216. Combination Sum III

Find all valid combinations of `k` numbers that sum up to `n` such that the following conditions are true:

- Only numbers `1` through `9` are used.
- Each number is used **at most once**.

Return *a list of all possible valid combinations*. The list must not contain the same combination twice, and the combinations may be returned in any order.

Example 1:

```
Input: k = 3, n = 7
Output: [[1,2,4]]
Explanation:
1 + 2 + 4 = 7
There are no other valid combinations.
```

Example 2:

```
Input: k = 3, n = 9
Output: [[1,2,6],[1,3,5],[2,3,4]]
Explanation:
1 + 2 + 6 = 9
1 + 3 + 5 = 9
2 + 3 + 4 = 9
There are no other valid combinations.
```

Example 3:

```
Input: k = 4, n = 1
Output: []
Explanation: There are no valid combinations.
Using 4 different numbers in the range [1,9], the smallest sum we can get
is 1+2+3+4 = 10 and since 10 > 1, there are no valid combination.
```

Example 4:

```
Input: k = 3, n = 2
Output: []
Explanation: There are no valid combinations.
```

Example 5:

```
Input: k = 9, n = 45
Output: [[1,2,3,4,5,6,7,8,9]]
Explanation:
1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 = 45
There are no other valid combinations.
```

```
class Solution:
    def combinationSum3(self, k: int, n: int) -> List[List[int]]:
        visited = [0]*10
        res=[]
        self.combinationSum3Util(visited,res,k,0,[],0,n,1)
        return res

    def combinationSum3Util(self,visited,res,k,box,asf,total,n,llb):
        if total==n:
            if box == k:
                temp = asf[:]
                res.append(temp)
            return



        for i in range(llb,10):
            if visited[i]==0:
                visited[i]=1
                self.combinationSum3Util(visited,res,k,box+1,asf+
[i],total+i,n,i+1)
                visited[i]=0
```