# Problem 3337: Total Characters in String After Transformations II

## Problem Information

**Difficulty:** Hard
**Acceptance Rate:** 58.06%
**Paid Only:** No
**Tags:** Hash Table, Math, String, Dynamic Programming, Counting

## Problem Description

You are given a string `s` consisting of lowercase English letters, an integer `t` representing the number of **transformations** to perform, and an array `nums` of size 26. In one **transformation** , every character in `s` is replaced according to the following rules:

* Replace `s[i]` with the **next** `nums[s[i] - 'a']` consecutive characters in the alphabet. For example, if `s[i] = 'a'` and `nums[0] = 3`, the character `'a'` transforms into the next 3 consecutive characters ahead of it, which results in `"bcd"`. * The transformation **wraps** around the alphabet if it exceeds `'z'`. For example, if `s[i] = 'y'` and `nums[24] = 3`, the character `'y'` transforms into the next 3 consecutive characters ahead of it, which results in `"zab"`.

Return the length of the resulting string after **exactly** `t` transformations.

Since the answer may be very large, return it **modulo** `109 + 7`.

**Example 1:**

**Input:** s = "abcyy", t = 2, nums = [1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,2]

**Output:** 7

**Explanation:**

* **First Transformation (t = 1):**

* `'a'` becomes `'b'` as `nums[0] == 1` * `'b'` becomes `'c'` as `nums[1] == 1` * `'c'` becomes `'d'` as `nums[2] == 1` * `'y'` becomes `'z'` as `nums[24] == 1` * `'y'` becomes `'z'` as `nums[24] == 1` * String after the first transformation: `"bcdzz"` * **Second Transformation (t = 2):**

* `'b'` becomes `'c'` as `nums[1] == 1` * `'c'` becomes `'d'` as `nums[2] == 1` * `'d'` becomes `'e'` as `nums[3] == 1` * `'z'` becomes `'ab'` as `nums[25] == 2` * `'z'` becomes `'ab'` as `nums[25] == 2` * String after the second transformation: `"cdeabab"` * **Final Length of the string:** The string is `"cdeabab"`, which has 7 characters.

**Example 2:**

**Input:** s = "azbk", t = 1, nums = [2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2]

**Output:** 8

**Explanation:**

* **First Transformation (t = 1):**

* `'a'` becomes `'bc'` as `nums[0] == 2` * `'z'` becomes `'ab'` as `nums[25] == 2` * `'b'` becomes `'cd'` as `nums[1] == 2` * `'k'` becomes `'lm'` as `nums[10] == 2` * String after the first transformation: `"bcabcdlm"` * **Final Length of the string:** The string is `"bcabcdlm"`, which has 8 characters.

**Constraints:**

* `1 <= s.length <= 105` * `s` consists only of lowercase English letters. * `1 <= t <= 109` * `nums.length == 26` * `1 <= nums[i] <= 25`

## Code Snippets

**C++:**

```
class Solution {
public:
int lengthAfterTransformations(string s, int t, vector<int>& nums) {

}
```

```
};
```

**Java:**

```
class Solution {
public int lengthAfterTransformations(String s, int t, List<Integer> nums) {

}
}
```

**Python3:**

```
class Solution:
def lengthAfterTransformations(self, s: str, t: int, nums: List[int]) -> int:
```