**Using Bash and Python wrap together a data processing pipeline**

There are a few considerations to be made whenever one uses software to process data.

1. What other software is required to run the program?
2. How is data input into the program?
3. How is data output from the program?

Python especially under the anaconda framework provides many of the tools necessary to solve these considerations.

**Anaconda provides a framework for distributing environments**

Using a .yml file and anaconda one can distribute different environments for running in python 2 or 3. See the link bellow to learn more about distributing software using anaconda.

<https://conda.io/docs/user-guide/tasks/manage-environments.html>

Essentially through a .yml file one can automatically distribute all the packages and programs needed to run a bioinformatics software pipeline

**Python provides a framework through which one can modify data files and submit them to software programs**

Certain libraries can be helpful in accessing files and reformatting them.

The sys library allows you to access system commands and select files of interest.

<https://docs.python.org/2/library/sys.html>

For example, the following script allows one to read through a test file.

import sys

file = sys.argv[1]

with open(file) as f:

for line in f:

tmp = line.rstrip()

print tmp

This script can be run as so:

Python script.py file.txt > outfile.txt

Modifications on the above scripts allow one to store your file in a data structure so that you can manipulate it. There are three main types of data structures in python that can help with this

1. Lists: denoted as [] in python. These are most easily described as typical arrays from other programing languages, but with some flexibility. Lists are not data type specific.
   1. Lists can be multidimensional: List = [[],[],[],] 2D, List = [[[],[],],[[],[]]] 3D
2. Sets: denoted as set() in python. These are most easily described as lists where non of the entries can be duplicated.
3. Dictionaries: denoted as {} in python. These are a very powerful tool. Dictionaries are indexed by keys, which can only appear once. Keys can be a string or number. Hint: Amino Acid sequences can be treated as strings and can be the key to your dictionary. These strings can be used to point to another data structure which can be a number, string, list, set, or even another dictionary!
   1. For example: dict[myindexnumber] = [[],[]]

Using most of the above string we can load a specific type of file like at tab delimited file into a data structure for future manipulation.

import sys

file = sys.argv[1]

NumberOfColumns = sys.argv[2]

FileInList = []

for i in range(0,NumberOfColumns):

FileInList.append([])

with open(file) as f:

for line in f:

tmp = line.rstrip()

tmp2 = tmp.split(“\t”)

for i in range(0, len(FileInList)):

FileInList[i].append(tmp2[i])

Run as

Python script.py filename.txt “Number of Columns”

For more information about these tools see the python documentation

<https://www.python.org/doc/>

**Selecting A Protein Database processing of Mass Spectrometry raw data for proteomics**

Untargeted mass spectrometry based proteomics is usually done bottom-up where proteins are digested into peptides using a protease with a defined digestion pattern, for example Trypsin digests proteins after lysine and arginine. Using the total mass of an isolated peptide the sequence of the peptide can then be determined by fragmenting the isolated peptide