Πανεπιστήμιου Δυτικής Αττικής
Σχολή Μηχανικών
Ανάκτηση Πληροφορίας
Εργαστήριο

Άσκηση 1

Ονοματεπώνυμο: Γάγγας Ιωάννης
Αριθμός Μητρώου: 19390038
Τμήμα Μηχανικών Πληροφορικής και Υπολογιστών
Ημερομηνία παράδωσης εργασίας: 30/11/2022

Βήμα 1

Ερώτημα 1

Α)

i)

Monty.py

```
from nltk.book import *
from collections import Counter

def Monty(text6):
    voc = len(set(text6))    #Δείκτης πλούτου λεξιλογίου
    Lanc = Counter(text6)    #Αριθμός φορών εμφάνισης της
λέξης LAUNCELOT
    PrintLanc = print(f'"LAUNCELOT" appears
{Lanc["LAUNCELOT"]} times')
    PercLanc = 100 * text6.count('LAUNCELOT') / len(text6)
#Ποσοστό εμφάνισης της λέξης
    return[voc, PrintLanc, PercLanc]
```

Αποτελέσματα εκτέλεσης

```
In [53]: Monty(text6)
"LAUNCELOT" appears 76 times
Out[53]: [2166, None, 0.4479283314669653]
```

ii)

Chat.py

```
from nltk.book import *
from collections import Counter

def Chat(text6):
    voc = len(set(text6))    #Δείκτης πλούτου λεξιλογίου
    Lanc = Counter(text6)    #Αριθμός φορών εμφάνισης της
λέξης LAUNCELOT
    PrintLanc = print(f'"omg" appears {Lanc["omg"]} times')
    PrintLanc = print(f'"OMG" appears {Lanc["OMG"]} times')
    PrintLanc = print(f'"lol" appears {Lanc["lol"]} times')
    PercLanc = 100 * text6.count('omg') / len(text6)
#Ποσοστό εμφάνισης της λέξης omg
    PercOMG = 100 * text6.count('OMG') / len(text6) #Ποσοστό
εμφάνισης της λέξης OMG
    Perclol = 100 * text6.count('lol') / len(text6) #Ποσοστό
εμφάνισης της λέξης lol
    return[voc, PrintLanc, PercLanc, PercOMG, Perclol]
```

Αποτελέσματα εκτέλεσης

```
In [9]: Chat(text5)
"omg" appears 29 times
"OMG" appears 6 times
"lol" appears 704 times
Out[9]: [6066, None, 0.06443012663852478, 0.013330371028660297, 1.5640968673628082]
```

β)

Αποτελέσματα εκτέλεσης

```
In [11]: Chat(text6)
"sword" appears 3 times
"I" appears 255 times
"rock" appears 2 times
Out[11]: [2166, None, 0.017681381505274946, 1.5029174279483704, 0.011787587670183296]
```

```
In [8]: Chat(text5)
"do" appears 168 times
"it" appears 332 times
"Corpus" appears 0 times
Out[8]: [6066, None, 0.37325038880248834, 0.7376138635858698, 0.0]
```

Από τα πειράματα, και ιδίως από το πείραμα ii), συμπαιρένεται ότι το πρόγραμμα αναγνωρίζει και τις συλλαβές που αντιστοιχούν στις λέξεις που δίνονται. Για παράδειγμα η λέξη “lol” εμφανίζεται 3 φορές εντός της λέξης “lolipop”.Παρόλ'αυτα, το nltk φαίνεται να αποτελέι ένα πολυ χρήσιμο εργαλείο για την αναγνώριση κειμένων και σχετικών διεργασιών σήμερα.

Β)

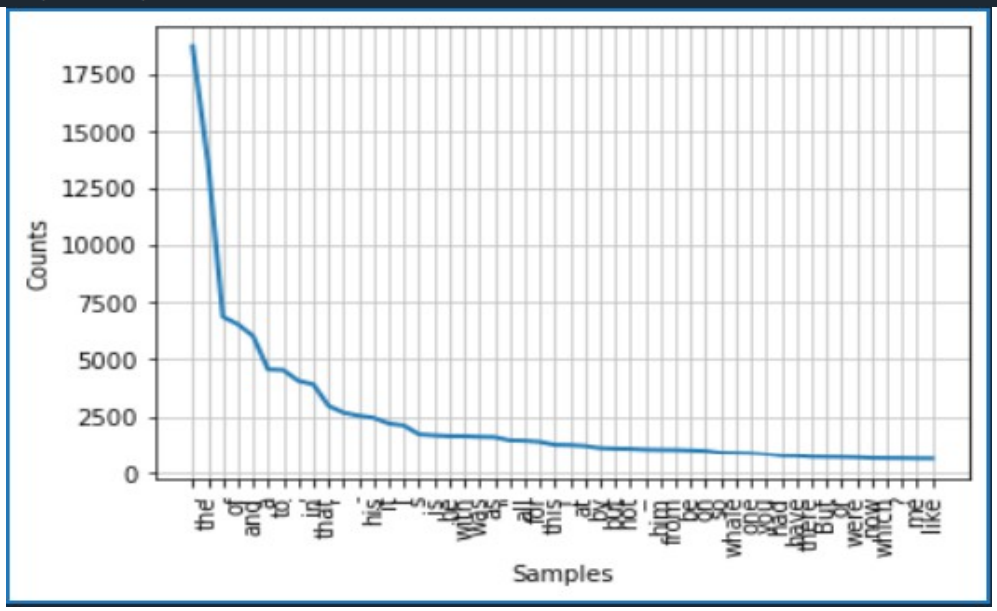
Αποτελέσματα εκτέλεσης

```
In [12]: sent1
Out[12]: ['Call', 'me', 'Ishmael', '.']
```

Γ)

Αποτελέσματα εκτέλεσης

```
In [18]: fdist1 = FreqDist(text1)

In [19]: fdist1
Out[19]: FreqDist({',': 18713, 'the': 13721, '.': 6862, 'of': 6536, 'and': 6024, 'a': 4569, 'to': 4542, ';': 4072,
'in': 3916, 'that': 2982, ...})

In [20]: fdist1.most_common(50)
Out[20]: [(',', 18713), ('the', 13721), ('.', 6862), ('of', 6536), ('and', 6024), ('a', 4569), ('to', 4542), (';',
4072), ('in', 3916), ('that', 2982), ("'", 2684), ('-', 2552), ('his', 2459), ('it', 2209), ('I', 2124), ('s',
1739), ('is', 1695), ('he', 1661), ('with', 1659), ('was', 1632), ('as', 1620), ('"', 1478), ('all', 1462), ('for',
1414), ('this', 1280), ('!', 1269), ('at', 1231), ('by', 1137), ('but', 1113), ('not', 1103), ('--', 1070), ('him',
1058), ('from', 1052), ('be', 1030), ('on', 1005), ('so', 918), ('whale', 906), ('one', 889), ('you', 841), ('had',
767), ('have', 760), ('there', 715), ('But', 705), ('or', 697), ('were', 680), ('now', 646), ('which', 640), ('?',
637), ('me', 627), ('like', 624)]
```



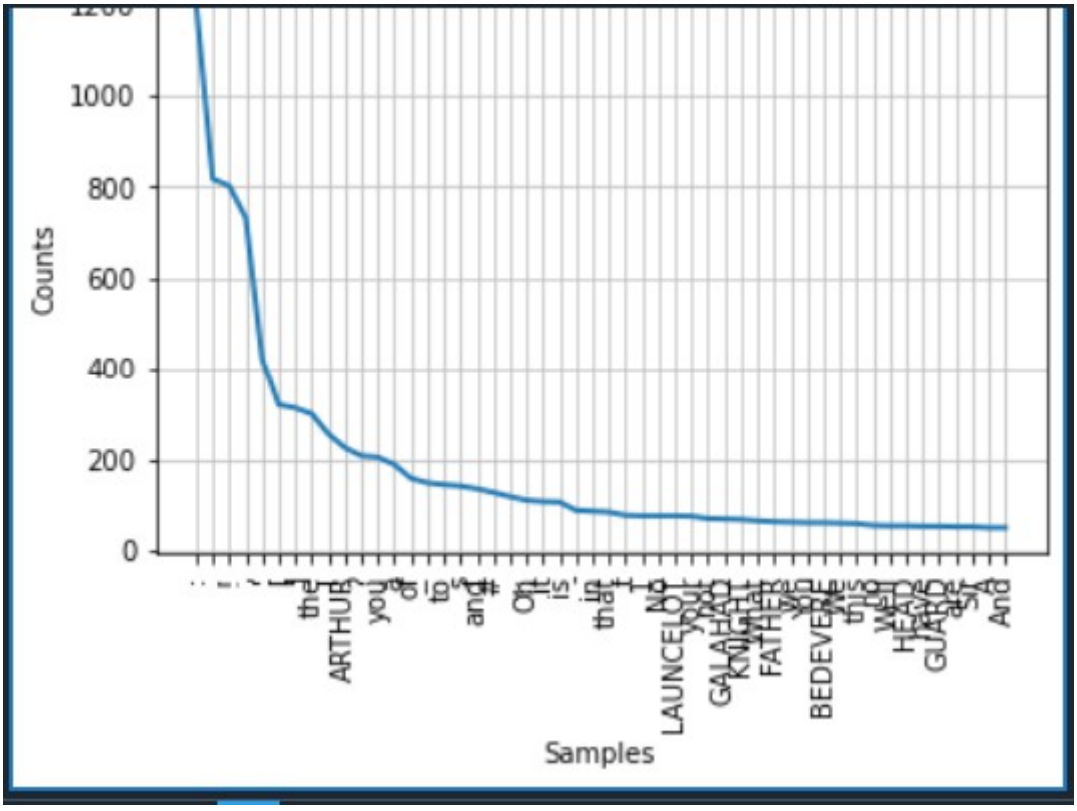
Ερώτημα 2

Από το γράφημα που εμφανίστηκε υπάρχει η δυνατότητα να βγουν συμπεράσματα σχετικά με το βιβλίο. Παρατηρώντας το γράφημα είναι φανερό ότι στον άξονα x'x υπάρχουν οι λέξεις με την μεγαλύτερη συχνότητα. Δύο από αυτές είναι “whale” και “witchy” από το οποίο μπορεί να γίνει αντιληπτό ότι το βιβλίο σχετίζεται με κάποια φάλαινα και μαγεία.

Ερώτημα 3

<u>Αποτελέσματα εκτέλεσης</u>

```
In [23]: fdist1 = FreqDist(text6)

In [24]: fdist6 = FreqDist(text6)

In [25]: fdist6
Out[25]: FreqDist({':': 1197, '.': 816, '!': 801, ',': 731, "'": 421, '[': 319, ']': 312, 'the': 299, 'I': 255,
'ARTHUR': 225, ...})

In [26]: fdist6.most_common(50)
Out[26]: [(':', 1197), ('.', 816), ('!', 801), (',', 731), ("'", 421), ('[', 319), (']', 312), ('the', 299), ('I',
255), ('ARTHUR', 225), ('?', 207), ('you', 204), ('a', 188), ('of', 158), ('--', 148), ('to', 144), ('s', 141),
('and', 135), ('#', 127), ('...', 118), ('Oh', 110), ('it', 107), ('is', 106), ('-', 88), ('in', 86), ('that', 84),
('t', 77), ('1', 76), ('No', 76), ('LAUNCELOT', 76), ('your', 75), ('not', 70), ('GALAHAD', 69), ('KNIGHT', 68),
('What', 65), ('FATHER', 63), ('we', 62), ('You', 61), ('BEDEVERE', 61), ('We', 60), ('this', 59), ('no', 55),
('Well', 54), ('HEAD', 54), ('have', 53), ('GUARD', 53), ('are', 52), ('Sir', 52), ('A', 50), ('And', 50)]

In [27]: fdist6.plot(50)
Out[27]: <AxesSubplot:xlabel='Samples', ylabel='Counts'>
```



Η λέξη που εμφανίζει την μεγαλύτερη συχνότητα εκτός της λέξης “LAUNCELOT” είναι η λέξη “ARTHUR”. Από το παραπάνω γράφημα γίνονται αντιληπτά τα ονομάτα των χαρακτήρων του έργου. Παράλληλα, λόγω των λέξεων σε κεφαλαία μπορεί να υποθεί ότι είναι ένα έργο με διαλόγους, επομένως μπορει να αποτελεί σενάριο μίας ταινίας ή ενος θεατρικού έργου. Τέλος, λόγω των ονομάτων συμπαιρένεται ότι κατα πάσα πιθανότητα το έργο διαδραματίζεται σε κάποια μεσσαιωνική εποχή.

Βήμα 2

Αποτελέσματα εκτέλεσης

```
In [28]: sent1
Out[28]: ['Call', 'me', 'Ishmael', '.']

In [29]: tokens1=sent1

In [30]: normalized_sent1=[x.lower() for x in tokens1]

In [31]: normalized_sent1
Out[31]: ['call', 'me', 'ishmael', '.']
```

Ερώτημα 4

Με την εκτέλεση των παραπάνω εντολών παρατηρείται ότι μεταρέπονται όλα τα κεφαλαία γράμματα της πρώτης πρότασης του βιβλίου σε μικρά. Όσον αφορα την επιρροή που έχει η εντολή αυτή στις παραπάνω ερωτησης είναι προφανής. Στην πιθανή περίπτωση που το έργο “Monty Python and the Holy Grail” έχει διαλόγους μεταξύ χαρακτήρων, τα στοιχεία που θα εκληφθούν από την έρευνα να δεν θα μπορουν να υποστηρίξουν τα συμπεράσματα αυτά και μπορεί να χαθεί σημαντικό ποσό πληροφορίας. Αντιθέτως, με αυτόν τον τρόπο το nltk μπορεί να αναγνωρίσει και τις λέξεις οι οποίες ξεκινούν με κεφαλαίο γράμμα σε περίπτωση που ψάχνονται, όπως π.χ. η λέξη “LAUNCELOT”.

Αποτελέσματα εκτέλεσης

```
In [35]: tokens1 = text2[:200]

In [35]:

In [36]: print(text2[:200]
    ...: )
['[', 'Sense', 'and', 'Sensibility', 'by', 'Jane', 'Austen', '1811', ']', 'CHAPTER', '1', 'The', 'family', 'of',
'Dashwood', 'had', 'long', 'been', 'settled', 'in', 'Sussex', '.', 'Their', 'estate', 'was', 'large', ',', 'and',
'their', 'residence', 'was', 'at', 'Norland', 'Park', ',', 'in', 'the', 'centre', 'of', 'their', 'property', ',',
'where', ',', 'for', 'many', 'generations', ',', 'they', 'had', 'lived', 'in', 'so', 'respectable', 'a', 'manner',
'as', 'to', 'engage', 'the', 'general', 'good', 'opinion', 'of', 'their', 'surrounding', 'acquaintance', '.',
'The', 'late', 'owner', 'of', 'this', 'estate', 'was', 'a', 'single', 'man', ',', 'who', 'lived', 'to', 'a',
'very', 'advanced', 'age', ',', 'and', 'who', 'for', 'many', 'years', 'of', 'his', 'life', ',', 'had', 'a',
'constant', 'companion', 'and', 'housekeeper', 'in', 'his', 'sister', '.', 'But', 'her', 'death', ',', 'which',
'happened', 'ten', 'years', 'before', 'his', 'own', ',', 'produced', 'a', 'great', 'alteration', 'in', 'his',
'home', ';', 'for', 'to', 'supply', 'her', 'loss', ',', 'he', 'invited', 'and', 'received', 'into', 'his', 'house',
'the', 'family', 'of', 'his', 'nephew', 'Mr', '.', 'Henry', 'Dashwood', ',', 'the', 'legal', 'inheritor', 'of',
'the', 'Norland', 'estate', ',', 'and', 'the', 'person', 'to', 'whom', 'he', 'intended', 'to', 'bequeath', 'it',
'.', 'In', 'the', 'society', 'of', 'his', 'nephew', 'and', 'niece', ',', 'and', 'their', 'children', ',', 'the',
'old', 'Gentleman', "'", 's', 'days', 'were', 'comfortably', 'spent', '.', 'His', 'attachment', 'to', 'them',
'all', 'increased', '.', 'The', 'constant']
```

```
In [38]: from nltk.stem.porter import PorterStemmer

In [39]: porter = PorterStemmer()

In [40]: [porter.stem(t) for t in tokens1]
Out[40]: ['[', 'sens', 'and', 'sensibl', 'by', 'jane', 'austen', '1811', ']', 'chapter', '1', 'the', 'famili',
'of', 'dashwood', 'had', 'long', 'been', 'settl', 'in', 'sussex', '.', 'their', 'estat', 'wa', 'larg', ',', 'and',
'their', 'resid', 'wa', 'at', 'norland', 'park', ',', 'in', 'the', 'centr', 'of', 'their', 'properti', ',',
'where', ',', 'for', 'mani', 'gener', ',', 'they', 'had', 'live', 'in', 'so', 'respect', 'a', 'manner', 'as', 'to',
'engag', 'the', 'gener', 'good', 'opinion', 'of', 'their', 'surround', 'acquaint', '.', 'the', 'late', 'owner',
'of', 'thi', 'estat', 'wa', 'a', 'singl', 'man', ',', 'who', 'live', 'to', 'a', 'veri', 'advanc', 'age', ',',
'and', 'who', 'for', 'mani', 'year', 'of', 'hi', 'life', ',', 'had', 'a', 'constant', 'companion', 'and',
'housekeep', 'in', 'hi', 'sister', '.', 'but', 'her', 'death', ',', 'which', 'happen', 'ten', 'year', 'befor',
'hi', 'own', ',', 'produc', 'a', 'great', 'alter', 'in', 'hi', 'home', ';', 'for', 'to', 'suppli', 'her', 'loss',
',', 'he', 'invit', 'and', 'receiv', 'into', 'hi', 'hous', 'the', 'famili', 'of', 'hi', 'nephew', 'mr', '.',
'henri', 'dashwood', ',', 'the', 'legal', 'inheritor', 'of', 'the', 'norland', 'estat', ',', 'and', 'the',
'person', 'to', 'whom', 'he', 'intend', 'to', 'bequeath', 'it', '.', 'in', 'the', 'societi', 'of', 'hi', 'nephew',
'and', 'niec', ',', 'and', 'their', 'children', ',', 'the', 'old', 'gentleman', "'", 's', 'day', 'were', 'comfort',
'spent', '.', 'hi', 'attach', 'to', 'them', 'all', 'increas', '.', 'the', 'constant']
```

```
In [16]: nltk.download('wordnet')
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\ranke\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
Out[16]: True

In [17]: tokens1 = text2[:200]

In [18]: wnl = nltk.WordNetLemmatizer()

In [19]: [wnl.lemmatize(t) for t in tokens1]
Out[19]: ['[', 'Sense', 'and', 'Sensibility', 'by', 'Jane', 'Austen', '1811', ']', 'CHAPTER', '1', 'The', 'family',
'of', 'Dashwood', 'had', 'long', 'been', 'settled', 'in', 'Sussex', '.', 'Their', 'estate', 'wa', 'large', ',',
'and', 'their', 'residence', 'wa', 'at', 'Norland', 'Park', ',', 'in', 'the', 'centre', 'of', 'their', 'property',
',', 'where', ',', 'for', 'many', 'generation', ',', 'they', 'had', 'lived', 'in', 'so', 'respectable', 'a',
'manner', 'a', 'to', 'engage', 'the', 'general', 'good', 'opinion', 'of', 'their', 'surrounding', 'acquaintance',
'.', 'The', 'late', 'owner', 'of', 'this', 'estate', 'wa', 'a', 'single', 'man', ',', 'who', 'lived', 'to', 'a',
'very', 'advanced', 'age', ',', 'and', 'who', 'for', 'many', 'year', 'of', 'his', 'life', ',', 'had', 'a',
'constant', 'companion', 'and', 'housekeeper', 'in', 'his', 'sister', '.', 'But', 'her', 'death', ',', 'which',
'happened', 'ten', 'year', 'before', 'his', 'own', ',', 'produced', 'a', 'great', 'alteration', 'in', 'his',
'home', ';', 'for', 'to', 'supply', 'her', 'loss', ',', 'he', 'invited', 'and', 'received', 'into', 'his', 'house',
'the', 'family', 'of', 'his', 'nephew', 'Mr', '.', 'Henry', 'Dashwood', ',', 'the', 'legal', 'inheritor', 'of',
'the', 'Norland', 'estate', ',', 'and', 'the', 'person', 'to', 'whom', 'he', 'intended', 'to', 'bequeath', 'it',
'.', 'In', 'the', 'society', 'of', 'his', 'nephew', 'and', 'niece', ',', 'and', 'their', 'child', ',', 'the',
'old', 'Gentleman', "'", 's', 'day', 'were', 'comfortably', 'spent', '.', 'His', 'attachment', 'to', 'them', 'all',
'increased', '.', 'The', 'constant']
```


Ερώτημα 5

Αποτελέσματα εκτέλεσης

```
In [55]: [porter.stem(t) for t in tokens1]
Out[55]: ['ήταν', 'ένας', 'γάιδαρος', 'με', 'μεγάλα', 'αυτιά,', 'το', 'μαντρί', 'δεν', 'του', 'άρεσε', 'ήθελε',
'αρχοντιά!']

In [56]: [wnl.lemmatize(t) for t in tokens1]
Out[56]: ['Ήταν', 'ένας', 'γάιδαρος', 'με', 'μεγάλα', 'αυτιά,', 'το', 'μαντρί', 'δεν', 'του', 'άρεσε', 'ήθελε',
'αρχοντιά!']

In [57]: normalized_sent=[x.lower() for x in tokens1]

In [58]: normalized_sent
Out[58]: ['ήταν', 'ένας', 'γάιδαρος', 'με', 'μεγάλα', 'αυτιά,', 'το', 'μαντρί', 'δεν', 'του', 'άρεσε', 'ήθελε',
'αρχοντιά!']
```

```
In [77]: sentence = "General Kenobiii! It is nice meeting you again."
```

```
In [82]: [porter.stem(t) for t in tokens1]
Out[82]: ['gener', 'kenobiii!', 'it', 'is', 'nice', 'meet', 'you', 'again.']

In [83]: [wnl.lemmatize(t) for t in tokens1]
Out[83]: ['General', 'Kenobiii!', 'It', 'is', 'nice', 'meeting', 'you', 'again.']
```

Από την επεξεργασία του αποσπάσματος του “Sense and Sensibility” καθώς και από το δεύτερο παράδειγμα που παρατήθεται στα αγγλικά, γίνεται αντιληπτό ότι η μέθοδος stemming αφαιρεί τις καταλήξεις από διάφρες λέξεις, ενώ με την μέθοδο lemmatization δεν προχωρά στην διαδικασία αυτή (π.χ. “meet” -stemming- “meeting” -lemmatization-). Ταυτόχρονα, όμως η πρώτη παρουσιάζει μερικά λάθη. Για παράδειγμα, στην περίπτωση του αποσπάσματος από το “Sense and Sensibility” έχει αφαιρεθεί από πολλές λέξεις η κατάληξη κρατώντας αποκλιστικά το σώμα της λέξης (π.χ. Sense → sens), ενώ ταυτόχρονα μετατρέπει τ κεφαλαία σε μικρά, όπως η μέθοδος της κανονικοποιήσης. Τα ελληνικά αντιθέτως, φαίνεται να διατηρούνται αμετάβλητα με κάθε μία από τις μεθόδους με εξέρεση την μετατροπή των κεφαλαίων γραμμάτων σε μικρά. Γενικά, η μέθοδος lemmatizion φαίονεται να αναγνωρίζει τις λέξεις με το αρνητικό ότι την λέξη “was” την κόβει σε “wa”. Από την άλλη, η μέθοδος της απλής κανονικοποίησης μπορεί να ξεχωρίσει απλά τις ίδιες λέξεις χωρίς να κολλάει στην διάκριση πεζών-κεφαλαίων. Τέλος, η μέθοδος stemming είναι η καλύτερη για την εύρεση της ρίζας μιας λέξης.

Βήμα 3

Αποτελέσματα εκτέλεσης

```
In [27]: sentence = "Monticello wasn't designed as UNESCO World Heritage Site until 1987."

In [28]: sentence.split()
Out[28]: ['Monticello', "wasn't", 'designed', 'as', 'UNESCO', 'World', 'Heritage', 'Site', 'until', '1987.']

In [29]:
```



```
In [29]: str.split(sentence)
Out[29]: ['Monticello', "wasn't", 'designed', 'as', 'UNESCO', 'World', 'Heritage', 'Site', 'until', '1987.']
```



```
In [30]: import nltk.tokenize

In [31]: nltk.word_tokenize(sentence)
Out[31]: ['Monticello', 'was', "n't", 'designed', 'as', 'UNESCO', 'World', 'Heritage', 'Site', 'until', '1987',
'.']
```


Ερώτημα 6

Αποτελέσματα εκτέλεσης

Ελληνικό κείμενο

```
In [10]: sentence = "Ήταν ένας γάιδαρος με μεγάλα αυτιά, το μαντρί δεν του άρεσε ήθελε αρχοντιά!"

In [11]: sentence.split()
Out[11]:
['Ήταν',
 'ένας',
 'γάιδαρος',
 'με',
 'μεγάλα',
 'αυτιά,',
 'το',
 'μαντρί',
 'δεν',
 'του',
 'άρεσε',
 'ήθελε',
 'αρχοντιά!']
```

```
In [12]: str.split(sentence)
Out[12]:
['Ήταν',
 'ένας',
 'γάιδαρος',
 'με',
 'μεγάλα',
 'αυτιά,',
 'το',
 'μαντρί',
 'δεν',
 'του',
 'άρεσε',
 'ήθελε',
 'αρχοντιά!']
```

```
In [13]: nltk.word_tokenize(sentence)
Out[13]:
['Ήταν',
 'ένας',
 'γάιδαρος',
 'με',
 'μεγάλα',
 'αυτιά',
 ',',
 'το',
 'μαντρί',
 'δεν',
 'του',
 'άρεσε',
 'ήθελε',
 'αρχοντιά',
 '!']
```

Αγγλικό κείμενο

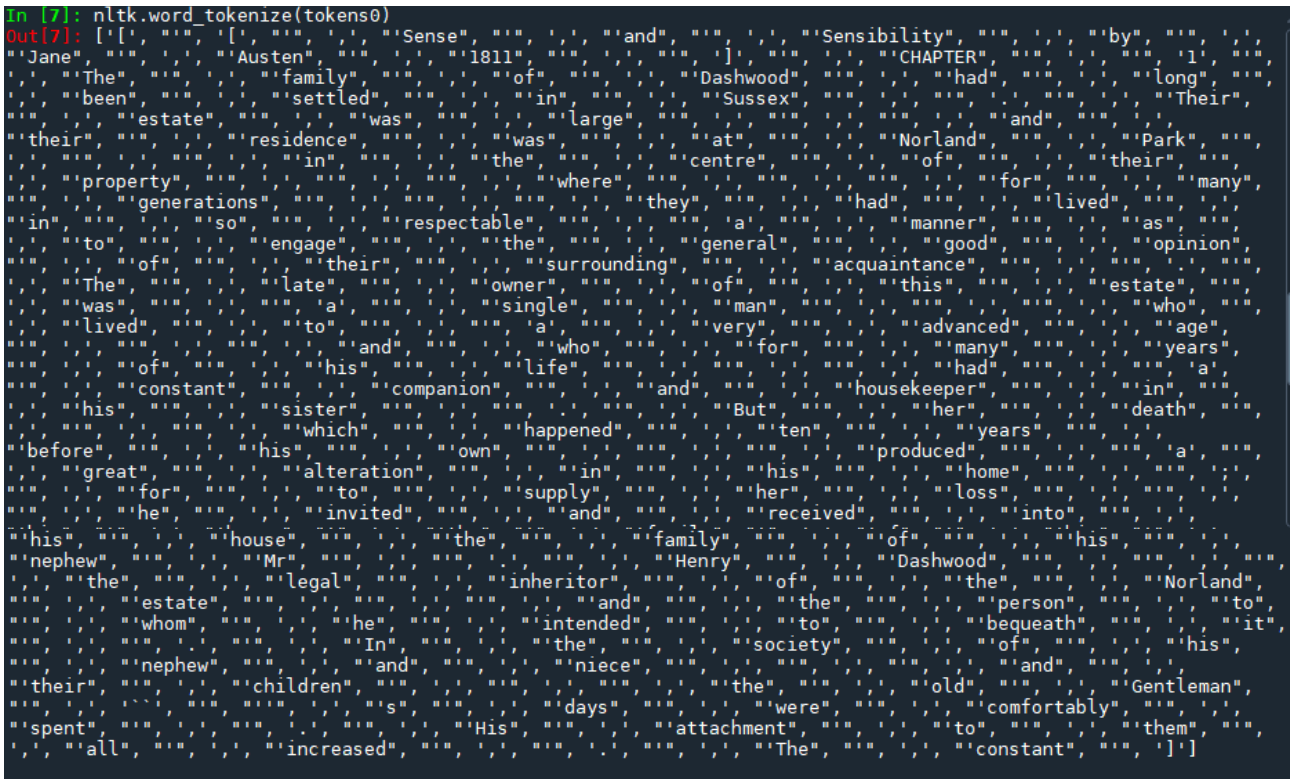
```
In [14]: sentence = "General Kenobiii! It is nice meeting you again."

In [15]: sentence.split()
Out[15]: ['General', 'Kenobiii!', 'It', 'is', 'nice', 'meeting', 'you', 'again.']

In [16]: str.split(sentence)
Out[16]: ['General', 'Kenobiii!', 'It', 'is', 'nice', 'meeting', 'you', 'again.']

In [17]: nltk.word_tokenize(sentence)
Out[17]:
['General',
 'Kenobiii',
 '!',
 'It',
 'is',
 'nice',
 'meeting',
 'you',
 'again',
 '.']
```

Απόσπασμα από “Sense and Sensibility”

```
In [3]: tokens0 = str(text2[:200])
```

```
In [5]: tokens0.split()
Out[5]: ["['[',", "'Sense',", "'and',", "'Sensibility',", "'by',", "'Jane',", "'Austen',", "'1811',", "']',",
"'CHAPTER',", "'1',", "'The',", "'family',", "'of',", "'Dashwood',", "'had',", "'long',", "'been',", "'settled',",
"'in',", "'Sussex',", "'.',", "'Their',", "'estate',", "'was',", "'large',", "',',", "'and',", "'their',",
"'residence',", "'was',", "'at',", "'Norland',", "'Park',", "',',", "'in',", "'the',", "'centre',", "'of',",
"'their',", "'property',", "',',", "'where',", "',',", "'for',", "'many',", "'generations',", "',',", "'they',",
"'had',", "'lived',", "'in',", "'so',", "'respectable',", "'a',", "'manner',", "'as',", "'to',", "'engage',",
"'the',", "'general',", "'good',", "'opinion',", "'of',", "'their',", "'surrounding',", "'acquaintance',", "'.',",
"'The',", "'late',", "'owner',", "'of',", "'this',", "'estate',", "'was',", "'a',", "'single',", "'man',", "',',",
"'who',", "'lived',", "'to',", "'a',", "'very',", "'advanced',", "'age',", "',',", "'and',", "'who',", "'for',",
"'many',", "'years',", "'of',", "'his',", "'life',", "',',", "'had',", "'a',", "'constant',", "'companion',",
"'and',", "'housekeeper',", "'in',", "'his',", "'sister',", "'.',", "'But',", "'her',", "'death',", "',',",
"'which',", "'happened',", "'ten',", "'years',", "'before',", "'his',", "'own',", "',',", "'produced',", "'a',",
"'great',", "'alteration',", "'in',", "'his',", "'home',", "';',", "'for',", "'to',", "'supply',", "'her',",
"'loss',", "',',", "'he',", "'invited',", "'and',", "'received',", "'into',", "'his',", "'house',", "'the',",
"'family',", "'of',", "'his',", "'nephew',", "'Mr',", "'.',", "'Henry',", "'Dashwood',", "',',", "'the',",
"'legal',", "'inheritor',", "'of',", "'the',", "'Norland',", "'estate',", "',',", "'and',", "'the',", "'person',",
"'to',", "'whom',", "'he',", "'intended',", "'to',", "'bequeath',", "'it',", "'.',", "'In',", "'the',",
"'society',", "'of',", "'his',", "'nephew',", "'and',", "'niece',", "',',", "'and',", "'their',", "'children',",
"',',", "'the',", "'old',", "'Gentleman',", '"\'",', "'s',", "'days',", "'were',", "'comfortably',", "'spent',",
"'.',", "'His',", "'attachment',", "'to',", "'them',", "'all',", "'increased',", "'.',", "'The',", "'constant']"]
```

```
In [6]: str.split(tokens0)
Out[6]: ["['[',", "'Sense',", "'and',", "'Sensibility',", "'by',", "'Jane',", "'Austen',", "'1811',", "']',",
"'CHAPTER',", "'1',", "'The',", "'family',", "'of',", "'Dashwood',", "'had',", "'long',", "'been',", "'settled',",
"'in',", "'Sussex',", "'.',", "'Their',", "'estate',", "'was',", "'large',", "',',", "'and',", "'their',",
"'residence',", "'was',", "'at',", "'Norland',", "'Park',", "',',", "'in',", "'the',", "'centre',", "'of',",
"'their',", "'property',", "',',", "'where',", "',',", "'for',", "'many',", "'generations',", "',',", "'they',",
"'had',", "'lived',", "'in',", "'so',", "'respectable',", "'a',", "'manner',", "'as',", "'to',", "'engage',",
"'the',", "'general',", "'good',", "'opinion',", "'of',", "'their',", "'surrounding',", "'acquaintance',", "'.',",
"'The',", "'late',", "'owner',", "'of',", "'this',", "'estate',", "'was',", "'a',", "'single',", "'man',", "',',",
"'who',", "'lived',", "'to',", "'a',", "'very',", "'advanced',", "'age',", "',',", "'and',", "'who',", "'for',",
"'many',", "'years',", "'of',", "'his',", "'life',", "',',", "'had',", "'a',", "'constant',", "'companion',",
"'and',", "'housekeeper',", "'in',", "'his',", "'sister',", "'.',", "'But',", "'her',", "'death',", "',',",
"'which',", "'happened',", "'ten',", "'years',", "'before',", "'his',", "'own',", "',',", "'produced',", "'a',",
"'great',", "'alteration',", "'in',", "'his',", "'home',", "';',", "'for',", "'to',", "'supply',", "'her',",
"'loss',", "',',", "'he',", "'invited',", "'and',", "'received',", "'into',", "'his',", "'house',", "'the',",
"'family',", "'of',", "'his',", "'nephew',", "'Mr',", "'.',", "'Henry',", "'Dashwood',", "',',", "'the',",
"'legal',", "'inheritor',", "'of',", "'the',", "'Norland',", "'estate',", "',',", "'and',", "'the',", "'person',",
"'to',", "'whom',", "'he',", "'intended',", "'to',", "'bequeath',", "'it',", "'.',", "'In',", "'the',",
"'society',", "'of',", "'his',", "'nephew',", "'and',", "'niece',", "',',", "'and',", "'their',", "'children',",
"',',", "'the',", "'old',", "'Gentleman',", '"\'",', "'s',", "'days',", "'were',", "'comfortably',", "'spent',",
"'.',", "'His',", "'attachment',", "'to',", "'them',", "'all',", "'increased',", "'.',", "'The',", "'constant']"]
```



Αρχικά, για τον έλεγχω της λειτουργίας των εντολών με είσοδο το απόσπασμα του βιβλίου χρειάστηκε μετατροπή της λίστας χαρακτήρων σε string, με την εντολή “tokens0 = str(text[:200]). Από εκεί και πέρα μπορούν να παρθούν τα παρακάτω συμπεράσματα:

Οι εντολές split() και str.split() έχουν την ίδια ακριβώς λειτουργία ανεξαρτήτως γλώσσας κειμένου.

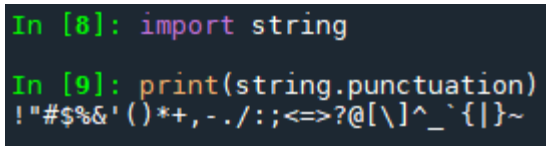
Η διαφορά, τώρα, της εντολής nltk.word_tokenize() είναι ότι ξεχωρίζει τις λέξεις με τα σημεία στίξης των κειμένων.

Παρατηρώντας το τελυταίο screenshot μπορεί να διακριθεί ότι λόγω της μετατροπής της λίστας σε string το κάθε λέξη περικλύεται σε << ' >>. Γίνεται αντιληπτό ότι με την χρήση της nlt.word_tokenize() κάθε δεύτερο << ' >> το εκλαμβάνει σαν ξεχωριστό token.

Συνοψίζοντας τα παραπάνω, η εντολη nltk.word_tokenize() είναι ιδανικότερη στην περίπτωση που χρειάζεται ξεκάθαρη ανάλυση λέξεων και χαρακτήρων σε δεδομένα μορφής string, καθώς δεν έχει την δυνατότητα να το πραγματοποιήση σε λίστα και στην μετατροπή λιστας σε string η κατάσταση μπορεί να γίνει λίγο μπερδευτική και χαοτική. Εν αντιθέση, οι εντολές split() φαντάζουν να έχουν μια σχετικά πιο πρακτική λειτουργία με πιο ξεκάθαρα αποτελέσματα, ακόμα και αν δεν ξεχωρίζουν τα σημεία στήξης με τις λέξεις.

Βήμα 4

<u>Αποτελέσματα εκτέλεσης</u>



```
In [14]: nltk.download('stopwords')
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\ranke\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
Out[14]: True

In [15]: stopwords = nltk.corpus.stopwords.words('english')

In [16]: print(stopwords)
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your',
'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it',
"it's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this',
'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had',
'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until',
'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before',
'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again',
'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few',
'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very',
's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o', 're', 've',
'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn',
"hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "needn't",
'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren', "weren't", 'won', "won't", 'wouldn',
"wouldn't"]
```

Ερώτημα 7

Αποτελέσματα εκτέλεσης

```
In [16]: print(stopwords)
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your',
'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it',
"it's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this',
'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had',
'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until',
'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before',
'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again',
'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few',
'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very',
's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o', 're', 've',
'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn',
"hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "needn't",
'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren', "weren't", 'won', "won't", 'wouldn',
"wouldn't"]

In [17]: len(stopwords)
Out[17]: 179
```

```
In [18]: stopwordsel = nltk.corpus.stopwords.words('greek')
```

```
In [20]: print(stopwordsel)
['αλλα', 'αν', 'αντι', 'απο', 'αυτα', 'αυτεσ', 'αυτη', 'αυτο', 'αυτοι', 'αυτοσ', 'αυτουσ', 'αυτων', 'αἱ', 'αἳ',
'αἵ', 'αὐτόσ', 'αὐτὸς', 'αὖ', 'γάρ', 'γα', 'γα^', 'γε', 'για', 'γοῦν', 'γὰρ', "δ'", 'δέ', 'δή', 'δαί', 'δαίσ',
'δαὶ', 'δαὶς', 'δε', 'δεν', "δι'", 'διά', 'διὰ', 'δὲ', 'δὴ', 'δ’', 'εαν', 'ειμαι', 'ειμαστε', 'ειναι', 'εισαι',
'ειστε', 'εκεινα', 'εκεινεσ', 'εκεινη', 'εκεινο', 'εκεινοι', 'εκεινοσ', 'εκεινουσ', 'εκεινων', 'ενω', 'επ', 'επι',
'ει', 'εἰμί', 'εἰμὶ', 'εἰς', 'εἰσ', 'εἳ', 'εἵμι', 'εἵτε', 'η', 'θα', 'ισωσ', 'κ', 'καί', 'καίτοι', 'καθ', 'και',
'κατ', 'κατά', 'κατα', 'κατὰ', 'καὶ', 'κι', 'κἀν', 'κἂν', 'μέν', 'μή', 'μήτε', 'μα', 'με', 'μεθ', 'μετ', 'μετά',
'μετα', 'μετὰ', 'μη', 'μην', 'μέν', 'μὲν', 'μὴ', 'μὴν', 'να', 'ο', 'οι', 'ομωσ', 'οπωσ', 'οσο', 'οτι', 'οἱ', 'οἳ',
'οἷς', 'οὐ', 'οὐδ', 'οὐδέ', 'οὐδείσ', 'οὐδεὶς', 'οὐδὲ', 'οὐδὲν', 'οὐκ', 'οὐχ', 'οὐχὶ', 'οὓς', 'οὔτε', 'οὕτω',
'οὕτως', 'οὕτωσ', 'οὖν', 'οὗ', 'οὗτος', 'οὗτοσ', 'παρ', 'παρά', 'παρα', 'παρὰ', 'περί', 'περὶ', 'ποια', 'ποιεσ',
'ποιο', 'ποιοι', 'ποιοσ', 'ποιουσ', 'ποιων', 'ποτε', 'που', 'ποῦ', 'προ', 'προσ', 'πρόσ', 'πρὸ', 'πρὸς', 'πωσ',
'πωσ', 'σε', 'στη', 'στην', 'στο', 'στον', 'σόσ', 'σύ', 'σύν', 'σὸς', 'σὺ', 'σὺν', 'τά', 'τήν', 'τί', 'τίς', 'τίσ',
'τα', 'ταῖς', 'τε', 'την', 'τησ', 'τι', 'τινα', 'τις', 'τισ', 'το', 'τοί', 'τοι', 'τοιοῦτος', 'τοιοῦτοσ', 'τον',
'τοτε', 'του', 'τούσ', 'τοὺς', 'τοῖς', 'τοῦ', 'των', 'τό', 'τόν', 'τότε', 'τὰ', 'τὰς', 'τὴν', 'τὸ', 'τὸν', 'τῆς',
'τῆσ', 'τῇ', 'τῶν', 'τῷ', 'ωσ', "ἀλλ'", 'ἀλλά', 'ἀλλὰ', 'ἀλλ’', 'ἀπ', 'ἀπό', 'ἀπὸ', 'ἀφ', 'ἂν', 'ἃ', 'ἄλλος',
'ἄλλοσ', 'ἄν', 'ἄρα', 'ἅμα', 'ἐάν', 'ἐγώ', 'ἐγὼ', 'ἐκ', 'ἐμόσ', 'ἐμὸς', 'ἐν', 'ἐξ', 'ἐπί', 'ἐπεὶ', 'ἐπὶ', 'ἐστι',
'ἐφ', 'ἐὰν', 'ἑαυτοῦ', 'ἔτι', 'ἡ', 'ἦ', 'ἦ', 'ἧ', 'ἥ', 'ἧς', 'ἵνα', 'ὁ', 'ὃ', 'ὃν', 'ὃς', 'ὅ', 'ὅδε', 'ὅθεν',
'ὅπερ', 'ὅς', 'ὅσ', 'ὅστις', 'ὅστισ', 'ὅτε', 'ὅτι', 'ὑμόσ', 'ὑπ', 'ὑπέρ', 'ὑπό', 'ὑπὲρ', 'ὑπὸ', 'ὡς', 'ὡσ', 'ὥς',
'ὥστε', 'ὥ', 'ὧ']

In [21]: len(stopwordsel)
Out[21]: 265
```

Τα stopwords της αγγλικής γλώσσας είναι 179 την στιγμή που η ελληνική έχει 265.

Ερώτημα 8

Punctuation.py

```
from nltk.book import *
import string
import nltk
nltk.download('stopwords')

def Punctuations():
    cleaned_tokens = []
    sentence = text2[:200]
    #sentence = "General Kenobi! Nice meeting you again."
    #sentence = "Ήταν ένας γάιδαρος με μεγάλα αυτιά, το
μαντρί δεν του άρεσε ήθελε αρχοντιά!"
    #sent = list(sentence.split())   #μετατροπή string σε
list
    stopwords = nltk.corpus.stopwords.words('english')
#καταχώριση stopwords αγγλικών
    #stopwords = nltk.corpus.stopwords.words('greek')
#καταχώριση stopwords ελληνικών
    for token in sentence:  #για το έτοιμο text
    #for token in sent:   #για προτάσεις
        #εκκαθάριση από σημεία στιξης και προθήματα
       if token not in string.punctuation:
           if token not in stopwords:
               cleaned_tokens.append(token)
    return[cleaned_tokens]
```

Αποτελέσματα εκτέλεσης

```
In [53]: Punctuations()
Out[53]: [['Sense', 'Sensibility', 'Jane', 'Austen', '1811', 'CHAPTER', '1', 'The', 'family', 'Dashwood', 'long',
'settled', 'Sussex', 'Their', 'estate', 'large', 'residence', 'Norland', 'Park', 'centre', 'property', 'many',
'generations', 'lived', 'respectable', 'manner', 'engage', 'general', 'good', 'opinion', 'surrounding',
'acquaintance', 'The', 'late', 'owner', 'estate', 'single', 'man', 'lived', 'advanced', 'age', 'many', 'years',
'life', 'constant', 'companion', 'housekeeper', 'sister', 'But', 'death', 'happened', 'ten', 'years', 'produced',
'great', 'alteration', 'home', 'supply', 'loss', 'invited', 'received', 'house', 'family', 'nephew', 'Mr', 'Henry',
'Dashwood', 'legal', 'inheritor', 'Norland', 'estate', 'person', 'intended', 'bequeath', 'In', 'society', 'nephew',
'niece', 'children', 'old', 'Gentleman', 'days', 'comfortably', 'spent', 'His', 'attachment', 'increased', 'The',
'constant']]
```

```
In [81]: Punctuations()
Out[81]: [['Ήταν', 'ένας', 'γάιδαρος', 'μεγάλα', 'αυτιά,', 'μαντρί', 'άρεσε', 'ήθελε', 'αρχοντιά!']]
```

```
In [84]: Punctuations()
Out[84]: [['General', 'Kenobi!', 'Nice', 'meeting', 'again.']]
```

Η παραπάνω συνάρτηση καταφέρνει να διαχωρίσει τα σημεία στίξης και τα προθήματα στο απόσπαμα του “Sense and Sensibility”, εκτός από τα σημεία όπου προθήματα είναι γραμμένα με κεφαλαίο αρχικό γράμμα. Επομένως, ο χρήστης θα πρέπει να αποφασίσει αν επιθυμεί να συνεχίσει με το σφάλμα αυτό ή να προβεί πρώτα σε μία μέθοδο κανονικοποίησης η οποία βέβαια θα μετατρέψει τα κεφαλαία γράμμαρα όλων των λέξεων

σε μικρά. Παρότι στο κείμενο ο διαχωρισμός των σημείων στίξης είναι επιτυχής, όταν το πρόγραμμα επεξεργάζεται προτάσεις (που από strings έχουν μετατραπεί σε list), αποτυγχάνει να τα διακρίνει ξεχωριστά από τις λέξεις. Τα προθήματα από την αλλή και σε ελληνικά και σε αγγλικά τα αναγνωρίζει κανονικά. Επομένως, είναι ασφαλές να συμπεράνουμε ότι η μέθοδος αυτή είναι αρκετά αξιόπιστη για χρήση σε κείμενα που έχουν ήδη την μορφή πίνακα ενώ είναι ελειπής όσον αφορά την επεξεργασία μεμονομένων strings.

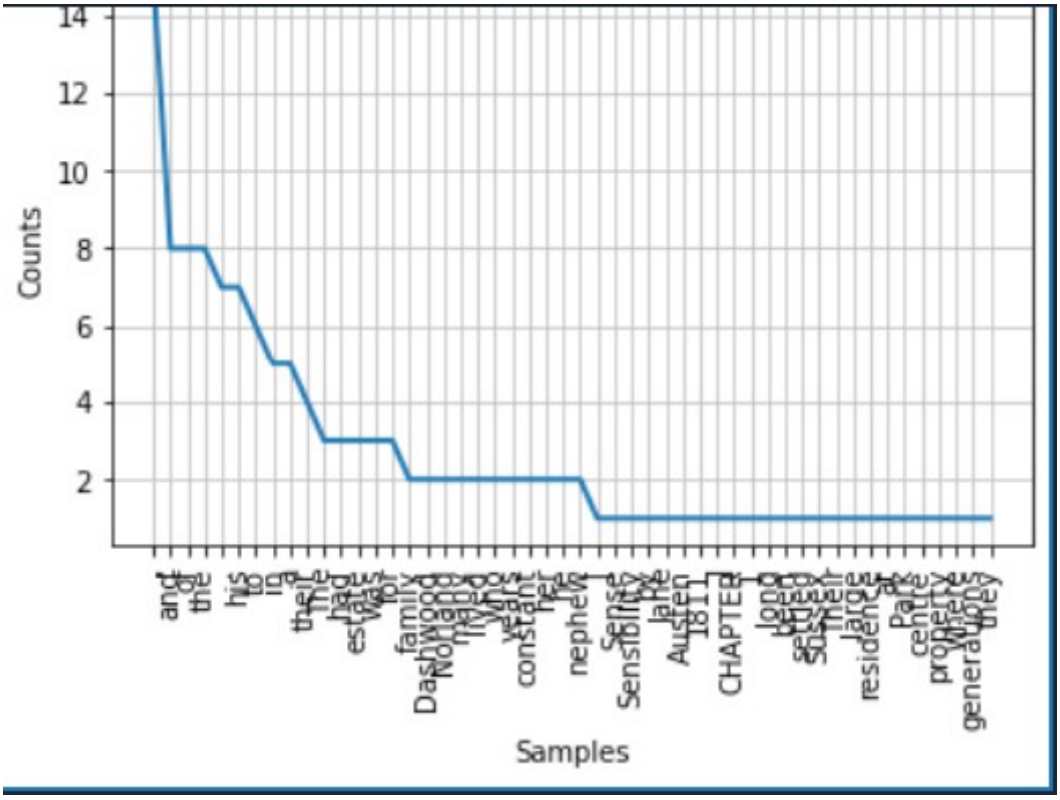
Ερώτημα 9

<u>Αποτελέσματα εκτέλεσης</u>

<u>Αρχικό κείμενο</u>

```
In [64]: fdist2 = FreqDIst(text2[:200])
```

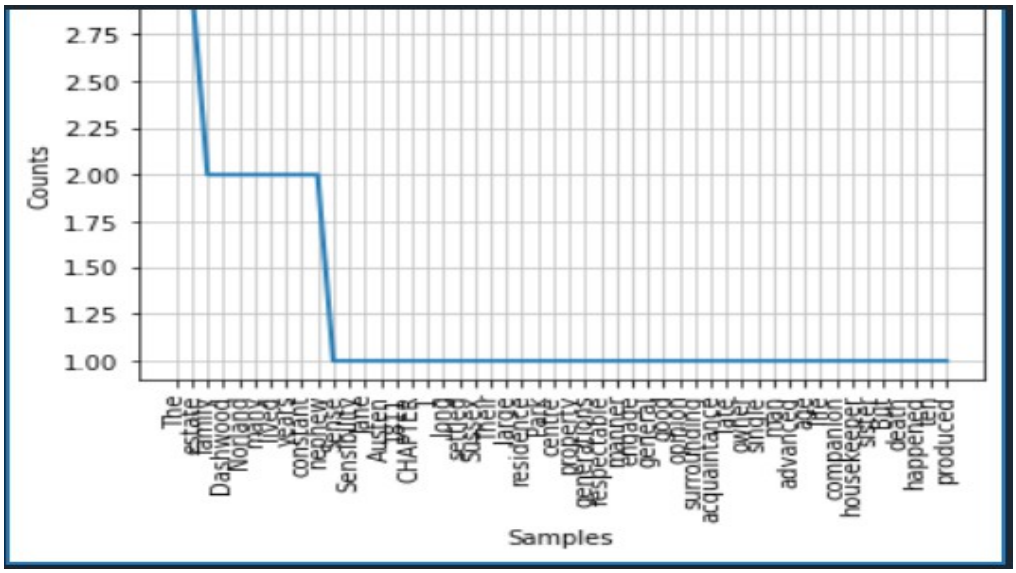
```
In [65]: fdist2 = FreqDist(text2[:200])

In [66]: fdist2.most_common(50)
Out[66]: [(',', 15), ('and', 8), ('of', 8), ('the', 8), ('.', 7), ('his', 7), ('to', 6), ('in', 5), ('a', 5),
('their', 4), ('The', 3), ('had', 3), ('estate', 3), ('was', 3), ('for', 3), ('family', 2), ('Dashwood', 2),
('Norland', 2), ('many', 2), ('lived', 2), ('who', 2), ('years', 2), ('constant', 2), ('her', 2), ('he', 2),
('nephew', 2), ('[', 1), ('Sense', 1), ('Sensibility', 1), ('by', 1), ('Jane', 1), ('Austen', 1), ('1811', 1),
(']', 1), ('CHAPTER', 1), ('1', 1), ('long', 1), ('been', 1), ('settled', 1), ('Sussex', 1), ('Their', 1),
('large', 1), ('residence', 1), ('at', 1), ('Park', 1), ('centre', 1), ('property', 1), ('where', 1),
('generations', 1), ('they', 1)]
```



<<Καθαρό>> κείμενο

```
In [69]: cleaned_tokens = []
```

```
In [71]: for token in tokens0:
    ...:     if token not in string.punctuation:
    ...:         if token not in stopwords:
    ...:             cleaned_tokens.append(token)

In [72]: fdist0 = FreqDist(cleaned_tokens)

In [73]: fdist0.most_common(50)
Out[73]: [('The', 3), ('estate', 3), ('family', 2), ('Dashwood', 2), ('Norland', 2), ('many', 2), ('lived', 2),
('years', 2), ('constant', 2), ('nephew', 2), ('Sense', 1), ('Sensibility', 1), ('Jane', 1), ('Austen', 1),
('1811', 1), ('CHAPTER', 1), ('1', 1), ('long', 1), ('settled', 1), ('Sussex', 1), ('Their', 1), ('large', 1),
('residence', 1), ('Park', 1), ('centre', 1), ('property', 1), ('generations', 1), ('respectable', 1), ('manner',
1), ('engage', 1), ('general', 1), ('good', 1), ('opinion', 1), ('surrounding', 1), ('acquaintance', 1), ('late',
1), ('owner', 1), ('single', 1), ('man', 1), ('advanced', 1), ('age', 1), ('life', 1), ('companion', 1),
('housekeeper', 1), ('sister', 1), ('But', 1), ('death', 1), ('happened', 1), ('ten', 1), ('produced', 1)]

In [74]: fdist0.plot(50)
Out[74]: <AxesSubplot:xlabel='Samples', ylabel='Counts'>
```



Όπως είναι αναμενόμενο το δεύτεο διάγραμμα έχει πολύ μικρότερη κατανομη συχνότητας από το πρώτο. Αυτό οφείλεται στο γεγονός ότι στα κείμενα τα σημεία στίξης και τα προθήματα είναι τα στοιχεία με την μεγαλύτερη χρήση. Παρατηρώντας το πρώτο και το δεύτερο διάγραμμα είναι φανερό ότι 10 διαφορετικά στοιχεία λεξιλογίου βρίσκουν μεγαλύτερη κατανομή συχνότητας από το πρώτο του <<καθαρού>> κειμένου που είναι το “The”.