Name:Viviyan Richards W

D no:205229133

In [2]:
```
import nltk
```

In [3]:
```
nltk.download('wordnet')
nltk.download('punkt')
```

```
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\Angelan\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\Angelan\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

Out[3]:
```
True
```

In [5]:
```
text = "This is Andrew's text, isn't it?"
```

1. How many tokens are there if you use WhitespaceTokenizer?. Print tokens.

In [6]:
```
tokenizer = nltk.tokenize.WhitespaceTokenizer()
tokens = tokenizer.tokenize(text)
print(len(tokens))
print(tokens)
```

```
6
['This', 'is', "Andrew's", 'text,', "isn't", 'it?']
```

2 . How many tokens are there if you use TreebankWordTokenizer?. Print tokens.

In [7]:
```
tokenizer = nltk.tokenize.TreebankWordTokenizer()
tokens = tokenizer.tokenize(text)
print(len(tokens))
print(tokens)
```

```
10
['This', 'is', 'Andrew', "'s", 'text', ',', 'is', "n't", 'it', '?']
```

3 . How many tokens there are if you use WordPunctTokenizer?. Print tokens.

In [8]:
```
tokenizer = nltk.tokenize.WordPunctTokenizer()
tokens = tokenizer.tokenize(text)
print(len(tokens))
print(tokens)
```

```
12
['This', 'is', 'Andrew', "'", 's', 'text', ',', 'isn', "'", 't', 'it', '?']
```

EXERCISE-2

1. Open the file: O. Henry's The Gift of the Magi (gift-of-magi.txt).

In [9]:
```
import re
f = open("gift-of-magi.txt", encoding='utf-8')
con=f.read()
print(con)
```

```
The Gift of the Magi
by O. Henry

One dollar and eighty-seven cents. That was all. And sixty cents of it was in
pennies. Pennies saved one and two at a time by bulldozing the grocer and the
vegetable man and the butcher until one's cheeks burned with the silent imput
ation of parsimony that such close dealing implied. Three times Della counted
it. One dollar and eighty-seven cents. And the next day would be Christmas.

There was clearly nothing left to do but flop down on the shabby little couch
and howl. So Della did it. Which instigates the moral reflection that life is
made up of sobs, sniffles, and smiles, with sniffles predominating.

While the mistress of the home is gradually subsiding from the first stage to
the second, take a look at the home. A furnished flat at $8 per week. It did
not exactly beggar description, but it certainly had that word on the look-ou
t for the mendicancy squad.

In the vestibule below was a letter-box into which no letter would go, and an
```

2 . Write a Python script to print out the following:

1. How many word tokens there are

In [10]:
```
tokenizer = nltk.tokenize.WhitespaceTokenizer()
tokens = tokenizer.tokenize(con)
print(len(tokens))
```

```
2074
```

2 . How many word types there are, (word types are a unique set of words)

In [11]:
```
from nltk import *
data=FreqDist(tokens)
data
```

Out[11]:
```
FreqDist({'the': 107, 'and': 74, 'a': 64, 'of': 51, 'to': 41, 'was': 26, 'she':
25, 'in': 24, 'had': 21, 'her': 21, ...})
```

3 . Top 20 most frequent words and their counts

In [24]:
```
data.most_common(20)
```

Out[24]:
```
[('the', 107),
 ('and', 74),
 ('a', 64),
 ('of', 51),
 ('to', 41),
 ('was', 26),
 ('she', 25),
 ('in', 24),
 ('had', 21),
 ('her', 21),
 ('that', 20),
 ('it', 19),
 ('at', 19),
 ('with', 19),
 ('for', 19),
 ('his', 17),
 ('on', 16),
 ('I', 14),
 ('Jim', 13),
 ('were', 11)]
```

4 . Words that are at least 10 characters long and their counts

In [22]:
```
from nltk import *
text=[w for w in tokens if len(w)>10]
print(text)
freq=FreqDist(text)
freq
```

```
['eighty-seven', 'eighty-seven', 'predominating.', 'description,', 'appertainin
g', '"Dillingham"', '"Dillingham"', 'contracting', 'calculated.', 'sterling--so
mething', 'longitudinal', 'brilliantly,', 'possessions', "grandfather's.", '"So
fronie."', 'proclaiming', 'meretricious', 'ornamentation--as', 'description',
'intoxication', 'close-lying', 'wonderfully', 'critically.', 'eighty-seven', 't
wenty-two--and', 'disapproval,', "Christmas!'", 'laboriously,', 'inconsequentia
l', 'difference?', 'mathematician', 'illuminated', 'necessitating', 'tortoise-s
hell,', 'possession.', 'men--wonderfully', 'duplication.']
```

Out[22]:
```
FreqDist({'eighty-seven': 3, '"Dillingham"': 2, 'predominating.': 1, 'descripti
on,': 1, 'appertaining': 1, 'contracting': 1, 'calculated.': 1, 'sterling--some
thing': 1, 'longitudinal': 1, 'brilliantly,': 1, ...})
```

5 . 10+ characters-long words that occur at least twice, sorted from most frequent to least

```
In [23]: text = [w for w in tokens if len(w)>10]
         s=FreqDist(text)
         s
```

```
Out[23]: FreqDist({'eighty-seven': 3, '"Dillingham"': 2, 'predominating.': 1, 'descripti
         on,': 1, 'appertaining': 1, 'contracting': 1, 'calculated.': 1, 'sterling--some
         thing': 1, 'longitudinal': 1, 'brilliantly,': 1, ...})
```

```
In [28]: for i,j in freq.items():
             if len(i) > 10 and j>2:
                 print(i,j)
```

```
eighty-seven 3
```

EXERCISE -3:

List Comprehension

STEP-1

```
In [38]: fname = "./data/austen-emma.txt"
         f = open("austen-emma.txt", encoding='utf-8')
         etxt=f.read()
         f.close()
```

```
In [39]: etxt[-200:]
```

```
Out[39]: 'e deficiencies, the wishes,\nthe hopes, the confidence, the predictions of the
         small band\nof true friends who witnessed the ceremony, were fully answered\nin
         the perfect happiness of the union.\n\n\nFINIS\n'
```

In [40]:
```
tokenizer = nltk.tokenize.WhitespaceTokenizer()
tokens = tokenizer.tokenize(etxt)
tokens[-20:]
```

Out[40]:
```
['small',
 'band',
 'of',
 'true',
 'friends',
 'who',
 'witnessed',
 'the',
 'ceremony,',
 'were',
 'fully',
 'answered',
 'in',
 'the',
 'perfect',
 'happiness',
 'of',
 'the',
 'union.',
 'FINIS']
```

In [41]:
```
etoks = nltk.word_tokenize(etxt.lower())
etoks[-20:]
```

Out[41]:
```
['of',
 'true',
 'friends',
 'who',
 'witnessed',
 'the',
 'ceremony',
 ',',
 'were',
 'fully',
 'answered',
 'in',
 'the',
 'perfect',
 'happiness',
 'of',
 'the',
 'union',
 '.',
 'finis']
```

In [42]:
```
len(etoks)
```

Out[42]:
```
191781
```

In [43]:
```
etypes=sorted(set(etoks))
```

```
In [44]: etypes[-10:]
```

```
Out[44]: ['younger',
          'youngest',
          'your',
          'yours',
          'yourself',
          'yourself.',
          'youth',
          'youthful',
          'zeal',
          'zigzags']
```

```
In [45]: len(etypes)
```

```
Out[45]: 7944
```

```
In [46]: efreq = nltk.FreqDist(etoks)
```

```
In [47]: efreq['beautiful']
```

```
Out[47]: 24
```

STEP 2: list-comprehend Emma

```
In [48]: etxt
```

```
Out[48]: '[Emma by Jane Austen 1816]\n\nVOLUME I\n\nCHAPTER I\n\n\nEmma Woodhouse, han
dsome, clever, and rich, with a comfortable home\nand happy disposition, seem
ed to unite some of the best blessings\nof existence; and had lived nearly tw
enty-one years in the world\nwith very little to distress or vex her.\n\nShe
was the youngest of the two daughters of a most affectionate,\nindulgent fath
er; and had, in consequence of her sister\'s marriage,\nbeen mistress of his
house from a very early period.  Her mother\nhad died too long ago for her to
have more than an indistinct\nremembrance of her caresses; and her place had
been supplied\nby an excellent woman as governess, who had fallen little shor
t\nof a mother in affection.\n\nSixteen years had Miss Taylor been in Mr. Woo
dhouse\'s family,\nless as a governess than a friend, very fond of both daugh
ters,\nbut particularly of Emma.  Between _them_ it was more the intimacy\nof
sisters.  Even before Miss Taylor had ceased to hold the nominal\noffice of g
overness, the mildness of her temper had hardly allowed\nher to impose any re
straint; and the shadow of authority being\nnow long passed away, they had be
en living together as friend and\nfriend very mutually attached, and Emma doi
ng just what she liked;\nhighly esteeming Miss Taylor\'s judgment, but direct
ed chiefly by\nher own.\n\nThe real evils, indeed, of Emma\'s situation were
the power of having\nrather too much her own way, and a disposition to think
```

Question 1: Words with prefix and suffix

What are the words that start with 'un' and end in 'able'?

```
In [49]: [word for word in tokens if word.startswith("un") & word.endswith("able")]
```

```
Out[49]: ['unexceptionable',
          'unsuitable',
          'unreasonable',
          'unreasonable',
          'uncomfortable',
          'unfavourable',
          'unexceptionable',
          'uncomfortable',
          'unpersuadable',
          'unavoidable',
          'unsuitable',
          'unmanageable',
          'unreasonable',
          'unobjectionable',
          'unpersuadable',
          'unexceptionable',
          'unpardonable',
          'unmanageable',
          'unfavourable',
          'unaccountable',
          'unable',
          'unable',
          'unpardonable',
          'unexceptionable',
          'unreasonable',
          'unreasonable',
          'unpardonable',
          'unexceptionable',
          'unreasonable']
```

Question 2: Length

How many Emma word types are 15 characters or longer? Exclude hyphenated words.

```
In [50]: tokenizer = nltk.tokenize.WordPunctTokenizer()
         toke= tokenizer.tokenize(etxt)
```

In [51]:
```
[word for word in toke if len(word)>15]
```

Out[51]:
```
['companionableness',
 'misunderstanding',
 'incomprehensible',
 'undistinguishing',
 'unceremoniousness',
 'Disingenuousness',
 'disagreeableness',
 'misunderstandings',
 'misunderstandings',
 'misunderstandings',
 'misunderstandings',
 'disinterestedness',
 'unseasonableness']
```

Average word length

What's the average length of all Emma word types?

In [53]:
```
average=sum(len(word)for word in toke)/len(toke)
average
```

Out[53]:
```
3.755268231589122
```

In [54]:
```
lg = []
for i in toke:
    if len(i)>15:
        lg.append(i)
print(lg)
```

```
['companionableness', 'misunderstanding', 'incomprehensible', 'undistinguishin
g', 'unceremoniousness', 'Disingenuousness', 'disagreeableness', 'misunderstand
ings', 'misunderstandings', 'misunderstandings', 'misunderstandings', 'disinter
estedness', 'unseasonableness']
```

Question 4: Word frequency

How many Emma word types have a frequency count of 200 or more?

In [57]:
```
from nltk import *
fdiemm = FreqDist(toke)
```

In [59]:
```
for i,j in fdiemm.items():
    if j > 200:
        print(i,j)
```

```
your 337
sure 204
will 559
are 447
You 303
may 213
me 564
do 580
about 246
Knightley 389
out 212
quite 269
," 421
has 243
should 366
can 270
nothing 237
Elton 385
Churchill 223
Frank 208
```

How many word types appear only once?

In [60]:
```
for i,j in fdiemm.items():
    if j == 1:
        print(i,j)
```

```
Austen 1
1816 1
] 1
vex 1
indistinct 1
caresses 1
nominal 1
mildness 1
impose 1
esteeming 1
disadvantages 1
misfortunes 1
Sorrow 1
mournful 1
debt 1
tenderer 1
valetudinarian 1
amounting 1
equals 1
```

STEP 3: bigrams in Emma

Question 6: Bigrams

What are the last 10 bigrams

In [62]:
```
e2grams = list(nltk.bigrams(toke))
e2gramfd = nltk.FreqDist(e2grams)
```

In [63]:
```
e2gramfd
```

Out[63]:
```
FreqDist({(',', 'and'): 1879, ('Mr', '.'): 1153, ("'", 's'): 932, (';', 'and'):
866, ('."', '"'): 757, ('Mrs', '.'): 699, ('to', 'be'): 595, ('.', 'I'): 570,
(',', 'I'): 568, ('of', 'the'): 556, ...})
```

In [64]:
```
last_ten = FreqDist(dict(e2gramfd.most_common()[-10:]))
last_ten
```

Out[64]:
```
FreqDist({('who', 'witnessed'): 1, ('witnessed', 'the'): 1, ('the', 'ceremon
y'): 1, ('were', 'fully'): 1, ('fully', 'answered'): 1, ('answered', 'in'): 1,
('the', 'perfect'): 1, ('the', 'union'): 1, ('union', '.'): 1, ('.', 'FINIS'):
1})
```

Question 7: Bigram top frequency

What are the top 20 most frequent bigrams?

In [65]:
```
tokenizer = nltk.tokenize.WhitespaceTokenizer()
tokes = tokenizer.tokenize(etxt)
```

In [66]:
```
e2grams = list(nltk.bigrams(tokes))
e2gramfd = nltk.FreqDist(e2grams)
```

In [67]:
```
e2gramfd.most_common(20)
```

Out[67]:
```
[(('to', 'be'), 562),
 (('of', 'the'), 556),
 (('in', 'the'), 431),
 (('I', 'am'), 302),
 (('had', 'been'), 299),
 (('could', 'not'), 270),
 (('it', 'was'), 253),
 (('she', 'had'), 242),
 (('to', 'the'), 236),
 (('have', 'been'), 233),
 (('of', 'her'), 230),
 (('I', 'have'), 214),
 (('and', 'the'), 208),
 (('would', 'be'), 208),
 (('she', 'was'), 206),
 (('do', 'not'), 196),
 (('of', 'his'), 182),
 (('that', 'she'), 178),
 (('to', 'have'), 176),
 (('such', 'a'), 176)]
```

Question 8: Bigram frequency count

#How many times does the bigram 'so happy' appear?

In [68]:
```
for i , j in e2gramfd.items():
    if i == ('so', 'happy'):
        print(i,j)
```

```
('so', 'happy') 3
```

Question 9: Word following 'so'

What are the words that follow 'so'? What are their frequency counts? (For loop will be easier; see if you can utilize list comprehension for this.)

In [69]:
```
import re
from collections import Counter
```

```
In [70]: words = re.findall(r'so+ \w+',open('austen-emma.txt').read())
         ab = Counter(zip(words))
         print(ab)
```

```
Counter({('so much',): 95, ('so very',): 76, ('so well',): 30, ('so many',): 2
7, ('so long',): 27, ('so little',): 20, ('so far',): 17, ('so I',): 14, ('so k
ind',): 13, ('so good',): 12, ('so often',): 10, ('so soon',): 9, ('so grea
t',): 8, ('so to',): 7, ('so fond',): 7, ('so she',): 7, ('so it',): 6, ('so an
xious',): 6, ('so as',): 6, ('so you',): 6, ('so truly',): 6, ('so completel
y',): 5, ('so obliging',): 5, ('so extremely',): 5, ('so entirely',): 4, ('so h
appy',): 4, ('so interesting',): 4, ('so fast',): 4, ('so near',): 4, ('so plea
sed',): 4, ('so few',): 4, ('so that',): 4, ('so strong',): 4, ('so liberal',):
4, ('so miserable',): 4, ('so happily',): 3, ('so proper',): 3, ('so pleasantl
y',): 3, ('so superior',): 3, ('so warmly',): 3, ('so bad',): 3, ('so odd',):
3, ('so ill',): 3, ('so delighted',): 3, ('so particularly',): 3, ('so easil
y',): 3, ('so on',): 3, ('so attentive',): 3, ('so fortunate',): 3, ('so gla
d',): 3, ('so shocked',): 3, ('so at',): 3, ('so obliged',): 2, ('so perfectl
y',): 2, ('so dear',): 2, ('so busy',): 2, ('so did',): 2, ('so forth',): 2,
('so totally',): 2, ('so remarkably',): 2, ('so plainly',): 2, ('so charmin
g',): 2, ('so surprized',): 2, ('so early',): 2, ('so too',): 2, ('so easy',):
2, ('so decidedly',): 2, ('so absolutely',): 2, ('so particular',): 2, ('so dec
eived',): 2, ('so palpably',): 2, ('so clever',): 2, ('so short',): 2, ('so col
d',): 2, ('so high',): 2, ('so happened',): 2, ('so full',): 2, ('so thoroughl
y',): 2, ('so equal',): 2, ('so off',): 2, ('so naturally',): 2, ('so afrai
d',): 2, ('so deep',): 2, ('so kindly',): 2, ('so pale',): 2, ('so noble',): 2,
('so lovely',): 2, ('so mad',): 2, ('so nearly',): 2, ('so sorry',): 2, ('so ch
eerful',): 2, ('so unfeeling',): 2, ('so ready',): 2, ('so unperceived',): 1,
('so mild',): 1, ('so constantly',): 1, ('so comfortably',): 1, ('so avowed',):
1, ('so deservedly',): 1, ('so convenient',): 1, ('so just',): 1, ('so apparen
t',): 1, ('so sorrowful',): 1, ('so spent',): 1, ('so artlessly',): 1, ('so pla
in',): 1, ('so firmly',): 1, ('so genteel',): 1, ('so _then_',): 1, ('so brilli
ant',): 1, ('so seldom',): 1, ('so nervous',): 1, ('so indeed',): 1, ('so pac
k',): 1, ('so doubtful',): 1, ('so with',): 1, ('so contemptible',): 1, ('so sl
ightingly',): 1, ('so by',): 1, ('so loudly',): 1, ('so materially',): 1, ('so
hard',): 1, ('so delightful',): 1, ('so pointed',): 1, ('so equalled',): 1, ('s
o evidently',): 1, ('so immediately',): 1, ('so sought',): 1, ('so excellen
t',): 1, ('so prettily',): 1, ('so extreme',): 1, ('so wonder',): 1, ('so alway
s',): 1, ('so silly',): 1, ('so satisfied',): 1, ('so smiling',): 1, ('so prosi
ng',): 1, ('so undistinguishing',): 1, ('so apt',): 1, ('so dreadful',): 1, ('s
o respected',): 1, ('so tenderly',): 1, ('so grieved',): 1, ('so shocking',):
1, ('so conceited',): 1, ('so before',): 1, ('so prevalent',): 1, ('so heav
y',): 1, ('so swiftly',): 1, ('so spoken',): 1, ('so or',): 1, ('so overcharge
d',): 1, ('so pleasant',): 1, ('so fenced',): 1, ('so hospitable',): 1, ('so in
terested',): 1, ('so sanguine',): 1, ('so sure',): 1, ('so careless',): 1, ('so
rapidly',): 1, ('so frequent',): 1, ('so sensible',): 1, ('so misled',): 1, ('s
o blind',): 1, ('so complaisant',): 1, ('so misinterpreted',): 1, ('so activ
e',): 1, ('so pointedly',): 1, ('so striking',): 1, ('so sudden',): 1, ('so ind
ustriously',): 1, ('so partial',): 1, ('so natural',): 1, ('so inevitable',):
1, ('so lately',): 1, ('so beautifully',): 1, ('so distinct',): 1, ('so conside
rate',): 1, ('so light',): 1, ('so intimate',): 1, ('so magnified',): 1, ('so c
autious',): 1, ('so confined',): 1, ('so wish',): 1, ('so he',): 1, ('so glorio
us',): 1, ('so quick',): 1, ('so sweetly',): 1, ('so inseparably',): 1, ('so de
serving',): 1, ('so disappointed',): 1, ('so ended',): 1, ('so sluggish',): 1,
('so amiable',): 1, ('so quiet',): 1, ('so idolized',): 1, ('so cried',): 1,
('so acceptable',): 1, ('so properly',): 1, ('so reasonable',): 1, ('so delight
fully',): 1, ('so rich',): 1, ('so warm',): 1, ('so large',): 1, ('so handsomel
y',): 1, ('so abundant',): 1, ('so outree',): 1, ('so thoughtful',): 1, ('so mu
st',): 1, ('so effectually',): 1, ('so beautiful',): 1, ('so Patty',): 1, ('so
honoured',): 1, ('so close',): 1, ('so imprudent',): 1, ('so limited',): 1, ('s
o from',): 1, ('so amusing',): 1, ('so indifferent',): 1, ('so indignant',): 1,
('so said',): 1, ('so right',): 1, ('so wretched',): 1, ('so now',): 1, ('so oc
cupied',): 1, ('so unhappy',): 1, ('so highly',): 1, ('so generally',): 1, ('so
exactly',): 1, ('so double',): 1, ('so secluded',): 1, ('so regular',): 1, ('so
determined',): 1, ('so motherly',): 1, ('so the',): 1, ('so glibly',): 1, ('so
calculated',): 1, ('so thrown',): 1, ('so exclusively',): 1, ('so disgustingl
y',): 1, ('so needlessly',): 1, ('so does',): 1, ('so resolutely',): 1, ('so wo
uld',): 1, ('so infinitely',): 1, ('so fluently',): 1, ('so they',): 1, ('so im
patient',): 1, ('so briskly',): 1, ('so vigorously',): 1, ('so young',): 1, ('s
o hardened',): 1, ('so gratified',): 1, ('so received',): 1, ('so then',): 1,
('so and',): 1, ('so gratefully',): 1, ('so found',): 1, ('so placed',): 1, ('s
o lain',): 1, ('so his',): 1, ('so arranged',): 1, ('so moving',): 1, ('so walk
ing',): 1, ('so when',): 1, ('so favourable',): 1, ('so late',): 1, ('so silen
t',): 1, ('so dull',): 1, ('so irksome',): 1, ('so agitated',): 1, ('so bruta
l',): 1, ('so cruel',): 1, ('so depressed',): 1, ('so no',): 1, ('so justly',):
1, ('so astonished',): 1, ('so will',): 1, ('so simple',): 1, ('so dignifie
d',): 1, ('so suddenly',): 1, ('so a',): 1, ('so herself',): 1, ('so peremptori
ly',): 1, ('so uneasy',): 1, ('so wonderful',): 1, ('so _very_',): 1, ('so expr
essly',): 1, ('so angry',): 1, ('so anxiously',): 1, ('so strange',): 1, ('so s
toutly',): 1, ('so mistake',): 1, ('so mistaken',): 1, ('so dreadfully',): 1,
('so voluntarily',): 1, ('so satisfactory',): 1, ('so disinterested',): 1, ('so
foolishly',): 1, ('so ingeniously',): 1, ('so entreated',): 1, ('so like',): 1,
('so cordially',): 1, ('so essential',): 1, ('so designedly',): 1, ('so hast
y',): 1, ('so richly',): 1, ('so grateful',): 1, ('so tenaciously',): 1, ('so f
eeling',): 1, ('so engaging',): 1, ('so engaged',): 1, ('so hot',): 1, ('so use
ful',): 1, ('so attached',): 1, ('so peculiarly',): 1, ('so singularly',): 1,
('so taken',): 1, ('so recently',): 1, ('so fresh',): 1, ('so hateful',): 1,
('so heartily',): 1, ('so steady',): 1, ('so complete',): 1, ('so in',): 1, ('s
o suffered',): 1})
```

Question 10: Trigrams¶

What are the last 10 trigrams

In [71]:
```
e3grams = list(nltk.trigrams(tokes))
e3gramfd = nltk.FreqDist(e3grams)
```

In [72]:
```
last_ten = FreqDist(dict(e3gramfd.most_common()[-10:]))
last_ten
```

Out[72]:
```
FreqDist({('the', 'ceremony,', 'were'): 1, ('ceremony,', 'were', 'fully'): 1,
('were', 'fully', 'answered'): 1, ('fully', 'answered', 'in'): 1, ('answered',
'in', 'the'): 1, ('in', 'the', 'perfect'): 1, ('the', 'perfect', 'happiness'):
1, ('perfect', 'happiness', 'of'): 1, ('of', 'the', 'union.'): 1, ('the', 'unio
n.', 'FINIS'): 1})
```

Question 11: Trigram top frequency

What are the top 10 most frequent trigrams?

```
In [74]: e3gramfd.most_common(10)

Out[74]: [(('I', 'do', 'not'), 94),
          (('I', 'am', 'sure'), 75),
          (('would', 'have', 'been'), 55),
          (('a', 'great', 'deal'), 55),
          (('she', 'could', 'not'), 49),
          (('could', 'not', 'be'), 45),
          (('she', 'had', 'been'), 44),
          (('it', 'would', 'be'), 43),
          (('do', 'not', 'know'), 43),
          (('Mr.', 'and', 'Mrs.'), 37)]
```

Question 12: Trigram frequency count

How many times does the trigram 'so happy to' appear?

```
In [78]: for i , j in e3gramfd.items():
             if i == ('so', 'happy','to'):
                 print(i,j)
```

```
In [ ]:
```