Project

Due: Friday, December 13, 2019 at 10:00am **Points:** 100

Please turn in your answer for this homework assignment on Canvas under Project in Assignments.

This exercise has you query the PubMed database for a list of publications related to a keyword. You will then take the list of publication numbers you get back and turn them into a list of papers with a second query to the PubMed database.

To access the PubMed database, go to the URL below, replacing *keyword* with the keyword you want to search for, and *num* the number of publications you would like returned:

```
https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?
     db=pubmed&retmode=json&retmax=num&sort=relevance&term=keyword
```

with no spaces and all on a single line.

So, for example, to find the 20 publications most relevant to "fever", the URL would be:

```
https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?
     db=pubmed&retmode=json&retmax=20&sort=relevance&term=fever
```

with no spaces and all on a single line.

When you read the contents of this web page, it is in the JSON format. You can turn this into a dictionary easily using the module `json`. The method `json.loads(contents)`, where *contents* is the contents of the web page, returns a dictionary with one entry, the key of which is "esearchresults". The associated value is another dictionary. The part you want is a list of the publication numbers. The key is "idlist" and the value is a list of the numbers. Save them as a sequence of numbers separated by commas (this is useful in the next part).

Then, for each publication, use the following URL to get the metadata:

```
http://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi?db=pubmed&retmode=xml&id=idlist
```

with no spaces and all on a single line, and *idlist* replaced with the ID list you got earlier. The web page you get back is an XML document giving details of the publications.

Your job is to print a bibliography from this record. Your entry for each journal should look like this:

A. Bester, R. Zelazny, and H. Ellison, "On the Role of Viruses in Future Epidemics," Journal of Irreproducible Results 3(4) pp. 29–35 (Mar. 2103). PUBMED: 23456789; DOI 12.1119/2847595.

Then print the abstract, if it is present in the record.

If there is no DOI, use the PII. If neither is there, omit that part of the entry.

You will need to look at the XML records to get the fields. These are delimited by tags with attributes, each of which may have a value. For example, the element

```
<ELocationID EIdType="doi" ValidYN="Y">10.1016/j.vaccine.2015.04.071</ELocationID>
```

has a tag of `ELocationID`, attributes of `EIdType` (with value `doi`) and `ValidYN` (with a value of `Y`), and the field contains `10.1016/j.vaccine.2015.04.071`, which (as the EIdType value indicates) is a DOI.

The easiest way to see what the records look like is to ask for a single entry. You can then see its structure. The fields of interest will have these tags:

- Article — contains the Journal, ArticleTitle (article title), Pagination (page numbers), ElocationID, which gives both the DOI and PII (if those exist), the Abstract, and the AuthorList.
- Journal — this consists of several elements, including JournalIssue, which contains the Volume, Issue, and PubDate (publication date), and Title (article title).
- AuthorList — this lists the authors, each author being in a field called Author. Subfields of interest are LastName and Initial (the initial of the first name)

Those will be enough to build the reference, as described above.

You can find methods for processing XML and JSON in the Python Library Reference at `https://docs.python.org/3.7/library/xml.etree.elementtree.html` and `https://docs.python.org/3.7/library/json.html`, respectively.

Call this program "pubmed.py" when you submit it.

A Problem You May Encounter, and Its Solution

If you get the following error (it will be on one line):

```
[SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed:
                    unable to get local issuer certificate (_ssl.c:1051)
```

that is a problem at the server end that, unfortunately, is causing your connection to PubMed to fail. To solve it, import the module "ssl" and then put the following anywhere *before* you go to the web site:

```
try:
    _create_unverified_https_context = ssl._create_unverified_context
except AttributeError:
    # Legacy Python that doesn't verify HTTPS certificates by default
    pass
else:
    # Handle target environment that doesn't support HTTPS verification
    ssl._create_default_https_context = _create_unverified_https_context
```

In case you want to know what's going on (and if you don't, skip this part), when you connect to a site using "https:", the server sends a *certificate* to your browser to verify that the client (your browser or this program) went to the right place. If this check fails, or the certificate cannot be validated for some reason, it will be rejected by your client. If the client is a browser, you usually get a message that says something like "Bad certificate" or "Unable to verify certificate". In this program, you will get the error message above. The above Python lines tell your program to ignore this error.

Here's what the above means. "ssl" is a module that handles secure connections; you can tell these by the "https:" in the URL. By default, it analyzes the certificate, and does the rejection as described above. The attribute "_create_unverified_context" says that the ssl module is to ignore the certificate (the "unverified" part). The `except` part is for versions of the ssl module that do not check certificate validity, and says to ignore that the attribute doesn't exist. If it does exist, then the `else` part sets the module to ignore any errors with the certificate.

In more detail, the ssl module checks certificate validity by default. If the attribute "_create_unverified_context" does not exist, the ssl module is an old module that does not check certificate validity; that the attribute does not exist causes an AttributeError, and in this case we don't need to do anything. If it does exist, the default context for the new instance of ssl is set to that attribute, meaning the ssl module will not check certificate validity.