Problem 3348: Smallest Divisible Digit Product II

Problem Information

Difficulty: Hard
Acceptance Rate: 13.23%
Paid Only: No
Tags: Math, String, Backtracking, Greedy, Number Theory

Problem Description

You are given a string `num` which represents a **positive** integer, and an integer `t`.

A number is called **zero-free** if _none_ of its digits are 0.

Return a string representing the **smallest** **zero-free** number greater than or equal to `num` such that the **product of its digits** is divisible by `t`. If no such number exists, return `"-1"`.

Example 1:

Input: num = "1234", t = 256

Output: "1488"

Explanation:

The smallest zero-free number that is greater than 1234 and has the product of its digits divisible by 256 is 1488, with the product of its digits equal to 256.

Example 2:

Input: num = "12355", t = 50

Output: "12355"

Explanation:

12355 is already zero-free and has the product of its digits divisible by 50, with the product of its digits equal to 150.

Example 3:

Input: num = "11111", t = 26

Output: "-1"

Explanation:

No number greater than 11111 has the product of its digits divisible by 26.

Constraints:

* `2 <= num.length <= 2 * 105` * `num` consists only of digits in the range `['0', '9']`. * `num` does not contain leading zeros. * `1 <= t <= 1014`

Code Snippets

C++:

```
class Solution {
public:
string smallestNumber(string num, long long t) {

}
};
```

Java:

```
class Solution {
public String smallestNumber(String num, long t) {

}
}
```

Python3:

```
class Solution:
def smallestNumber(self, num: str, t: int) -> str:
```