Hanjiao Qiu
CSCIE63 Big Data Analytics
Assignment 12

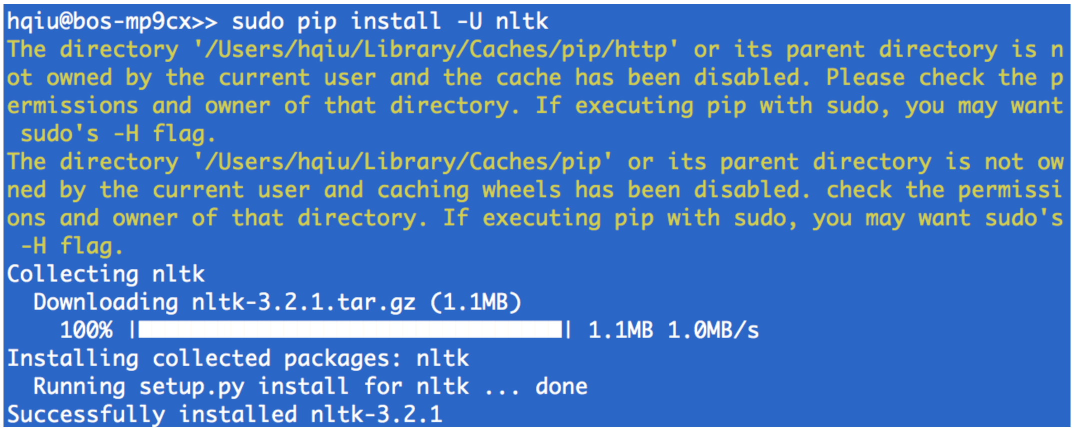
Assignment 12 Solution

Please, describe every step of your work and present all intermediate and final results in a Word document. Please, copy past text version of all essential command and snippets of results into the Word document with explanations of the purpose of those commands. We cannot retype text that is in JPG images. Please, always submit a separate copy of the original, working scripts and/or class files you used. Sometimes we need to run your code and retyping is too costly. Please include in your MS Word document only relevant portions of the console output or output files. Sometime either console output or the result file is too long and including it into the MS Word document makes that document too hard to read. PLEASE DO NOT EMBED files into your MS Word document. For issues and comments visit the class Discussion Board. If you use some other language other than Python in your daily work with NLP, please be free to use that language and a framework of your choice to do this assignment.

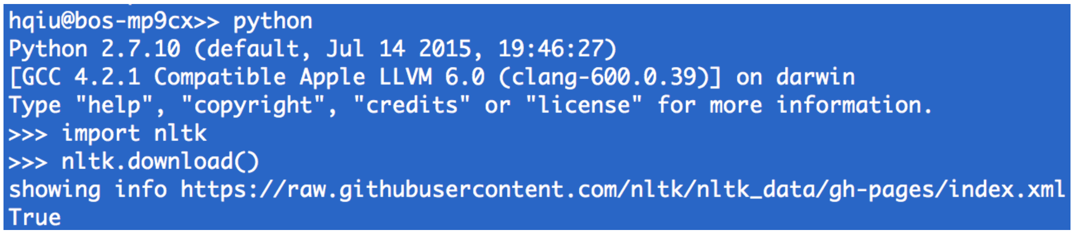
Problem 1. Create a table displaying **relative** frequencies with which "modals" (can, could, may, might, will, would and should) are used in 18 texts provided by NLTK in their extract from Gutenberg Corpus. For two modals with the largest span of relative frequencies (most used minus least used), select a text which uses it the most and the text that uses it the least. Compare usage in both texts by examining the concordances of those modals in two texts. Perhaps try to understand how are those words used in different texts.

Solution:

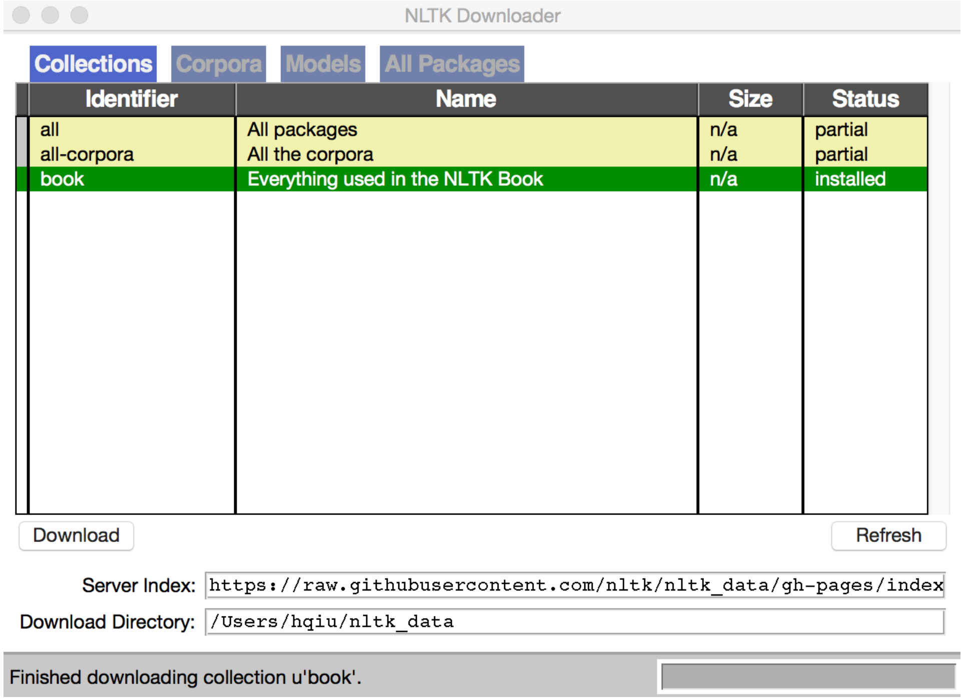
1. Install "nltk" and download the book.



```
hqiu@bos-mp9cx>> sudo pip install -U nltk
The directory '/Users/hqiu/Library/Caches/pip/http' or its parent directory is n
ot owned by the current user and the cache has been disabled. Please check the p
ermissions and owner of that directory. If executing pip with sudo, you may want
 sudo's -H flag.
The directory '/Users/hqiu/Library/Caches/pip' or its parent directory is not ow
ned by the current user and caching wheels has been disabled. check the permissi
ons and owner of that directory. If executing pip with sudo, you may want sudo's
 -H flag.
Collecting nltk
  Downloading nltk-3.2.1.tar.gz (1.1MB)
    100% |████████████████████████████████| 1.1MB 1.0MB/s
Installing collected packages: nltk
  Running setup.py install for nltk ... done
Successfully installed nltk-3.2.1
```



```
hqiu@bos-mp9cx>> python
Python 2.7.10 (default, Jul 14 2015, 19:46:27)
[GCC 4.2.1 Compatible Apple LLVM 6.0 (clang-600.0.39)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import nltk
>>> nltk.download()
showing info https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index.xml
True
```



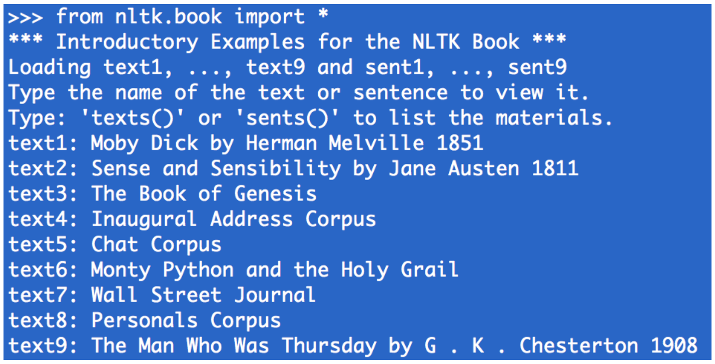
Identifier	Name	Size	Status
all	All packages	n/a	partial
all-corpora	All the corpora	n/a	partial
book	Everything used in the NLTK Book	n/a	installed

Server Index: https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index

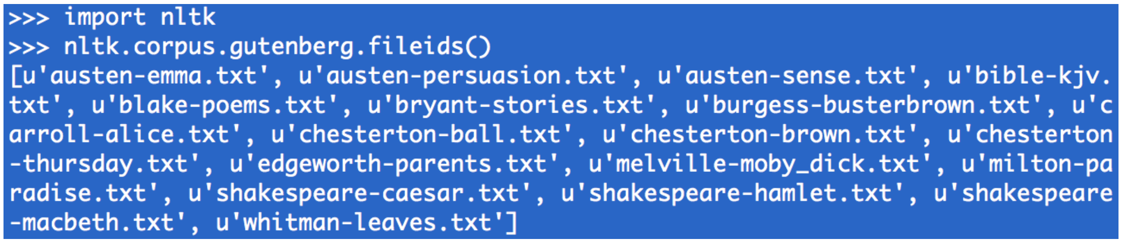
Download Directory: /Users/hqiu/nltk_data

Finished downloading collection u'book'.

2. Import the book and get the texts. Calculate the relative frequencies with which “modals” (can, could, may, might, will, would and should) are used in the text.

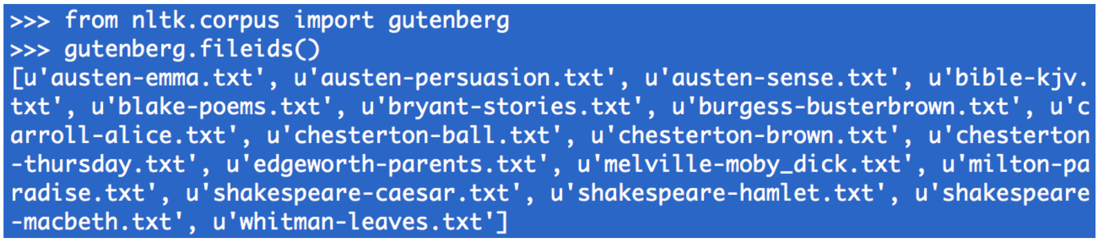


```
>>> from nltk.book import *
*** Introductory Examples for the NLTK Book ***
Loading text1, ..., text9 and sent1, ..., sent9
Type the name of the text or sentence to view it.
Type: 'texts()' or 'sents()' to list the materials.
text1: Moby Dick by Herman Melville 1851
text2: Sense and Sensibility by Jane Austen 1811
text3: The Book of Genesis
text4: Inaugural Address Corpus
text5: Chat Corpus
text6: Monty Python and the Holy Grail
text7: Wall Street Journal
text8: Personals Corpus
text9: The Man Who Was Thursday by G . K . Chesterton 1908
```



```
>>> import nltk
>>> nltk.corpus.gutenberg.fileids()
[u'austen-emma.txt', u'austen-persuasion.txt', u'austen-sense.txt', u'bible-kjv.
txt', u'blake-poems.txt', u'bryant-stories.txt', u'burgess-busterbrown.txt', u'c
arroll-alice.txt', u'chesterton-ball.txt', u'chesterton-brown.txt', u'chesterton
-thursday.txt', u'edgeworth-parents.txt', u'melville-moby_dick.txt', u'milton-pa
radise.txt', u'shakespeare-caesar.txt', u'shakespeare-hamlet.txt', u'shakespeare
-macbeth.txt', u'whitman-leaves.txt']
```

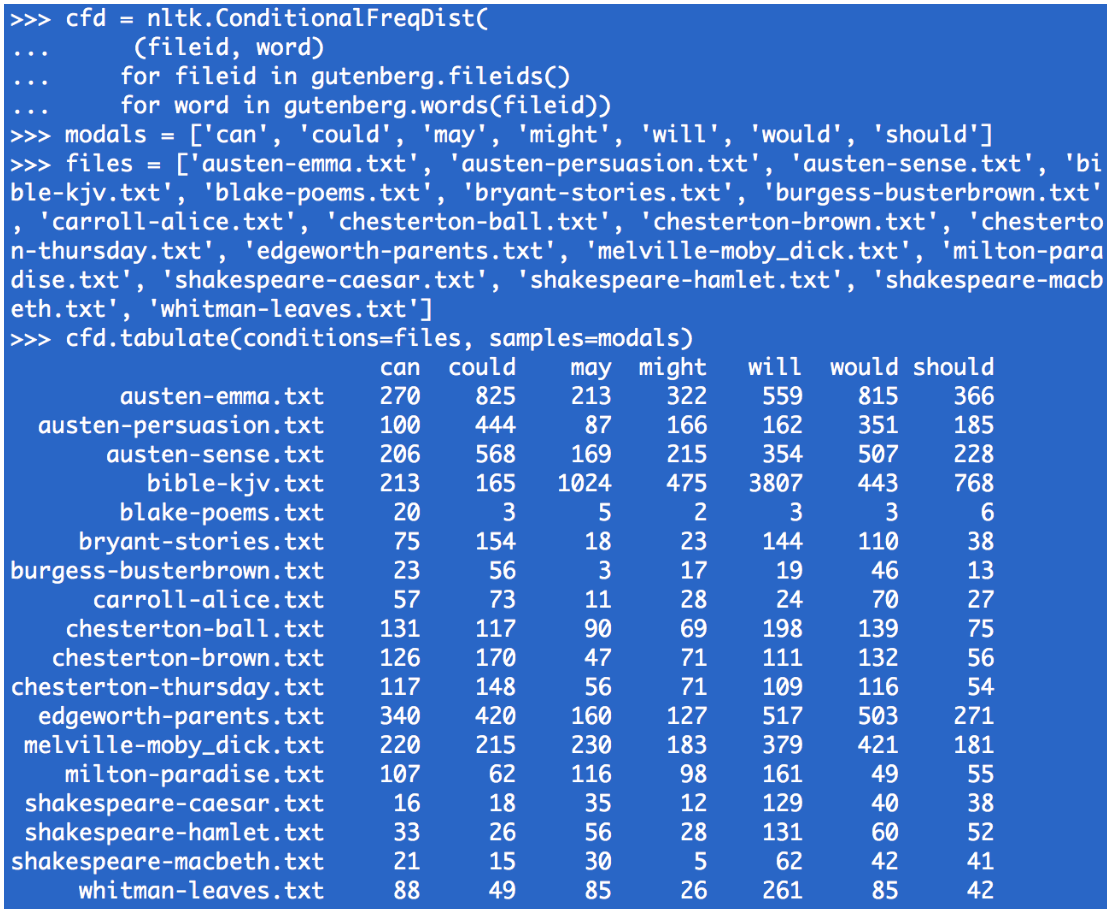
```
>>> from nltk.corpus import gutenberg
>>> gutenberg.fileids()
```



```
>>> from nltk.corpus import gutenberg
>>> gutenberg.fileids()
[u'austen-emma.txt', u'austen-persuasion.txt', u'austen-sense.txt', u'bible-kjv.
txt', u'blake-poems.txt', u'bryant-stories.txt', u'burgess-busterbrown.txt', u'c
arroll-alice.txt', u'chesterton-ball.txt', u'chesterton-brown.txt', u'chesterton
-thursday.txt', u'edgeworth-parents.txt', u'melville-moby_dick.txt', u'milton-pa
radise.txt', u'shakespeare-caesar.txt', u'shakespeare-hamlet.txt', u'shakespeare
-macbeth.txt', u'whitman-leaves.txt']
```

We can use the “nltk.ConditionalFreqDist()” to get the frequencies.

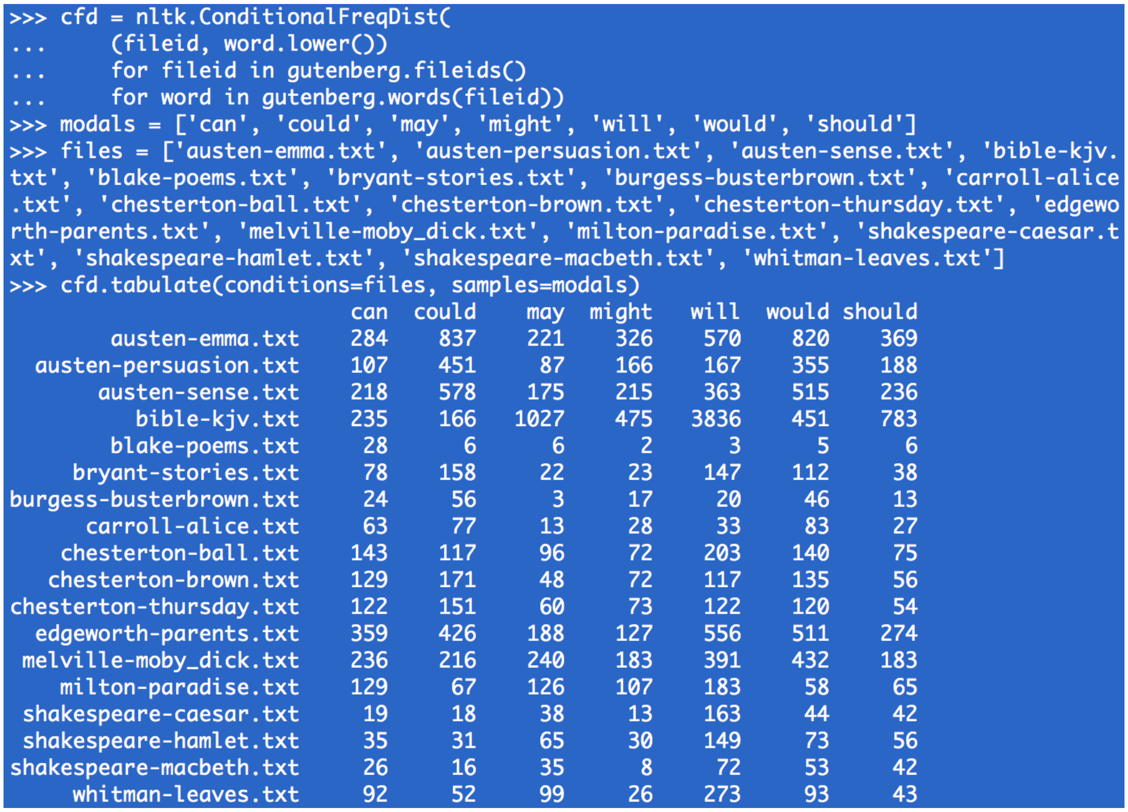
```
>>> cfd = nltk.ConditionalFreqDist(
...     (fileid, word)
...     for fileid in gutenberg.fileids()
...     for word in gutenberg.words(fileid))
>>> modals = ['can', 'could', 'may', 'might', 'will', 'would', 'should']
>>> files = ['austen-emma.txt', 'austen-persuasion.txt', 'austen-sense.txt',
'bible-kjv.txt', 'blake-poems.txt', 'bryant-stories.txt', 'burgess-
busterbrown.txt', 'carroll-alice.txt', 'chesterton-ball.txt', 'chesterton-
brown.txt', 'chesterton-thursday.txt', 'edgeworth-parents.txt', 'melville-
moby_dick.txt', 'milton-paradise.txt', 'shakespeare-caesar.txt', 'shakespeare-
hamlet.txt', 'shakespeare-macbeth.txt', 'whitman-leaves.txt']
>>> cfd.tabulate(conditions=files, samples=modals)
```



```
>>> cfd = nltk.ConditionalFreqDist(
...      (fileid, word)
...     for fileid in gutenberg.fileids()
...     for word in gutenberg.words(fileid))
>>> modals = ['can', 'could', 'may', 'might', 'will', 'would', 'should']
>>> files = ['austen-emma.txt', 'austen-persuasion.txt', 'austen-sense.txt', 'bi
ble-kjv.txt', 'blake-poems.txt', 'bryant-stories.txt', 'burgess-busterbrown.txt'
, 'carroll-alice.txt', 'chesterton-ball.txt', 'chesterton-brown.txt', 'chesterto
n-thursday.txt', 'edgeworth-parents.txt', 'melville-moby_dick.txt', 'milton-para
dise.txt', 'shakespeare-caesar.txt', 'shakespeare-hamlet.txt', 'shakespeare-macb
eth.txt', 'whitman-leaves.txt']
>>> cfd.tabulate(conditions=files, samples=modals)
                           can  could    may  might   will  would should
        austen-emma.txt    270    825    213    322    559    815    366
  austen-persuasion.txt    100    444     87    166    162    351    185
       austen-sense.txt    206    568    169    215    354    507    228
          bible-kjv.txt    213    165   1024    475   3807    443    768
        blake-poems.txt     20      3      5      2      3      3      6
     bryant-stories.txt     75    154     18     23    144    110     38
burgess-busterbrown.txt     23     56      3     17     19     46     13
      carroll-alice.txt     57     73     11     28     24     70     27
    chesterton-ball.txt    131    117     90     69    198    139     75
   chesterton-brown.txt    126    170     47     71    111    132     56
chesterton-thursday.txt    117    148     56     71    109    116     54
  edgeworth-parents.txt    340    420    160    127    517    503    271
 melville-moby_dick.txt    220    215    230    183    379    421    181
    milton-paradise.txt    107     62    116     98    161     49     55
 shakespeare-caesar.txt     16     18     35     12    129     40     38
 shakespeare-hamlet.txt     33     26     56     28    131     60     52
shakespeare-macbeth.txt     21     15     30      5     62     42     41
     whitman-leaves.txt     88     49     85     26    261     85     42
```

Convert the words to lower case and calculate again.

```
>>> cfd = nltk.ConditionalFreqDist(
...     (fileid, word.lower())
...     for fileid in gutenberg.fileids()
...     for word in gutenberg.words(fileid))
>>> modals = ['can', 'could', 'may', 'might', 'will', 'would', 'should']
>>> files = ['austen-emma.txt', 'austen-persuasion.txt', 'austen-sense.txt',
'bible-kjv.txt', 'blake-poems.txt', 'bryant-stories.txt', 'burgess-
busterbrown.txt', 'carroll-alice.txt', 'chesterton-ball.txt', 'chesterton-
brown.txt', 'chesterton-thursday.txt', 'edgeworth-parents.txt', 'melville-
moby_dick.txt', 'milton-paradise.txt', 'shakespeare-caesar.txt', 'shakespeare-
hamlet.txt', 'shakespeare-macbeth.txt', 'whitman-leaves.txt']
>>> cfd.tabulate(conditions=files, samples=modals)
```



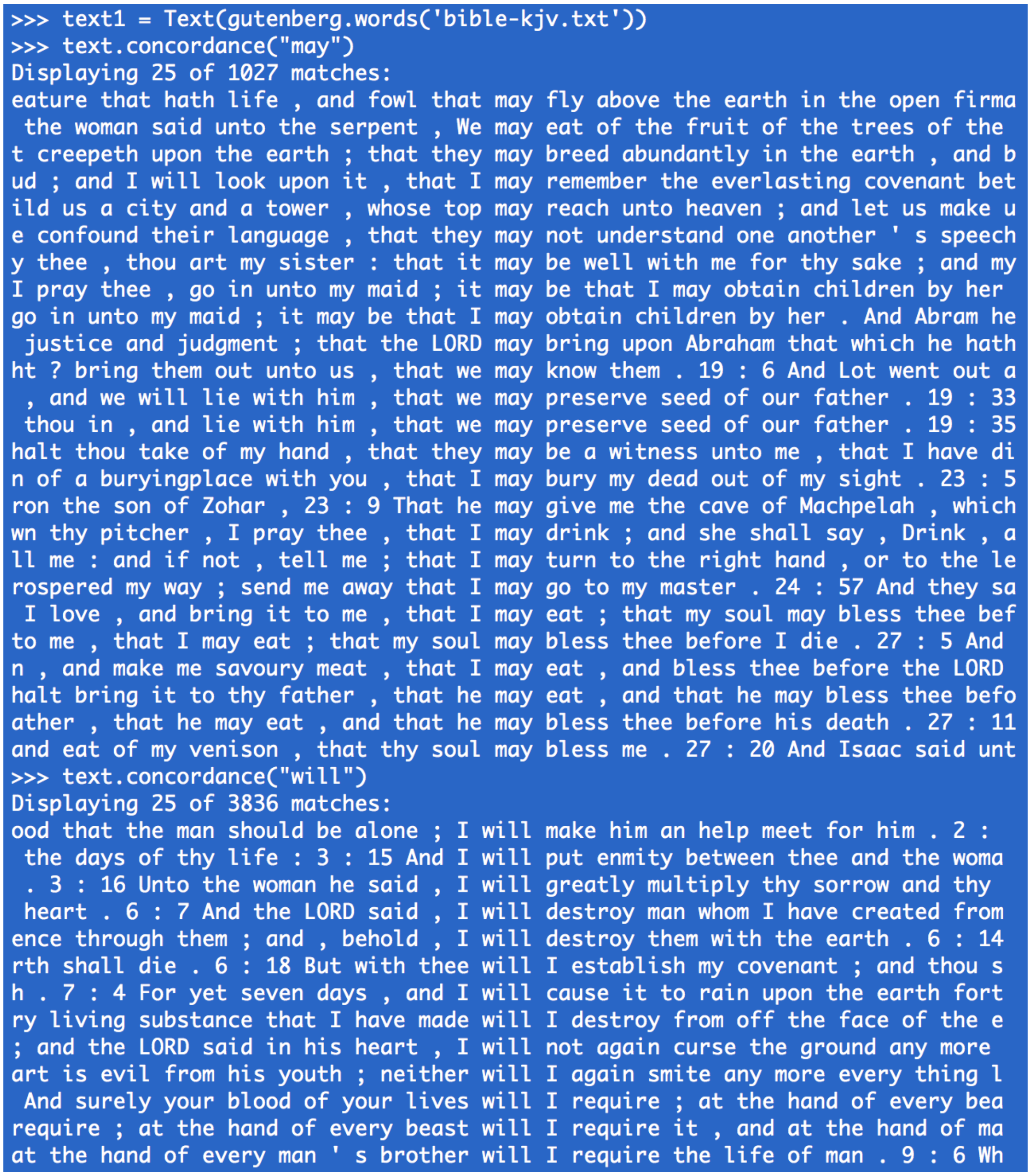
```
>>> cfd = nltk.ConditionalFreqDist(
...     (fileid, word.lower())
...     for fileid in gutenberg.fileids()
...     for word in gutenberg.words(fileid))
>>> modals = ['can', 'could', 'may', 'might', 'will', 'would', 'should']
>>> files = ['austen-emma.txt', 'austen-persuasion.txt', 'austen-sense.txt', 'bible-kjv.
txt', 'blake-poems.txt', 'bryant-stories.txt', 'burgess-busterbrown.txt', 'carroll-alice
.txt', 'chesterton-ball.txt', 'chesterton-brown.txt', 'chesterton-thursday.txt', 'edgewo
rth-parents.txt', 'melville-moby_dick.txt', 'milton-paradise.txt', 'shakespeare-caesar.t
xt', 'shakespeare-hamlet.txt', 'shakespeare-macbeth.txt', 'whitman-leaves.txt']
>>> cfd.tabulate(conditions=files, samples=modals)
                           can  could    may  might   will  would should
        austen-emma.txt    284    837    221    326    570    820    369
  austen-persuasion.txt    107    451     87    166    167    355    188
       austen-sense.txt    218    578    175    215    363    515    236
          bible-kjv.txt    235    166   1027    475   3836    451    783
        blake-poems.txt     28      6      6      2      3      5      6
     bryant-stories.txt     78    158     22     23    147    112     38
burgess-busterbrown.txt     24     56      3     17     20     46     13
      carroll-alice.txt     63     77     13     28     33     83     27
    chesterton-ball.txt    143    117     96     72    203    140     75
   chesterton-brown.txt    129    171     48     72    117    135     56
chesterton-thursday.txt    122    151     60     73    122    120     54
  edgeworth-parents.txt    359    426    188    127    556    511    274
 melville-moby_dick.txt    236    216    240    183    391    432    183
    milton-paradise.txt    129     67    126    107    183     58     65
 shakespeare-caesar.txt     19     18     38     13    163     44     42
 shakespeare-hamlet.txt     35     31     65     30    149     73     56
shakespeare-macbeth.txt     26     16     35      8     72     53     42
     whitman-leaves.txt     92     52     99     26    273     93     43
```

3. The word “may” and “will” are used the most in “buble-kjv.txt” and used the least in “blake-poems.txt”. Examine the concordances of the two words in two texts.

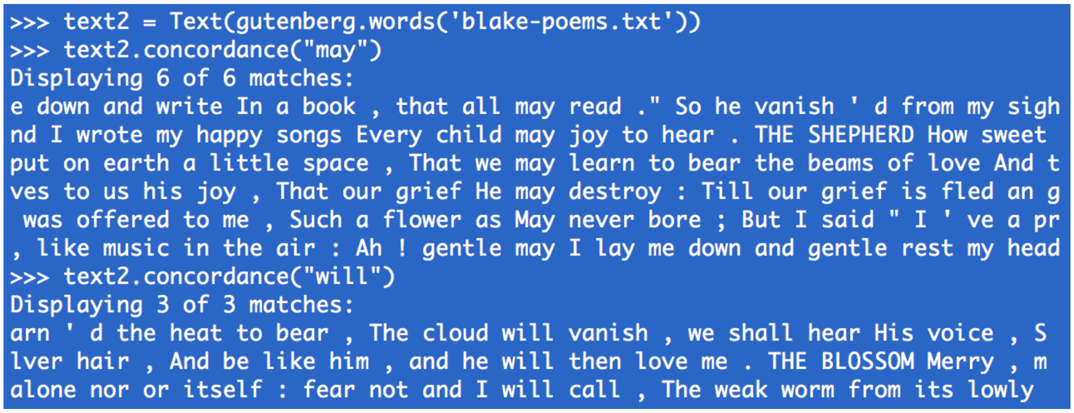
```
>>> text1 = Text(gutenberg.words('bible-kjv.txt'))
>>> text1.concordance("may")
>>> text1.concordance("will")
```

The “may” is often used as “we may”, “I may”. The “will” is used with “I will”.

```
>>> text2 = Text(gutenberg.words('blake-poems.txt'))
>>> text2.concordance("may")
>>> text2.concordance("will")
```

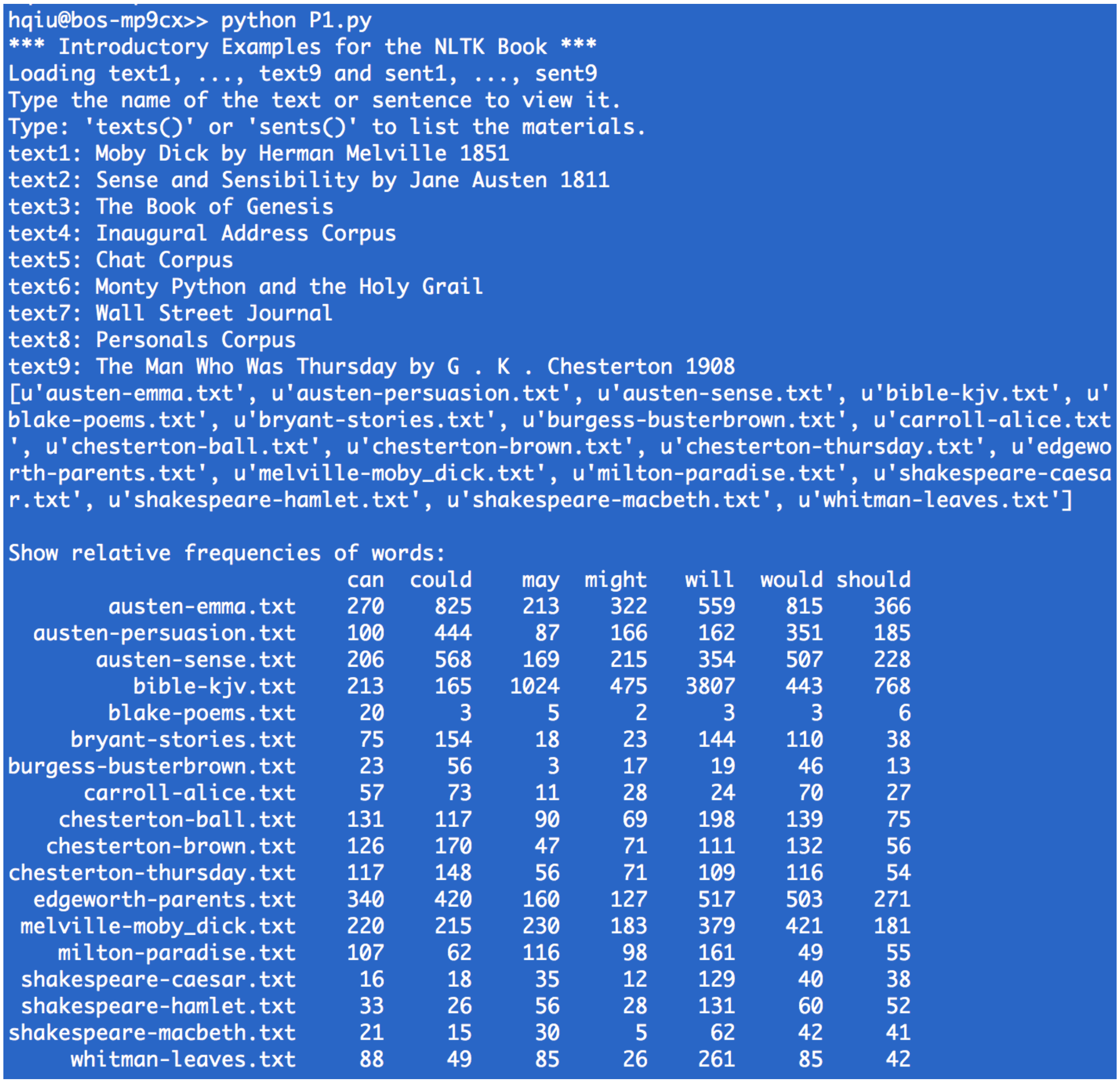


```
>>> text1 = Text(gutenberg.words('bible-kjv.txt'))
>>> text.concordance("may")
Displaying 25 of 1027 matches:
eature that hath life , and fowl that may fly above the earth in the open firma
 the woman said unto the serpent , We may eat of the fruit of the trees of the
t creepeth upon the earth ; that they may breed abundantly in the earth , and b
ud ; and I will look upon it , that I may remember the everlasting covenant bet
ild us a city and a tower , whose top may reach unto heaven ; and let us make u
e confound their language , that they may not understand one another ' s speech
y thee , thou art my sister : that it may be well with me for thy sake ; and my
I pray thee , go in unto my maid ; it may be that I may obtain children by her
go in unto my maid ; it may be that I may obtain children by her . And Abram he
 justice and judgment ; that the LORD may bring upon Abraham that which he hath
ht ? bring them out unto us , that we may know them . 19 : 6 And Lot went out a
 , and we will lie with him , that we may preserve seed of our father . 19 : 33
 thou in , and lie with him , that we may preserve seed of our father . 19 : 35
halt thou take of my hand , that they may be a witness unto me , that I have di
n of a buryingplace with you , that I may bury my dead out of my sight . 23 : 5
ron the son of Zohar , 23 : 9 That he may give me the cave of Machpelah , which
wn thy pitcher , I pray thee , that I may drink ; and she shall say , Drink , a
ll me : and if not , tell me ; that I may turn to the right hand , or to the le
rospered my way ; send me away that I may go to my master . 24 : 57 And they sa
 I love , and bring it to me , that I may eat ; that my soul may bless thee bef
to me , that I may eat ; that my soul may bless thee before I die . 27 : 5 And
n , and make me savoury meat , that I may eat , and bless thee before the LORD
halt bring it to thy father , that he may eat , and that he may bless thee befo
ather , that he may eat , and that he may bless thee before his death . 27 : 11
and eat of my venison , that thy soul may bless me . 27 : 20 And Isaac said unt
>>> text.concordance("will")
Displaying 25 of 3836 matches:
ood that the man should be alone ; I will make him an help meet for him . 2 :
 the days of thy life : 3 : 15 And I will put enmity between thee and the woma
 . 3 : 16 Unto the woman he said , I will greatly multiply thy sorrow and thy
 heart . 6 : 7 And the LORD said , I will destroy man whom I have created from
ence through them ; and , behold , I will destroy them with the earth . 6 : 14
rth shall die . 6 : 18 But with thee will I establish my covenant ; and thou s
h . 7 : 4 For yet seven days , and I will cause it to rain upon the earth fort
ry living substance that I have made will I destroy from off the face of the e
; and the LORD said in his heart , I will not again curse the ground any more
art is evil from his youth ; neither will I again smite any more every thing l
 And surely your blood of your lives will I require ; at the hand of every bea
require ; at the hand of every beast will I require it , and at the hand of ma
at the hand of every man ' s brother will I require the life of man . 9 : 6 Wh
```



```
>>> text2 = Text(gutenberg.words('blake-poems.txt'))
>>> text2.concordance("may")
Displaying 6 of 6 matches:
e down and write In a book , that all may read ." So he vanish ' d from my sigh
nd I wrote my happy songs Every child may joy to hear . THE SHEPHERD How sweet
put on earth a little space , That we may learn to bear the beams of love And t
ves to us his joy , That our grief He may destroy : Till our grief is fled an g
 was offered to me , Such a flower as May never bore ; But I said " I ' ve a pr
, like music in the air : Ah ! gentle may I lay me down and gentle rest my head
>>> text2.concordance("will")
Displaying 3 of 3 matches:
arn ' d the heat to bear , The cloud will vanish , we shall hear His voice , S
lver hair , And be like him , and he will then love me . THE BLOSSOM Merry , m
alone nor or itself : fear not and I will call , The weak worm from its lowly
```

Source code: P1.py



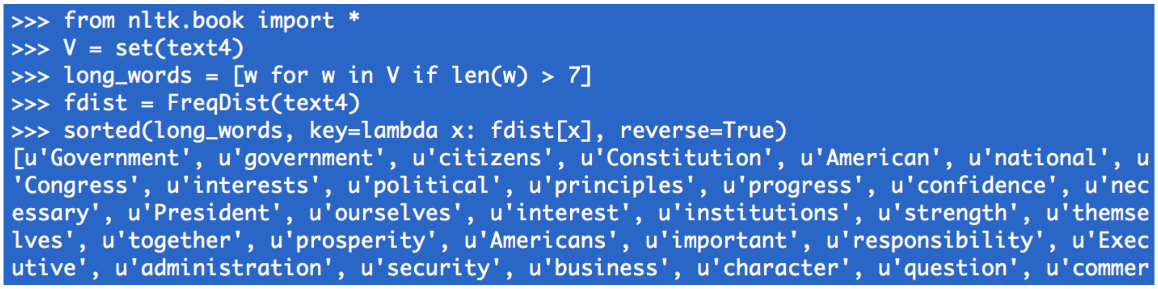
```
hqiu@bos-mp9cx>> python P1.py
*** Introductory Examples for the NLTK Book ***
Loading text1, ..., text9 and sent1, ..., sent9
Type the name of the text or sentence to view it.
Type: 'texts()' or 'sents()' to list the materials.
text1: Moby Dick by Herman Melville 1851
text2: Sense and Sensibility by Jane Austen 1811
text3: The Book of Genesis
text4: Inaugural Address Corpus
text5: Chat Corpus
text6: Monty Python and the Holy Grail
text7: Wall Street Journal
text8: Personals Corpus
text9: The Man Who Was Thursday by G . K . Chesterton 1908
[u'austen-emma.txt', u'austen-persuasion.txt', u'austen-sense.txt', u'bible-kjv.txt', u'
blake-poems.txt', u'bryant-stories.txt', u'burgess-busterbrown.txt', u'carroll-alice.txt
', u'chesterton-ball.txt', u'chesterton-brown.txt', u'chesterton-thursday.txt', u'edgewo
rth-parents.txt', u'melville-moby_dick.txt', u'milton-paradise.txt', u'shakespeare-caesa
r.txt', u'shakespeare-hamlet.txt', u'shakespeare-macbeth.txt', u'whitman-leaves.txt']

Show relative frequencies of words:
                           can  could   may  might   will  would should
        austen-emma.txt    270   825    213   322    559    815    366
  austen-persuasion.txt    100   444     87   166    162    351    185
       austen-sense.txt    206   568    169   215    354    507    228
          bible-kjv.txt    213   165   1024   475   3807    443    768
        blake-poems.txt     20     3      5     2      3      3      6
     bryant-stories.txt     75   154     18    23    144    110     38
burgess-busterbrown.txt     23    56      3    17     19     46     13
      carroll-alice.txt     57    73     11    28     24     70     27
    chesterton-ball.txt    131   117     90    69    198    139     75
   chesterton-brown.txt    126   170     47    71    111    132     56
chesterton-thursday.txt    117   148     56    71    109    116     54
  edgeworth-parents.txt    340   420    160   127    517    503    271
 melville-moby_dick.txt    220   215    230   183    379    421    181
    milton-paradise.txt    107    62    116    98    161     49     55
 shakespeare-caesar.txt     16    18     35    12    129     40     38
 shakespeare-hamlet.txt     33    26     56    28    131     60     52
shakespeare-macbeth.txt     21    15     30     5     62     42     41
     whitman-leaves.txt     88    49     85    26    261     85     42
```

Problem 2. In the Inaugural corpus identify 10 most frequently used words longer than 7 characters. Which one of those has the largest number of synonyms? List all synonyms for those 10 words. Which one of those 10 words has the largest number of hyponyms? List all hyponyms of those 10 most frequently used "long" words.

Solution:

1. Get the 10 mist frequently used words longer than 7 characters.

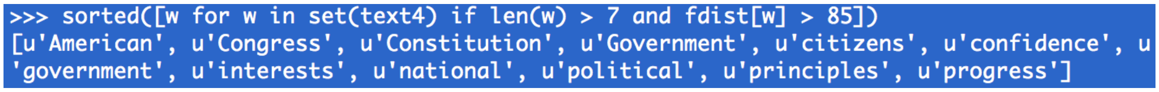
```
>>> from nltk.book import *
>>> V = set(text4)
>>> long_words = [w for w in V if len(w) > 7]
>>> fdist = FreqDist(text4)
>>> sorted(long_words, key=lambda x: fdist[x], reverse=True)
```



```
>>> from nltk.book import *
>>> V = set(text4)
>>> long_words = [w for w in V if len(w) > 7]
>>> fdist = FreqDist(text4)
>>> sorted(long_words, key=lambda x: fdist[x], reverse=True)
[u'Government', u'government', u'citizens', u'Constitution', u'American', u'national', u
'Congress', u'interests', u'political', u'principles', u'progress', u'confidence', u'nec
essary', u'President', u'ourselves', u'interest', u'institutions', u'strength', u'themse
lves', u'together', u'prosperity', u'Americans', u'important', u'responsibility', u'Exec
utive', u'administration', u'security', u'business', u'character', u'question', u'commer
```

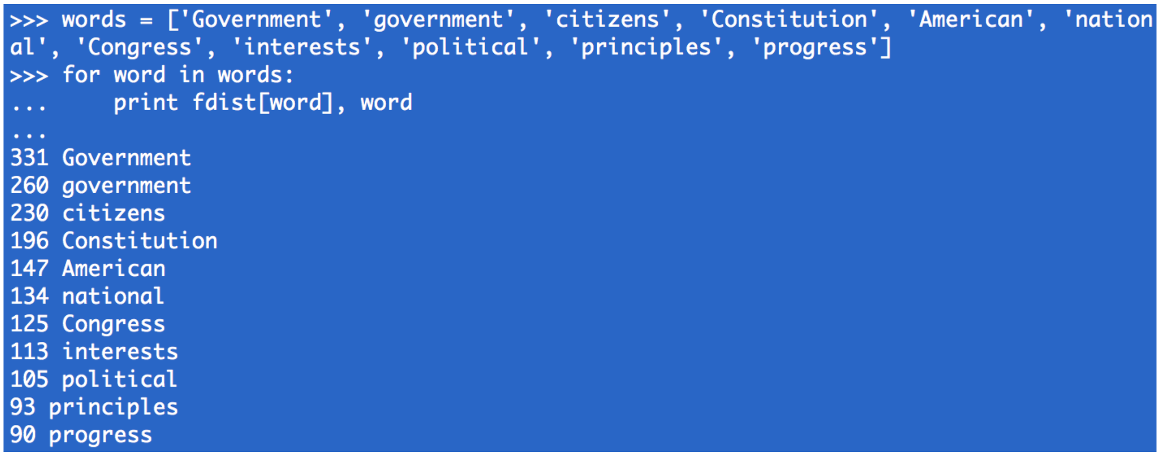
Validate the results above:

```
>>> sorted([w for w in set(text4) if len(w) > 7 and fdist[w] > 85])
```



```
>>> sorted([w for w in set(text4) if len(w) > 7 and fdist[w] > 85])
[u'American', u'Congress', u'Constitution', u'Government', u'citizens', u'confidence', u
'government', u'interests', u'national', u'political', u'principles', u'progress']
```

```
>>> words = ['Government', 'government', 'citizens', 'Constitution', 'American',
'national', 'Congress', 'interests', 'political', 'principles', 'progress']
>>> for word in words:
...     print fdist[word], word
...
```

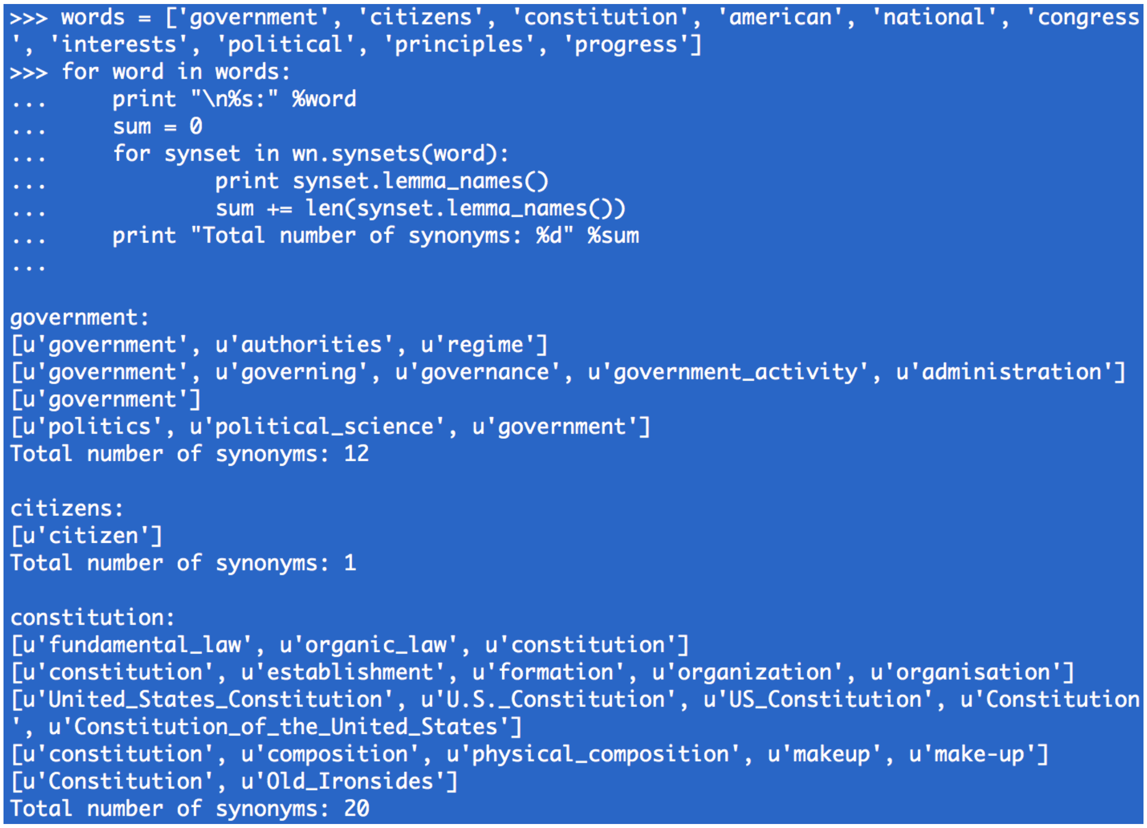


```
>>> words = ['Government', 'government', 'citizens', 'Constitution', 'American', 'nation
al', 'Congress', 'interests', 'political', 'principles', 'progress']
>>> for word in words:
...     print fdist[word], word
...
331 Government
260 government
230 citizens
196 Constitution
147 American
134 national
125 Congress
113 interests
105 political
93 principles
90 progress
```

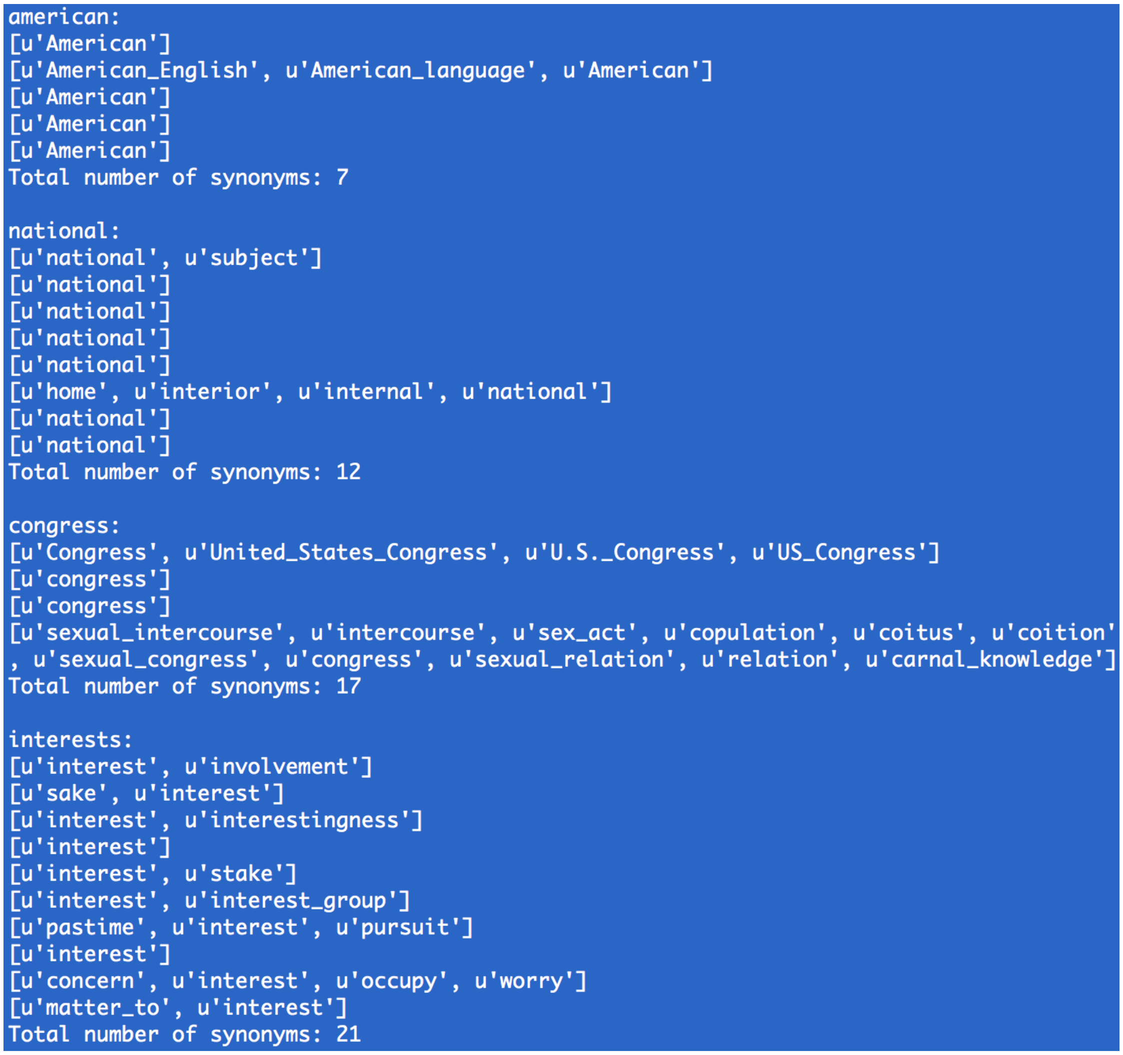
2. List all synonyms for those 10 words. The word "progress" has the largest number of synonyms, which is 29.

```
>>> from nltk.corpus import wordnet as wn
>>> words = ['government', 'citizens', 'constitution', 'american', 'national',
'congress', 'interests', 'political', 'principles', 'progress']
>>> for word in words:
...     print "\n%s:" %word
...     sum = 0
...     for synset in wn.synsets(word):
...             print synset.lemma_names()
...             sum += len(synset.lemma_names())
...     print "Total number of synonyms: %d" %sum
```

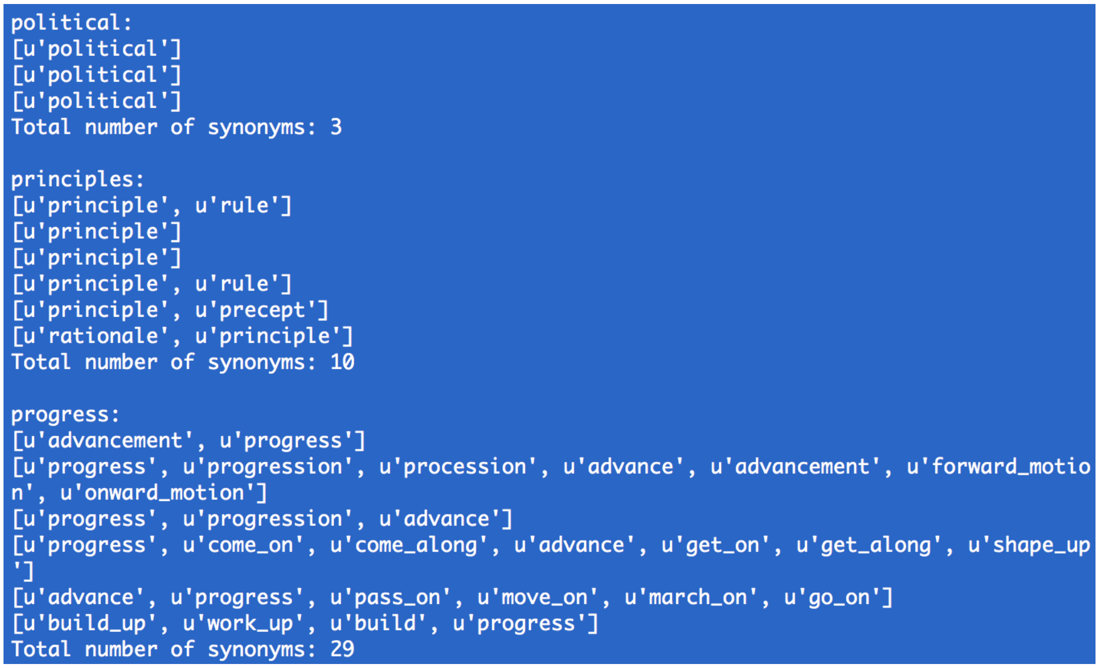
```
...
```



```
>>> words = ['government', 'citizens', 'constitution', 'american', 'national', 'congress
', 'interests', 'political', 'principles', 'progress']
>>> for word in words:
...     print "\n%s:" %word
...     sum = 0
...     for synset in wn.synsets(word):
...             print synset.lemma_names()
...             sum += len(synset.lemma_names())
...     print "Total number of synonyms: %d" %sum
...

government:
[u'government', u'authorities', u'regime']
[u'government', u'governing', u'governance', u'government_activity', u'administration']
[u'government']
[u'politics', u'political_science', u'government']
Total number of synonyms: 12

citizens:
[u'citizen']
Total number of synonyms: 1

constitution:
[u'fundamental_law', u'organic_law', u'constitution']
[u'constitution', u'establishment', u'formation', u'organization', u'organisation']
[u'United_States_Constitution', u'U.S._Constitution', u'US_Constitution', u'Constitution
', u'Constitution_of_the_United_States']
[u'constitution', u'composition', u'physical_composition', u'makeup', u'make-up']
[u'Constitution', u'Old_Ironsides']
Total number of synonyms: 20
```



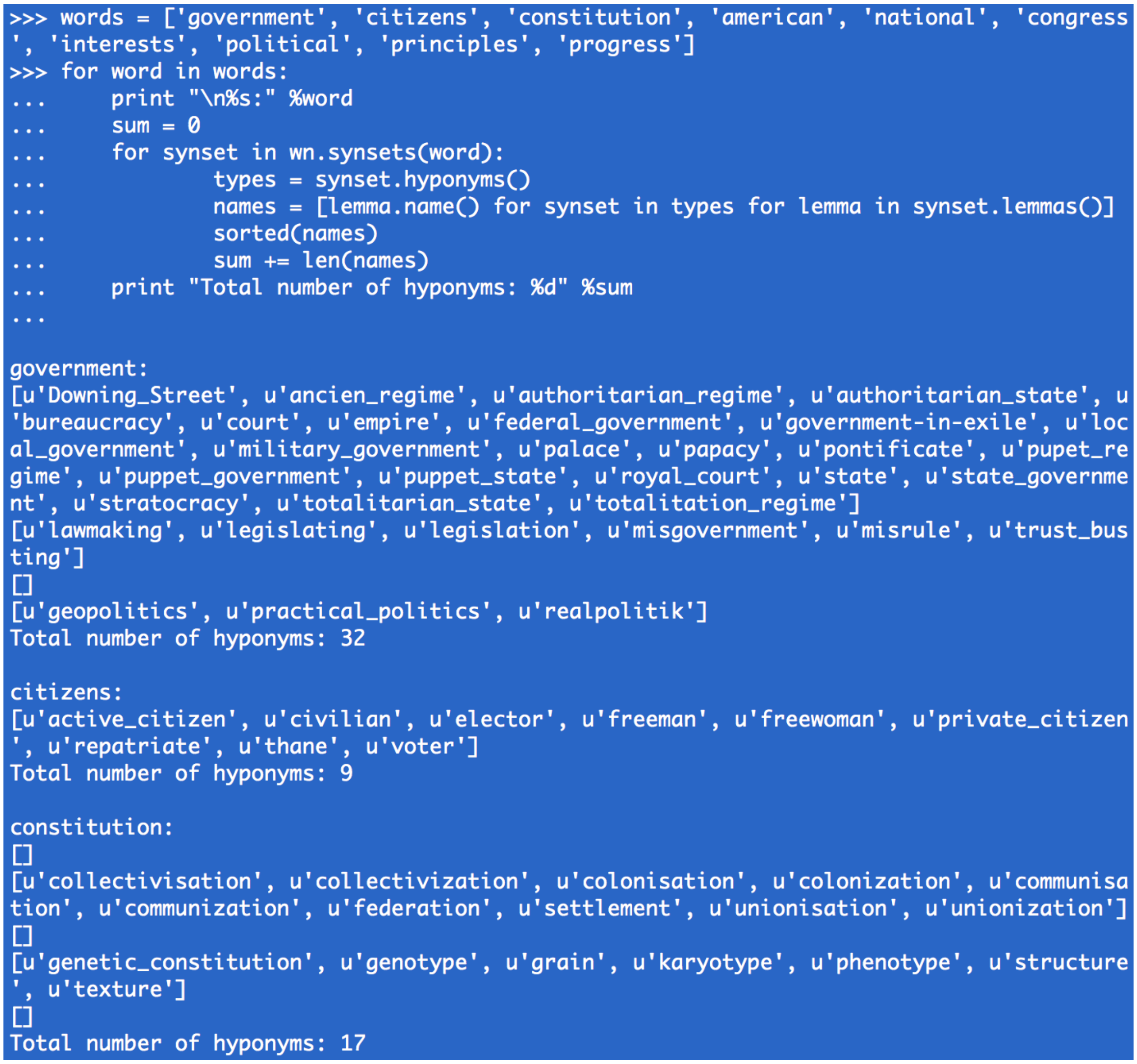
```
american:
[u'American']
[u'American_English', u'American_language', u'American']
[u'American']
[u'American']
[u'American']
Total number of synonyms: 7

national:
[u'national', u'subject']
[u'national']
[u'national']
[u'national']
[u'national']
[u'home', u'interior', u'internal', u'national']
[u'national']
[u'national']
Total number of synonyms: 12

congress:
[u'Congress', u'United_States_Congress', u'U.S._Congress', u'US_Congress']
[u'congress']
[u'congress']
[u'sexual_intercourse', u'intercourse', u'sex_act', u'copulation', u'coitus', u'coition'
, u'sexual_congress', u'congress', u'sexual_relation', u'relation', u'carnal_knowledge']
Total number of synonyms: 17

interests:
[u'interest', u'involvement']
[u'sake', u'interest']
[u'interest', u'interestingness']
[u'interest']
[u'interest', u'stake']
[u'interest', u'interest_group']
[u'pastime', u'interest', u'pursuit']
[u'interest']
[u'concern', u'interest', u'occupy', u'worry']
[u'matter_to', u'interest']
Total number of synonyms: 21
```



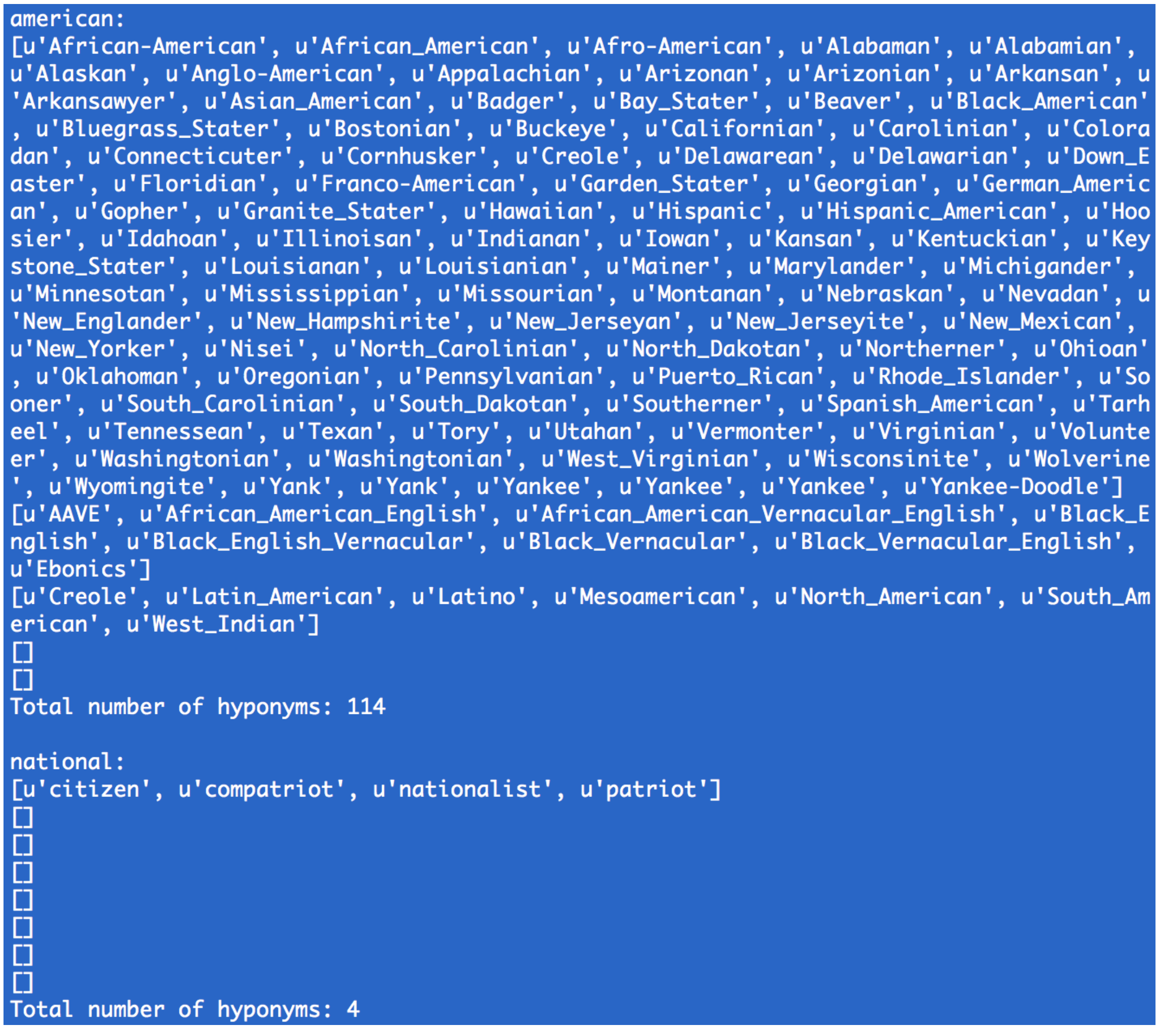
```
political:
[u'political']
[u'political']
[u'political']
Total number of synonyms: 3

principles:
[u'principle', u'rule']
[u'principle']
[u'principle']
[u'principle', u'rule']
[u'principle', u'precept']
[u'rationale', u'principle']
Total number of synonyms: 10

progress:
[u'advancement', u'progress']
[u'progress', u'progression', u'procession', u'advance', u'advancement', u'forward_motio
n', u'onward_motion']
[u'progress', u'progression', u'advance']
[u'progress', u'come_on', u'come_along', u'advance', u'get_on', u'get_along', u'shape_up
']
[u'advance', u'progress', u'pass_on', u'move_on', u'march_on', u'go_on']
[u'build_up', u'work_up', u'build', u'progress']
Total number of synonyms: 29
```

3. List all hyponyms for those 10 words. The word “american” has the largest number of hyponyms, which is 114.

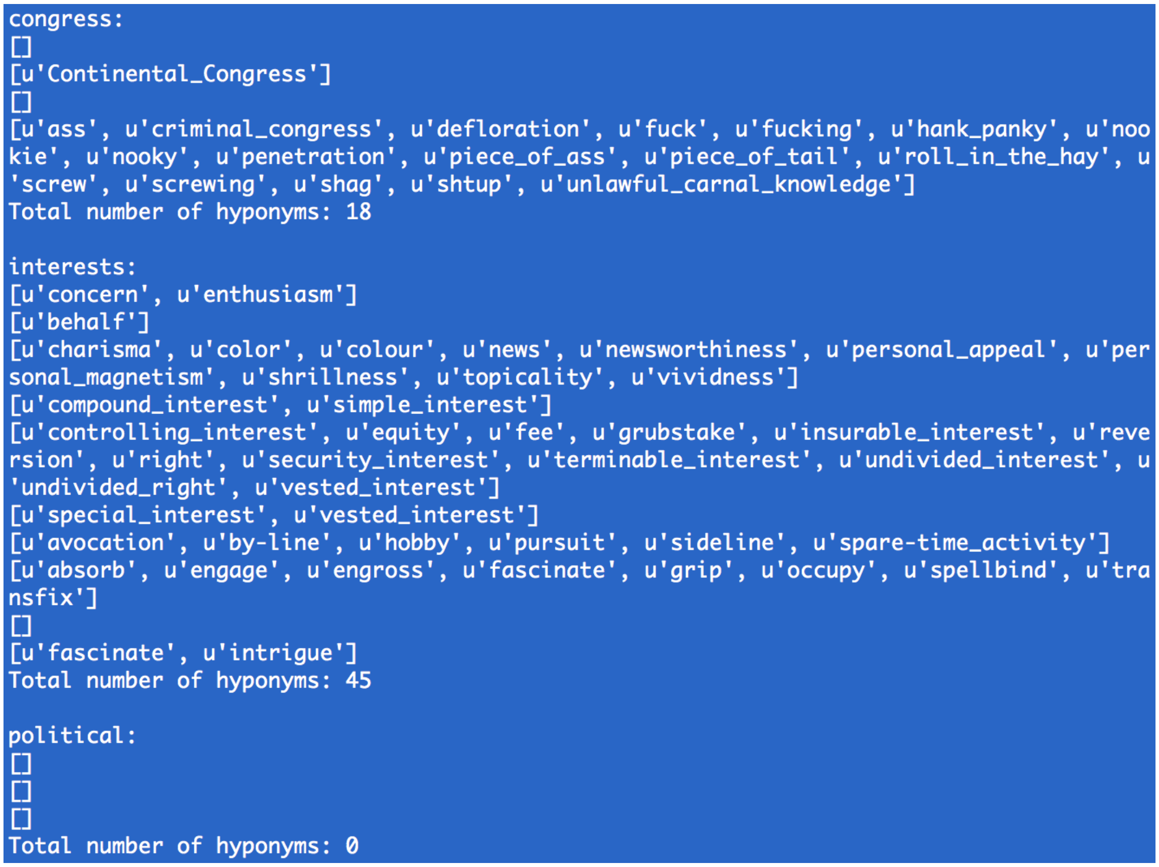
```
>>> words = ['government', 'citizens', 'constitution', 'american', 'national',
'congress', 'interests', 'political', 'principles', 'progress']
>>> for word in words:
...     print "\n%s:" %word
...     sum = 0
...     for synset in wn.synsets(word):
...           types = synset.hyponyms()
...           names = [lemma.name() for synset in types for lemma in
synset.lemmas()]
...           sorted(names)
...           sum += len(names)
...     print "Total number of hyponyms: %d" %sum
...
```



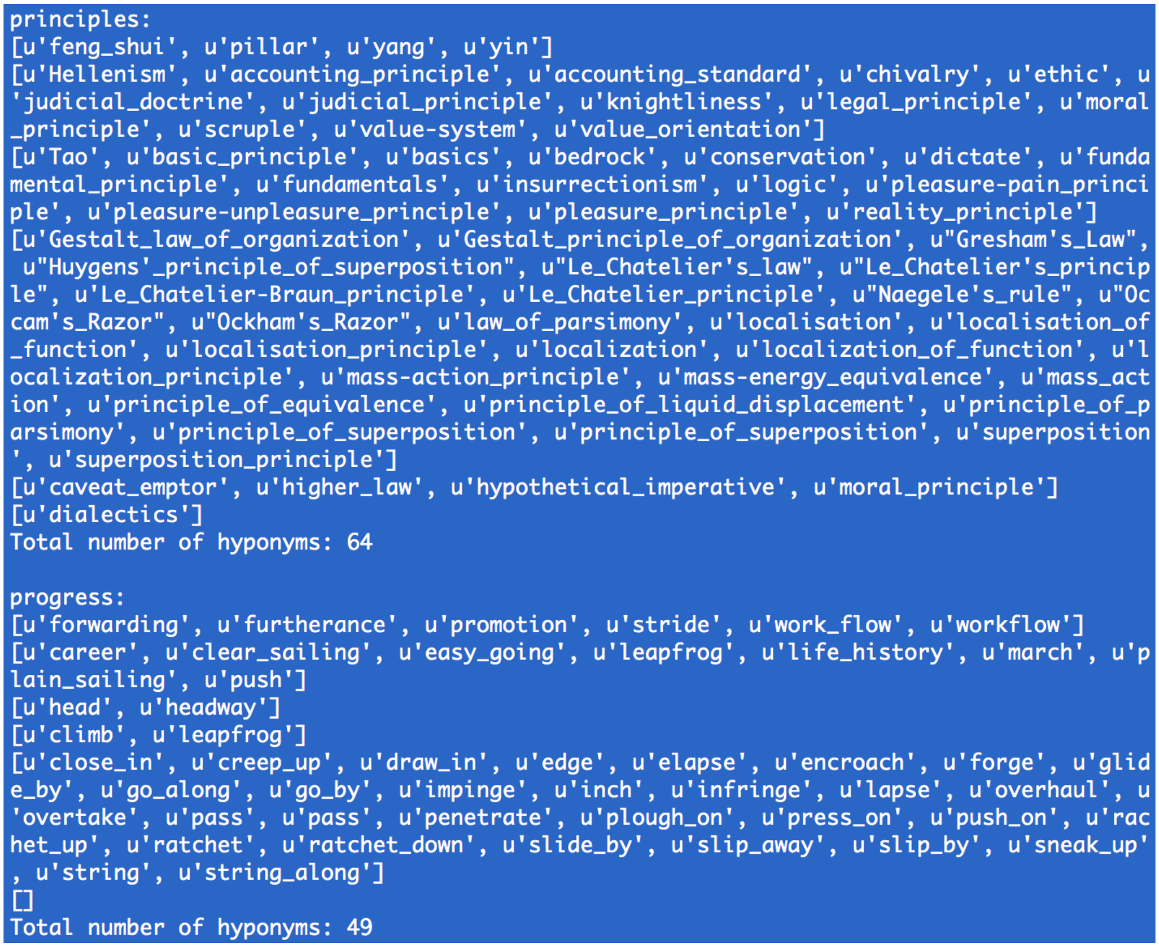
```
>>> words = ['government', 'citizens', 'constitution', 'american', 'national', 'congress
', 'interests', 'political', 'principles', 'progress']
>>> for word in words:
...     print "\n%s:" %word
...     sum = 0
...     for synset in wn.synsets(word):
...             types = synset.hyponyms()
...             names = [lemma.name() for synset in types for lemma in synset.lemmas()]
...             sorted(names)
...             sum += len(names)
...     print "Total number of hyponyms: %d" %sum
...

government:
[u'Downing_Street', u'ancien_regime', u'authoritarian_regime', u'authoritarian_state', u
'bureaucracy', u'court', u'empire', u'federal_government', u'government-in-exile', u'loc
al_government', u'military_government', u'palace', u'papacy', u'pontificate', u'pupet_re
gime', u'puppet_government', u'puppet_state', u'royal_court', u'state', u'state_governme
nt', u'stratocracy', u'totalitarian_state', u'totalitation_regime']
[u'lawmaking', u'legislating', u'legislation', u'misgovernment', u'misrule', u'trust_bus
ting']
[]
[u'geopolitics', u'practical_politics', u'realpolitik']
Total number of hyponyms: 32

citizens:
[u'active_citizen', u'civilian', u'elector', u'freeman', u'freewoman', u'private_citizen
', u'repatriate', u'thane', u'voter']
Total number of hyponyms: 9

constitution:
[]
[u'collectivisation', u'collectivization', u'colonisation', u'colonization', u'communisa
tion', u'communization', u'federation', u'settlement', u'unionisation', u'unionization']
[]
[u'genetic_constitution', u'genotype', u'grain', u'karyotype', u'phenotype', u'structure
', u'texture']
[]
Total number of hyponyms: 17
```



```
american:
[u'African-American', u'African_American', u'Afro-American', u'Alabaman', u'Alabamian',
u'Alaskan', u'Anglo-American', u'Appalachian', u'Arizonan', u'Arizonian', u'Arkansan', u
'Arkansawyer', u'Asian_American', u'Badger', u'Bay_Stater', u'Beaver', u'Black_American'
, u'Bluegrass_Stater', u'Bostonian', u'Buckeye', u'Californian', u'Carolinian', u'Colora
dan', u'Connecticuter', u'Cornhusker', u'Creole', u'Delawarean', u'Delawarian', u'Down_E
aster', u'Floridian', u'Franco-American', u'Garden_Stater', u'Georgian', u'German_Americ
an', u'Gopher', u'Granite_Stater', u'Hawaiian', u'Hispanic', u'Hispanic_American', u'Hoo
sier', u'Idahoan', u'Illinoisan', u'Indianan', u'Iowan', u'Kansan', u'Kentuckian', u'Key
stone_Stater', u'Louisianan', u'Louisianian', u'Mainer', u'Marylander', u'Michigander',
u'Minnesotan', u'Mississippian', u'Missourian', u'Montanan', u'Nebraskan', u'Nevadan', u
'New_Englander', u'New_Hampshirite', u'New_Jerseyan', u'New_Jerseyite', u'New_Mexican',
u'New_Yorker', u'Nisei', u'North_Carolinian', u'North_Dakotan', u'Northerner', u'Ohioan'
, u'Oklahoman', u'Oregonian', u'Pennsylvanian', u'Puerto_Rican', u'Rhode_Islander', u'So
oner', u'South_Carolinian', u'South_Dakotan', u'Southerner', u'Spanish_American', u'Tarh
eel', u'Tennessean', u'Texan', u'Tory', u'Utahan', u'Vermonter', u'Virginian', u'Volunte
er', u'Washingtonian', u'Washingtonian', u'West_Virginian', u'Wisconsinite', u'Wolverine
', u'Wyomingite', u'Yank', u'Yank', u'Yankee', u'Yankee', u'Yankee', u'Yankee-Doodle']
[u'AAVE', u'African_American_English', u'African_American_Vernacular_English', u'Black_E
nglish', u'Black_English_Vernacular', u'Black_Vernacular', u'Black_Vernacular_English',
u'Ebonics']
[u'Creole', u'Latin_American', u'Latino', u'Mesoamerican', u'North_American', u'South_Am
erican', u'West_Indian']
[]
[]
Total number of hyponyms: 114

national:
[u'citizen', u'compatriot', u'nationalist', u'patriot']
[]
[]
[]
[]
[]
[]
[]
Total number of hyponyms: 4
```

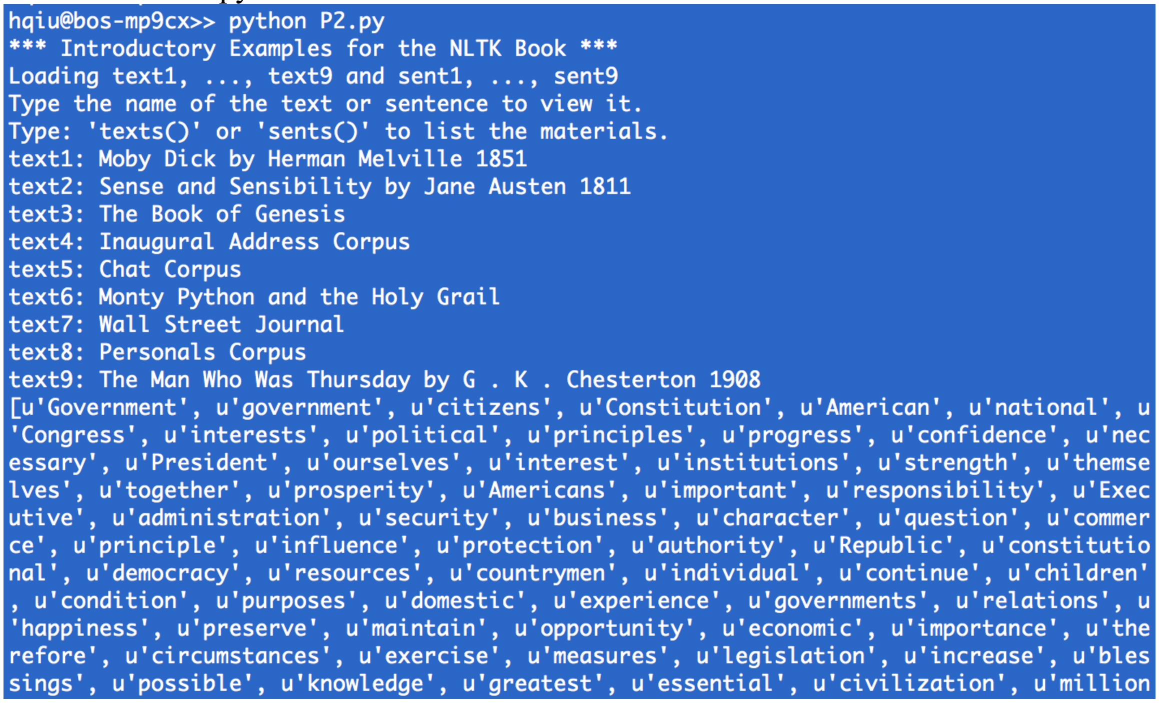


```
congress:
[]
[u'Continental_Congress']
[]
[u'ass', u'criminal_congress', u'defloration', u'fuck', u'fucking', u'hank_panky', u'noo
kie', u'nooky', u'penetration', u'piece_of_ass', u'piece_of_tail', u'roll_in_the_hay', u
'screw', u'screwing', u'shag', u'shtup', u'unlawful_carnal_knowledge']
Total number of hyponyms: 18

interests:
[u'concern', u'enthusiasm']
[u'behalf']
[u'charisma', u'color', u'colour', u'news', u'newsworthiness', u'personal_appeal', u'per
sonal_magnetism', u'shrillness', u'topicality', u'vividness']
[u'compound_interest', u'simple_interest']
[u'controlling_interest', u'equity', u'fee', u'grubstake', u'insurable_interest', u'reve
rsion', u'right', u'security_interest', u'terminable_interest', u'undivided_interest', u
'undivided_right', u'vested_interest']
[u'special_interest', u'vested_interest']
[u'avocation', u'by-line', u'hobby', u'pursuit', u'sideline', u'spare-time_activity']
[u'absorb', u'engage', u'engross', u'fascinate', u'grip', u'occupy', u'spellbind', u'tra
nsfix']
[]
[u'fascinate', u'intrigue']
Total number of hyponyms: 45

political:
[]
[]
[]
Total number of hyponyms: 0
```

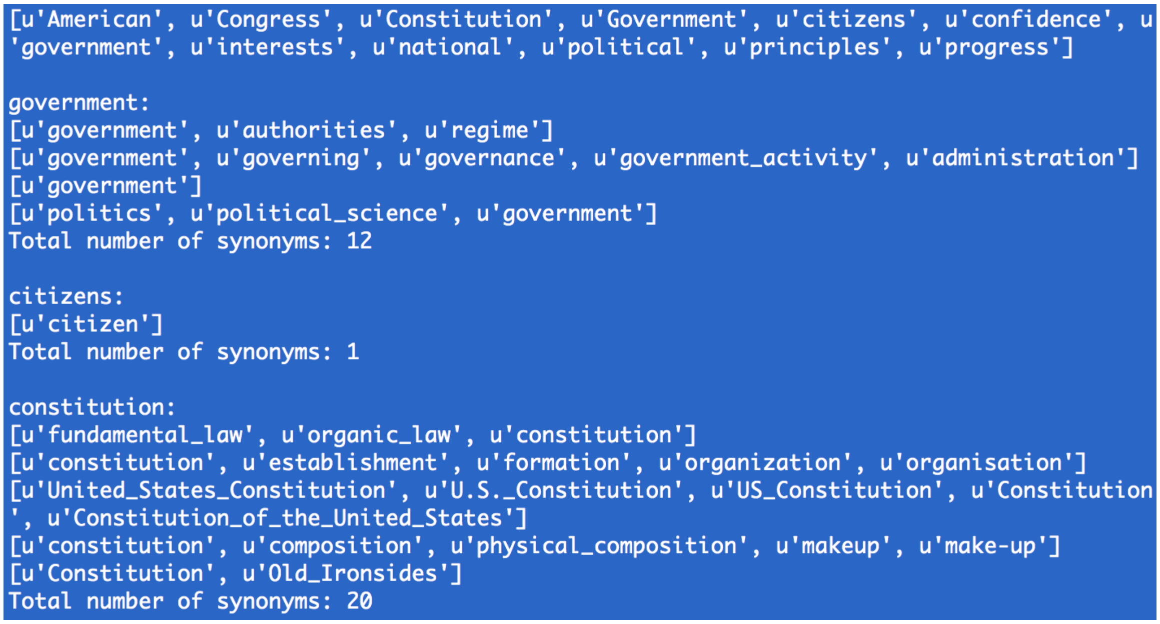


```
principles:
[u'feng_shui', u'pillar', u'yang', u'yin']
[u'Hellenism', u'accounting_principle', u'accounting_standard', u'chivalry', u'ethic', u
'judicial_doctrine', u'judicial_principle', u'knightliness', u'legal_principle', u'moral
_principle', u'scruple', u'value-system', u'value_orientation']
[u'Tao', u'basic_principle', u'basics', u'bedrock', u'conservation', u'dictate', u'funda
mental_principle', u'fundamentals', u'insurrectionism', u'logic', u'pleasure-pain_princi
ple', u'pleasure-unpleasure_principle', u'pleasure_principle', u'reality_principle']
[u'Gestalt_law_of_organization', u'Gestalt_principle_of_organization', u"Gresham's_Law",
 u"Huygens'_principle_of_superposition", u"Le_Chatelier's_law", u"Le_Chatelier's_princip
le", u'Le_Chatelier-Braun_principle', u'Le_Chatelier_principle', u"Naegele's_rule", u"Oc
cam's_Razor", u"Ockham's_Razor", u'law_of_parsimony', u'localisation', u'localisation_of
_function', u'localisation_principle', u'localization', u'localization_of_function', u'l
ocalization_principle', u'mass-action_principle', u'mass-energy_equivalence', u'mass_act
ion', u'principle_of_equivalence', u'principle_of_liquid_displacement', u'principle_of_p
arsimony', u'principle_of_superposition', u'principle_of_superposition', u'superposition
', u'superposition_principle']
[u'caveat_emptor', u'higher_law', u'hypothetical_imperative', u'moral_principle']
[u'dialectics']
Total number of hyponyms: 64

progress:
[u'forwarding', u'furtherance', u'promotion', u'stride', u'work_flow', u'workflow']
[u'career', u'clear_sailing', u'easy_going', u'leapfrog', u'life_history', u'march', u'p
lain_sailing', u'push']
[u'head', u'headway']
[u'climb', u'leapfrog']
[u'close_in', u'creep_up', u'draw_in', u'edge', u'elapse', u'encroach', u'forge', u'glid
e_by', u'go_along', u'go_by', u'impinge', u'inch', u'infringe', u'lapse', u'overhaul', u
'overtake', u'pass', u'pass', u'penetrate', u'plough_on', u'press_on', u'push_on', u'rac
het_up', u'ratchet', u'ratchet_down', u'slide_by', u'slip_away', u'slip_by', u'sneak_up'
, u'string', u'string_along']
[]
Total number of hyponyms: 49
```

Source code: P2.py



```
hqiu@bos-mp9cx>> python P2.py
*** Introductory Examples for the NLTK Book ***
Loading text1, ..., text9 and sent1, ..., sent9
Type the name of the text or sentence to view it.
Type: 'texts()' or 'sents()' to list the materials.
text1: Moby Dick by Herman Melville 1851
text2: Sense and Sensibility by Jane Austen 1811
text3: The Book of Genesis
text4: Inaugural Address Corpus
text5: Chat Corpus
text6: Monty Python and the Holy Grail
text7: Wall Street Journal
text8: Personals Corpus
text9: The Man Who Was Thursday by G . K . Chesterton 1908
[u'Government', u'government', u'citizens', u'Constitution', u'American', u'national', u
'Congress', u'interests', u'political', u'principles', u'progress', u'confidence', u'nec
essary', u'President', u'ourselves', u'interest', u'institutions', u'strength', u'themse
lves', u'together', u'prosperity', u'Americans', u'important', u'responsibility', u'Exec
utive', u'administration', u'security', u'business', u'character', u'question', u'commer
ce', u'principle', u'influence', u'protection', u'authority', u'Republic', u'constitutio
nal', u'democracy', u'resources', u'countrymen', u'individual', u'continue', u'children'
, u'condition', u'purposes', u'domestic', u'experience', u'governments', u'relations', u
'happiness', u'preserve', u'maintain', u'opportunity', u'economic', u'importance', u'the
refore', u'circumstances', u'exercise', u'measures', u'legislation', u'increase', u'bles
sings', u'possible', u'knowledge', u'greatest', u'essential', u'civilization', u'million
```



```
[u'American', u'Congress', u'Constitution', u'Government', u'citizens', u'confidence', u
'government', u'interests', u'national', u'political', u'principles', u'progress']

government:
[u'government', u'authorities', u'regime']
[u'government', u'governing', u'governance', u'government_activity', u'administration']
[u'government']
[u'politics', u'political_science', u'government']
Total number of synonyms: 12

citizens:
[u'citizen']
Total number of synonyms: 1

constitution:
[u'fundamental_law', u'organic_law', u'constitution']
[u'constitution', u'establishment', u'formation', u'organization', u'organisation']
[u'United_States_Constitution', u'U.S._Constitution', u'US_Constitution', u'Constitution
', u'Constitution_of_the_United_States']
[u'constitution', u'composition', u'physical_composition', u'makeup', u'make-up']
[u'Constitution', u'Old_Ironsides']
Total number of synonyms: 20
```

~~**Problem 3.** Create for us one graph displaying cumulative word length distribution for six different genres in Brown corpus. Create a tabular display of basic word statistics for all genres in Brown corpus. Include: average word length, average sentence length, number of concurrences in each genre, percentage of the text consumed by conditional words: would, could and should.~~

You literature for this assignment are chapters 1 and 2 of Natural Language Processing with Python book by Steven Bird et al.