All Palindromic Permutations

1. You are given a string of length n.
2. You have to print all the palindromic permutations of the given string.
3. If no palindromic permutation exists for the given string, print "-1".
Note -> Check out the question video and write the recursive code as it is intended without changing signature. The judge can't force you but intends you to teach a concept.

1 <= length of string <= 15

aaabb

ababa
baaab

```
import collections


def palindromicPermutation(string):
    fmap = collections.Counter(string)
    odd = ''
    oddNum = 0
    length = 0
    for key, val in fmap.items():
        if val % 2 != 0:
            odd = key
            oddNum += 1
        fmap[key] = val // 2
        length += val // 2

    if oddNum > 1:
        return -1
    helper(1, length, fmap, odd, '')
    return


def helper(idx, length, fmap, odd, ssf):
    if idx > length:
        right = ssf[::-1]
        if odd != '':
            print(ssf + odd + right)
        else:
```

```
            print(ssf + right)
        return

    for key in fmap:
        if fmap[key] > 0:
            fmap[key] = fmap[key] - 1
            helper(idx + 1, length, fmap, odd, ssf + key)
            fmap[key] = fmap[key] + 1

string = 'aabbcc'

palindromicPermutation(string)
```