Homework 7:
Lexical Semantics: WordNet

Robert Litschko*
Symbolische Programmiersprache

Due: Thursday December 8, 2022, 12:00 (noon)

In this exercise you will:

- measure semantic similarity of words using WordNet
- find hyponyms of the given hypernyms in the text

This part of the homework will be graded using unit tests by running:

```
python3 -m unittest -v hw07_wordnet/test_wordnet.py
```

Exercise 1: Exercise 1: WordNet semantic similarity [4 points]

In `noun_similarity.py` implement the function `get_similarity_scores(pairs)` so that it ranks word pairs presented as list of tuples [("word1", "word2"), ...] in order of decreasing similarity. The similarity of a word pair should correspond to the similarity of their most similar synsets; use the predefined path-based similarity measures (`synset1.path_similarity(synset2)`) for that. [4 points]

Exercise 2: Exercise 2: Finding Hyponyms with WordNet [10 points]

In this exercise, you will write a program to find nouns (hyponyms) that belong to certain categories (hypernyms) in wordnet. These categories are *relative, science* and *illness*. You will need the file ada_lovelace.txt. Take a look at the file hw07_wordnet/find_hyponyms.py. Complete some methods to find hyponyms:

1. In the class constructor determine all noun lemmas from ada_lovelace.txt following the steps:
   - Read text as a string
   - Split text into sentences: use `nltk.sent_tokenize()`

*Credit: Exercises are based on previous iterations from Katerina Kalouli.

- Split sentences into tokens: use `nltk.word_tokenize()`
- Perform POS tagging of tokens (Note that the POS tagging should be called on all tokens of all sentences at the same time and not on each sentence separately; otherwise, your results will not match the expected output of the unit test.)
  - Use `WordNetLemmatizer()` to lemmatize nouns (any token whose POS tags start with "N")
  - Determine all noun lemmas [6 points]

2. Implement the class method `hypernym_of(self, synset1, synset2)` by returning True if `synset2` is a hypernym of `synset1`, or if they are the same synsets. Return False otherwise. Hint: use `synset1.hypernyms()` and do not forget to check whether the hypernym of `synset1` is hypernym of `synset2`. [1 point]

3. Implement the class method `get_hyponyms(self,hypernym)`. This method should return the set of noun lemmas in ada_lovelace.txt that are hyponyms (subordinates) to the hypernym. Hint: the `noun_lemmas` are already stored as an attribute of the class – use them. Also make sure you use the implemented methof `hypernym_of(self,synset1, synset2)`. [3 points]

The output would then look as follows:

```
>>> from nltk.corpus import wordnet as wn
>>>
>>> hyponym_searcher = HyponymSearcher('ada_lovelace.txt')
>>>
>>> wn.synsets("relatives")
[Synset('relative.n.01'), Synset('relative.n.02')]
>>>
>>> hyponym_searcher.get_hyponyms(wn.synsets('relatives')[0]) # relative.n.01
'grandchild', 'wife', 'boy', 'half-sister', 'husband', 'relation',
'Family', 'relative', 'daughter', 'son', 'child', 'baby', 'father',
'parent', 'girl', 'mother'
>>>
>>> hyponym_searcher.get_hyponyms(wn.synsets('science')) # science.n.01
'calculus', 'phrenology', 'analysis', 'Science', 'anatomy',
'Magnetism', 'math', 'government', 'science', 'thermodynamics',
'mathematics'
>>>
>>> hyponym_searcher.get_hyponyms(wn.synsets('illness')) # illness.n.01
'measles', 'madness', 'cancer', 'disease', 'illness'
```