Lord_of_the_flies_solution

January 30, 2016

0.1 Find the author that published the most papers on Drosophila virilis.

```
In [1]: from Bio import Entrez
        import re
```

0.1.1 We first want to know now many publications have D. virilis in their title or abstract. We use the NCBI history function in order to refer to this search in our subsequent efetch call.

```
In [2]: # Remember to edit the e-mail address
        Entrez.email = "your_name@yourmailhost.com" # Always tell NCBI who you are
        handle = Entrez.esearch(db="pubmed", term="Drosophila virilis[Title/Abstract]", usehistory="y")
        record = Entrez.read(handle)
        # generate a Python list with all Pubmed IDs of articles about D. virilis
        id_list = record["IdList"]
        record["Count"]

Out[2]: '528'
```

```
In [3]: webenv = record["WebEnv"]
        query_key = record["QueryKey"]
```

0.1.2 Retrieve the PubMed entries using our search history

```
In [4]: handle = Entrez.efetch(db="pubmed",rettype="medline", retmode="text", retstart=0,
        retmax=528, webenv=webenv, query_key=query_key)
```

```
In [5]: out_handle = open("D_virilis_pubs.txt", "w")
        data = handle.read()
        handle.close()
        out_handle.write(data)
        out_handle.close()
```

0.1.3 We construct a dictionary with all authors as keys and author occurance as value.

```
In [6]: with open("D_virilis_pubs.txt") as datafile:
            author_dict = {}
            for line in datafile:
                if re.match("AU", line):
                    # capture author
                    author = line.split("-", 1)[1]
                    # remove leading and trailing whitespace
                    author = author.strip()
                    # if key is present, add 1
                    # if it's not present, initialize at 1
                    author_dict[author] = 1 + author_dict.get(author, 0)
```

0.1.4 Dictionaries do not have a natural order but we can sort a dictionary based on the values.

```
In [7]: # use the values (retrieved by author_dict.get) for sorting the dictionary
        # The function "sorted" returns a list that can be indexed to return only some elements, e.g. t
        for author in sorted(author_dict, key = author_dict.get, reverse = True)[:5]:
            print(author, ":", author_dict[author])
```

```
Gruntenko NE : 36
Evgen'ev MB : 29
Raushenbakh IIu : 24
Hoikkala A : 23
Korochkin LI : 22
```