Protein Finder

This is a Python script for pulling data out of https://www.ncbi.nlm.nih.gov/ and https://www.uniprot.org/ using their APIs. It programmatically downloads all information found for given search terms in the databases. It saves biologists hours or even days of time.

Usage

API-KEY

Before proteinFinder.py can be used, a line in the code has to be changed. Since NCBI search requires an identification key or API-KEY, the first thing to do is to create an account on NCBI. Next it is best to follow this guide to acquire the key. The API key needs to be inserted into the following line of code in proteinFinder.py:

```
apiKey = "INSERT YOUR KEY HERE"
```

Once this is done, the program has its full functionality.

Basic Functionality

Let's assume we have a csv File containing gene loci. The loci were originally from the NCBI database. Now we want to have more information about the gene product. We start with the input file, named 'genes.csv'. It stores the following information:

Gene loci:

gene_loci
Dshi_0051
Dshi_0052
Dshi_0053
Dshi_0054
Dshi_0055
Dshi_0056
Dshi_0057
Dshi_0057
Dshi_0058
Dshi_0059
Dshi_0060
Dshi_0061
CC_2_9

We start the command line prompt and navigate to the folder containing proteinFinder.py. To use the Protein Finder module we need to know what kind of arguments the script is expecting. We can go look it up in documentation or we just type in:

```
python3 proteinFinder.py -h
```

The flag -h is short for help. This command shows all arguments the script is able to process and a short description:

```
usage: proteinFinder.py [-h] [--noheaders] [-db DATABASE] [-c COLUMNS]
                    [--organism ORGANISM] [-i {text,id}] [--idType IDTYPE]
                    inputFile outputFile {ncbi,uniprot,ncbi+uniprot}

positional arguments:
inputFile             Path to csvfile with query strings
outputFile            Path to file the results should besaved
{ncbi,uniprot,ncbi+uniprot}
                        Which website should be queried

optional arguments:
-h, --help            show this help message and exit
--noheaders           When reading the input file it does notskip the first row
-db DATABASE, --database DATABASE
                        Database which shouldbe queried
-c COLUMNS, --columns COLUMNS
                        When quiering Uniprot,depicts, which type of
                        information should be downloaded. Terms have to be
                        comma seperated
--organism ORGANISM   Narrow down the search tospecified organism
-i {text,id}, --inputDataType {text,id}
                        Type of data stored in input File
--idType IDTYPE       What database the IDs are from?
```

For the script working proberly, the positional arguments are required. The minimum working script, would be:

```
python3 proteinFinder.py inputFilePath outputFilePath websiteToQuery
```

In our case the data is stored in a csv file called 'genes.csv'. And we want the endresult to be saved in 'proteinInformation.csv'. Since Uniprot does not recognise our gene loci we first need to query NCBI, for the ID. And with that ID we query the Uniprot database. luckily the script is smart and handles this with the simple argument 'ncbi+uniprot'. So we end up with the following command:

```
python3 proteinFinder.py genes.csv proteinInformation.csv ncbi+uniprot
```

The script informs us that it ignores the first line in the input file, which contains the headers. If you don't want this behaviour, you need to pass the argument '–noheaders'. We also get a feedback on the status of the querys:

```
Skipping headers
Query 1 of 12
Query 2 of 12
Query 3 of 12
Query 4 of 12
Query 5 of 12
Query 6 of 12
Query 7 of 12
Query 8 of 12
Query 9 of 12
Query 10 of 12
Query 11 of 12
Query 12 of 12
```

The information found by the search are summed up in the table. The resulting table in 'proteinInformation.csv' should look similar to this:

query	ID_NCBI	Entry	Gene names	...
Dshi_0052	157910368	A8LJX5	Dshi_0052	...
Dshi_0053	157910369	A8LJX6	Dshi_0053	...
Dshi_0054	157910370	A8LJX7	gst2 Dshi_0054	...
Dshi_0055	157910371	A8LJX8	holA Dshi_0055	...
Dshi_0056	157910372	A8LJX9	Dshi_0056	...
Dshi_0057	189082998	A8LJY0	leuS Dshi_0057	...
Dshi_0057	157910373	A8LJY0	leuS Dshi_0057	...
Dshi_0058	157910374	A8LJY1	lolA Dshi_0058	...
Dshi_0059	157910375	A8LJY2	ftsK Dshi_0059	...
Dshi_0060	157910376	A8LJY3	Dshi_0060	...
Dshi_0061	157910377	A8LJY4	Dshi_0061	...
Dshi_0062	157910378	A8LJY5	ubiH Dshi_0062	...
Dshi_0063	157910379	A8LJY6	suhB2 Dshi_0063	...

The table stores the original query, the IDs found in NCBI and the Uniport entry. Overall the search was successful. However there are 2 interesting cases. First, when the script does not find some entrys, it will fill the row with a 'Not Found' tag. Second, Dshi_0057 seems to appear twice. This is by design. When multiple IDs or entries are found for a query, each ID or entry, will be represented in a unique row. The reason is, there is no way for the programmer to safely determine what entry is relevant. Therefore every table, has to be inspected for such duplicates and be eliminated by hand.

Searching NCBI

To query NCBI you need to specify which database should be queried. A full list of all names can be found here. Right now only text search is available for NCBI.

```
python3 proteinFinder.py inputFile outputFile ncbi -db protein
```

If no -db flag is typed in the default value will be the protein database.

Example

Starting from a table genes.csv with gene loci, the corresponding ncbi accession numbers should be acquire.

gene_loci
Dshi_0051
Dshi_0052
Dshi_0053
Dshi_0054
Dshi_0055
Dshi_0056
Dshi_0057
Dshi_0057
Dshi_0058
Dshi_0059
Dshi_0060
Dshi_0061
CC_2_9

```
python3 proteinFinder.py genes.csv geneInfo.csv ncbi
```

This command outputs geneInfo.csv:

query	ID_NCBI
Dshi_0051	157910367
Dshi_0052	157910368
Dshi_0053	157910369
Dshi_0054	157910370
Dshi_0055	157910371
Dshi_0056	157910372
Dshi_0057	189082998
Dshi_0057	157910373
Dshi_0058	157910374
Dshi_0059	157910375

query	ID_NCBI
Dshi_0060	157910376
Dshi_0061	157910377

Searching Uniprot

Uniprot can be searched in two ways.

Query with search terms

```
python3 proteinFinder.py inputFile outputFile uniprot
```

First it can be searched by search terms. This takes a long time since every term needs to make its own request to the database.

The results can be filtered for a specific organism with the –organism flag followed by the species name. For trustworthy results it is recommended to input the UniProt taxon id, found on the website.

```
python3 proteinFinder.py inputFile outputFile uniprot --organism "Dinoroseobacter shibae"
python3 proteinFinder.py inputFile outputFile uniprot --organism 398580
```

Query with IDs

```
python3 proteinFinder.py inputFile outputFile uniprot --inputDataType id
```

The second way is to search by ID. To search with IDs the parameter –inputDataType needs to be set to ‘id’.

The IDs can be from variouse databases. Uniprot is able to map (translate) a given ID to its own accession number. For this the database has to specified from which the data come from. The database can be set with the argument –idType. A list of all available databases can be found here.

Columns

With UniProt querys the information, which should be retrieved, can be controled with the –columns flag. The column names have to be seperated by a comma. All possible columns are listed on the [UniProt website] (https://www.uniprot.org/help/uniprotkb_column_names).

```
python3 proteinFinder.py inputFile outputFile
    uniprot --columns "id,keywords,genes,organism"
```

If not specified the default values are: * id * protein names * genes * existence * organism * ec * feature(METAL BINDING) * keywords * comment(PATHWAY) * comment(SUBCELLULAR LOCATION) * comment(DOMAIN) * families * sequence

Examples

The starting point will be the following table, which is a list of uniprot accesion numbers of proteins:

uniprot_id
A8LJX5
A8LJX6
A8LJX7
A8LJX8
A8LJX9
A8LJY0
A8LJY0
A8LJY1
A8LJY2
A8LJY3
A8LJY4
A8LJY5
A8LJY6

To extract the information from UniProt, the following command will be used:

```
python3 proteinFinder.py proteins.csv proteinInformation.csv uniprot -i id
```

The program requests information about the accesion numbers, which after success results in the following table:

Entry	Protein names	Gene names	…
A8LJX5	Uncharacterized protein	Dshi_0052	…
A8LJX6	HI0933 family protein	Dshi_0053	…
A8LJX7	Glutathione S-transferase like protein (EC 2.5.1.18)	gst2 Dshi_0054	…
A8LJX8	DNA polymerase III, delta subunit (EC 2.7.7.7)	holA Dshi_0055	…
A8LJX9	Uncharacterized protein	Dshi_0056	…
A8LJY0	Leucine–tRNA ligase (EC 6.1.1.4) (Leucyl-tRNA synthetase) (LeuRS)	leuS Dshi_0057	…
A8LJY0	Leucine–tRNA ligase (EC 6.1.1.4) (Leucyl-tRNA synthetase) (LeuRS)	leuS Dshi_0057	…

Entry	Protein names	Gene names	...
A8LJY1	Outer-membrane lipoprotein carrier protein	lolA Dshi_0058	...
A8LJY2	DNA translocase	ftsK Dshi_0059	...
A8LJY3	Aminotransferase class I and II	Dshi_0060	...
A8LJY4	Amidase	Dshi_0061	...
A8LJY5	2-octaprenyl-6-methoxyphenol hydroxylase (EC 1.14.13.-)	ubiH Dshi_0062	...
A8LJY6	Inositol-phosphate phosphatase (EC 3.1.3.25)	suhB2 Dshi_0063	...

With alternating the columns using the –columns flag the information requested will be changed:

```
python3 proteinFinder.py proteins.csv proteinInformation.csv
    uniprot -i id --columns "id,keywords,genes"
```

Entry	Keywords	Gene names
A8LJX5	Complete proteome;Reference proteome	Dshi_0052
A8LJX6	Complete proteome;Reference proteome	Dshi_0053
A8LJX7	Complete proteome;Reference proteome;Transferase	gst2 Dshi_0054
A8LJX8	Complete proteome;Nucleotidyltransferase;Reference proteome;Transferase	holA Dshi_0055
A8LJX9	Complete proteome;Reference proteome	Dshi_0056
A8LJY0	ATP-binding;Aminoacyl-tRNA synthetase;Complete proteome	leuS Dshi_0057
A8LJY0	ATP-binding;Aminoacyl-tRNA synthetase;Complete proteome	leuS Dshi_0057
A8LJY1	Chaperone;Complete proteome;Lipoprotein;Protein transport	lolA Dshi_0058
A8LJY2	ATP-binding;Cell cycle;Cell division;Cell membrane;Chromosome partition	ftsK Dshi_0059
A8LJY3	Aminotransferase;Complete proteome;Reference proteome;Transferase	Dshi_0060
A8LJY4	Complete proteome;Reference proteome	Dshi_0061
A8LJY5	Complete proteome;Oxidoreductase;Reference proteome	ubiH Dshi_0062
A8LJY6	Complete proteome;Hydrolase;Reference proteome	suhB2 Dshi_0063

Searching NCBI + Uniprot

To search both databases consecutive there is a shortcut.

```
python3 proteinFinder.py inputFile outputFile ncbi+uniprot
```

The results of the NCBI search will be the input of the UniProt search and the final results will be saved in the outputFile. The same flags are avialable for this shortcut as for the single commands.

```
python3 proteinFinder.py genes.csv proteinInformation.csv  ncbi+uniprot
    --inputDataType id --idType --database db --organism species
    --columns columns
```

More detailed information can be found in the documentation.