# 6244. Number of Beautiful Partitions

My Submissions (/contest/weekly-contest-320/problems/number-of-beautiful-partitions/submissions/)

Back to Contest (/contest/weekly-contest-320/)

You are given a string s that consists of the digits '1' to '9' and two integers k and minLength.

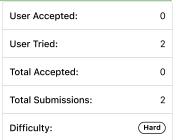
User Accepted: 0

A partition of s is called beautiful if:

- s is partitioned into k non-intersecting substrings.
- Each substring has a length of at least minLength.
- Each substring starts with a **prime** digit and ends with a **non-prime** digit. Prime digits are '2', '3', '5', and '7', and the rest of the digits are non-prime.

Return the number of **beautiful** partitions of s. Since the answer may be very large, return it **modulo**  $10^9 + 7$ .

A substring is a contiguous sequence of characters within a string.



### Example 1:

```
Input: s = "23542185131", k = 3, minLength = 2
Output: 3
Explanation: There exists three ways to create a beautiful partition:
"2354 | 218 | 5131"
"2354 | 21851 | 31"
"2354218 | 51 | 31"
```

#### Example 2:

```
Input: s = "23542185131", k = 3, minLength = 3
Output: 1
Explanation: There exists one way to create a beautiful partition: "2354 | 218 | 5131".
```

# Example 3:

```
Input: s = "3312958", k = 3, minLength = 1
Output: 1
Explanation: There exists one way to create a beautiful partition: "331 | 29 | 58".
```

### Constraints:

- 1 <= k, minLength <= s.length <= 1000
- s consists of the digits '1' to '9'.

```
JavaScript
                                                                                                                                  ক
                                                                                                                                         \mathfrak{C}
    const initialize2DArray = (n, m) \Rightarrow \{ let d = []; for (let i = 0; i < n; i++) \{ let t = Array(m).fill(0); d.push(t); \} \}
    return d; };
 2
 3
    const mod = 1e9 + 7;
    const beautiful
Partitions = (s, k, minLength) => { let n = s.length, primes = new Set(['2', '3', '5', '7']), dp = initialize2DArray(k + 1, n + 1);
 4
 5
 6
         if (!primes.has(s[0]) || primes.has(s[n - 1])) return 0;
 7
         s = s + '2';
         dp[0][0] = 1;
 8
 9.
         for (let i = 0; i < k; i++) {
              let cnt = 0;
10
11 ▼
              for (let j = 1; j <= n; j++) {
                  if (j - minLength >= 0) {
12
13
                       cnt += dp[i][j - minLength];
14
                       cnt %= mod:
15
                  if (primes.has(s[j]) \& !primes.has(s[j - 1])) dp[i + 1][j] = cnt;
16
17
             }
18
19
20
         return dp[k][n];
```

Help Center (/support) | Jobs (/jobs) | Bug Bounty (/bugbounty) | Online Interview (/interview/) | Students (/student) | Terms (/terms) | Privacy Policy (/privacy)

Copyright © 2022 LeetCode

United States (/region)