

2851. String Transformation

Description

You are given two strings `s` and `t` of equal length `n`. You can perform the following operation on the string `s`:

- Remove a **suffix** of `s` of length `l` where $0 < l < n$ and append it at the start of `s`.
For example, let `s = 'abcd'` then in one operation you can remove the suffix `'cd'` and append it in front of `s` making `s = 'cdab'`.

You are also given an integer `k`. Return *the number of ways in which `s` can be transformed into `t` in exactly `k` operations*.

Since the answer can be large, return it **modulo** `$10^9 + 7$` .

Example 1:

```
Input: s = "abcd", t = "cdab", k = 2
Output: 2
Explanation:
First way:
In first operation, choose suffix from index = 3, so resulting s = "dabc".
In second operation, choose suffix from index = 3, so resulting s = "cdab".

Second way:
In first operation, choose suffix from index = 1, so resulting s = "bcda".
In second operation, choose suffix from index = 1, so resulting s = "cdab".
```

Example 2:

```
Input: s = "ababab", t = "ababab", k = 1
Output: 2
Explanation:
First way:
Choose suffix from index = 2, so resulting s = "ababab".

Second way:
Choose suffix from index = 4, so resulting s = "ababab".
```

Constraints:

- `2 <= s.length <= 5 * 105`
- `1 <= k <= 1015`
- `s.length == t.length`
- `s` and `t` consist of only lowercase English alphabets.

