode" by - moving 2 units to the right to reach index 3. - moving 1 unit to the left to reach index 3. The shortest distance to

reach "leetcode" is 1.

Example 3:

Input: words = ["i","eat","leetcode"], target = "ate", startIndex = 0 **Output:** -1
Explanation: Since "ate" does not exist in words, we return -1.

Constraints:

* `1 <= words.length <= 100` * `1 <= words[i].length <= 100` * `words[i]` and `target` consist of only lowercase English letters. * `0 <= startIndex < words.length`

Code Snippets

C++:

```
class Solution {
public:
int closestTarget(vector<string>& words, string target, int startIndex) {

}
};
```

Java:

```
class Solution {
public int closestTarget(String[] words, String target, int startIndex) {

}
}
```

Python3:

```
class Solution:
def closestTarget(self, words: List[str], target: str, startIndex: int) ->
int:
```