Assignment4 - WordNet and Regular Expressions

Due November 16th at 12:30pm.

There are two parts to this assignment: one on WordNet and one on regular expressions.

Part 1: Using WordNet for Query Expansion

Query expansion is a common technique in information retrieval. It involves reformulating a given query to improve retrieval performance by expanding the search query to match additional documents. Typical expansion techniques are:

1. Finding synonyms of words.
2. Finding other semantically related words like hyponyms and hypernyms.
3. Finding morphological forms of words by stemming or lemmatizing.
4. Fixing spelling errors.

In this part of the assignment we will focus on the first two of these techniques using relations between synsets in WordNet. To that end, complete the function `get_siblings()` in `wordnet.py`. This function takes a word and returns a dictionary with for each sense of the word a list of lemmas that are siblings of the input word. For example, if the input is `orca` we should create a dictionary with only one key, `killer_whale.n.01`, which is the name of the only synset that `orca` occurs in. The value of that key should be a list with the following elements:

```
bottlenose_dolphin, bottle-nosed_dolphin, bottlenose, common_dolphin,
Delphinus_delphis,
grampus, Grampus_griseus, killer_whale, killer, orca, grampus, sea_wolf,
Orcinus_orca,
pilot_whale, black_whale, common_blackfish, blackfish,
Globicephala_melaena, porpoise,
river_dolphin, white_whale, beluga, Delphinapterus_leucas
```

Note that the list includes synonyms of `orca`, that is, other lemmas from the `killer_whale.n.01` synset like `killer_whale` and `grampus`, but also lemmas like `bottlenose`, a member of the `bottlenose_dolphin.n.01` synset, which is a sibling of the `killer_whale.n.01` synset because both have the `dolphin.n.02` synset as a hypernym.

Part 2: Regular expressions

Your task here is to write a set of functions that use regular expressions to perform some tasks. Some of these exercises are based on exercises from chapter 3 of the NLTK book. For each exercise, you complete a function in the `regular.py` Python script. Do not change the names of these functions since that will break our tests.

For this exercise, you must use Python's `re` module and you are not allowed to use any other imports unless stated otherwise.

Complete the following functions in `regular.py`:

1. `is_camel_case(s)`. Returns True or False depending on whether the string `s` is in camel case, which is when a string acts like a string of capitalized words strung together. For example: `CamelCase`, `TrueOrFalse`. You may assume that the input has no spaces.
2. `is_arithmetic_expression(s)`. Returns True or False depending on whether the string `s` is an arithmetic expression using integers, addition, and multiplication, such as `2*3+8` and `-2*3-7`. You do not need to deal with brackets. Assume there are no spaces in the expression.
3. `remove_html(url)`. Reads the url, removes all HTML tags and prints the result to the standard output. You can use `html.unescape()` to remove all HTML character references like '>'. The functionality to read the url is included. You can run the function on http://nltk.org/ or any other webpage of your liking.
4. `short_words(s)`. Takes a string and returns a list of all words (that is, sequences of letters) in the string that are shorter than 5 characters. The words in the output list should be in the same order as found in the text. You are not allowed to use `split()`.
5. `is_date(s)`. Takes a string and returns True or False depending on whether the string is a date. This is an open-ended question and you can earn extra points by recognizing more kinds of dates than the minimum requirement. You should at least recognize dates in the following formats: `12/31/2018`, `12/31/18` and `December 31st 2018`. For this function, add a class with unit tests to `test_regular.py`.

In `regular.py`, all these functions have some content that does nothing but producing some output that passes the doctests. You should add code that still allows the code to pass the tests (except perhaps for `remove_html()` since that test is very very specific).