Problem Statement

Did you konw taht slcrabmed wdros can be raed wtih vrey ltlite erfoft? Oh, sorry. Did you know that scrambled words can be read with very little effort? We can interpret most heavily mangled words because our brains "see" the similarity with the original words and catch the meaning of the text. It is claimed that as long as the first and last letters in every word remain unchanged, we can shuffle the rest of the letters within each word to obtain a reasonably readable text. We want to try the following strategy of scrambling letters in words:

1. Leave the first and last letters in their original positions, and remove the rest of the letters. For example, "alphabet" becomes "a_____t" (underscores represent empty positions).
2. Sort the removed letters alphabetically. In this example, the removed letters are "lphabe", and they are sorted to become "abehlp".
3. If any removed letters still exist, take the first one and move it to the leftmost empty position in the string. In this case, the string becomes "aa_____t", and the remaining removed letters are now "behlp".
4. If any removed letters still exist, take the first one and move it to the rightmost empty position in the string. So, the string becomes "aa____bt", and the remaining removed letters are now "ehlp".
5. Repeat steps 3 and 4 until no removed letters remain. In this example, the string becomes: "aae___bt" -> "aae__hbt" -> "aael_hbt" -> "aaelphbt".

You are given a **text** containing a lowercase word. Return the scrambled word following the described procedure.

Definition

Class:

Scramble
Method:

scrambleWord
Parameters:

string
Returns:

string
Method signature:

def scrambleWord(self, text):

Limits

Time limit (s):

2.000
Memory limit (MB):

64

Constraints

- **text** will contain between 2 and 50 characters, inclusive.
- **text** will contain only lowercase letters ('a'-'z').

Examples

0)
"alphabet"
Returns: "aaelphbt"
Example from the problem statement.
1)
"abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwx"
Returns: "aabcdefghijklmnopqrstuvwyzxwvutsrqponmlkjihgfedcbx"
2)
"aa"
Returns: "aa"
3)
"abdfheca"
Returns: "abdfheca"
4)
"aaaaaaaaa"
Returns: "aaaaaaaaa"
5)
"uodoutivesbegckw"
Returns: "ubdeiosuvtokgecw"
6)
"zaabz"
Returns: "zabaz"


```
class Scramble:

  def scrambleWord(self, text):

    n=len(text)

    if(n<=3):

      return text
```

```
    p1=text[0]
    #print(p1)
    p4=text[n-1]
    #print(p4)
    t=text[1:n-1]
    t=''.join(sorted(t))
    #print(t)
    p2,p3="",""

    for i in range(n-2):
      if i%2==0:
        p2+=t[i]
      else:
        p3+=t[i]

    p3=p3[::-1]
    return p1+p2+p3+p4


obj=Scramble()
print(obj.scrambleWord("karine"))
```