```
1.  Write a Python program to add 'ing' at the end of a given string (length shoul
d be at least 3. If the given string already ends with 'ing' then add 'ly' instead.
If the string length of the given string is less than 3, leave it unchanged
Sample String : 'abc'
Expected Result : 'abcing'
Sample String : 'string'
Expected Result : 'stringly'
```

In [24]:

```python
# Time complexity: O(1)
def modify_string(string):
    if len(string) < 3:
        return string
    elif string[-3:] == "ing":
        return string + "ly"
    else:
        return string + "ing"

print("Case-1:")
string = 'abc'
result = modify_string(string)
print(f"The original string is {string} || and The modified string is {result}")
print("-------------------------------------------------------------------------")
print("Case-2:")
string = 'string'
result = modify_string(string)
print(f"The original string is {string} || and The modified string is {result}")
print("-------------------------------------------------------------------------")
print("Case-3:")
string = 'st'
result = modify_string(string)
print(f"The original string is {string} || and The modified string is {result}")
print("-------------------------------------------------------------------------")
```

```
Case-1:
The original string is abc || and The modified string is abcing
-------------------------------------------------------------------------
Case-2:
The original string is string || and The modified string is stringly
-------------------------------------------------------------------------
Case-3:
The original string is st || and The modified string is st
-------------------------------------------------------------------------
```

```
2.Write a Python program to find the first appearance of the substring 'not' and 'p
oor' from a given string, if 'not' follows the 'poor', replace the whole 'not'...'p
oor' substring with 'good'. Return the resulting string.
Sample String : 'The lyrics is not that poor!'
'The lyrics is poor!'
Expected Result : 'The lyrics is good!'
'The lyrics is poor!'
```

In [102]:

```python
def find_pos(string, key1, key2, not_pos, poor_pos):
    str_list = string.split(" ")
    for i in range(len(str_list)):
        if key1 in str_list[i]:
            not_pos = i
        if key2 in str_list[i]:
            poor_pos = i
    return [not_pos, poor_pos, str_list]


string = "The lyrics is not that poor!"
not_pos = 1
poor_pos = 1
positions = find_pos(string.lower(), "not", "poor", not_pos, poor_pos)

not_pos, poor_pos, str_list = positions[0], positions[1], positions[2]
new_str = ""

if not_pos < poor_pos:
    for i in range(not_pos):
        if i == 0:
            new_str = new_str + " " + str_list[i].capitalize()
        else:
            new_str = new_str + " " + str_list[i]
    print(new_str + " " + "good!")

else:
    for i in str_list:
         if i == 0:
            new_str = new_str + " " + str_list[i].capitalize()
         else:
            new_str = new_str + " " + str_list[i]
    print(new_str)
```

```
 The lyrics is good!
```

3. Write a Python program to count the occurrences of each word in a given sentence

In [3]:

```
def count_occurances(sentence):
    sent_array = sentence.split(" ")
    counts = {}

    for i in sent_array:
        if i in counts:
            counts[i] = counts[i] + 1
        else:
            counts[i] = 1
    return counts

sentence = "Hello ra how u Hello"
print("The count of occurance is each word is as follows:")
print(count_occurances(sentence))
```

```
The count of occurance is each word is as follows:
{'Hello': 2, 'ra': 1, 'how': 1, 'u': 1}
```

4. Write a Python program that accepts a comma separated sequence of words as input and prints the unique words in sorted form (alphanumerically).
Sample Words : red, white, black, red, green, black
Expected Result : black, green, red, white,red

In [20]:

```
""" Time complexity: (O(nlogn))"""
def mergesort(array):
    if len(array) == 1:
        return array
    middle = len(array) // 2
    leftside = mergesort(array[: middle])
    rightside = mergesort(array[middle:])

    # call the recursive functions
    return mergesortarray(leftside, rightside)

def mergesortarray(leftside, rightside):
    sorted_array = [None] * (len(leftside) + len(rightside))
    i = j = k = 0
    while i < len(leftside) and j < len(rightside):
        if  leftside[i] <= rightside[j]:
            sorted_array[k] = leftside[i]
            i = i + 1
        else:
            sorted_array[k] = rightside[j]
            j = j + 1
        k = k + 1

    # if any element left on leftside
    while i < len(leftside):
        sorted_array[k] = leftside[i]
        i = i + 1
        k = k + 1

    while j < len(rightside):
        sorted_array[k] = rightside[j]
        j = j + 1
        k = k + 1

    return sorted_array


sentence = input("Enter the sentence:")
print("sentence before sorting..........................")
print(sentence)
sort = mergesort(sentence.split(", "))
print("**************************************************")
print("sentence after sorting............................")
sorted_sentence = ""
for i in sort:
    print(f"{i}, ", end="")
```

```
Enter the sentence:red, white, black, red, green, black
sentence before sorting..........................
red, white, black, red, green, black
**************************************************
sentence after sorting............................
black, black, green, red, red, white, 
```

```
5.Write a Python program to get a string made of the first 2 and the last
2 chars from a given a string. If the string length is less than 2, return instead

of the empty string.
Sample String : 'w3resource'
Expected Result : 'w3ce'
Sample String : 'w3'
Expected Result : 'w3w3'
Sample String : ' w'
Expected Result : Empty String
```

In [33]:

```
def create_string(string):
    if len(string) < 2:
        return "Empty String"
    else:
        return string[ :2] + string[-2: ]

print("Case1:")
string = 'w3resource'
print(f"Sample string:{string}")
print(f"Generated string:{create_string(string)}")
print("-------------------------------------------")
print("Case2:")
string = 'w3'
print(f"Sample string:{string}")
print(f"Generated string:{create_string(string)}")
print("-------------------------------------------")
print("Case3:")
string = 'w'
print(f"Sample string:{string}")
print(f"Generated string:{create_string(string)}")
```

```
Case1:
Sample string:w3resource
Generated string:w3ce
-------------------------------------------
Case2:
Sample string:w3
Generated string:w3w3
-------------------------------------------
Case3:
Sample string:w
Generated string:Empty String
```