WordNet organizes words into sets of synonyms called synsets, which are connected by lexical and semantic relations. Each synset represents a distinct concept, and all the words in a synset share the same meaning. In addition to providing information about word meanings and relationships, WordNet also includes information about word usage and examples of how words are used in context.

```
from nltk.corpus import wordnet as wn
```

```
wn.synsets('people')
```

```
[Synset('people.n.01'),
 Synset('citizenry.n.01'),
 Synset('people.n.03'),
 Synset('multitude.n.03'),
 Synset('people.v.01'),
 Synset('people.v.02')]
```

```
people = wn.synset('people.n.01')
print('definition: ', people.definition())
print('examples: ', people.examples())
print('lemmas: ', people.lemmas())
```

```
definition:  (plural) any group of human beings (men or women or children) collectively
examples:  ['old people', 'there were at least 200 people in the audience']
lemmas:  [Lemma('people.n.01.people')]
```

```
hyp = people.hypernyms()[0]
top = wn.synset('entity.n.01')
while hyp:
  print(hyp)
  if hyp == top:
    break
  if hyp.hypernyms():
    hyp = hyp.hypernyms()[0]
```

```
print(people.hypernyms())
print(people.hyponyms())
print(people.part_meronyms())
print(people.part_holonyms())
print(people.lemmas()[0].antonyms)
```

```
Synset('group.n.01')
Synset('abstraction.n.06')
Synset('entity.n.01')
[Synset('group.n.01')]
[Synset('age_group.n.01'), Synset('ancients.n.01'), Synset('baffled.n.01'), Synset('blind.n.01'), Synset('blood.n.05'), Synse
[]
[]
<bound method Lemma.antonyms of Lemma('people.n.01.people')>
```

```
# selecting verb
verb = "walk"
walk = wn.synsets(verb)
```

```
walk
```

```
[Synset('walk.n.01'),
 Synset('base_on_balls.n.01'),
 Synset('walk.n.03'),
 Synset('walk.n.04'),
 Synset('walk.n.05'),
 Synset('walk.n.06'),
 Synset('walk_of_life.n.01'),
 Synset('walk.v.01'),
 Synset('walk.v.02'),
 Synset('walk.v.03'),
 Synset('walk.v.04'),
 Synset('walk.v.05'),
 Synset('walk.v.06'),
 Synset('walk.v.07'),
 Synset('walk.v.08'),
 Synset('walk.v.09'),
 Synset('walk.v.10')]
```

```
walk = wn.synset('walk.v.01')
print('definition: ', walk.definition())
print('examples: ', walk.examples())
print('lemmas: ', walk.lemmas())
```

```
definition:  use one's feet to advance; advance by steps
examples:  ["Walk, don't run!", 'We walked instead of driving', 'She walks with a slight limp', 'The patient cannot walk yet
lemmas:  [Lemma('walk.v.01.walk')]
```

```
hyper = lambda s:s.hypernyms()
list(walk.closure(hyper))
```

```
[Synset('travel.v.01')]
```

```
morph_verb = wn.morphy(verb)
print(wn.synsets(morph_verb))
```

```
[Synset('walk.n.01'), Synset('base_on_balls.n.01'), Synset('walk.n.03'), Synset('walk.n.04'), Synset('walk.n.05'), Synset('wa
```

```
from nltk.corpus import sentiwordnet as swn
```

```
word1 = wn.synsets('help')
word2 = wn.synsets('assist')
```

```
word1
```

```
[Synset('aid.n.02'),
 Synset('assistant.n.01'),
 Synset('aid.n.01'),
 Synset('avail.n.01'),
 Synset('help.v.01'),
 Synset('help.v.02'),
 Synset('help.v.03'),
 Synset('help_oneself.v.01'),
 Synset('serve.v.05'),
 Synset('help.v.06'),
 Synset('avail.v.03'),
 Synset('help.v.08')]
```

```
word2
```

```
[Synset('aid.n.02'),
 Synset('assist.n.02'),
 Synset('help.v.01'),
 Synset('assist.v.02'),
 Synset('serve.v.10')]
```

```
wup = wn.wup_similarity(word1[0], word2[0])
print("similarity: ", wup)
```

```
similarity:  1.0
```

From my observation I observed that the reason that the WUP index is one is because if we print the hierarchy for both help and assist they are only one level up before they share the same synset and since the WUP looks at common path it makes sense that the index is one

SentiWordNet is an online lexical resource . It provides an annotated resource of words and their associated sentiment scores based on the Princeton WordNet structure. SentiWordNet is a valuable tool for natural language processing and sentiment analysis.

```
from nltk.corpus import sentiwordnet as swn
wn.synsets('sad')
```

```
[Synset('sad.a.01'), Synset('sad.s.02'), Synset('deplorable.s.01')]
```

```
import nltk
nltk.download('sentiwordnet')
```

```
[nltk_data] Downloading package sentiwordnet to /root/nltk_data...
[nltk_data]   Package sentiwordnet is already up-to-date!
True
```

```
senti_list = list(swn.senti_synsets('sad'))
for item in senti_list:
  print(item)
sent = 'I am sad and angry'
tokens = sent.split()
for token in tokens:
  print('For the Word: ', token)
  syn_list = list(swn.senti_synsets(token))
  if syn_list:
    for item in syn_list:
      print(item)
    print()
```

```
<sad.a.01: PosScore=0.125 NegScore=0.75>
<sad.s.02: PosScore=0.0 NegScore=0.25>
<deplorable.s.01: PosScore=0.0 NegScore=1.0>
For the Word:  I
<iodine.n.01: PosScore=0.0 NegScore=0.0>
<one.n.01: PosScore=0.0 NegScore=0.0>
<i.n.03: PosScore=0.0 NegScore=0.0>
<one.s.01: PosScore=0.0 NegScore=0.25>

For the Word:  am
<americium.n.01: PosScore=0.0 NegScore=0.0>
<master_of_arts.n.01: PosScore=0.0 NegScore=0.125>
<amplitude_modulation.n.01: PosScore=0.0 NegScore=0.0>
<be.v.01: PosScore=0.25 NegScore=0.125>
<be.v.02: PosScore=0.0 NegScore=0.0>
<be.v.03: PosScore=0.0 NegScore=0.0>
<exist.v.01: PosScore=0.0 NegScore=0.0>
<be.v.05: PosScore=0.0 NegScore=0.0>
<equal.v.01: PosScore=0.125 NegScore=0.125>
<constitute.v.01: PosScore=0.0 NegScore=0.0>
<be.v.08: PosScore=0.0 NegScore=0.0>
<embody.v.02: PosScore=0.0 NegScore=0.0>
<be.v.10: PosScore=0.0 NegScore=0.0>
<be.v.11: PosScore=0.0 NegScore=0.0>
<be.v.12: PosScore=0.0 NegScore=0.0>
<cost.v.01: PosScore=0.0 NegScore=0.0>

For the Word:  sad
<sad.a.01: PosScore=0.125 NegScore=0.75>
<sad.s.02: PosScore=0.0 NegScore=0.25>
<deplorable.s.01: PosScore=0.0 NegScore=1.0>

For the Word:  and
For the Word:  angry
<angry.a.01: PosScore=0.375 NegScore=0.375>
<angry.s.02: PosScore=0.375 NegScore=0.5>
<angry.s.03: PosScore=0.0 NegScore=0.875>
```

A collocation is a sequence of words that frequently occur together in a language and have a tendency to co-occur more often than would be expected by chance. Collocations are an important aspect of language learning and can help improve one's vocabulary and overall fluency in a language.

```
import nltk
nltk.download('genesis')
```

```
[nltk_data] Downloading package genesis to /root/nltk_data...
[nltk_data]   Unzipping corpora/genesis.zip.
True
```

```
import nltk
from nltk.book import *
import math
text4
```

```
<Text: Inaugural Address Corpus>
```

```
import nltk
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

```
text4.collocations()
```

```
United States; fellow citizens; years ago; four years; Federal
Government; General Government; American people; Vice President; God
bless; Chief Justice; one another; fellow Americans; Old World;
Almighty God; Fellow citizens; Chief Magistrate; every citizen; Indian
tribes; public debt; foreign nations
```

```
text = ' '.join(text4.tokens)
vocab = len(set(text4))
hg = text.count('Federal Government')/vocab
print("p(Federal Government) = ",hg )
h = text.count('Federal')/vocab
print("p(Federal) = ", h)
g = text.count('Government')/vocab
print('p(Government) = ', g)
pmi = math.log2(hg/(h*g))
print('pmi = ', pmi)
```

```
p(Federal Government) =  0.0031920199501246885
p(Federal) =  0.006483790523690773
p(Government) =  0.03371571072319202
pmi =  3.868067366919006
```

The results show a positive pmi and rather high positive pmi which suggests that "Federal Government" is most likely a collocation