Book 2 (Accessing Text Corpora and Lexical Resources)

SIDE-39-GAB

Bastomy - 1301178418 - Text Mining

1 Accessing Text Corpora

1.1 Gutenberg Corpus

In [1]:
```
import nltk
nltk.corpus.gutenberg.fileids()
```

Out[1]:
```
['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']
```

contoh mengakses corpus kemudia menyimpannya dalam sebuah variabel

In [2]:
```
emma = nltk.corpus.gutenberg.words('austen-emma.txt')
persuasion = nltk.corpus.gutenberg.words('austen-persuasion.txt')
sense = nltk.corpus.gutenberg.words('austen-sense.txt')
poems = nltk.corpus.gutenberg.words('blake-poems.txt')
stories = nltk.corpus.gutenberg.words('bryant-stories.txt')
```

contoh isi dari corpus yang telah di simpan dalam sebuah variabel

In [3]:
```
emma,persuasion,sense,poems,stories
```

Out[3]:
```
(['[', 'Emma', 'by', 'Jane', 'Austen', '1816', ']', ...],
 ['[', 'Persuasion', 'by', 'Jane', 'Austen', '1818', ...],
 ['[', 'Sense', 'and', 'Sensibility', 'by', 'Jane', ...],
 ['[', 'Poems', 'by', 'William', 'Blake', '1789', ']', ...],
 ['[', 'Stories', 'to', 'Tell', 'to', 'Children', 'by', ...])
```

panjang dari setiap corpus yang tersedia

In [4]:
```
print("Panjang corpus emma ",len(emma))
print("Panjang corpus persuasion ",len(persuasion))
print("Panjang corpus sense ",len(sense))
print("Panjang corpus poems ",len(poems))
print("Panjang corpus stories ",len(stories))
```

```
Panjang corpus emma  192427
Panjang corpus persuasion  98171
Panjang corpus sense  141576
Panjang corpus poems  8354
Panjang corpus stories  55563
```

detail dari setiap informasi mengetai text tersebut

In [5]:
```
from nltk.corpus import gutenberg
```

In [6]:
```
 for fileid in gutenberg.fileids():
        num_chars = len(gutenberg.raw(fileid))
        num_words = len(gutenberg.words(fileid))
        num_sents = len(gutenberg.sents(fileid))
        num_vocab = len(set(w.lower() for w in gutenberg.words(fileid)))
        print(round(num_chars/num_words), round(num_words/num_sents), round(num_v
```

```
5 25 26 austen-emma.txt
5 26 17 austen-persuasion.txt
5 28 22 austen-sense.txt
4 34 79 bible-kjv.txt
5 19 5 blake-poems.txt
4 19 14 bryant-stories.txt
4 18 12 burgess-busterbrown.txt
4 20 13 carroll-alice.txt
5 20 12 chesterton-ball.txt
5 23 11 chesterton-brown.txt
5 18 11 chesterton-thursday.txt
4 21 25 edgeworth-parents.txt
5 26 15 melville-moby_dick.txt
5 52 11 milton-paradise.txt
4 12 9 shakespeare-caesar.txt
4 12 8 shakespeare-hamlet.txt
4 12 7 shakespeare-macbeth.txt
5 36 12 whitman-leaves.txt
```

cara mengakses mengakses text

In [7]:
```
macbeth_sentences = gutenberg.sents('shakespeare-macbeth.txt')
emma = gutenberg.sents('austen-emma.txt')
```

In [8]:
```
macbeth_sentences,emma
```

Out[8]:
```
([['[', 'The', 'Tragedie', 'of', 'Macbeth', 'by', 'William', 'Shakespeare', '16
03', ']'], ['Actus', 'Primus', '.'], ...],
 [['[', 'Emma', 'by', 'Jane', 'Austen', '1816', ']'], ['VOLUME', 'I'], ...])
```

In [9]:
```
print("kalimat 1 pada macbeth_sentences ", macbeth_sentences[1])
print("kalimat 2 pada macbeth_sentences ", macbeth_sentences[2])
print("kalimat 1 pada emma ", emma[1])
print("kalimat 2 pada emma ", emma[2])
```

```
kalimat 1 pada macbeth_sentences  ['Actus', 'Primus', '.']
kalimat 2 pada macbeth_sentences  ['Scoena', 'Prima', '.']
kalimat 1 pada emma  ['VOLUME', 'I']
kalimat 2 pada emma  ['CHAPTER', 'I']
```

In [10]:
```
longest_len = max(len(s) for s in emma)
```

In [11]:
```
a= [s for s in emma if len(s) == longest_len]
print(a)
```

```
[['While', 'he', 'lived', ',', 'it', 'must', 'be', 'only', 'an', 'engagement',
';', 'but', 'she', 'flattered', 'herself', ',', 'that', 'if', 'divested', 'of',
'the', 'danger', 'of', 'drawing', 'her', 'away', ',', 'it', 'might', 'become',
'an', 'increase', 'of', 'comfort', 'to', 'him', '.--', 'How', 'to', 'do', 'he
r', 'best', 'by', 'Harriet', ',', 'was', 'of', 'more', 'difficult', 'decision',
';--', 'how', 'to', 'spare', 'her', 'from', 'any', 'unnecessary', 'pain', ';',
'how', 'to', 'make', 'her', 'any', 'possible', 'atonement', ';', 'how', 'to',
'appear', 'least', 'her', 'enemy', '?--', 'On', 'these', 'subjects', ',', 'he
r', 'perplexity', 'and', 'distress', 'were', 'very', 'great', '--', 'and', 'he
r', 'mind', 'had', 'to', 'pass', 'again', 'and', 'again', 'through', 'every',
'bitter', 'reproach', 'and', 'sorrowful', 'regret', 'that', 'had', 'ever', 'sur
rounded', 'it', '.--', 'She', 'could', 'only', 'resolve', 'at', 'last', ',', 't
hat', 'she', 'would', 'still', 'avoid', 'a', 'meeting', 'with', 'her', ',', 'an
d', 'communicate', 'all', 'that', 'need', 'be', 'told', 'by', 'letter', ';', 't
hat', 'it', 'would', 'be', 'inexpressibly', 'desirable', 'to', 'have', 'her',
'removed', 'just', 'now', 'for', 'a', 'time', 'from', 'Highbury', ',', 'and',
'--', 'indulging', 'in', 'one', 'scheme', 'more', '--', 'nearly', 'resolve',
',', 'that', 'it', 'might', 'be', 'practicable', 'to', 'get', 'an', 'invitatio
n', 'for', 'her', 'to', 'Brunswick', 'Square', '.--', 'Isabella', 'had', 'bee
n', 'pleased', 'with', 'Harriet', ';', 'and', 'a', 'few', 'weeks', 'spent', 'i
n', 'London', 'must', 'give', 'her', 'some', 'amusement', '.--', 'She', 'did',
'not', 'think', 'it', 'in', 'Harriet', "'", 's', 'nature', 'to', 'escape', 'bei
ng', 'benefited', 'by', 'novelty', 'and', 'variety', ',', 'by', 'the', 'street
s', ',', 'the', 'shops', ',', 'and', 'the', 'children', '.--', 'At', 'any', 'ra
te', ',', 'it', 'would', 'be', 'a', 'proof', 'of', 'attention', 'and', 'kindnes
s', 'in', 'herself', ',', 'from', 'whom', 'every', 'thing', 'was', 'due', ';',
'a', 'separation', 'for', 'the', 'present', ';', 'an', 'averting', 'of', 'the',
'evil', 'day', ',', 'when', 'they', 'must', 'all', 'be', 'together', 'again',
'.']]
```

In [12]:
```
print("text terpanjang pada emma memiliki ",len(a[0])," kata")
```

```
text terpanjang pada emma memiliki  274  kata
```

1.2 Web and Chat Text

In [13]:
```
from nltk.corpus import webtext
```

In [14]:
```
for fileid in webtext.fileids():
    print(fileid, webtext.raw(fileid)[:65], '...')
```

```
firefox.txt Cookie Manager: "Don't allow sites that set removed cookies to se
...
grail.txt SCENE 1: [wind] [clop clop clop]
KING ARTHUR: Whoa there!  [clop ...
overheard.txt White guy: So, do you have any plans for this evening?
Asian girl ...
pirates.txt PIRATES OF THE CARRIBEAN: DEAD MAN'S CHEST, by Ted Elliott & Terr
...
singles.txt 25 SEXY MALE, seeks attrac older single lady, for discreet encoun
...
wine.txt Lovely delicate, fragrant Rhone wine. Polished leather and strawb ...
```

menampilkan 10 chat room pertama dari file xml

In [15]:
```
from nltk.corpus import nps_chat
```

In [16]:
```
chatroom = nps_chat.posts('10-19-20s_706posts.xml')
for i in range(len(chatroom)):
    if i< 10:
        print(chatroom[i])
```

```
['now', 'im', 'left', 'with', 'this', 'gay', 'name']
[':P']
['PART']
['hey', 'everyone']
['ah', 'well']
['NICK', ':', 'U7']
['U7', 'is', 'a', 'gay', 'name', '.']
['.', 'ACTION', 'gives', 'U121', 'a', 'golf', 'clap', '.']
[':)']
['JOIN']
```

1.3 Brown Corpus

daftar dari brown corpus

ID	File	Genre	Description
A16	ca16	news	Chicago Tribune: *Society Reportage*
B02	cb02	editorial	Christian Science Monitor: *Editorials*
C17	cc17	reviews	Time Magazine: *Reviews*
D12	cd12	religion	Underwood: *Probing the Ethics of Realtors*
E36	ce36	hobbies	Norling: *Renting a Car in Europe*
F25	cf25	lore	Boroff: *Jewish Teenage Culture*
G22	cg22	belles_lettres	Reiner: *Coping with Runaway Technology*
H15	ch15	government	US Office of Civil and Defence Mobilization: *The Family Fallout Shelter*
J17	cj19	learned	Mosteller: *Probability with Statistical Applications*
K04	ck04	fiction	W.E.B. Du Bois: *Worlds of Color*
L13	cl13	mystery	Hitchens: *Footsteps in the Night*
M01	cm01	science_fiction	Heinlein: *Stranger in a Strange Land*
N14	cn15	adventure	Field: *Rattlesnake Ridge*
P12	cp12	romance	Callaghan: *A Passion in Rome*
R06	cr06	humor	Thurber: *The Future, If Any, of Comedy*

sebelum menggunakan brown corpus kita harus mengimport terlebih dahulu seperi code dibawah ini

In [17]:
```
from nltk.corpus import brown
```

untuk mengecek ketersediaan kategori brown bisa dengan menggunakan code di bawah ini

In [18]:
```
brown.categories()
```

Out[18]:
```
['adventure',
 'belles_lettres',
 'editorial',
 'fiction',
 'government',
 'hobbies',
 'humor',
 'learned',
 'lore',
 'mystery',
 'news',
 'religion',
 'reviews',
 'romance',
 'science_fiction']
```

untuk menampilkan kategori tertentu bisa menggunakan code dibawah ini sesuai dengan kategori yang di inginkan

In [19]:
```
print("humor ",brown.words(categories='humor'))
print("hobbies ",brown.words(categories='hobbies'))
print("editorial ",brown.words(categories='editorial'))
print("mystery ",brown.words(categories='mystery'))
```

```
humor  ['It', 'was', 'among', 'these', 'that', 'Hinkle', ...]
hobbies  ['Too', 'often', 'a', 'beginning', 'bodybuilder', ...]
editorial  ['Assembly', 'session', 'brought', 'much', 'good', ...]
mystery  ['There', 'were', 'thirty-eight', 'patients', 'on', ...]
```

untuk melihat beberapa kategori sekaligus kita dapat menggunakan kode dibawah ini

untuk melihat daftar field kita bisa menggunakan kode dibawah ini

In [20]:
```
brown.sents(categories=['religion', 'science_fiction', 'hobbies','learned'])
```

Out[20]:
```
[['As', 'a', 'result', ',', 'although', 'we', 'still', 'make', 'use', 'of', 'th
is', 'distinction', ',', 'there', 'is', 'much', 'confusion', 'as', 'to', 'the',
'meaning', 'of', 'the', 'basic', 'terms', 'employed', '.'], ['Just', 'what', 'i
s', 'meant', 'by', '``', 'spirit', "''", 'and', 'by', '``', 'matter', "''",
'?', '?'], ...]
```

In [21]:
```
print(brown.fileids())
```

```
['ca01', 'ca02', 'ca03', 'ca04', 'ca05', 'ca06', 'ca07', 'ca08', 'ca09', 'ca1
0', 'ca11', 'ca12', 'ca13', 'ca14', 'ca15', 'ca16', 'ca17', 'ca18', 'ca19', 'ca
20', 'ca21', 'ca22', 'ca23', 'ca24', 'ca25', 'ca26', 'ca27', 'ca28', 'ca29', 'c
a30', 'ca31', 'ca32', 'ca33', 'ca34', 'ca35', 'ca36', 'ca37', 'ca38', 'ca39',
'ca40', 'ca41', 'ca42', 'ca43', 'ca44', 'cb01', 'cb02', 'cb03', 'cb04', 'cb05',
'cb06', 'cb07', 'cb08', 'cb09', 'cb10', 'cb11', 'cb12', 'cb13', 'cb14', 'cb15',
'cb16', 'cb17', 'cb18', 'cb19', 'cb20', 'cb21', 'cb22', 'cb23', 'cb24', 'cb25',
'cb26', 'cb27', 'cc01', 'cc02', 'cc03', 'cc04', 'cc05', 'cc06', 'cc07', 'cc08',
'cc09', 'cc10', 'cc11', 'cc12', 'cc13', 'cc14', 'cc15', 'cc16', 'cc17', 'cd01',
'cd02', 'cd03', 'cd04', 'cd05', 'cd06', 'cd07', 'cd08', 'cd09', 'cd10', 'cd11',
'cd12', 'cd13', 'cd14', 'cd15', 'cd16', 'cd17', 'ce01', 'ce02', 'ce03', 'ce04',
'ce05', 'ce06', 'ce07', 'ce08', 'ce09', 'ce10', 'ce11', 'ce12', 'ce13', 'ce14',
'ce15', 'ce16', 'ce17', 'ce18', 'ce19', 'ce20', 'ce21', 'ce22', 'ce23', 'ce24',
'ce25', 'ce26', 'ce27', 'ce28', 'ce29', 'ce30', 'ce31', 'ce32', 'ce33', 'ce34',
'ce35', 'ce36', 'cf01', 'cf02', 'cf03', 'cf04', 'cf05', 'cf06', 'cf07', 'cf08',
'cf09', 'cf10', 'cf11', 'cf12', 'cf13', 'cf14', 'cf15', 'cf16', 'cf17', 'cf18',
'cf19', 'cf20', 'cf21', 'cf22', 'cf23', 'cf24', 'cf25', 'cf26', 'cf27', 'cf28',
'cf29', 'cf30', 'cf31', 'cf32', 'cf33', 'cf34', 'cf35', 'cf36', 'cf37', 'cf38',
'cf39', 'cf40', 'cf41', 'cf42', 'cf43', 'cf44', 'cf45', 'cf46', 'cf47', 'cf48',
'cg01', 'cg02', 'cg03', 'cg04', 'cg05', 'cg06', 'cg07', 'cg08', 'cg09', 'cg10',
'cg11', 'cg12', 'cg13', 'cg14', 'cg15', 'cg16', 'cg17', 'cg18', 'cg19', 'cg20',
'cg21', 'cg22', 'cg23', 'cg24', 'cg25', 'cg26', 'cg27', 'cg28', 'cg29', 'cg30',
'cg31', 'cg32', 'cg33', 'cg34', 'cg35', 'cg36', 'cg37', 'cg38', 'cg39', 'cg40',
'cg41', 'cg42', 'cg43', 'cg44', 'cg45', 'cg46', 'cg47', 'cg48', 'cg49', 'cg50',
'cg51', 'cg52', 'cg53', 'cg54', 'cg55', 'cg56', 'cg57', 'cg58', 'cg59', 'cg60',
'cg61', 'cg62', 'cg63', 'cg64', 'cg65', 'cg66', 'cg67', 'cg68', 'cg69', 'cg70',
'cg71', 'cg72', 'cg73', 'cg74', 'cg75', 'ch01', 'ch02', 'ch03', 'ch04', 'ch05',
'ch06', 'ch07', 'ch08', 'ch09', 'ch10', 'ch11', 'ch12', 'ch13', 'ch14', 'ch15',
'ch16', 'ch17', 'ch18', 'ch19', 'ch20', 'ch21', 'ch22', 'ch23', 'ch24', 'ch25',
'ch26', 'ch27', 'ch28', 'ch29', 'ch30', 'cj01', 'cj02', 'cj03', 'cj04', 'cj05',
'cj06', 'cj07', 'cj08', 'cj09', 'cj10', 'cj11', 'cj12', 'cj13', 'cj14', 'cj15',
'cj16', 'cj17', 'cj18', 'cj19', 'cj20', 'cj21', 'cj22', 'cj23', 'cj24', 'cj25',
'cj26', 'cj27', 'cj28', 'cj29', 'cj30', 'cj31', 'cj32', 'cj33', 'cj34', 'cj35',
'cj36', 'cj37', 'cj38', 'cj39', 'cj40', 'cj41', 'cj42', 'cj43', 'cj44', 'cj45',
'cj46', 'cj47', 'cj48', 'cj49', 'cj50', 'cj51', 'cj52', 'cj53', 'cj54', 'cj55',
'cj56', 'cj57', 'cj58', 'cj59', 'cj60', 'cj61', 'cj62', 'cj63', 'cj64', 'cj65',
'cj66', 'cj67', 'cj68', 'cj69', 'cj70', 'cj71', 'cj72', 'cj73', 'cj74', 'cj75',
'cj76', 'cj77', 'cj78', 'cj79', 'cj80', 'ck01', 'ck02', 'ck03', 'ck04', 'ck05',
'ck06', 'ck07', 'ck08', 'ck09', 'ck10', 'ck11', 'ck12', 'ck13', 'ck14', 'ck15',
'ck16', 'ck17', 'ck18', 'ck19', 'ck20', 'ck21', 'ck22', 'ck23', 'ck24', 'ck25',
'ck26', 'ck27', 'ck28', 'ck29', 'cl01', 'cl02', 'cl03', 'cl04', 'cl05', 'cl06',
'cl07', 'cl08', 'cl09', 'cl10', 'cl11', 'cl12', 'cl13', 'cl14', 'cl15', 'cl16',
'cl17', 'cl18', 'cl19', 'cl20', 'cl21', 'cl22', 'cl23', 'cl24', 'cm01', 'cm02',
'cm03', 'cm04', 'cm05', 'cm06', 'cn01', 'cn02', 'cn03', 'cn04', 'cn05', 'cn06',
'cn07', 'cn08', 'cn09', 'cn10', 'cn11', 'cn12', 'cn13', 'cn14', 'cn15', 'cn16',
'cn17', 'cn18', 'cn19', 'cn20', 'cn21', 'cn22', 'cn23', 'cn24', 'cn25', 'cn26',
'cn27', 'cn28', 'cn29', 'cp01', 'cp02', 'cp03', 'cp04', 'cp05', 'cp06', 'cp07',
'cp08', 'cp09', 'cp10', 'cp11', 'cp12', 'cp13', 'cp14', 'cp15', 'cp16', 'cp17',
'cp18', 'cp19', 'cp20', 'cp21', 'cp22', 'cp23', 'cp24', 'cp25', 'cp26', 'cp27',
'cp28', 'cp29', 'cr01', 'cr02', 'cr03', 'cr04', 'cr05', 'cr06', 'cr07', 'cr08',
'cr09']
```

untuk menampilkan field tertentu bisa menggunakan code dibawah ini sesuai dengan field yang di inginkan

In [22]:
```
print("cg22 ",brown.words(fileids=['cg22']))
print("cj77 ",brown.words(fileids=['cj77']))
print("cr06 ",brown.words(fileids=['cr06']))
print("cr09 ",brown.words(fileids=['cr09']))
print("ck13 ",brown.words(fileids=['ck13']))
```

```
cg22  ['Does', 'our', 'society', 'have', 'a', 'runaway', ',', ...]
cj77  ['Temperature', 'of', 'the', 'wash', 'and', 'rinse', ...]
cr06  ['I', 'called', 'the', 'other', 'afternoon', 'on', ...]
cr09  ['Dear', 'Sirs', ':', 'Let', 'me', 'begin', 'by', ...]
ck13  ['In', 'the', 'dim', 'underwater', 'light', 'they', ...]
```

untuk melihat jumlah kemunculan kata dalam sebuah kategori kita dapat menggunakan kode di bawah ini

In [23]:
```
cfd = nltk.ConditionalFreqDist(
    (genre, word)
    for genre in brown.categories()
    for word in brown.words(categories=genre))
genres = ['news', 'religion', 'hobbies', 'science_fiction', 'romance', 'humor']
modals = ['can', 'could', 'may', 'might', 'must', 'will','be']
```

In [24]:
```
cfd.tabulate(conditions=genres, samples=modals)
```

```
                  can could   may might  must  will    be
           news    93    86    66    38    50   389   526
       religion    82    59    78    12    54    71   243
        hobbies   268    58   131    22    83   264   508
science_fiction    16    49     4    12     8    16    80
        romance    74   193    11    51    45    43   289
          humor    16    30     8     8     9    13    78
```

1.4 Reuters Corpus

In [25]:
```
from nltk.corpus import reuters
```

untuk melihat semua field dari reuters bisa menggunakan kode dibawah ini

In [26]:
```
a = reuters.fileids()
```

In [27]:
```
for i in range(30):
    print(a[i])
```

```
test/14826
test/14828
test/14829
test/14832
test/14833
test/14839
test/14840
test/14841
test/14842
test/14843
test/14844
test/14849
test/14852
test/14854
test/14858
test/14859
test/14860
test/14861
test/14862
test/14863
test/14865
test/14867
test/14872
test/14873
test/14875
test/14876
test/14877
test/14881
test/14882
test/14885
```

untuk melihat semua categori dari reuters bisa menggunakan kode dibawah ini

In [28]:
```
reuters.categories()
```

Out[28]:
```
['acq',
 'alum',
 'barley',
 'bop',
 'carcass',
 'castor-oil',
 'cocoa',
 'coconut',
 'coconut-oil',
 'coffee',
 'copper',
 'copra-cake',
 'corn',
 'cotton',
 'cotton-oil',
 'cpi',
 'cpu',
 'crude',
 'dfl',
 'dlr',
 'dmk',
 'earn',
 'fuel',
 'gas',
 'gnp',
 'gold',
 'grain',
 'groundnut',
 'groundnut-oil',
 'heat',
 'hog',
 'housing',
 'income',
 'instal-debt',
 'interest',
 'ipi',
 'iron-steel',
 'jet',
 'jobs',
 'l-cattle',
 'lead',
 'lei',
 'lin-oil',
 'livestock',
 'lumber',
 'meal-feed',
 'money-fx',
 'money-supply',
 'naphtha',
 'nat-gas',
 'nickel',
 'nkr',
 'nzdlr',
 'oat',
 'oilseed',
```

```
 'orange',
 'palladium',
 'palm-oil',
 'palmkernel',
 'pet-chem',
 'platinum',
 'potato',
 'propane',
 'rand',
 'rape-oil',
 'rapeseed',
 'reserves',
 'retail',
 'rice',
 'rubber',
 'rye',
 'ship',
 'silver',
 'sorghum',
 'soy-meal',
 'soy-oil',
 'soybean',
 'strategic-metal',
 'sugar',
 'sun-meal',
 'sun-oil',
 'sunseed',
 'tea',
 'tin',
 'trade',
 'veg-oil',
 'wheat',
 'wpi',
 'yen',
 'zinc']
```

untuk melihat training tertentu masuk ke dalam kategori yang mana bisa menggunakan kode di bawah ini

In [29]:
```
print("training/9865 kategori = ",reuters.categories('training/9865'))
print("test/14960 kategori = ",reuters.categories('test/14960'))
print("training/10752 kategori = ",reuters.categories('training/10752'))
print("training/11302 kategori = ",reuters.categories('training/11302'))
print("training/11643 kategori = ",reuters.categories('training/11643'))
```

```
training/9865 kategori =  ['barley', 'corn', 'grain', 'wheat']
test/14960 kategori =  ['earn']
training/10752 kategori =  ['coffee']
training/11302 kategori =  ['earn']
training/11643 kategori =  ['earn']
```

untuk melihat langsung beberapa training atau testing masuk ke kategori mana bisa menggunakan kode dibawah ini

In [30]:
```
reuters.categories(['training/9865', 'training/9880','training/10752','training/
```

Out[30]:
```
['barley', 'coffee', 'corn', 'earn', 'grain', 'money-fx', 'wheat']
```

jika di balik kita ingin mengetahui training atau test mana saja yang termasuk ke dalam kategori tertentu

In [31]:
```
reuters.fileids('rand')
```

Out[31]:
```
['test/21535', 'training/7043', 'training/9336']
```

jika di balik kita ingin mengetahui training atau test mana saja yang termasuk ke dalam beberapa kategori tertentu

In [32]:
```
reuters.fileids(['oat','rice'])
```

Out[32]:
```
['test/14832',
 'test/14858',
 'test/15206',
 'test/15367',
 'test/15567',
 'test/15871',
 'test/15875',
 'test/15906',
 'test/15928',
 'test/15952',
 'test/16071',
 'test/16079',
 'test/16099',
 'test/16147',
 'test/16601',
 'test/16784',
 'test/17962',
 'test/18609',
 'test/18642',
 'test/19059',
 'test/19165',
 'test/19275',
 'test/19388',
 'test/19668',
 'test/19835',
 'test/19947',
 'test/20052',
 'test/21570',
 'training/10519',
 'training/10830',
 'training/11216',
 'training/11536',
 'training/11640',
 'training/11655',
 'training/13123',
 'training/13129',
 'training/13294',
 'training/13852',
 'training/1405',
 'training/14515',
 'training/1590',
 'training/1882',
 'training/1915',
 'training/197',
 'training/228',
 'training/235',
 'training/3358',
 'training/3445',
 'training/4133',
 'training/417',
 'training/4314',
 'training/4549',
 'training/4678',
 'training/4719',
 'training/5',
```

```
 'training/5266',
 'training/5610',
 'training/6114',
 'training/6153',
 'training/6269',
 'training/7160',
 'training/7545',
 'training/7917',
 'training/8161',
 'training/8413',
 'training/855',
 'training/8759',
 'training/9018',
 'training/97',
 'training/9705']
```

menampilkan 10 kata pertama pada data training atau test tertentu bisa menggunakan kode dibawah ini

In [33]:
```
reuters.words('training/12067')[:10]
```

Out[33]:
```
['WAXMAN',
 'INDUSTRIES',
 'INC',
 '&',
 'lt',
 ';',
 'WAXM',
 '>',
 'REGULAR',
 'PAYOUT']
```

jika ingin langsung dari beberapa data bisa menggunakan kode dibawah ini

In [34]:
```
reuters.words(['training/9865', 'training/9880','training/12067'])
```

Out[34]:
```
['FRENCH', 'FREE', 'MARKET', 'CEREAL', 'EXPORT', ...]
```

jika ingin mencari kata yang terdapat dalam suatu kategori dapat menggunakan kode dibawah ini

In [35]:
```
reuters.words(categories='tea')
```

Out[35]:
```
['PAKISTAN', 'CONFIRMS', 'KENYA', 'TEA', 'IMPORT', ...]
```

jika ingin mencari kata yang terdapat dalam beberapa kategori dapat menggunakan kode dibawah ini

In [36]:
```
reuters.words(categories=['barley', 'rice'])
```

Out[36]:
```
['THAI', 'TRADE', 'DEFICIT', 'WIDENS', 'IN', 'FIRST', ...]
```

1.5 Inaugural Address Corpus

In [37]:
```
from nltk.corpus import inaugural
```

field yang tersedia pada inagural corpus

In [38]:
```
inaugural.fileids()
```

Out[38]:
```
['1789-Washington.txt',
 '1793-Washington.txt',
 '1797-Adams.txt',
 '1801-Jefferson.txt',
 '1805-Jefferson.txt',
 '1809-Madison.txt',
 '1813-Madison.txt',
 '1817-Monroe.txt',
 '1821-Monroe.txt',
 '1825-Adams.txt',
 '1829-Jackson.txt',
 '1833-Jackson.txt',
 '1837-VanBuren.txt',
 '1841-Harrison.txt',
 '1845-Polk.txt',
 '1849-Taylor.txt',
 '1853-Pierce.txt',
 '1857-Buchanan.txt',
 '1861-Lincoln.txt',
 '1865-Lincoln.txt',
 '1869-Grant.txt',
 '1873-Grant.txt',
 '1877-Hayes.txt',
 '1881-Garfield.txt',
 '1885-Cleveland.txt',
 '1889-Harrison.txt',
 '1893-Cleveland.txt',
 '1897-McKinley.txt',
 '1901-McKinley.txt',
 '1905-Roosevelt.txt',
 '1909-Taft.txt',
 '1913-Wilson.txt',
 '1917-Wilson.txt',
 '1921-Harding.txt',
 '1925-Coolidge.txt',
 '1929-Hoover.txt',
 '1933-Roosevelt.txt',
 '1937-Roosevelt.txt',
 '1941-Roosevelt.txt',
 '1945-Roosevelt.txt',
 '1949-Truman.txt',
 '1953-Eisenhower.txt',
 '1957-Eisenhower.txt',
 '1961-Kennedy.txt',
 '1965-Johnson.txt',
 '1969-Nixon.txt',
 '1973-Nixon.txt',
 '1977-Carter.txt',
 '1981-Reagan.txt',
 '1985-Reagan.txt',
 '1989-Bush.txt',
 '1993-Clinton.txt',
 '1997-Clinton.txt',
 '2001-Bush.txt',
```

```
 '2005-Bush.txt',
 '2009-Obama.txt']
```

untuk membuat sebuah plot dari kemunculan kata tertentu pada beberapa tahun tertentu dapat menggunakan kode dibawah ini

In [39]:
```
cfd = nltk.ConditionalFreqDist(
    (target, fileid[:4])
    for fileid in inaugural.fileids()
    for w in inaugural.words(fileid)
    for target in ['america', 'citizen','people']
    if w.lower().startswith(target))
```

In [40]:
```
cfd.plot()
```

```
<Figure size 640x480 with 1 Axes>
```

1.6 Annotated Text Corpora

beberapa corpus yang tersedia

Corpus	Compiler	Contents
Brown Corpus	Francis, Kucera	15 genres, 1.15M words, tagged, categorized
CESS Treebanks	CLiC-UB	1M words, tagged and parsed (Catalan, Spanish)
Chat-80 Data Files	Pereira & Warren	World Geographic Database
CMU Pronouncing Dictionary	CMU	127k entries
CoNLL 2000 Chunking Data	CoNLL	270k words, tagged and chunked
CoNLL 2002 Named Entity	CoNLL	700k words, pos- and named-entity-tagged (Dutch, Spanish)
CoNLL 2007 Dependency Treebanks (sel)	CoNLL	150k words, dependency parsed (Basque, Catalan)
Dependency Treebank	Narad	Dependency parsed version of Penn Treebank sample
FrameNet	Fillmore, Baker et al	10k word senses, 170k manually annotated sentences
Floresta Treebank	Diana Santos et al	9k sentences, tagged and parsed (Portuguese)
Gazetteer Lists	Various	Lists of cities and countries
Genesis Corpus	Misc web sources	6 texts, 200k words, 6 languages
Gutenberg (selections)	Hart, Newby, et al	18 texts, 2M words
Inaugural Address Corpus	CSpan	US Presidential Inaugural Addresses (1789-present)
Indian POS-Tagged Corpus	Kumaran et al	60k words, tagged (Bangla, Hindi, Marathi, Telugu)
MacMorpho Corpus	NILC, USP, Brazil	1M words, tagged (Brazilian Portuguese)
Movie Reviews	Pang, Lee	2k movie reviews with sentiment polarity classification
Names Corpus	Kantrowitz, Ross	8k male and female names
NIST 1999 Info Extr (selections)	Garofolo	63k words, newswire and named-entity SGML markup
Nombank	Meyers	115k propositions, 1400 noun frames
NPS Chat Corpus	Forsyth, Martell	10k IM chat posts, POS-tagged and dialogue-act tagged
Open Multilingual WordNet	Bond et al	15 languages, aligned to English WordNet
PP Attachment Corpus	Ratnaparkhi	28k prepositional phrases, tagged as noun or verb modifiers
Proposition Bank	Palmer	113k propositions, 3300 verb frames
Question Classification	Li, Roth	6k questions, categorized
Reuters Corpus	Reuters	1.3M words, 10k news documents, categorized
Roget's Thesaurus	Project Gutenberg	200k words, formatted text
RTE Textual Entailment	Dagan et al	8k sentence pairs, categorized
SEMCOR	Rus, Mihalcea	880k words, part-of-speech and sense tagged
Senseval 2 Corpus	Pedersen	600k words, part-of-speech and sense tagged
SentiWordNet	Esuli, Sebastiani	sentiment scores for 145k WordNet synonym sets
Shakespeare texts (selections)	Bosak	8 books in XML format
State of the Union Corpus	CSPAN	485k words, formatted text
Stopwords Corpus	Porter et al	2,400 stopwords for 11 languages
Swadesh Corpus	Wiktionary	comparative wordlists in 24 languages
Switchboard Corpus (selections)	LDC	36 phonecalls, transcribed, parsed
Univ Decl of Human Rights	United Nations	480k words, 300+ languages
Penn Treebank (selections)	LDC	40k words, tagged and parsed
TIMIT Corpus (selections)	NIST/LDC	audio files and transcripts for 16 speakers
VerbNet 2.1	Palmer et al	5k verbs, hierarchically organized, linked to WordNet
Wordlist Corpus	OpenOffice.org et al	960k words and 20k affixes for 8 languages
WordNet 3.0 (English)	Miller, Fellbaum	145k synonym sets

1.7 Corpora in Other Languages

untuk menggunakan corpus di bahasa lain kita bisa menggunakan kode dibawah ini

In [41]:
```
import nltk
nltk.download('cess_esp')
nltk.download('floresta')
nltk.download('indian')
nltk.download('udhr')
```

```
[nltk_data] Downloading package cess_esp to
[nltk_data]     C:\Users\Bastomy\AppData\Roaming\nltk_data...
[nltk_data]   Package cess_esp is already up-to-date!
[nltk_data] Downloading package floresta to
[nltk_data]     C:\Users\Bastomy\AppData\Roaming\nltk_data...
[nltk_data]   Package floresta is already up-to-date!
[nltk_data] Downloading package indian to
[nltk_data]     C:\Users\Bastomy\AppData\Roaming\nltk_data...
[nltk_data]   Package indian is already up-to-date!
[nltk_data] Downloading package udhr to
[nltk_data]     C:\Users\Bastomy\AppData\Roaming\nltk_data...
[nltk_data]   Package udhr is already up-to-date!
```

Out[41]:
```
True
```

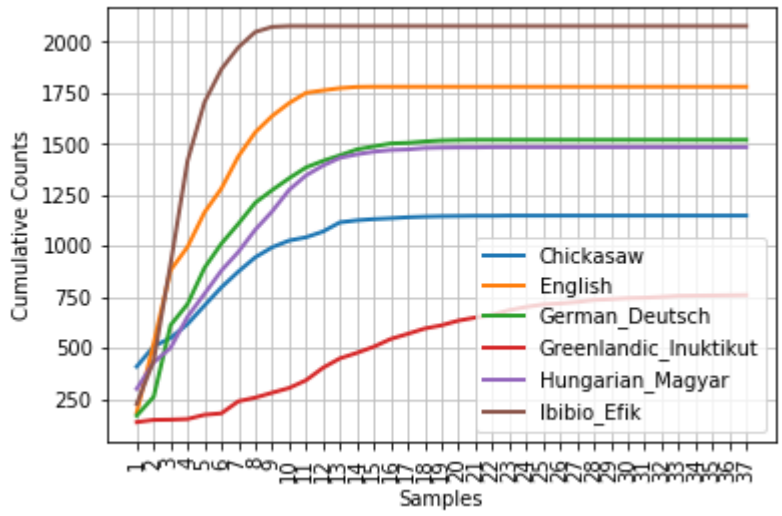
In [42]:
```
print("Spanyol ",nltk.corpus.cess_esp.words())
print("Floresta ",nltk.corpus.floresta.words())
print("Indian ",nltk.corpus.indian.words('hindi.pos'))
```

```
Spanyol  ['El', 'grupo', 'estatal', 'Electricité_de_France', ...]
Floresta  ['Um', 'revivalismo', 'refrescante', 'O', '7_e_Meio', ...]
Indian  ['पूर्ण', 'प्रतिबंध', 'हटाओ', ':', 'इराक', 'संयुक्त', ...]
```

In [43]:
```
from nltk.corpus import udhr
languages = ['Chickasaw', 'English', 'German_Deutsch','Greenlandic_Inuktikut', 'I
```

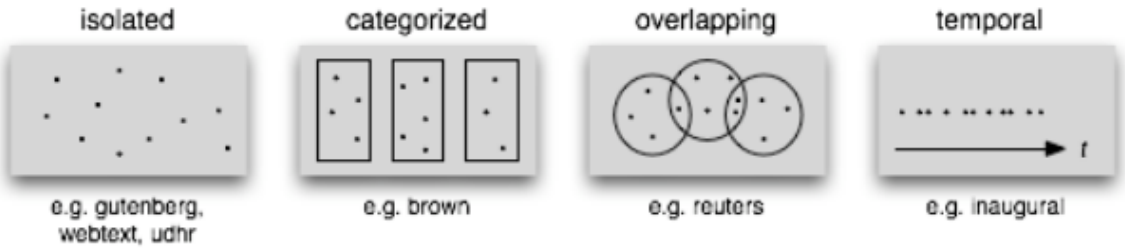
In [44]:
```
cfd = nltk.ConditionalFreqDist(
    (lang, len(word))
    for lang in languages
    for word in udhr.words(lang + '-Latin1'))
```

In [45]:
```
cfd.plot(cumulative=True)
```



1.8 Text Corpus Structure

berikut beberapa struktur dari corpus yang dimiliki NLTK



fungsi dasar dari nltk corpus

Example	Description
`fileids()`	the files of the corpus
`fileids([categories])`	the files of the corpus corresponding to these categories
`categories()`	the categories of the corpus
`categories([fileids])`	the categories of the corpus corresponding to these files
`raw()`	the raw content of the corpus
`raw(fileids=[f1,f2,f3])`	the raw content of the specified files
`raw(categories=[c1,c2])`	the raw content of the specified categories
`words()`	the words of the whole corpus
`words(fileids=[f1,f2,f3])`	the words of the specified fileids
`words(categories=[c1,c2])`	the words of the specified categories
`sents()`	the sentences of the whole corpus
`sents(fileids=[f1,f2,f3])`	the sentences of the specified fileids
`sents(categories=[c1,c2])`	the sentences of the specified categories
`abspath(fileid)`	the location of the given file on disk
`encoding(fileid)`	the encoding of the file (if known)
`open(fileid)`	open a stream for reading the given corpus file
`root`	if the path to the root of locally installed corpus
`readme()`	the contents of the README file of the corpus

1.9 Loading your own Corpus

untuk membaca corpus sendiri kita dapat menggunakan kode dibawah ini

In [46]:
```
from nltk.corpus import PlaintextCorpusReader
corpus_root = './my_corpus'
wordlists = PlaintextCorpusReader(corpus_root, '.*')
```

In [47]:
```
wordlists.fileids()
```

Out[47]:
```
['malin_kundang.txt']
```

2 Conditional Frequency Distributions

2.1 Conditions and Events

In [48]:
```
text = ['The', 'Fulton', 'County', 'Grand', 'Jury', 'said', ...]
pairs = [('news', 'The'), ('news', 'Fulton'), ('news', 'County'), ...]
```

2.2 Counting Words by Genre

In [49]:
```
import nltk
from nltk.corpus import brown
cfd = nltk.ConditionalFreqDist(
            (genre, word)
            for genre in brown.categories()
            for word in brown.words(categories=genre))
```

In [50]:
```
genre_word = [(genre, word)
              for genre in ['news', 'romance']
              for word in brown.words(categories=genre)]
len(genre_word)
```

Out[50]:
```
170576
```

In [51]:
```
genre_word[:4]
```

Out[51]:
```
[('news', 'The'), ('news', 'Fulton'), ('news', 'County'), ('news', 'Grand')]
```

In [52]:
```
genre_word[-4:]
```

Out[52]:
```
[('romance', 'afraid'),
 ('romance', 'not'),
 ('romance', "'"),
 ('romance', '.')]
```

In [53]:
```
cfd = nltk.ConditionalFreqDist(genre_word)
```

In [54]:
```
cfd
```

Out[54]:
```
<ConditionalFreqDist with 2 conditions>
```

In [55]:
```
cfd.conditions()
```

Out[55]:
```
['news', 'romance']
```

In [56]:
```
print(cfd['news'])
```

```
<FreqDist with 14394 samples and 100554 outcomes>
```

In [57]:
```
print(cfd['romance'])
```

```
<FreqDist with 8452 samples and 70022 outcomes>
```

In [58]:
```
cfd['romance'].most_common(20)
```

Out[58]:
```
[(',', 3899),
 ('.', 3736),
 ('the', 2758),
 ('and', 1776),
 ('to', 1502),
 ('a', 1335),
 ('of', 1186),
 ('``', 1045),
 ("''", 1044),
 ('was', 993),
 ('I', 951),
 ('in', 875),
 ('he', 702),
 ('had', 692),
 ('?', 690),
 ('her', 651),
 ('that', 583),
 ('it', 573),
 ('his', 559),
 ('she', 496)]
```

In [59]:
```
cfd['romance']['could']
```

Out[59]:
```
193
```

2.3 Plotting and Tabulating Distributions

In [60]:
```
from nltk.corpus import inaugural
cfd = nltk.ConditionalFreqDist(
          (target, fileid[:4])
          for fileid in inaugural.fileids()
          for w in inaugural.words(fileid)
          for target in ['america', 'citizen']
          if w.lower().startswith(target))
```

In [61]:
```
from nltk.corpus import udhr
languages = ['Chickasaw', 'English', 'German_Deutsch',
    'Greenlandic_Inuktikut', 'Hungarian_Magyar', 'Ibibio_Efik']
cfd = nltk.ConditionalFreqDist(
          (lang, len(word))
          for lang in languages
          for word in udhr.words(lang + '-Latin1'))
```

In [62]:
```
cfd.tabulate(conditions=['English', 'German_Deutsch'],
             samples=range(10), cumulative=True)
```

```
                 0    1    2    3    4    5    6    7    8    9
       English   0  185  525  883  997 1166 1283 1440 1558 1638
German_Deutsch   0  171  263  614  717  894 1013 1110 1213 1275
```

2.4 Generating Random Text with Bigrams

In [63]:
```
sent = ['In', 'the', 'beginning', 'God', 'created', 'the', 'heaven','and', 'the'
```

In [64]:
```
list(nltk.bigrams(sent))
```

Out[64]:
```
[('In', 'the'),
 ('the', 'beginning'),
 ('beginning', 'God'),
 ('God', 'created'),
 ('created', 'the'),
 ('the', 'heaven'),
 ('heaven', 'and'),
 ('and', 'the'),
 ('the', 'earth'),
 ('earth', '.')]
```

In [65]:
```
def generate_model(cfdist, word, num=15):
    for i in range(num):
        print(word, end=' ')
        word = cfdist[word].max()

text = nltk.corpus.genesis.words('english-kjv.txt')
bigrams = nltk.bigrams(text)
cfd = nltk.ConditionalFreqDist(bigrams)
```

In [66]:
```
cfd['living']
```

Out[66]:
```
FreqDist({'creature': 7, 'thing': 4, 'substance': 2, 'soul': 1, '.': 1, ',':
1})
```

In [67]:
```
generate_model(cfd, 'living')
```

```
living creature that he said , and the land of the land of the land
```

3 More Python: Reusing Code

3.1 Creating Programs with a Text Editor

In [68]:
```
print('Selamat menunaikan ibadah puasa')
```

```
Selamat menunaikan ibadah puasa
```

3.2 Functions

function sangat berguna jika kita menggunakan fungsi yang digunakan berulang-ulang, hal ini akan berguna agar kita tidak menulis kode program yang sama secara berulang

In [69]:
```
import math
```

In [70]:
```
def faktorial(x):
    print('faktorial dari ',x,' adalah ', math.factorial(x))
```

In [71]:
```
faktorial(1)
faktorial(2)
faktorial(3)
faktorial(4)
faktorial(5)
```

```
faktorial dari  1  adalah  1
faktorial dari  2  adalah  2
faktorial dari  3  adalah  6
faktorial dari  4  adalah  24
faktorial dari  5  adalah  120
```

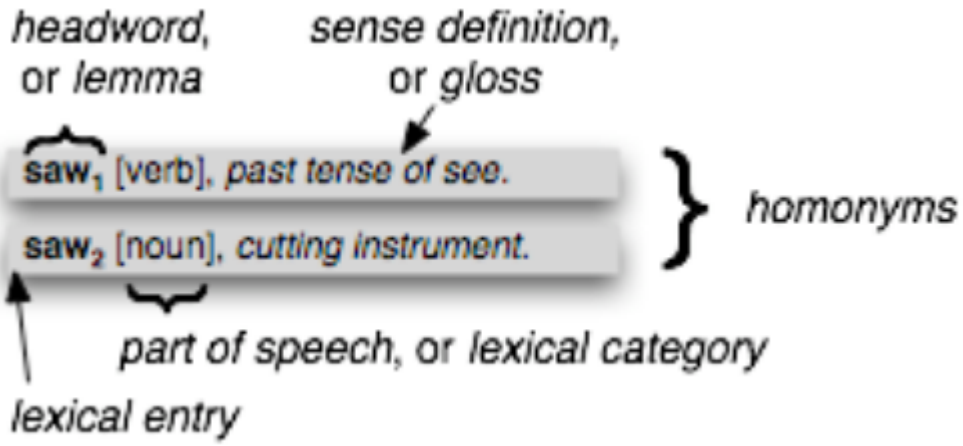
di atas adalah contoh dari fungsi membuat faktorial, kita hanya tinggal memanggil fungsi yang sudah dibuat dan memasukan parameter yang dibutuhkan

3.3 Modules

module adalah mengimport dari file python lain untuk digunakan fungsinya

contoh jika fungsi faktorial diatas d save dalam sebuah file bernama perhitungan maka cara memanggil modulnya adalah
import perhitungan
perhitungan.faktorial(1)

4 Lexical Resources



4.1 Wordlist Corpora

dalam nltk terdapat list dari stopword atau kata yang tidak penting, untuk memanggil wordlist tersebut kita harus mengimport stopwords seperti pada code dibawah ini

In [72]:
```
from nltk.corpus import stopwords
```

berikut adalah daftar stopword bahasa inggris

In [73]:
```
stopwords.words('english')
```

Out[73]:
```
['i',
 'me',
 'my',
 'myself',
 'we',
 'our',
 'ours',
 'ourselves',
 'you',
 "you're",
 "you've",
 "you'll",
 "you'd",
 'your',
 'yours',
 'yourself',
 'yourselves',
 'he',
 'him',
```

Untuk bahasa indonesia bisa menggunakan code seperti dibawah ini

In [74]:
```
print(stopwords.words('indonesian'))
```

```
['ada', 'adalah', 'adanya', 'adapun', 'agak', 'agaknya', 'agar', 'akan', 'akank
ah', 'akhir', 'akhiri', 'akhirnya', 'aku', 'akulah', 'amat', 'amatlah', 'anda',
'andalah', 'antar', 'antara', 'antaranya', 'apa', 'apaan', 'apabila', 'apakah',
'apalagi', 'apatah', 'artinya', 'asal', 'asalkan', 'atas', 'atau', 'ataukah',
'ataupun', 'awal', 'awalnya', 'bagai', 'bagaikan', 'bagaimana', 'bagaimanakah',
'bagaimanapun', 'bagi', 'bagian', 'bahkan', 'bahwa', 'bahwasanya', 'baik', 'bak
al', 'bakalan', 'balik', 'banyak', 'bapak', 'baru', 'bawah', 'beberapa', 'begin
i', 'beginian', 'beginikah', 'beginilah', 'begitu', 'begitukah', 'begitulah',
'begitupun', 'bekerja', 'belakang', 'belakangan', 'belum', 'belumlah', 'benar',
'benarkah', 'benarlah', 'berada', 'berakhir', 'berakhirlah', 'berakhirnya', 'be
rapa', 'berapakah', 'berapalah', 'berapapun', 'berarti', 'berawal', 'berbagai',
'berdatangan', 'beri', 'berikan', 'berikut', 'berikutnya', 'berjumlah', 'berkal
i-kali', 'berkata', 'berkehendak', 'berkeinginan', 'berkenaan', 'berlainan', 'b
erlalu', 'berlangsung', 'berlebihan', 'bermacam', 'bermacam-macam', 'bermaksu
d', 'bermula', 'bersama', 'bersama-sama', 'bersiap', 'bersiap-siap', 'bertany
a', 'bertanya-tanya', 'berturut', 'berturut-turut', 'bertutur', 'berujar', 'ber
upa', 'besar', 'betul', 'betulkah', 'biasa', 'biasanya', 'bila', 'bilakah', 'bi
sa', 'bisakah', 'boleh', 'bolehkah', 'bolehlah', 'buat', 'bukan', 'bukankah',
'bukanlah', 'bukannya', 'bulan', 'bung', 'cara', 'caranya', 'cukup', 'cukupka
h', 'cukuplah', 'cuma', 'dahulu', 'dalam', 'dan', 'dapat', 'dari', 'daripada',
'datang', 'dekat', 'demi', 'demikian', 'demikianlah', 'dengan', 'depan', 'di',
'dia', 'diakhiri', 'diakhirinya', 'dialah', 'diantara', 'diantaranya', 'diber
i', 'diberikan', 'diberikannya', 'dibuat', 'dibuatnya', 'didapat', 'didatangka
n', 'digunakan', 'diibaratkan', 'diibaratkannya', 'diingat', 'diingatkan', 'dii
nginkan', 'dijawab', 'dijelaskan', 'dijelaskannya', 'dikarenakan', 'dikatakan',
'dikatakannya', 'dikerjakan', 'diketahui', 'diketahuinya', 'dikira', 'dilakuka
n', 'dilalui', 'dilihat', 'dimaksud', 'dimaksudkan', 'dimaksudkannya', 'dimaksu
dnya', 'diminta', 'dimintai', 'dimisalkan', 'dimulai', 'dimulailah', 'dimulainy
a', 'dimungkinkan', 'dini', 'dipastikan', 'diperbuat', 'diperbuatnya', 'dipergu
nakan', 'diperkirakan', 'diperlihatkan', 'diperlukan', 'diperlukannya', 'dipers
oalkan', 'dipertanyakan', 'dipunyai', 'diri', 'dirinya', 'disampaikan', 'disebu
t', 'disebutkan', 'disebutkannya', 'disini', 'disinilah', 'ditambahkan', 'ditan
daskan', 'ditanya', 'ditanyai', 'ditanyakan', 'ditegaskan', 'ditujukan', 'ditun
juk', 'ditunjuki', 'ditunjukkan', 'ditunjukkannya', 'ditunjuknya', 'dituturka
n', 'dituturkannya', 'diucapkan', 'diucapkannya', 'diungkapkan', 'dong', 'dua',
'dulu', 'empat', 'enggak', 'enggaknya', 'entah', 'entahlah', 'guna', 'gunakan',
'hal', 'hampir', 'hanya', 'hanyalah', 'hari', 'harus', 'haruslah', 'harusnya',
'hendak', 'hendaklah', 'hendaknya', 'hingga', 'ia', 'ialah', 'ibarat', 'ibaratk
an', 'ibaratnya', 'ibu', 'ikut', 'ingat', 'ingat-ingat', 'ingin', 'inginkah',
'inginkan', 'ini', 'inikah', 'inilah', 'itu', 'itukah', 'itulah', 'jadi', 'jadi
lah', 'jadinya', 'jangan', 'jangankan', 'janganlah', 'jauh', 'jawab', 'jawaba
n', 'jawabnya', 'jelas', 'jelaskan', 'jelaslah', 'jelasnya', 'jika', 'jikalau',
'juga', 'jumlah', 'jumlahnya', 'justru', 'kala', 'kalau', 'kalaulah', 'kalaupu
n', 'kalian', 'kami', 'kamilah', 'kamu', 'kamulah', 'kan', 'kapan', 'kapankah',
'kapanpun', 'karena', 'karenanya', 'kasus', 'kata', 'katakan', 'katakanlah', 'k
atanya', 'ke', 'keadaan', 'kebetulan', 'kecil', 'kedua', 'keduanya', 'keingina
n', 'kelamaan', 'kelihatan', 'kelihatannya', 'kelima', 'keluar', 'kembali', 'ke
mudian', 'kemungkinan', 'kemungkinannya', 'kenapa', 'kepada', 'kepadanya', 'kes
ampaian', 'keseluruhan', 'keseluruhannya', 'keterlaluan', 'ketika', 'khususny
a', 'kini', 'kinilah', 'kira', 'kira-kira', 'kiranya', 'kita', 'kitalah', 'ko
k', 'kurang', 'lagi', 'lagian', 'lah', 'lain', 'lainnya', 'lalu', 'lama', 'lama
nya', 'lanjut', 'lanjutnya', 'lebih', 'lewat', 'lima', 'luar', 'macam', 'maka',
'makanya', 'makin', 'malah', 'malahan', 'mampu', 'mampukah', 'mana', 'manakal
a', 'manalagi', 'masa', 'masalah', 'masalahnya', 'masih', 'masihkah', 'masing',
'masing-masing', 'mau', 'maupun', 'melainkan', 'melakukan', 'melalui', 'meliha
```

```
t', 'melihatnya', 'memang', 'memastikan', 'memberi', 'memberikan', 'membuat',
'memerlukan', 'memihak', 'meminta', 'memintakan', 'memisalkan', 'memperbuat',
'mempergunakan', 'memperkirakan', 'memperlihatkan', 'mempersiapkan', 'mempersoa
lkan', 'mempertanyakan', 'mempunyai', 'memulai', 'memungkinkan', 'menaiki', 'me
nambahkan', 'menandaskan', 'menanti', 'menanti-nanti', 'menantikan', 'menanya',
'menanyai', 'menanyakan', 'mendapat', 'mendapatkan', 'mendatang', 'mendatangi',
'mendatangkan', 'menegaskan', 'mengakhiri', 'mengapa', 'mengatakan', 'mengataka
nnya', 'mengenai', 'mengerjakan', 'mengetahui', 'menggunakan', 'menghendaki',
'mengibaratkan', 'mengibaratkannya', 'mengingat', 'mengingatkan', 'menginginka
n', 'mengira', 'mengucapkan', 'mengucapkannya', 'mengungkapkan', 'menjadi', 'me
njawab', 'menjelaskan', 'menuju', 'menunjuk', 'menunjuki', 'menunjukkan', 'menu
njuknya', 'menurut', 'menuturkan', 'menyampaikan', 'menyangkut', 'menyatakan',
'menyebutkan', 'menyeluruh', 'menyiapkan', 'merasa', 'mereka', 'merekalah', 'me
rupakan', 'meski', 'meskipun', 'meyakini', 'meyakinkan', 'minta', 'mirip', 'mis
al', 'misalkan', 'misalnya', 'mula', 'mulai', 'mulailah', 'mulanya', 'mungkin',
'mungkinkah', 'nah', 'naik', 'namun', 'nanti', 'nantinya', 'nyaris', 'nyatany
a', 'oleh', 'olehnya', 'pada', 'padahal', 'padanya', 'pak', 'paling', 'panjan
g', 'pantas', 'para', 'pasti', 'pastilah', 'penting', 'pentingnya', 'per', 'per
cuma', 'perlu', 'perlukah', 'perlunya', 'pernah', 'persoalan', 'pertama', 'pert
ama-tama', 'pertanyaan', 'pertanyakan', 'pihak', 'pihaknya', 'pukul', 'pula',
'pun', 'punya', 'rasa', 'rasanya', 'rata', 'rupanya', 'saat', 'saatnya', 'saj
a', 'sajalah', 'saling', 'sama', 'sama-sama', 'sambil', 'sampai', 'sampai-sampa
i', 'sampaikan', 'sana', 'sangat', 'sangatlah', 'satu', 'saya', 'sayalah', 's
e', 'sebab', 'sebabnya', 'sebagai', 'sebagaimana', 'sebagainya', 'sebagian', 's
ebaik', 'sebaik-baiknya', 'sebaiknya', 'sebaliknya', 'sebanyak', 'sebegini', 's
ebegitu', 'sebelum', 'sebelumnya', 'sebenarnya', 'seberapa', 'sebesar', 'sebetu
lnya', 'sebisanya', 'sebuah', 'sebut', 'sebutlah', 'sebutnya', 'secara', 'secuk
upnya', 'sedang', 'sedangkan', 'sedemikian', 'sedikit', 'sedikitnya', 'seenakny
a', 'segala', 'segalanya', 'segera', 'seharusnya', 'sehingga', 'seingat', 'seja
k', 'sejauh', 'sejenak', 'sejumlah', 'sekadar', 'sekadarnya', 'sekali', 'sekali
-kali', 'sekalian', 'sekaligus', 'sekalipun', 'sekarang', 'sekarang', 'sekeci
l', 'seketika', 'sekiranya', 'sekitar', 'sekitarnya', 'sekurang-kurangnya', 'se
kurangnya', 'sela', 'selain', 'selaku', 'selalu', 'selama', 'selama-lamanya',
'selamanya', 'selanjutnya', 'seluruh', 'seluruhnya', 'semacam', 'semakin', 'sem
ampu', 'semampunya', 'semasa', 'semasih', 'semata', 'semata-mata', 'semaunya',
'sementara', 'semisal', 'semisalnya', 'sempat', 'semua', 'semuanya', 'semula',
'sendiri', 'sendirian', 'sendirinya', 'seolah', 'seolah-olah', 'seorang', 'sepa
njang', 'sepantasnya', 'sepantasnyalah', 'seperlunya', 'seperti', 'sepertinya',
'sepihak', 'sering', 'seringnya', 'serta', 'serupa', 'sesaat', 'sesama', 'sesam
pai', 'sesegera', 'sesekali', 'seseorang', 'sesuatu', 'sesuatunya', 'sesudah',
'sesudahnya', 'setelah', 'setempat', 'setengah', 'seterusnya', 'setiap', 'setib
a', 'setibanya', 'setidak-tidaknya', 'setidaknya', 'setinggi', 'seusai', 'sewak
tu', 'siap', 'siapa', 'siapakah', 'siapapun', 'sini', 'sinilah', 'soal', 'soaln
ya', 'suatu', 'sudah', 'sudahkah', 'sudahlah', 'supaya', 'tadi', 'tadinya', 'ta
hu', 'tahun', 'tak', 'tambah', 'tambahnya', 'tampak', 'tampaknya', 'tandas', 't
andasnya', 'tanpa', 'tanya', 'tanyakan', 'tanyanya', 'tapi', 'tegas', 'tegasny
a', 'telah', 'tempat', 'tengah', 'tentang', 'tentu', 'tentulah', 'tentunya', 't
epat', 'terakhir', 'terasa', 'terbanyak', 'terdahulu', 'terdapat', 'terdiri',
'terhadap', 'terhadapnya', 'teringat', 'teringat-ingat', 'terjadi', 'terjadila
h', 'terjadinya', 'terkira', 'terlalu', 'terlebih', 'terlihat', 'termasuk', 'te
rnyata', 'tersampaikan', 'tersebut', 'tersebutlah', 'tertentu', 'tertuju', 'ter
us', 'terutama', 'tetap', 'tetapi', 'tiap', 'tiba', 'tiba-tiba', 'tidak', 'tida
kkah', 'tidaklah', 'tiga', 'tinggi', 'toh', 'tunjuk', 'turut', 'tutur', 'tuturn
ya', 'ucap', 'ucapnya', 'ujar', 'ujarnya', 'umum', 'umumnya', 'ungkap', 'ungkap
nya', 'untuk', 'usah', 'usai', 'waduh', 'wah', 'wahai', 'waktu', 'waktunya', 'w
alau', 'walaupun', 'wong', 'yaitu', 'yakin', 'yakni', 'yang']
```

daftar nama dan gender yang disediakan oleh nltk

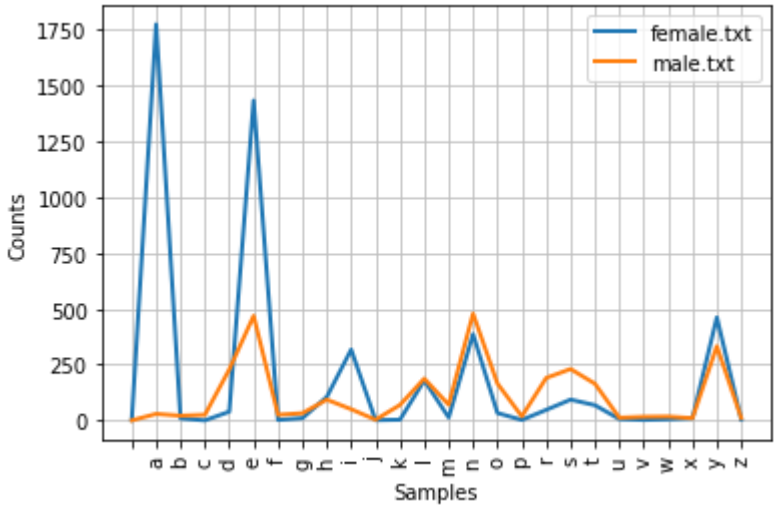
In [75]:
```
names = nltk.corpus.names
```

In [76]:
```
male_names = names.words('male.txt')
female_names = names.words('female.txt')
print([w for w in male_names if w in female_names])
```

```
['Abbey', 'Abbie', 'Abby', 'Addie', 'Adrian', 'Adrien', 'Ajay', 'Alex', 'Alexi
s', 'Alfie', 'Ali', 'Alix', 'Allie', 'Allyn', 'Andie', 'Andrea', 'Andy', 'Ange
l', 'Angie', 'Ariel', 'Ashley', 'Aubrey', 'Augustine', 'Austin', 'Averil', 'Bar
rie', 'Barry', 'Beau', 'Bennie', 'Benny', 'Bernie', 'Bert', 'Bertie', 'Bill',
'Billie', 'Billy', 'Blair', 'Blake', 'Bo', 'Bobbie', 'Bobby', 'Brandy', 'Bret
t', 'Britt', 'Brook', 'Brooke', 'Brooks', 'Bryn', 'Cal', 'Cam', 'Cammy', 'Care
y', 'Carlie', 'Carlin', 'Carmine', 'Carroll', 'Cary', 'Caryl', 'Casey', 'Cass',
'Cat', 'Cecil', 'Chad', 'Chris', 'Chrissy', 'Christian', 'Christie', 'Christy',
'Clair', 'Claire', 'Clare', 'Claude', 'Clem', 'Clemmie', 'Cody', 'Connie', 'Con
stantine', 'Corey', 'Corrie', 'Cory', 'Courtney', 'Cris', 'Daffy', 'Dale', 'Dal
las', 'Dana', 'Dani', 'Daniel', 'Dannie', 'Danny', 'Darby', 'Darcy', 'Darryl',
'Daryl', 'Deane', 'Del', 'Dell', 'Demetris', 'Dennie', 'Denny', 'Devin', 'Devo
n', 'Dion', 'Dionis', 'Dominique', 'Donnie', 'Donny', 'Dorian', 'Dory', 'Drew',
'Eddie', 'Eddy', 'Edie', 'Elisha', 'Emmy', 'Erin', 'Esme', 'Evelyn', 'Felice',
'Fran', 'Francis', 'Frank', 'Frankie', 'Franky', 'Fred', 'Freddie', 'Freddy',
'Gabriel', 'Gabriell', 'Gail', 'Gale', 'Gay', 'Gayle', 'Gene', 'George', 'Georg
ia', 'Georgie', 'Geri', 'Germaine', 'Gerri', 'Gerry', 'Gill', 'Ginger', 'Glen',
'Glenn', 'Grace', 'Gretchen', 'Gus', 'Haleigh', 'Haley', 'Hannibal', 'Harley',
'Hazel', 'Heath', 'Henrie', 'Hilary', 'Hillary', 'Holly', 'Ike', 'Ikey', 'Ira',
'Isa', 'Isador', 'Isadore', 'Jackie', 'Jaime', 'Jamie', 'Jan', 'Jean', 'Jere',
'Jermaine', 'Jerrie', 'Jerry', 'Jess', 'Jesse', 'Jessie', 'Jo', 'Jodi', 'Jodi
e', 'Jody', 'Joey', 'Jordan', 'Juanita', 'Jude', 'Judith', 'Judy', 'Julie', 'Ju
stin', 'Karel', 'Kellen', 'Kelley', 'Kelly', 'Kelsey', 'Kerry', 'Kim', 'Kip',
'Kirby', 'Kit', 'Kris', 'Kyle', 'Lane', 'Lanny', 'Lauren', 'Laurie', 'Lee', 'Le
igh', 'Leland', 'Lesley', 'Leslie', 'Lin', 'Lind', 'Lindsay', 'Lindsey', 'Lind
y', 'Lonnie', 'Loren', 'Lorne', 'Lorrie', 'Lou', 'Luce', 'Lyn', 'Lynn', 'Maddi
e', 'Maddy', 'Marietta', 'Marion', 'Marlo', 'Martie', 'Marty', 'Mattie', 'Matt
y', 'Maurise', 'Max', 'Maxie', 'Mead', 'Meade', 'Mel', 'Meredith', 'Merle', 'Me
rrill', 'Merry', 'Meryl', 'Michal', 'Michel', 'Michele', 'Mickie', 'Micky', 'Mi
llicent', 'Morgan', 'Morlee', 'Muffin', 'Nat', 'Nichole', 'Nickie', 'Nicky', 'N
iki', 'Nikki', 'Noel', 'Ollie', 'Page', 'Paige', 'Pat', 'Patrice', 'Patsy', 'Pa
ttie', 'Patty', 'Pen', 'Pennie', 'Penny', 'Perry', 'Phil', 'Pooh', 'Quentin',
'Quinn', 'Randi', 'Randie', 'Randy', 'Ray', 'Regan', 'Reggie', 'Rene', 'Rey',
'Ricki', 'Rickie', 'Ricky', 'Rikki', 'Robbie', 'Robin', 'Ronnie', 'Ronny', 'Ror
y', 'Ruby', 'Sal', 'Sam', 'Sammy', 'Sandy', 'Sascha', 'Sasha', 'Saundra', 'Sayr
e', 'Scotty', 'Sean', 'Shaine', 'Shane', 'Shannon', 'Shaun', 'Shawn', 'Shay',
'Shayne', 'Shea', 'Shelby', 'Shell', 'Shelley', 'Sibyl', 'Simone', 'Sonnie', 'S
onny', 'Stacy', 'Sunny', 'Sydney', 'Tabbie', 'Tabby', 'Tallie', 'Tally', 'Tammi
e', 'Tammy', 'Tate', 'Ted', 'Teddie', 'Teddy', 'Terri', 'Terry', 'Theo', 'Tim',
'Timmie', 'Timmy', 'Tobe', 'Tobie', 'Toby', 'Tommie', 'Tommy', 'Tony', 'Torey',
'Trace', 'Tracey', 'Tracie', 'Tracy', 'Val', 'Vale', 'Valentine', 'Van', 'Vin',
'Vinnie', 'Vinny', 'Virgie', 'Wallie', 'Wallis', 'Wally', 'Whitney', 'Willi',
'Willie', 'Willy', 'Winnie', 'Winny', 'Wynn']
```

code dibawah ini untuk melihat plot frequensi data dari nama laki-laki atau perempuan

In [77]:
```
cfd = nltk.ConditionalFreqDist(
    (fileid, name[-1])
    for fileid in names.fileids()
    for name in names.words(fileid))
cfd.plot()
```



4.2 A Pronouncing Dictionary

In [78]:
```
entries = nltk.corpus.cmudict.entries()
```

untuk melihat panjang dari entries bisa menggunakan kode dibawah ini

In [79]:
```
len(entries)
```

Out[79]:
```
133737
```

untuk melihat 15 data entries bisa menggunakan kode dibawah ini
tetapi kita juga bisa melihat di range tertentu dengan cara entries**[index_awal:index_akhir]**

In [80]:
```
for entry in entries[:15]:
    print(entry)
```

```
('a', ['AH0'])
('a.', ['EY1'])
('a', ['EY1'])
('a42128', ['EY1', 'F', 'AO1', 'R', 'T', 'UW1', 'W', 'AH1', 'N', 'T', 'UW1', 'E
Y1', 'T'])
('aaa', ['T', 'R', 'IH2', 'P', 'AH0', 'L', 'EY1'])
('aaberg', ['AA1', 'B', 'ER0', 'G'])
('aachen', ['AA1', 'K', 'AH0', 'N'])
('aachener', ['AA1', 'K', 'AH0', 'N', 'ER0'])
('aaker', ['AA1', 'K', 'ER0'])
('aalseth', ['AA1', 'L', 'S', 'EH0', 'TH'])
('aamodt', ['AA1', 'M', 'AH0', 'T'])
('aancor', ['AA1', 'N', 'K', 'AO2', 'R'])
('aardema', ['AA0', 'R', 'D', 'EH1', 'M', 'AH0'])
('aardvark', ['AA1', 'R', 'D', 'V', 'AA2', 'R', 'K'])
('aaron', ['EH1', 'R', 'AH0', 'N'])
```

untuk melihat jumlah pron tertentu kita dapat menggunakan kode dibawah ini, dan juka kita dapat mengecek untuk yang berawalan huruf R dan berakhiran huruf S yang memiliki 4 pronon

In [81]:
```
for word, pron in entries:
    if len(pron) == 4:
        ph1, ph2, ph3, ph4 = pron
        if ph1 == 'R' and ph4 == 'S':
            print(word, ph2, end=' ')
```

```
hritz IH1 raatz AA1 racks AE1 raetz IY1 raikes EY1 rakes EY1 rance AE1 rapes EY
1 raps AE1 rate's EY1 rates EY1 rat's AE1 rats AE1 ratts AE1 ratz AE1 rax AE1 r
eaps IY1 reefs IY1 reeks IY1 reetz IY1 reich's AY1 reicks AY1 reits AY1 reitz A
Y1 rep's EH1 reps EH1 retz EH1 rewis UW1 rex EH1 reyes EY1 rick's IH1 ricks IH1
rietz IY1 riffs IH1 right's AY1 rights AY1 rights' AY1 rinse IH1 rios IY1 rips
IH1 rite's AY1 rites AY1 ritt's IH1 ritts IH1 ritz IH1 rix IH1 rock's AA1 rocks
AA1 roofs UW1 rooks UH1 roots UW1 ropes OW1 rops AA1 roth's AA1 roths AO1 rots
AA1 rotz AA1 rough's AH1 routes UW1 routes AW1 routes UH1 rox AA1 rucks AH1 rue
tz UW1 rusts AH1 ruth's UW1 ruts AH1 rutts AH1 rutz AH1 rux AH1 wraps AE1 wreak
s IY1 wreaths IY1 wrecks EH1 wright's AY1 wrights AY1 writes AY1 writs IH1
```

untuk menampilkan syllable tertentu bisa menggunakan kode dibawah ini

In [82]:
```
syllable = ['N', 'IH0', 'K', 'S']
[word for word, pron in entries if pron[-4:] == syllable]
```

Out[82]:
```
["atlantic's",
 'audiotronics',
 'avionics',
 'beatniks',
 'calisthenics',
 'centronics',
 'chamonix',
 'chetniks',
 "clinic's",
 'clinics',
 'conics',
 'conics',
 'cryogenics',
 'cynics',
 'diasonics',
 "dominic's",
 'ebonics',
 'electronics',
 "electronics'",
 "endotronics'",
 'endotronics',
 'enix',
 'environics',
 'ethnics',
 'eugenics',
 'fibronics',
 'flextronics',
 'harmonics',
 'hispanics',
 'histrionics',
 'identics',
 'ionics',
 'kibbutzniks',
 'lasersonics',
 'lumonics',
 'mannix',
 'mechanics',
 "mechanics'",
 'microelectronics',
 'minix',
 'minnix',
 'mnemonics',
 'mnemonics',
 'molonicks',
 'mullenix',
 'mullenix',
 'mullinix',
 'mulnix',
 "munich's",
 'nucleonics',
 'onyx',
 'organics',
 "panic's",
 'panics',
```

```
 'penix',
 'pennix',
 'personics',
 'phenix',
 "philharmonic's",
 'phoenix',
 'phonics',
 'photronics',
 'pinnix',
 'plantronics',
 'pyrotechnics',
 'refuseniks',
 "resnick's",
 'respironics',
 'sconnix',
 'siliconix',
 'skolniks',
 'sonics',
 'sputniks',
 'technics',
 'tectonics',
 'tektronix',
 'telectronics',
 'telephonics',
 'tonics',
 'unix',
 "vinick's",
 "vinnick's",
 'vitronics']
```

untuk menampilkan pronon tertentu dan berakhiran huruf tertentu bisa menggunakan kode dibawah ini, seperti mengecek yang pronon S dan berakhiran huruf t

In [83]:
```
[w for w, pron in entries if pron[-1] == 'S' and w[-1] == 't']
```

Out[83]:
```
['cataract', 'corexit', 'last', 'leubert', 'most', 'next']
```

untuk mengurutkan sebuah data kita bisa menggunakan **sorted** seperti pada kode dibawah ini

In [84]:
```
sorted(set(w[:2] for w, pron in entries if pron[0] == 'N' and w[0] != 'n'))
```

Out[84]:
```
['gn', 'kn', 'mn', 'pn']
```

In [85]:
```
def stress(pron):
    return [char for phone in pron for char in phone if char.isdigit()]
```

In [86]:
```
data = [w for w, pron in entries if stress(pron) == ['0', '1', '0', '2', '0']]
print(data[:10])
```

```
['abbreviated', 'abbreviated', 'abbreviating', 'accelerated', 'accelerating',
'accelerator', 'accelerators', 'accentuated', 'accentuating', 'accommodated']
```

untuk mengecek pronon tertentu dan kata yang menggunakan pronon tersebut bisa menggunakan

kode dibawah ini

In [87]:
```
p3 = [(pron[0]+'-'+pron[2], word)
      for (word, pron) in entries
      if pron[0] == 'P' and len(pron) == 3]
```

In [88]:
```
cfd = nltk.ConditionalFreqDist(p3)
```

In [89]:
```
for template in sorted(cfd.conditions()):
    if len(cfd[template]) > 10:
        words = sorted(cfd[template])
        wordstring = ' '.join(words)
        print(template, wordstring[:70] + "...")
```

```
P-CH patch pautsch peach perch petsch petsche piche piech pietsch pitch pit...
P-K pac pack paek paik pak pake paque peak peake pech peck peek perc perk ...
P-L pahl pail paille pal pale pall paul paule paull peal peale pearl pearl...
P-N paign pain paine pan pane pawn payne peine pen penh penn pin pine pinn...
P-P paap paape pap pape papp paup peep pep pip pipe pipp poop pop pope pop...
P-R paar pair par pare parr pear peer pier poor poore por pore porr pour...
P-S pace pass pasts peace pearse pease perce pers perse pesce piece piss p...
P-T pait pat pate patt peart peat peet peete pert pet pete pett piet piett...
P-UW1 peru peugh pew plew plue prew pru prue prugh pshew pugh...
P-Z p's p.'s p.s pais paiz pao's pas pause paws pays paz peas pease pei's ...
```

4.3 Comparative Wordlists

untuk melihat komparatif wordlist kita bisa menggunakan kode dibawah ini, dimana ini disediakan oleh NLTK

In [90]:
```
from nltk.corpus import swadesh
```

terdapat beberapa bahasa yang disediakan seperti english spanyol dll, bisa di lihat dari pilihan yang tersedia

In [91]:
```
print(swadesh.fileids())
```

```
['be', 'bg', 'bs', 'ca', 'cs', 'cu', 'de', 'en', 'es', 'fr', 'hr', 'it', 'la',
'mk', 'nl', 'pl', 'pt', 'ro', 'ru', 'sk', 'sl', 'sr', 'sw', 'uk']
```

berikut beberapa contoh dari kata yg tersedia di bahasa spanyol dan inggris

In [92]:
```
print("Spanyol ",swadesh.words('es'))
```

```
Spanyol  ['yo', 'tú, usted', 'él', 'nosotros', 'vosotros, ustedes', 'ellos, ell
as', 'este', 'ese, aquel', 'aquí, acá', 'ahí, allí, allá', 'quien', 'que', 'don
de', 'cuando', 'como', 'no', 'todo', 'muchos', 'algunos, unos', 'poco', 'otro',
'uno', 'dos', 'tres', 'cuatro', 'cinco', 'grande', 'largo', 'ancho', 'gordo',
'pesado', 'pequeño', 'corto', 'estrecho, angosto', 'delgado, flaco', 'mujer',
'hombre', 'hombre', 'niño', 'esposa, mujer', 'esposo, marido', 'madre', 'padr
e', 'animal', 'pez, pescado', 'ave, pájaro', 'perro', 'piojo', 'serpiente, cule
bra', 'gusano', 'árbol', 'bosque', 'palo', 'fruta', 'semilla', 'hoja', 'raíz',
'corteza', 'flor', 'hierba, pasto', 'cuerda', 'piel', 'carne', 'sangre', 'hues
o', 'grasa', 'huevo', 'cuerno', 'cola', 'pluma', 'cabello, pelo', 'cabeza', 'or
eja', 'ojo', 'nariz', 'boca', 'diente', 'lengua', 'uña', 'pie', 'pierna', 'rodi
lla', 'mano', 'ala', 'barriga, vientre, panza', 'entrañas, tripas', 'cuello',
'espalda', 'pecho, seno', 'corazón', 'hígado', 'beber, tomar', 'comer', 'morde
r', 'chupar', 'escupir', 'vomitar', 'soplar', 'respirar', 'reír', 'ver', 'oír',
'saber', 'pensar', 'oler', 'temer', 'dormir', 'vivir', 'morir', 'matar', 'pelea
r', 'cazar', 'golpear', 'cortar', 'partir', 'apuñalar', 'arañar, rascar', 'cava
r', 'nadar', 'volar', 'caminar', 'venir', 'echarse, acostarse, tenderse', 'sent
arse', 'estar de pie', 'voltear', 'caer', 'dar', 'sostener', 'apretar', 'frota
r', 'lavar', 'limpiar', 'tirar', 'empujar', 'tirar', 'atar', 'coser', 'contar',
'decir', 'cantar', 'jugar', 'flotar', 'fluir', 'helar', 'hincharse', 'sol', 'lu
na', 'estrella', 'agua', 'lluvia', 'río', 'lago', 'mar', 'sal', 'piedra', 'aren
a', 'polvo', 'tierra', 'nube', 'niebla', 'cielo', 'viento', 'nieve', 'hielo',
'humo', 'fuego', 'cenizas', 'quemar', 'camino', 'montaña', 'rojo', 'verde', 'am
arillo', 'blanco', 'negro', 'noche', 'día', 'año', 'cálido, tibio', 'frío', 'll
eno', 'nuevo', 'viejo', 'bueno', 'malo', 'podrido', 'sucio', 'recto', 'redond
o', 'afilado', 'desafilado', 'suave, liso', 'mojado', 'seco', 'correcto', 'cerc
a', 'lejos', 'derecha', 'izquierda', 'a, en, ante', 'en', 'con', 'y', 'si', 'po
rque', 'nombre']
```

In [93]:
```
print("English ",swadesh.words('en'))
```

```
English  ['I', 'you (singular), thou', 'he', 'we', 'you (plural)', 'they', 'thi
s', 'that', 'here', 'there', 'who', 'what', 'where', 'when', 'how', 'not', 'al
l', 'many', 'some', 'few', 'other', 'one', 'two', 'three', 'four', 'five', 'bi
g', 'long', 'wide', 'thick', 'heavy', 'small', 'short', 'narrow', 'thin', 'woma
n', 'man (adult male)', 'man (human being)', 'child', 'wife', 'husband', 'mothe
r', 'father', 'animal', 'fish', 'bird', 'dog', 'louse', 'snake', 'worm', 'tre
e', 'forest', 'stick', 'fruit', 'seed', 'leaf', 'root', 'bark (from tree)', 'fl
ower', 'grass', 'rope', 'skin', 'meat', 'blood', 'bone', 'fat (noun)', 'egg',
'horn', 'tail', 'feather', 'hair', 'head', 'ear', 'eye', 'nose', 'mouth', 'toot
h', 'tongue', 'fingernail', 'foot', 'leg', 'knee', 'hand', 'wing', 'belly', 'gu
ts', 'neck', 'back', 'breast', 'heart', 'liver', 'drink', 'eat', 'bite', 'suc
k', 'spit', 'vomit', 'blow', 'breathe', 'laugh', 'see', 'hear', 'know (a fac
t)', 'think', 'smell', 'fear', 'sleep', 'live', 'die', 'kill', 'fight', 'hunt',
'hit', 'cut', 'split', 'stab', 'scratch', 'dig', 'swim', 'fly (verb)', 'walk',
'come', 'lie', 'sit', 'stand', 'turn', 'fall', 'give', 'hold', 'squeeze', 'ru
b', 'wash', 'wipe', 'pull', 'push', 'throw', 'tie', 'sew', 'count', 'say', 'sin
g', 'play', 'float', 'flow', 'freeze', 'swell', 'sun', 'moon', 'star', 'water',
'rain', 'river', 'lake', 'sea', 'salt', 'stone', 'sand', 'dust', 'earth', 'clou
d', 'fog', 'sky', 'wind', 'snow', 'ice', 'smoke', 'fire', 'ashes', 'burn', 'roa
d', 'mountain', 'red', 'green', 'yellow', 'white', 'black', 'night', 'day', 'ye
ar', 'warm', 'cold', 'full', 'new', 'old', 'good', 'bad', 'rotten', 'dirty', 's
traight', 'round', 'sharp', 'dull', 'smooth', 'wet', 'dry', 'correct', 'near',
'far', 'right', 'left', 'at', 'in', 'with', 'and', 'if', 'because', 'name']
```

perbandingan bahasa inggris dan espanyol

In [94]:
```
fr2en = swadesh.entries(['en', 'es'])
fr2en[:30]
```

Out[94]:
```
[('I', 'yo'),
 ('you (singular), thou', 'tú, usted'),
 ('he', 'él'),
 ('we', 'nosotros'),
 ('you (plural)', 'vosotros, ustedes'),
 ('they', 'ellos, ellas'),
 ('this', 'este'),
 ('that', 'ese, aquel'),
 ('here', 'aquí, acá'),
 ('there', 'ahí, allí, allá'),
 ('who', 'quien'),
 ('what', 'que'),
 ('where', 'donde'),
 ('when', 'cuando'),
 ('how', 'como'),
 ('not', 'no'),
 ('all', 'todo'),
 ('many', 'muchos'),
 ('some', 'algunos, unos'),
 ('few', 'poco'),
 ('other', 'otro'),
 ('one', 'uno'),
 ('two', 'dos'),
 ('three', 'tres'),
 ('four', 'cuatro'),
 ('five', 'cinco'),
 ('big', 'grande'),
 ('long', 'largo'),
 ('wide', 'ancho'),
 ('thick', 'gordo')]
```

dengan memanfaatkan list tersebut kita bisa menggunakan nya sebagai sebuah kamus dengan cara penggunaan seperti kode dibawah ini

kita juga bisa menggunakan translate antar bahasa dengan menggunakan kode bahasa yang tersedia di atas dan dengan kode seperti diatas sesuai dengan yang kita butuhkan

In [95]:
```
translate = dict(fr2en)
translate['how']
```

Out[95]:
```
'como'
```

Contoh kita akan membuat translate inggris ke german maka bisa menggunakan kode ini

In [96]:
```
en2de = swadesh.entries(['en', 'de'])
```

In [97]:
```
translate.update(dict(en2de))
```

In [98]:
```
translate['what']
```

Out[98]:
```
'was'
```

4.4 Shoebox and Toolbox Lexicons

In [99]:
```
from nltk.corpus import toolbox
```

In [100]:
```
toolbox.entries('rotokas.dic')[:2]
```

Out[100]:
```
[('kaa',
  [('ps', 'V'),
   ('pt', 'A'),
   ('ge', 'gag'),
   ('tkp', 'nek i pas'),
   ('dcsv', 'true'),
   ('vx', '1'),
   ('sc', '???'),
   ('dt', '29/Oct/2005'),
   ('ex', 'Apoka ira kaaroi aioa-ia reoreopaoro.'),
   ('xp', 'Kaikai i pas long nek bilong Apoka bikos em i kaikai na toktok.'),
   ('xe', 'Apoka is gagging from food while talking.')]),
 ('kaa',
  [('ps', 'V'),
   ('pt', 'B'),
   ('ge', 'strangle'),
   ('tkp', 'pasim nek'),
   ('arg', 'O'),
   ('vx', '2'),
   ('dt', '07/Oct/2006'),
   ('ex', 'Rera rauroro rera kaarevoi.'),
   ('xp', 'Em i holim pas em na nekim em.'),
   ('xe', 'He is holding him and strangling him.'),
   ('ex', 'Iroiro-ia oirato okoearo kaaivoi uvare rirovira kaureoparoveira.'),
   ('xp', 'Ol i pasim nek bilong man long rop bikos em i save bikhet tumas.'),
   ('xe',
    "They strangled the man's neck with rope because he was very stubborn and a
rrogant."),
   ('ex',
    'Oirato okoearo kaaivoi iroiro-ia. Uva viapau uvuiparoi ra vovouparo uva ko
piiroi.'),
   ('xp',
    'Ol i pasim nek bilong man long rop. Olsem na em i no pulim win olsem na em
i dai.'),
   ('xe',
    "They strangled the man's neck with a rope. And he couldn't breathe and he
died.")])]
```

5 WordNet

5.1 Senses and Synonyms

nltk menyediakan sebuah data synonyms untuk mengetahui sebuah makna yang mirip, untuk menggunakannya kita bisa menggunakan kode seperti dibawah ini

In [101]:
```
from nltk.corpus import wordnet as wn
```

untuk mengecek kesamaan makna atau synonym bisa menggunakan kode seperti dibawah ini

In [102]:
```
print(wn.synsets('motorcar'))
print(wn.synset('car.n.01').lemma_names())
```

```
[Synset('car.n.01')]
['car', 'auto', 'automobile', 'machine', 'motorcar']
```

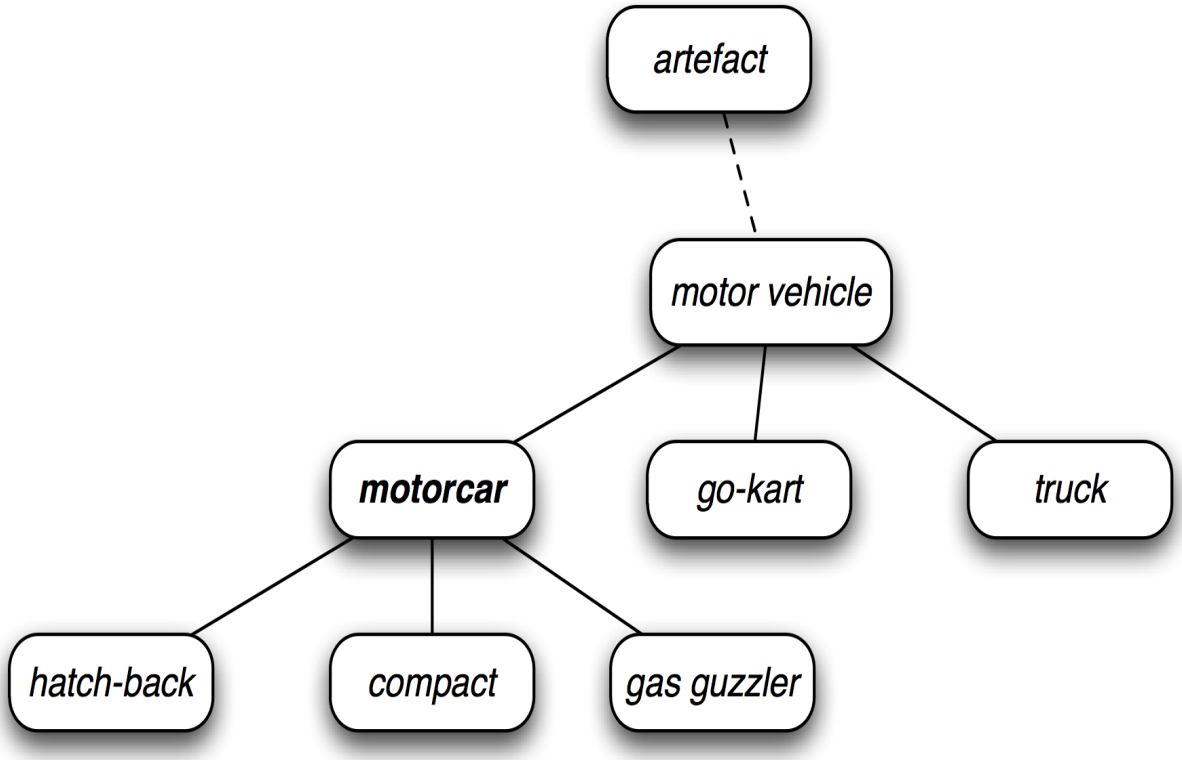
In [103]:
```
print(wn.synsets("cake"))
print(wn.synset('cake.n.01').lemma_names())
print(wn.synset('patty.n.01').lemma_names())
print(wn.synset('cake.n.03').lemma_names())
print(wn.synset('coat.v.03').lemma_names())
```

```
[Synset('cake.n.01'), Synset('patty.n.01'), Synset('cake.n.03'), Synset('coat.
v.03')]
['cake', 'bar']
['patty', 'cake']
['cake']
['coat', 'cake']
```

In [104]:
```
print(wn.synsets("drink"))
print(wn.synset('drink.n.01').lemma_names())
print(wn.synset('beverage.n.01').lemma_names())
print(wn.synset('drink.n.04').lemma_names())
print(wn.synset('swallow.n.02').lemma_names())
print(wn.synset('drink.v.01').lemma_names())
print(wn.synset('drink.v.02').lemma_names())
print(wn.synset('toast.v.02').lemma_names())
print(wn.synset('drink_in.v.01').lemma_names())
print(wn.synset('drink.v.05').lemma_names())
```

```
[Synset('drink.n.01'), Synset('drink.n.02'), Synset('beverage.n.01'), Synset('d
rink.n.04'), Synset('swallow.n.02'), Synset('drink.v.01'), Synset('drink.v.0
2'), Synset('toast.v.02'), Synset('drink_in.v.01'), Synset('drink.v.05')]
['drink']
['beverage', 'drink', 'drinkable', 'potable']
['drink']
['swallow', 'drink', 'deglutition']
['drink', 'imbibe']
['drink', 'booze', 'fuddle']
['toast', 'drink', 'pledge', 'salute', 'wassail']
['drink_in', 'drink']
['drink', 'tope']
```

5.2 The WordNet Hierarchy



In [105]:
```
drink = wn.synset('drink.v.01')
drink_type = drink.hyponyms()
drink_type[0]
```

Out[105]:
```
Synset('drain_the_cup.v.01')
```

In [106]:
```
sorted(lemma.name() for synset in drink_type for lemma in synset.lemmas())
```

Out[106]:
```
['belt_down',
 'bolt_down',
 'down',
 'drain_the_cup',
 'drink_down',
 'drink_up',
 'guggle',
 'gulp',
 'gurgle',
 'guzzle',
 'kill',
 'lap',
 'lap_up',
 'lick',
 'pop',
 'pour_down',
 'quaff',
 'sip',
 'suck',
 'swig',
 'swill',
 'swill_down',
 'toss_off']
```

In [107]:
```
drink.hypernyms()
```

Out[107]:
```
[Synset('consume.v.02')]
```

In [108]:
```
paths = drink.hypernym_paths()
len(paths)
```

Out[108]:
```
1
```

In [109]:
```
[synset.name() for synset in paths[0]]
```

Out[109]:
```
['consume.v.02', 'drink.v.01']
```

In [110]:
```
 drink.root_hypernyms()
```

Out[110]:
```
[Synset('consume.v.02')]
```

5.3 More Lexical Relations

In [111]:
```
wn.synset('tree.n.01').part_meronyms()
```

Out[111]:
```
[Synset('burl.n.02'),
 Synset('crown.n.07'),
 Synset('limb.n.02'),
 Synset('stump.n.01'),
 Synset('trunk.n.01')]
```

In [112]:
```
 wn.synset('tree.n.01').substance_meronyms()
```

Out[112]:
```
[Synset('heartwood.n.01'), Synset('sapwood.n.01')]
```

In [113]:
```
wn.synset('tree.n.01').member_holonyms()
```

Out[113]:
```
[Synset('forest.n.01')]
```

In [114]:
```
for synset in wn.synsets('mint', wn.NOUN):
    print(synset.name() + ':', synset.definition())
```

```
batch.n.02: (often followed by `of') a large number or amount or extent
mint.n.02: any north temperate plant of the genus Mentha with aromatic leaves a
nd small mauve flowers
mint.n.03: any member of the mint family of plants
mint.n.04: the leaves of a mint plant used fresh or candied
mint.n.05: a candy that is flavored with a mint oil
mint.n.06: a plant where money is coined by authority of the government
```

In [115]:
```
wn.synset('mint.n.04').part_holonyms()
```

Out[115]:
```
[Synset('mint.n.02')]
```

In [116]:
```
wn.synset('mint.n.04').substance_holonyms()
```

Out[116]:
```
[Synset('mint.n.05')]
```

In [117]:
```
wn.synset('walk.v.01').entailments()
```

Out[117]:
```
[Synset('step.v.01')]
```

In [118]:
```
wn.synset('eat.v.01').entailments()
```

Out[118]:
```
[Synset('chew.v.01'), Synset('swallow.v.01')]
```

In [119]:
```
wn.synset('tease.v.03').entailments()
```

Out[119]:
```
[Synset('arouse.v.07'), Synset('disappoint.v.01')]
```

In [120]:
```
wn.lemma('supply.n.02.supply').antonyms()
```

Out[120]:
```
[Lemma('demand.n.02.demand')]
```

In [121]:
```
wn.lemma('rush.v.01.rush').antonyms()
```

Out[121]:
```
[Lemma('linger.v.04.linger')]
```

In [122]:
```
wn.lemma('horizontal.a.01.horizontal').antonyms()
```

Out[122]:
```
[Lemma('vertical.a.01.vertical'), Lemma('inclined.a.02.inclined')]
```

In [123]:
```
wn.lemma('staccato.r.01.staccato').antonyms()
```

Out[123]:
```
[Lemma('legato.r.01.legato')]
```

5.4 Semantic Similarity

In [124]:
```
right = wn.synset('right_whale.n.01')
orca = wn.synset('orca.n.01')
minke = wn.synset('minke_whale.n.01')
tortoise = wn.synset('tortoise.n.01')
novel = wn.synset('novel.n.01')
```

In [125]:
```
print("minke" , right.lowest_common_hypernyms(minke))
print("orca" , right.lowest_common_hypernyms(orca))
print("tortoise" , right.lowest_common_hypernyms(tortoise))
print("novel" , right.lowest_common_hypernyms(novel))
```

```
minke [Synset('baleen_whale.n.01')]
orca [Synset('whale.n.02')]
tortoise [Synset('vertebrate.n.01')]
novel [Synset('entity.n.01')]
```

In [126]:
```
print("right" , orca.lowest_common_hypernyms(right))
print("minke" , orca.lowest_common_hypernyms(minke))
print("tortoise" , orca.lowest_common_hypernyms(tortoise))
print("novel" , orca.lowest_common_hypernyms(novel))
```

```
right [Synset('whale.n.02')]
minke [Synset('whale.n.02')]
tortoise [Synset('vertebrate.n.01')]
novel [Synset('entity.n.01')]
```

In [127]:
```
right.lowest_common_hypernyms(tortoise)
```

Out[127]:
```
[Synset('vertebrate.n.01')]
```

In [128]:
```
right.lowest_common_hypernyms(novel)
```

Out[128]:
```
[Synset('entity.n.01')]
```

In [129]:
```
wn.synset('baleen_whale.n.01').min_depth()
```

Out[129]:
```
14
```

In [130]:
```
wn.synset('whale.n.02').min_depth()
```

Out[130]:
```
13
```

In [131]:
```
wn.synset('vertebrate.n.01').min_depth()
```

Out[131]:
```
8
```

In [132]:
```
wn.synset('entity.n.01').min_depth()
```

Out[132]:
```
0
```

In [133]:
```
right.path_similarity(minke)
```

Out[133]:
```
0.25
```

In [134]:
```
right.path_similarity(orca)
```

Out[134]:
```
0.16666666666666666
```

In [135]:
```
right.path_similarity(tortoise)
```

Out[135]:
```
0.07692307692307693
```

In [136]:
```
right.path_similarity(novel)
```

Out[136]:
```
0.043478260869565216
```

6 Summary

- Korpus teks adalah kumpulan teks yang besar dan terstruktur. NLTK hadir dengan banyak korpora, mis., Brown Corpus, nltk.corpus.brown.
- Beberapa korpora teks dikategorikan, misalnya, berdasarkan genre atau topik; terkadang kategori korpus saling tumpang tindih.
- Distribusi frekuensi bersyarat adalah kumpulan distribusi frekuensi, masing-masing untuk kondisi yang berbeda. Mereka dapat digunakan untuk menghitung frekuensi kata, diberikan konteks atau genre.
- Program python yang panjangnya lebih dari beberapa baris harus dimasukkan menggunakan editor teks, disimpan ke file dengan ekstensi .py, dan diakses menggunakan pernyataan impor.
- Fungsi Python memungkinkan Anda untuk mengaitkan nama dengan blok kode tertentu, dan menggunakan kembali kode itu sesering yang diperlukan.
- Beberapa fungsi, yang dikenal sebagai "metode", dikaitkan dengan objek dan kami memberikan nama objek diikuti dengan periode yang diikuti oleh fungsi, seperti ini: x.funct (y), mis., Word.isalpha ().
- Untuk mencari tahu tentang beberapa variabel v, ketik help (v) dalam interpreter interaktif Python untuk membaca entri bantuan untuk objek semacam ini.
- WordNet adalah kamus bahasa Inggris yang berorientasi semantik, yang terdiri dari set sinonim - atau sinkronisasi - dan disusun dalam suatu jaringan.
- Beberapa fungsi tidak tersedia secara default, tetapi harus diakses menggunakan pernyataan impor Python.