Chemistry Programming with Python – Retrieving InChI From PubChem (Tutorial)

Andrew P. Cornell[1], Robert E. Belford[1]

[1]Department of Chemistry, University of Arkansas at Little Rock, 2801 S. University Avenue, Little Rock, AR 72204, USA

Abstract

In this tutorial, a program written in Python will take a user specified chemical name and retrieve the associated chemical identifier or basic property using an online chemical database. This program can be used as both an aid for learning to programmatically work with chemical data and as a short general lesson for using Python with an API (Application Programming Interface). The database that will be used for this lesson is known as PubChem, which is a publicly accessible platform run by NCBI (National Center for Biotechnology Information).

PubChem offers public REST (Representational State Transfer) based programmatic access to a lot of the data contained on the servers which is defined with a specific syntax.[1] Review the recommended reading to become familiar with the syntax if the need arises to pull data from PubChem that differs from this tutorial. This tutorial will use the InChI (International Chemical Identifier), InChI Key, molecular formula, Canonical SMILES (Simplified Molecular-Input Line-Entry System) and molecular weight with InChI being used in the demonstration section.[2]

Learning Objectives

- Import Python Library
- Create and Define Functions
- Make API Request with Python
- Parse and Display Results

Recommended Reading

- Internet of Chemistry Things Activity 1 (https://ioct.tech/edu/ioct1) Page that explains basic Instructions for setting up Python on a computer. The Python Activities listed in the sidebar may also help to explain some of the background information.
- Spring 2017 ChemInformatics OLCC Course (http://olcc.ccce.divched.org/Spring2017OLCC) This site provides lots of information on working with chemical data.
- Python Documentation (https://docs.python.org/3/) Python 3 documentation that correlates to the version used within this tutorial.
- PubChem REST Documentation (https://pubchemdocs.ncbi.nlm.nih.gov/pug-rest) Provides instructions on syntax structure, methods and request procedures for accessing data.

Methods

The files used in this tutorial can be located within the following GitHub Page (https://github.com/boots7458/Cheminformatics-Python) along with a DOI on FigShare (https://doi.org/10.6084/m9.figshare.7255901).[3] Python will run on many different operating systems, however this tutorial will use the Thonny IDE (Integrated Development Environment) to design, run and test the code.[4] The following code performs the commands that will retrieve either an InChI, InChI Key, molecular formula, SMILES or the molecular weight of a compound from PubChem. Each code section will be explained in detail with the full completed code located at the end of the tutorial. The completed code can be copied directly into a Python File and run as a fully functional program.

Python 3 has been used for all code in this tutorial so make sure to consult the correct version documentation if additional reference is needed. Should the syntax or format change with future updates to the Python Language, it may be necessary to approach the task in a different way. The steps are broken down into sections which should be placed into the file one after the other from top to bottom.

Step 1

When you run a python file it typically does not load user added modules. These modules are stored in specific libraries and are only loaded when programs need those features. So the first step will be to load a user requested library, the urllib.request - Extensible library for opening URLs (https://docs.python.org/3/library/urllib.request.html), which will define several functions and classes to help the program open URLs. This is done through the import command as shown in the following code.

```
import urllib.request
```

Step 2

After making the library import declaration, it is time to start defining a few functions that will build the different parts of the program. The first function is called “firstChoice” and will declare the “string2” variable as global. This will allow the variable to be called from any function without having to specifically pass it within the code. The function will be responsible for asking the user to input a text string that will be stored in memory as a variable for later use.

```
def firstChoice():
    global string2
    string2 = input("Enter a chemical name: ")
```

Step 3

The second function is much longer, despite performing a simple task. The function is called “choices” and it will ask the user to pick a number for which option to retrieve based on what the program is asking. The numbers inside the parenthesis next to the print statements define some formatting that will be displayed such as indents, line dividers and spacing for aesthetic purposes. The most important section is where “idChoice” is set as a global variable to store the value chosen by the user. The “if” and “elif” commands define what to do, depending on whether the user selects a valid number from the options or not. A valid option will simply pass the choice to be used in constructing the API (Application Programming Interface) URL for retrieving the user’s request. An incorrect option will simply display a message stating the problem and to try again.

```python
def choices():
    print(40 * "_")
    print(3 * " " + "Select the value below to retrieve")
    print(40 * "_")
    print('{:>23}'.format("INCHI[0]"))
    print('{:>23}'.format("INCHIKEY[1]"))
    print('{:>23}'.format("MOLECULAR FORMULA[2]"))
    print('{:>23}'.format("SMILES[3]"))
    print('{:>23}'.format("MOLECULAR WEIGHT[4]"))
    print(40 * "_")
    global idChoice
    idChoice = int(input("Enter a number choice? "))
    if 0 <= idChoice <= 4:
        choiceID()
    elif idChoice != range(0,4):
        print(2 * '\n' + 38 * '*')
        print("* Incorrect Number Choice, Try Again *")
        print(38 * '*' + 2 * '\n')
        choices()
```

Step 4

The next function in the program is called “choiceID” and this will take an array of values for the input and combine what the user has chosen into a string. The string will then be used for building the URL that will pull in the requested data value at the end. This is the function that will utilize the library from step 1 that was declared at the top of the file. The last part of the function will ask the user if another value should be requested or not. The program will end if the user types in “no” and will repeat the entire program if “yes” is entered as the user’s choice.

```
 def choiceID():
    inputList = ['INCHI', 'INCHIKEY', 'MolecularFormula','CanonicalSMILES',
'MolecularWeight']
    selChoice = inputList[idChoice]

    string1 = "https://pubchem.ncbi.nlm.nih.gov/rest/pug/compound/name/"
    string3 = "/property/"
    string4 = "/TXT"
    html = urllib.request.urlopen(string1 + string2 + string3 + selChoice +
string4).read()
    html2 = html.decode('UTF-8')
    print(2 * '\n')
    print(html2)
    print(2 * '\n')
    redoProg = input("Would you like to check another compound, yes or no? ")
    if redoProg == "yes":
        print ("\n" * 2)
        main()
    else:
        print("Done!")
```

Step 5

This function will be called “main” and it simply defines the order in which all other functions should be run. The program will run this first even though it is located towards the bottom of the Python File.

```
def main():
    firstChoice()
    choices()
```

Step 6

Here is where the program is told which function to start with and it says to start with the function titled “main” which will define the order of which individual functions to run. This line of code is the starting point at which the program will read from after all libraries have been loaded. It is very important that this line of code take place after the functions along with being loaded after “Import Time” so that libraries are loaded before any code is executed. For this reason “main()” is written last in the program.

```
## Program Assignments ##
main()
```

Completed Code Example

```
import urllib.request

def firstChoice():
    global string2
    string2 = input("Enter a chemical name: ")

def choices():
    print(40 * "_")
    print(3 * " " + "Select the value below to retrieve")
    print(40 * "_")
    print('{:>23}'.format("INCHI[0]"))
    print('{:>23}'.format("INCHIKEY[1]"))
    print('{:>23}'.format("MOLECULAR FORMULA[2]"))
    print('{:>23}'.format("SMILES[3]"))
    print('{:>23}'.format("MOLECULAR WEIGHT[4]"))
    print(40 * "_")
    global idChoice
    idChoice = int(input("Enter a number choice? "))
    if 0 <= idChoice <= 4:
        choiceID()
    elif idChoice != range(0,4):
        print(2 * '\n' + 38 * '*')
        print("* Incorrect Number Choice, Try Again *")
        print(38 * '*' + 2 * '\n')
        choices()

def choiceID():
    inputList = ['INCHI', 'INCHIKEY', 'MolecularFormula','CanonicalSMILES',
'MolecularWeight']
    selChoice = inputList[idChoice]

    string1 = "https://pubchem.ncbi.nlm.nih.gov/rest/pug/compound/name/"
    string3 = "/property/"
    string4 = "/TXT"
    html = urllib.request.urlopen(string1 + string2 + string3 + selChoice +
string4).read()
    html2 = html.decode('UTF-8')
    print(2 * '\n')
    print(html2)
    print(2 * '\n')
    redoProg = input("Would you like to check another compound, yes or no? ")
    if redoProg == "yes":
        print ("\n" * 2)
        main()
    else:
        print("Done!")

def main():
    firstChoice()
    choices()

## Program Assignments ##
```

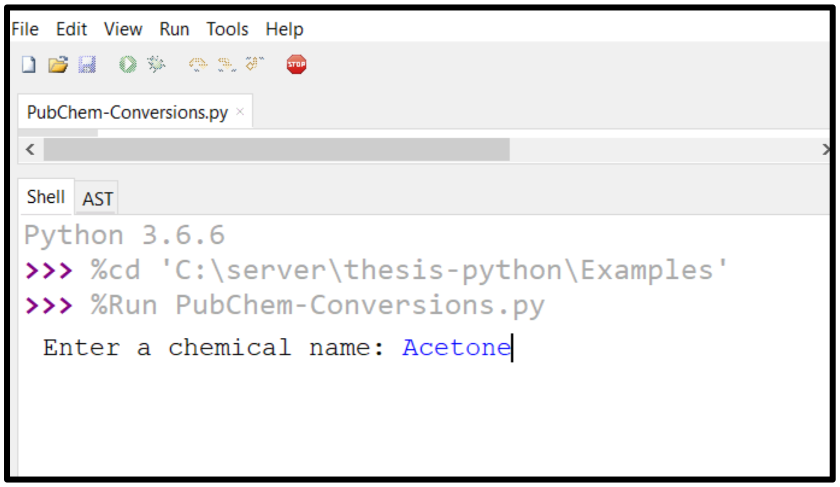
```
main()
```

NOTES ON USING THE PROGRAM

This program will take a single chemical name without any modification needed. For example, if the user wants to look for data related to acetone, then the term can be entered as just “acetone”. However, if a user wants to enter a chemical name that contains more than one word, then a place marker must be put in so that the spaces can be accounted for in the URL Syntax. To enter the term acetic acid, the user should enter “acetic%20acid” by putting %20 anywhere that should contain a space, or the program will throw several errors as a result.

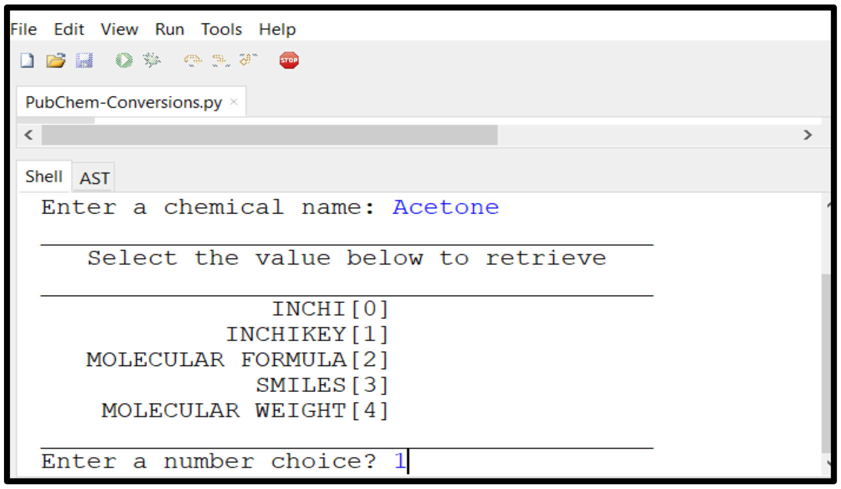
Program Demonstration

An interactive demo of this program is provided by Trinket in the online publication.[5] The trinket can be visited at this location (https://trinket.io/python3/c845b6bdbd). The stored and printable copies will only contain screenshots below with descriptions as they cannot display the trinket program in a live environment.



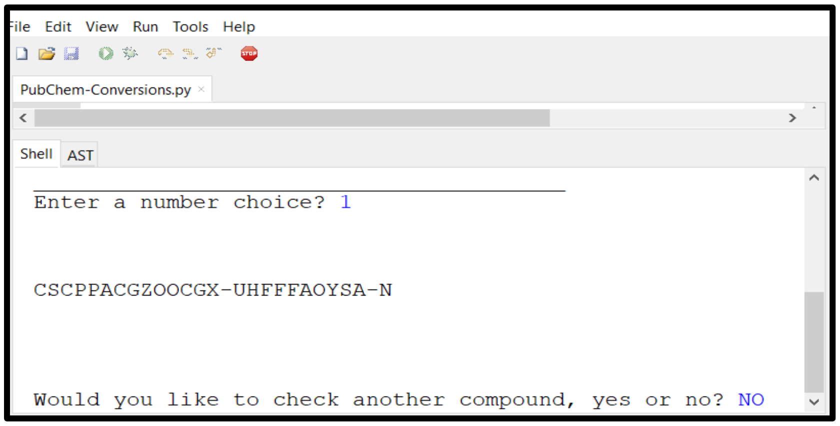
```
Python 3.6.6
>>> %cd 'C:\server\thesis-python\Examples'
>>> %Run PubChem-Conversions.py
 Enter a chemical name: Acetone
```

Figure 1 Shown above, "Acetone" has been inserted into the text input.



```
Enter a chemical name: Acetone
_________________________________________
   Select the value below to retrieve
_________________________________________
               INCHI[0]
            INCHIKEY[1]
   MOLECULAR FORMULA[2]
              SMILES[3]
    MOLECULAR WEIGHT[4]
_________________________________________
Enter a number choice? 1
```

Figure 2 Shown above, option 1 for INCHIKEY has been selected



```
_________________________________________
Enter a number choice? 1



CSCPPACGZOOCGX-UHFFFAOYSA-N




Would you like to check another compound, yes or no? NO
```

Figure 3 Shown above is the InChI Key Result from the PubChem API.

Suggested Questions for Classroom Use: Type answers in the boxes that follow each question

1. What is a library in python, and why would python use libraries?

2. What library was used in the above code, and what does it do?

3. What command allows you to make a function in python?

4. What are the names of the functions the above program created?

5. What does the command "global" do? Why is it needed?

6. What is the URL that is generated to get the molar mass of aspirin?

7. What does the code '{:>23}' do?

8. For the following code

```
idChoice = int(input("Enter a number choice? "))
    if 0 <= idChoice <= 4:
        choiceID()
    elif idChoice != range(0,4):
        print(2 * '\n' + 38 * '*')
        print("* Incorrect Number Choice, Try Again *")
        print(38 * '*' + 2 * '\n')
        choices()
```

(a) What does the following statements say?

```
if 0 <= idChoice <= 4:
    choiceID()
```

(b) What does the following statements say? Can you think of another way of doing this?

```
idChoice != range(0,4):
```

9. Go to the Compound Property Tables on this page
https://pubchemdocs.ncbi.nlm.nih.gov/pug-rest$_Toc494865567

Add a 6th option (option 5) that would allow you to get the IUPAC Name for a compound
Add a 7th option (option 6) that would give you an indication if the compound is water or fat soluble.

Save the revised code as a python file and submit that with your answers.

References

(1) Kim, S.; Thiessen, P. A.; Cheng, T.; Yu, B.; Bolton, E. E. An Update on PUG-REST: RESTful Interface for Programmatic Access to PubChem. *Nucleic Acids Research* **2018**, *46* (W1), W563–W570. https://doi.org/10.1093/nar/gky294.

(2) Heller, S. R.; McNaught, A.; Pletnev, I.; Stein, S.; Tchekhovskoi, D. InChI, the IUPAC International Chemical Identifier. *Journal of Cheminformatics* **2015**, *7* (1). https://doi.org/10.1186/s13321-015-0068-4.

(3) Cornell, A. Cheminformatics-Python. Figshare 2018. https://doi.org/10.6084/m9.figshare.7255901.

(4) Annamaa, A. Introducing Thonny, a Python IDE for Learning Programming. In *Proceedings of the 15th Koli Calling Conference on Computing Education Research - Koli Calling '15*; ACM Press: Koli, Finland, 2015; pp 117–121. https://doi.org/10.1145/2828959.2828969.

(5) Elliott Hauser; Brian Marks; Ben Wheeler. *Trinket*; 2019.