1850. Minimum Adjacent Swaps to Reach the Kth Smallest Number

Description

You are given a string `num`, representing a large integer, and an integer `k`.

We call some integer **wonderful** if it is a **permutation** of the digits in `num` and is **greater in value** than `num`. There can be many wonderful integers. However, we only care about the **smallest-valued** ones.

- For example, when `num = "5489355142"`:
  - The 1 st smallest wonderful integer is `"5489355214"`.
  - The 2 nd smallest wonderful integer is `"5489355241"`.
  - The 3 rd smallest wonderful integer is `"5489355412"`.
  - The 4 th smallest wonderful integer is `"5489355421"`.

Return *the* ***minimum number of adjacent digit swaps*** *that needs to be applied to* `num` *to reach the* `k th` ***smallest wonderful*** *integer*.

The tests are generated in such a way that `k th` smallest wonderful integer exists.

Example 1:

```
Input: num = "5489355142", k = 4
Output: 2
Explanation: The 4 th smallest wonderful number is "5489355421". To get this number:
- Swap index 7 with index 8: "5489355 14 2" -> "5489355 41 2"
- Swap index 8 with index 9: "54893554 12" -> "54893554 21"
```

Example 2:

```
Input: num = "11112", k = 4
Output: 4
Explanation: The 4 th smallest wonderful number is "21111". To get this number:
- Swap index 3 with index 4: "111 12" -> "111 21"
- Swap index 2 with index 3: "11 12 1" -> "11 21 1"
- Swap index 1 with index 2: "1 12 11" -> "1 21 11"
- Swap index 0 with index 1: " 12 111" -> " 21 111"
```

Example 3:

```
Input: num = "00123", k = 1
Output: 1
Explanation: The 1 st smallest wonderful number is "00132". To get this number:
- Swap index 3 with index 4: "001 23" -> "001 32"
```

Constraints:

- `2 <= num.length <= 1000`
- `1 <= k <= 1000`
- `num` only consists of digits.