恢复数字序列题目描述：

对于一个连续正整数组成的序列，可以将其拼接成一个字符串，再将字符串里的部分字符打乱顺序。如序列 8 9 10 11 12，拼接成的字符串为 891011112，打乱一部分字符后得到 90811211。注意打乱后原来的正整数可能被拆开，比如在 90811211 中，原来的正整数 10 就被拆成了 0 和 1。

现给定一个按如上规则得到的打乱了字符的字符串，请将其还原成连续正整数序列，并输出序列中最小的数字。

输入描述：

输入一行，为打乱字符的字符串和正整数序列的长度，两者间用空格分隔，字符串长度不超过 200，正整数不超过 1000，保证输入可以还原成唯一序列。

输出描述：

输出一个数字，为序列中最小的数字。

补充说明：

示例 1

输入：

```
19801211 5
```

输出：

```
8
```

说明：

还原出的序列为 8 9 10 11 12，故输出 8

示例 2

输入：

```
432111111111 4
```

输出：

```
111
```

说明：

还原出的序列为 111 112 113 114，故输出 111

```
import sys



def get_count(s):
    res = []
    for i in range(10):
        res.append(s.count(str(i)))

    return res


for line in sys.stdin:
    a = line.strip().split()
    num_str = a[0]
    raw_count = get_count(num_str)
    count =int(a[1])
    avg_size = len(num_str)//count
    min_num = 0
    if avg_size>1:
        min_num = int('1' + '0' * (avg_size-1))
```

```
max_size = avg_size if  len(num_str)% count==0 else avg_size +1


max_num = int('9' * max_size)
ans = 0
for i in range(min_num,max_num +1):
    str_res = ''
    for idx in range(count):
        str_res += str((i +idx))
    t_res = get_count(str_res)
    if t_res == raw_count:
        ans = i
        break
print(ans)
```