Largest number in K swaps

Given a number **K** and string **str** of digits denoting a positive integer, build the largest number possible by performing swap operations on the digits of **str** at most **K** times.

Example 1:

```
Input: K = 4
str = "1234567"
Output:
7654321 Explanation: Three swaps can make the
input 1234567 to 7654321, swapping 1
with 7, 2 with 6 and finally 3 with 5
```

Example 2:

```
Input: K = 3
str = "3435335"
Output: 5543333
Explanation: Three swaps can make the input
3435335 to 5543333, swapping 3
with 5, 4 with 5 and finally 3 with 4
```

Your task:
You don't have to read input or print anything. Your task is to complete the function **findMaximumNum()** which takes the string and an integer as input and returns a string containing the largest number formed by perfoming the swap operation at most k times.

Expected Time Complexity: O(n!/(n-k)!) , where n = length of input string
Expected Auxiliary Space: O(n)

```
def findMaximumNum(self,s,k):
        #code here
        maxm = [s]
        s = list(s)

        self.findMaximumNumUtil(s,k,maxm)
        return maxm[0]
    def findMaximumNumUtil(self,s,k,maxm):
        if k == 0:
            return s
        n = len(s)
        # consider every digit
        for i in range(n - 1):

            # and compare it with all digits after it
            for j in range(i + 1, n):

                # if digit at position i is less than
                # digit at position j, swap it and
                # check for maximum number so far and
                # recurse for remaining swaps
                if s[i] < s[j]:

                    # swap string[i] with string[j]
                    self.swap(s, i, j)

                    # If current num is more than
                    # maximum so far
                    if ''.join(s) > maxm[0]:
                        maxm[0] = ''.join(s)

                    # recurse of the other k - 1 swaps
                    self.findMaximumNumUtil(s, k - 1, maxm)

                    # backtrack
                    self.swap(s, i, j)
    def swap(self,string, i, j):

        string[i],string[j] = string[j],string[i]
```