**Exercise**

**Text Analysis**

You have been recruited by your friend, a linguistic enthusiast, to create a utility tool that can perform analysis on a given piece of text. Complete the class 'analysedText' with the following methods -

* Constructor (\_\_init\_\_) - This method should take the argument text, make it lower case, and remove all punctuation. Assume only the following punctuation is used: period (.), exclamation mark (!), comma (,) and question mark (?). Assign this newly formatted text to a new attribute called fmtText.
* freqAll - This method should create and **return** dictionary of all unique words in the text, along with the number of times they occur in the text. Each key in the dictionary should be the unique word appearing in the text and the associated value should be the number of times it occurs in the text. Create this dictionary from the fmtText attribute.
* freqOf - This method should take a word as an argument and **return** the number of occurrences of that word in fmtText.

The skeleton code has been given to you. Docstrings can be ignored for the purpose of the exercise.  
***Hint: Some useful functions are replace(), lower(), split(), count()***

**Hint for** **implementing Constructor**

The lower() function converts all characters in the string to lowercase.

The replace() function takes two arguments: the text to search for and the text to replace it with. Try calling this function for each punctuation you want to remove and replace it with a blank character, ''

You can define a class attribute and assign it a value with the following generic recipe: self.attribute\_name = value

**Hint for implementing freqAll**

You can create a list of all words in fmtText using the split() and by using the whitespace character, ' ' as the delimiter.

Using set() with a list as the argument will return a set with all the unique elements in the list. Try iterating over the elements in this set to create the keys for a dictionary. The count() function will return the number of occurrences of the argument in list. For example, ["hi", "hi", "hello"].count("hi") will return 2. This can be used to set the values for each key-value pair in the dictionary.

**Hint for implementing freqOf**

Try calling the freqAll method you implemented above and assign it to a variable. You will now have a dictionary with the unique words that appear in fmtText as the keys, and the number of times they appear as the value.

You can use this dictionary to return the number of occurrences of the word that was given as an argument to the freqOf method.

If the word given as an argument does not appear in the text, return 0. You can check if a string is a key in the dictionary using the following code recipe: if item in my\_dictionary:

**MY CODE:**

**class** analysedText(object):

**def** \_\_init\_\_ (self, text):

​

*# TODO: Remove the punctuation from <text> and make it lower case.*

lower\_text **=** text.replace(".","").replace("!","").replace(",","").replace("?","")

lower\_text **=** lower\_text.lower()

*# TODO: Assign the formatted text to a new attribute called "fmtText"*

self.fmtText **=** lower\_text

**def** freqAll(self):

​

*# TODO: Split the text into a list of words*

split\_text **=** self.fmtText.split(' ')

*# TODO: Create a dictionary with the unique words in the text as keys*

*# And the number of times they occur in the text as values*

fmtDict **=** {}

**for** text **in** set(split\_text):

fmtDict[text] **=** split\_text.count(text)

*# return the created dictionary*

**return** fmtDict

**def** freqOf(self, word):

​

*# TODO: return the number of occurrences of <word> in <fmtText>*

dict **=** self.freqAll()

**if** word **in** dict:

**return** dict[word]

**else**:

**return** 0

**TEST CODE:**

**import** sys

​

sampleMap **=** {'eirmod': 1,'sed': 1, 'amet': 2, 'diam': 5, 'consetetur': 1, 'labore': 1, 'tempor': 1, 'dolor': 1, 'magna': 2, 'et': 3, 'nonumy': 1, 'ipsum': 1, 'lorem': 2}

​

**def** testMsg(passed):

**if** passed:

**return** 'Test Passed'

**else** :

**return** 'Test Failed'

​

print("Constructor: ")

**try**:

samplePassage **=** analysedText("Lorem ipsum dolor! diam amet, consetetur Lorem magna. sed diam nonumy eirmod tempor. diam et labore? et diam magna. et diam amet.")

print(testMsg(samplePassage.fmtText **==** "lorem ipsum dolor diam amet consetetur lorem magna sed diam nonumy eirmod tempor diam et labore et diam magna et diam amet"))

**except**:

print("Error detected. Recheck your function " )

print("freqAll: ")

**try**:

wordMap **=** samplePassage.freqAll()

print(testMsg(wordMap**==**sampleMap))

**except**:

print("Error detected. Recheck your function " )

print("freqOf: ")

**try**:

passed **=** **True**

**for** word **in** sampleMap:

**if** samplePassage.freqOf(word) **!=** sampleMap[word]:

passed **=** **False**

**break**

print(testMsg(passed))

**except**:

print("Error detected. Recheck your function " )

**OUTPUT:**

Constructor:

Test Passed

freqAll:

Test Passed

freqOf:

Test Passed

**SOLUTION:**

**class** analysedText(object):

**def** \_\_init\_\_ (self, text):

*# Remove punctuation*

formattedText **=** text.replace('.','').replace('!','').replace('?','').replace(',','')

*# make text lowercase*

formattedText **=** formattedText.lower()

self.fmtText **=** formattedText

**def** freqAll(self):

*# split text into words*

wordList **=** self.fmtText.split(' ')

*# Create dictionary*

freqMap **=** {}

**for** word **in** set(wordList): *# use set to remove duplicates in list*

freqMap[word] **=** wordList.count(word)

**return** freqMap

**def** freqOf(self,word):

*# get frequency map*

freqDict **=** self.freqAll()

**if** word **in** freqDict:

**return** freqDict[word]

**else**:

**return** 0