Documentation: Permutation Sequence

Problem Statement:

The problem involves generating the kth permutation sequence of numbers from 1 to n. For example, given n = 3 and k = 3, the output should be "213", which is the third permutation of the numbers 1, 2, and 3.

Approach:

The solution utilizes the concept of factorial and iterative permutation generation to find the kth permutation sequence efficiently.

1. **Numbers Generation:** First, create a list of numbers from 1 to n represented as strings.
2. **Factorial Calculation:** Calculate the factorial of n. This will be used to determine the permutations.
3. **Decrement k:** Since indices start from 0 but the problem statement indexes from 1, decrement k by 1.
4. **Permutation Generation:** Iterate through each position from the rightmost to the leftmost.
    - Calculate the index of the number to pick for the current position.
    - Append the selected number to the result.
    - Update k and factorial accordingly.

Time Complexity:

The time complexity of the solution is O(n) since it involves iterating over the range of n.

Space Complexity:

The space complexity is also O(n) due to the list of numbers generated.

Example Usage:

```
sol = Solution()

print(sol.getPermutation(3, 3))  # Output: "213"

print(sol.getPermutation(4, 9))  # Output: "2314"

print(sol.getPermutation(3, 1))  # Output: "123"
```

Constraints:

- 1 <= n <= 9
- 1 <= k <= n!

References:

- [LeetCode Problem - 60. Permutation Sequence](https://leetcode.com/problems/permutation-sequence/)