

Problem 2227: Encrypt and Decrypt Strings

Problem Information

Difficulty: Hard

Acceptance Rate: 37.74%

Paid Only: No

Tags: Array, Hash Table, String, Design, Trie

Problem Description

You are given a character array `keys` containing **unique** characters and a string array `values` containing strings of length 2. You are also given another string array `dictionary` that contains all permitted original strings after decryption. You should implement a data structure that can encrypt or decrypt a **0-indexed** string.

A string is **encrypted** with the following process:

1. For each character `c` in the string, we find the index `i` satisfying `keys[i] == c` in `keys` . 2. Replace `c` with `values[i]` in the string.

Note that in case a character of the string is **not present** in `keys`, the encryption process cannot be carried out, and an empty string `""` is returned.

A string is **decrypted** with the following process:

1. For each substring `s` of length 2 occurring at an even index in the string, we find an `i` such that `values[i] == s` . If there are multiple valid `i` , we choose **any** one of them. This means a string could have multiple possible strings it can decrypt to. 2. Replace `s` with `keys[i]` in the string.

Implement the `Encrypter` class:

* `Encrypter(char[] keys, String[] values, String[] dictionary)` Initializes the `Encrypter` class with `keys`, `values` , and `dictionary` . * `String encrypt(String word1)` Encrypts `word1` with the encryption process described above and returns the encrypted string. * `int decrypt(String word2)` Returns the number of possible strings `word2` could decrypt to that also appear in

`dictionary`.

****Example 1:****

****Input**** ["Encrypter", "encrypt", "decrypt"] [[[{"a": "e", "b": "z", "c": "f", "d": "a"}, {"ei": "ei", "zf": "am"}], [{"abcd": "abcd", "acbd": "acbd", "adbc": "adbc", "badc": "badc", "dabc": "dabc", "cadb": "cadb", "cbda": "cbda", "abad": "abad"}], [{"abcd": "abcd", "eizfeiam": "eizfeiam"}]] ****Output**** [null, "eizfeiam", 2] ****Explanation**** Encrypter encrypter = new Encrypter([{"a": "e", "b": "z", "c": "f", "d": "a"}, {"ei": "ei", "zf": "am"}], [{"abcd": "abcd", "acbd": "acbd", "adbc": "adbc", "badc": "badc", "dabc": "dabc", "cadb": "cadb", "cbda": "cbda", "abad": "abad"}]); encrypter.encrypt("abcd"); // return "eizfeiam". // 'a' maps to "ei", 'b' maps to "zf", 'c' maps to "ei", and 'd' maps to "am". encrypter.decrypt("eizfeiam"); // return 2. // "ei" can map to 'a' or 'c', "zf" maps to 'b', and "am" maps to 'd'. // Thus, the possible strings after decryption are "abad", "cbad", "abcd", and "cbc". // 2 of those strings, "abad" and "abcd", appear in dictionary, so the answer is 2.

****Constraints:****

* `1 <= keys.length == values.length <= 26` * `values[i].length == 2` * `1 <= dictionary.length <= 100` * `1 <= dictionary[i].length <= 100` * All `keys[i]` and `dictionary[i]` are **unique**. * `1 <= word1.length <= 2000` * `2 <= word2.length <= 200` * All `word1[i]` appear in `keys`. * `word2.length` is even. * `keys`, `values[i]`, `dictionary[i]`, `word1`, and `word2` only contain lowercase English letters. * At most `200` calls will be made to `encrypt` and `decrypt` **in total**.

Code Snippets

C++:

```
class Encrypter {
public:
    Encrypter(vector<char>& keys, vector<string>& values, vector<string>& dictionary) {

    }

    string encrypt(string word1) {

    }

    int decrypt(string word2) {
```

```
}

};

/***
* Your Encrypter object will be instantiated and called as such:
* Encrypter* obj = new Encrypter(keys, values, dictionary);
* string param_1 = obj->encrypt(word1);
* int param_2 = obj->decrypt(word2);
*/

```

Java:

```
class Encrypter {

public Encrypter(char[] keys, String[] values, String[] dictionary) {

}

public String encrypt(String word1) {

}

public int decrypt(String word2) {

}

/***
* Your Encrypter object will be instantiated and called as such:
* Encrypter obj = new Encrypter(keys, values, dictionary);
* String param_1 = obj.encrypt(word1);
* int param_2 = obj.decrypt(word2);
*/

```

Python3:

```
class Encrypter:

def __init__(self, keys: List[str], values: List[str], dictionary:
List[str]):
```

```
def encrypt(self, word1: str) -> str:

def decrypt(self, word2: str) -> int:

# Your Encrypter object will be instantiated and called as such:
# obj = Encrypter(keys, values, dictionary)
# param_1 = obj.encrypt(word1)
# param_2 = obj.decrypt(word2)
```