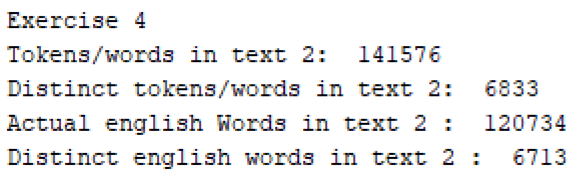
CSCI 8450 Assignment: Chapter 1

Mohan Sai Ambati

Exercise 4.

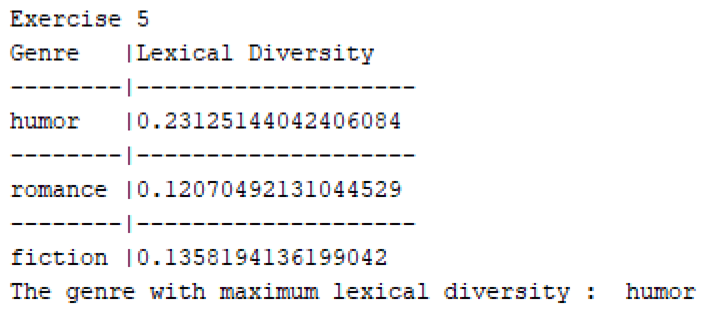
```
def exercise4():
    print('Tokens/words in text 2: ', len(text2))
    print('Distinct tokens/words in text 2: ', len(set(text2)))
    words = []
    for word in text2:
        m = re.search('([A-Z a-z])+', word)
        if m:
            words.append(m[0])
    print('Actual english Words in text 2 : ', len(words))
    print('Distinct english words in text 2 : ', len(set(words)))
    return
```



```
Exercise 4
Tokens/words in text 2:  141576
Distinct tokens/words in text 2:  6833
Actual english Words in text 2 :  120734
Distinct english words in text 2 :  6713
```

Exercise 5.

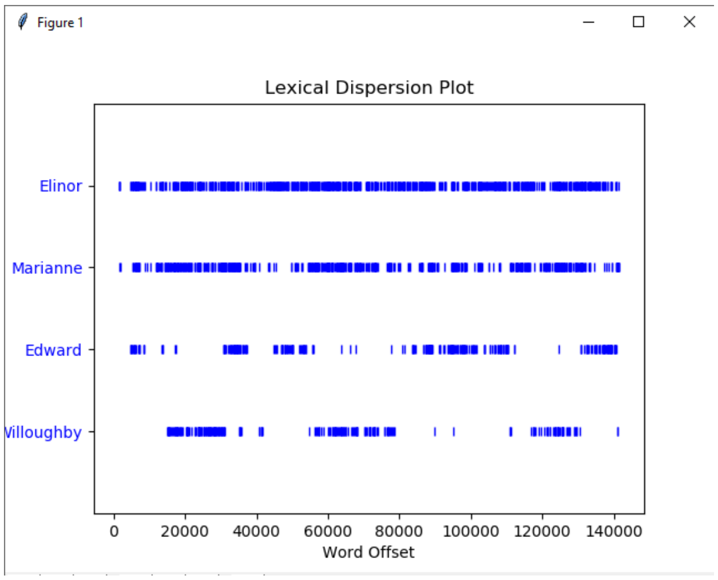
```
def lexical_diversity(text):
    return len(set(text))/len(text)

def exercise5():
    words = ['humor', 'romance', 'fiction']
    lexical_vals = []
    titles = ['Genre', 'Lexical Diversity']

    for word in words:
        lexical_vals.append(lexical_diversity(brown.words(categories=word)))

    print('{:<8}|{:<8}'.format(*titles))
    for item in zip(words, lexical_vals):
        print('{0}|{1}'.format('-' * 8, '-' * 20))
        print ('{:<8}|{:<8}'.format(*item))
    print('The genre with maximum lexical diversity : ',
words[lexical_vals.index(max(lexical_vals))])
    return
```



```
Exercise 5
Genre   |Lexical Diversity
--------|--------------------
humor   |0.23125144042406084
--------|--------------------
romance |0.12070492131044529
--------|--------------------
fiction |0.1358194136199042
The genre with maximum lexical diversity :  humor
```

Exercise 6.

```
def exercise6():
    words = ['Elinor', 'Marianne', 'Edward', 'Willoughby']
    text2.dispersion_plot(words)
    return
```



- If we observe the graph it is obvious that female character Elinor is important in text2 and closely related to another female character Marianne.
- Male characters Edward and Willoughby doesn't coincide in text2.
- If we observe the graph, Female character Marianne has high frequency if male character Willoughby has high frequency, it seems like frequency of word Marianne is directly proportional to word Willoughby. So, Willoughby and Marianne could be a couple.
- Even though female character Elinor seems to be main character in text2, we are clear that (Marianne, Willoughby) is a couple, only option we have for another couple is Edward and Elinor. Since, there is no further evidence to disprove this argument I consider that (Edward, Elinor) could be another couple.

Exercise 7.

```
def exercise7():
    collocations = text5.collocations()
    if collocations:
        print(collocations)
    return
```

```
Exercise 7
wanna chat; PART JOIN; MODE #14-19teens; JOIN PART; PART PART;
cute.-ass MP3; MP3 player; JOIN JOIN; times .. .; ACTION watches; guys
wanna; song lasts; last night; ACTION sits; -...)...- S.M.R.; Lime
Player; Player 12%; dont know; lez gurls; long time
```

Exercise 17.

```
def exercise17():
    # trail and error method for identifying sentence.
    move_front = move_back = True
    i = j = text9.index('sunset')
    while move_back or move_front:
        if text9[i] == '.':
            move_back = False
        if move_back: i -= 1
        if text9[j] == '.':
            move_front = False
        if move_front: j += 1
    i += 1
    j += 1
    print('The slice is : ', str(i) + ':' + str(j))
    print('Sliced sentence : ')
    print(" ".join(text9[i:j]))

    print("After further refinement.")
    # filter sentence
    sentence = text9[i:j]
    for k in range(1, len(sentence)):
        if sentence[k - 1].isupper() and sentence[k].isupper():
            i += 1
    print('The slice is : ', str(i) + ':' + str(j))
    print('Sliced sentence : ')
    print(" ".join(text9[i:j]))
    return
```

```
Exercise 17
The slice is :  613:644
Sliced sentence :
CHAPTER I THE TWO POETS OF SAFFRON PARK THE suburb of Saffron Park lay on the sunset side of London , as red and ragged as a cloud of sunset .
After further refinement.
The slice is :  621:644
Sliced sentence :
THE suburb of Saffron Park lay on the sunset side of London , as red and ragged as a cloud of sunset .
```

Exercise 18.

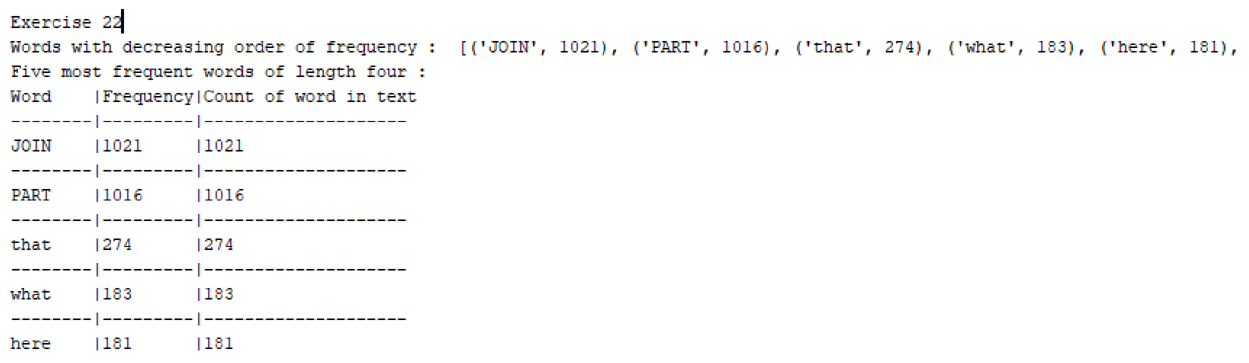
```
def exercise18():
    sent1_8 = []
    for i in range(1, 9):
        sent1_8 += globals()['sent' + str(i)]
    print('Vocabulary size : ', len(set(sent1_8)), 'Vocabulary : ',
sorted(set(sent1_8)))
    return
```

Exercise 18

Vocabulary size : 75 Vocabulary : ['!', ',', '-', '.', '1', '25', '29', '61', ':', 'ARTHUR', 'Call', 'Citizens', 'Dashwood', 'Fellow', 'God', 'House', 'I', 'In', 'Ishmael', 'JOIN', 'KING', 'MALE', 'Nov.', 'PMing', 'Pierre', 'Representatives', 'SCENE', 'SEXY', 'Senate', 'Sussex', 'The', 'Vinken', 'Whoa', '[', ']', 'a', 'and', 'as', 'attrac', 'been', 'beginning', 'board', 'clop', 'created', 'director', 'discreet', 'earth', 'encounters', 'family', 'for', 'had', 'have', 'heaven', 'in', 'join', 'lady', 'lol', 'long', 'me', 'nonexecutive', 'of', 'old', 'older', 'people', 'problem', 'seeks', 'settled', 'single', 'the', 'there', 'to', 'will', 'wind', 'with', 'years']

Exercise 22.

```
def exercise22():
    four_letter_words = [word for word in text5 if len(word) == 4]
    frequency_distribution = FreqDist(four_letter_words)
    print('Words with decreasing order of frequency : ',
          frequency_distribution.most_common(len(set(four_letter_words))))
    titles = ['Word', 'Frequency', 'Count of word in text']
    print("Five most frequent words of length four : ")
    print('{:<8}|{:<8}|{:<8}'.format(*titles))
    for word in frequency_distribution.most_common(5):
        print('{0}|{1}|{2}'.format('-' * 8, '-' * 9, '-' * 20))
        print('{:<8}|{:<9}|{:<8}'.format(word[0], word[1], text5.count(word[0])))
    return
```



```
Exercise 22
Words with decreasing order of frequency :  [('JOIN', 1021), ('PART', 1016), ('that', 274), ('what', 183), ('here', 181),
Five most frequent words of length four :
Word    |Frequency|Count of word in text
--------|---------|--------------------
JOIN    |1021     |1021
--------|---------|--------------------
PART    |1016     |1016
--------|---------|--------------------
that    |274      |274
--------|---------|--------------------
what    |183      |183
--------|---------|--------------------
here    |181      |181
```