Lab - Basic Data Structures

Lists

Exercise 1

Given the list, **Celsius**, use list comprehension to create a new list, **Fahrenheit**, which contains converts $^\circ C$ to $^\circ F$. Hint: the conversion equation is: T * 9/5 + 32.

```
Celsius = [39.2, 36.5, 37.3, 37.8]

Fahrenheit = [temperature * 9/5 + 32 for temperature in Celsius]
print(Fahrenheit)
```

```
[102.56, 97.7, 99.14, 100.03999999999999]
```

Exercise 2

Iterate through your Fahrenheit list to count the number of temperatures above the average temperature. Hint: use the numpy **.mean()** function to calculate the function. Use the following pseudo code as an example.

```
set counter for temperatures > average to 0
for each temperature in Fahrenheit list:
    check if temperature is > average:
        if true, then increment counter

print counter
```

```
import numpy as np

counter = 0
for temperature in Fahrenheit:
    if temperature > np.mean(Fahrenheit):
        counter += 1

print(counter)
```

```
2
```

Dictionaries

Exercise 1

Given the sentence, **woodchuck**, store each word as a key in a dictionary and the number of times

that word appears as the key's value. Then print how many times each word occurs.

Hint: See the pseudo code and use the method .get().

```
create empty dictionary
for each word in sentence:
    set each word as a key and set the number of occurances as its value

for each word in dictionary:
    print the value of each key
```

```
woodchuck = "How much wood could a woodchuck chuck if a woodchuck could chuck wood?
As much wood as a woodchuck could chuck if a woodchuck could chuck wood.".split()
```

```
words = {}
for word in woodchuck:
    words[word] = words.get(word, 0) + 1

for word in words:
    print("The word", word, "occurs", words[word], "times in the woodchuck riddle")
```

```
('The word', 'a', 'occurs', 4, 'times in the woodchuck riddle')
('The word', 'wood', 'occurs', 2, 'times in the woodchuck riddle')
('The word', 'could', 'occurs', 4, 'times in the woodchuck riddle')
('The word', 'chuck', 'occurs', 4, 'times in the woodchuck riddle')
('The word', 'wood.', 'occurs', 1, 'times in the woodchuck riddle')
('The word', 'How', 'occurs', 1, 'times in the woodchuck riddle')
('The word', 'As', 'occurs', 1, 'times in the woodchuck riddle')
('The word', 'much', 'occurs', 2, 'times in the woodchuck riddle')
('The word', 'woodchuck', 'occurs', 4, 'times in the woodchuck riddle')
('The word', 'wood?', 'occurs', 1, 'times in the woodchuck riddle')
('The word', 'as', 'occurs', 1, 'times in the woodchuck riddle')
('The word', 'if', 'occurs', 2, 'times in the woodchuck riddle')
```

Exercise 2

Remember that I said a dictionary is used for FITS header information? Well, let's have some dictionary fun with NIRSPEC data from the Keck Observatory.

```
# Read the FITS header. We will cover this in more detail in lession 2.
from astropy.io import fits
fitsFile = 'NS.20100429.40280.fits'
header = fits.getheader(fitsFile)
```

Use a for loop to print all of the header keywords

```
for keyword in header:
    print(keyword)
```

```
SIMPLE
BITPIX
NAXIS
NAXIS1
NAXIS2
BSCALE
BZERO
TELESCOP
OBSERVER
OBJECT
COMMENT
ROOTNAME
FILENUM
FILENAME
OUTDIR
SCRIPTNA
FILNAME
SLITNAME
SLITPA
SLITX
SLITY
SLITANG
SCAMPA
ITIME
COADDS
SAMPMODE
DETBIAS
MULTISPE
SAMPRATE
Q1OFFSET
Q2OFFSET
Q3OFFSET
Q4OFFSET
GAIN.SPE
FREQ.SPE
CRYOTEMP
FIL1POS
FIL2POS
SLITPOS
ECHLPOS
DISPPOS
IROTLOC
CALMPOS
CALCPOS
CALPPOS
NEON
ARGON
KRYPTON
XENON
ETALON
FLAT
AIRMASS
AXESTAT
AZ
CALOCAL
```

```
CELOCAL
CURRINST
DATE-OBS
DCSSTAT
DCSVERS
DEC
DOMEPOSN
DOMESTAT
EL
EQUINOX
FOCALSTN
HA
LST
MJD-OBS
PARANG
PARANTEL
DRA
DDEC
DTRACK
PONAME
POXOFF
POYOFF
POXPOS
POYPOS
PONAME1
POXPOS1
POYPOS1
PONAME2
POXPOS2
POYPOS2
PONAME3
POXPOS3
POYPOS3
PONAME4
POXPOS4
POYPOS4
PONAME5
POXPOS5
POYPOS5
PONAME6
POXPOS6
POYPOS6
PONAME7
POXPOS7
POYPOS7
PONAME8
POXPOS8
POYPOS8
PONAME9
RA
ROTCALAN
ROTMODE
ROTPPOSN
ROTPOSN
ROTREFAN
SECFOCUS
SECTHETX
```

```
SECTHETY
TARGDEC
TARGEPOC
TARGEQUI
TARGFRAM
TARGNAME
TARGPLAX
TARGPMDC
TARGPMRA
TARGRA
TARGRADV
TELESCOP
TELFOCUS
TUBETEMP
INSTRUME
COMMENT
COMMENT
COMMENT
UTC
UT
ELAPTIME
FRAMENO
COADD
BLANK
CRASH
KOAID
IMAGETYP
DATLEVEL
DQA_DATE
DQA_VERS
DETMODE
DETGAIN
DETRN
DISPERS
DISPSCAL
GUIDFWHM
GUIDTIME
IMAGEMD
IMAGEMN
IMAGESD
INSTSTAT
ISAO
NLINEAR
NPIXSAT
OA
PROGID
PROGINST
PROGPI
PROGTL1
PROGTL2
PROGTL3
SEMESTER
SIG2NOIS
SLITLEN
SLITWIDT
SPATSCAL
SPECRES
```

```
WAVEBLUE
WAVECNTR
WAVERED
WXDOMHUM
WXDOMTMP
WXDWPT
WXOUTHUM
WXOUTTMP
WXPRESS
WXTIME
WXWNDIR
WXWNDSP
```

Print the value of your favorite keyword

```
header['TARGNAME']
```

```
'511_Davida 11UT'
```

Change the value of your favorite keyword

```
header['TARGNAME'] = '511 Davida'
header['TARGNAME']
```

```
'511 Davida'
```

For learning reinforcement purposes, use list comprehension to create a list of all header keywords

```
keywords = [keyword for keyword in header]
keywords
```

```
['SIMPLE',
 'BITPIX',
 'NAXIS',
 'NAXIS1',
 'NAXIS2',
 'BSCALE',
 'BZERO',
 'TELESCOP',
 'OBSERVER',
 'OBJECT',
 'COMMENT',
 'ROOTNAME',
 'FILENUM',
 'FILENAME',
 'OUTDIR',
 'SCRIPTNA',
```

```
  'FILNAME',
  'SLITNAME',
  'SLITPA',
  'SLITX',
  'SLITY',
  'SLITANG',
  'SCAMPA',
  'ITIME',
  'COADDS',
  'SAMPMODE',
  'DETBIAS',
  'MULTISPE',
  'SAMPRATE',
  'Q1OFFSET',
  'Q2OFFSET',
  'Q3OFFSET',
  'Q4OFFSET',
  'GAIN.SPE',
  'FREQ.SPE',
  'CRYOTEMP',
  'FIL1POS',
  'FIL2POS',
  'SLITPOS',
  'ECHLPOS',
  'DISPPOS',
  'IROTLOC',
  'CALMPOS',
  'CALCPOS',
  'CALPPOS',
  'NEON',
  'ARGON',
  'KRYPTON',
  'XENON',
  'ETALON',
  'FLAT',
  'AIRMASS',
  'AXESTAT',
  'AZ',
  'CALOCAL',
  'CELOCAL',
  'CURRINST',
  'DATE-OBS',
  'DCSSTAT',
  'DCSVERS',
  'DEC',
  'DOMEPOSN',
  'DOMESTAT',
  'EL',
  'EQUINOX',
  'FOCALSTN',
  'HA',
  'LST',
  'MJD-OBS',
  'PARANG',
  'PARANTEL',
  'DRA',
  'DDEC',
```

```
  'DTRACK',
  'PONAME',
  'POXOFF',
  'POYOFF',
  'POXPOS',
  'POYPOS',
  'PONAME1',
  'POXPOS1',
  'POYPOS1',
  'PONAME2',
  'POXPOS2',
  'POYPOS2',
  'PONAME3',
  'POXPOS3',
  'POYPOS3',
  'PONAME4',
  'POXPOS4',
  'POYPOS4',
  'PONAME5',
  'POXPOS5',
  'POYPOS5',
  'PONAME6',
  'POXPOS6',
  'POYPOS6',
  'PONAME7',
  'POXPOS7',
  'POYPOS7',
  'PONAME8',
  'POXPOS8',
  'POYPOS8',
  'PONAME9',
  'RA',
  'ROTCALAN',
  'ROTMODE',
  'ROTPPOSN',
  'ROTPOSN',
  'ROTREFAN',
  'SECFOCUS',
  'SECTHETX',
  'SECTHETY',
  'TARGDEC',
  'TARGEPOC',
  'TARGEQUI',
  'TARGFRAM',
  'TARGNAME',
  'TARGPLAX',
  'TARGPMDC',
  'TARGPMRA',
  'TARGRA',
  'TARGRADV',
  'TELESCOP',
  'TELFOCUS',
  'TUBETEMP',
  'INSTRUME',
  'COMMENT',
  'COMMENT',
  'COMMENT',
```

```
 'UTC',
 'UT',
 'ELAPTIME',
 'FRAMENO',
 'COADD',
 'BLANK',
 'CRASH',
 'KOAID',
 'IMAGETYP',
 'DATLEVEL',
 'DQA_DATE',
 'DQA_VERS',
 'DETMODE',
 'DETGAIN',
 'DETRN',
 'DISPERS',
 'DISPSCAL',
 'GUIDFWHM',
 'GUIDTIME',
 'IMAGEMD',
 'IMAGEMN',
 'IMAGESD',
 'INSTSTAT',
 'ISAO',
 'NLINEAR',
 'NPIXSAT',
 'OA',
 'PROGID',
 'PROGINST',
 'PROGPI',
 'PROGTL1',
 'PROGTL2',
 'PROGTL3',
 'SEMESTER',
 'SIG2NOIS',
 'SLITLEN',
 'SLITWIDT',
 'SPATSCAL',
 'SPECRES',
 'WAVEBLUE',
 'WAVECNTR',
 'WAVERED',
 'WXDOMHUM',
 'WXDOMTMP',
 'WXDWPT',
 'WXOUTHUM',
 'WXOUTTMP',
 'WXPRESS',
 'WXTIME',
 'WXWNDIR',
 'WXWNDSP']
```

Create a new header keyword and value then verify that it exists

```
header['foo'] = 'bar'
```

```
header['foo']
```

```
'bar'
```

NumPy Arrays

Exercise 1

Let's create some basic arrays and manupulate them.

First, create a 1d array with 10 elements and store it in a variable

```
x = np.array(range(10))
x
```

```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

Reshape the original array so it has 2 rows an 5 columns

```
x.reshape(2, 5)
```

```
array([[0, 1, 2, 3, 4],
       [5, 6, 7, 8, 9]])
```

Reshape the original array so it has 5 rows and 2 colums

```
x.reshape(5,2)
```

```
array([[0, 1],
       [2, 3],
       [4, 5],
       [6, 7],
       [8, 9]])
```

Exercise 2

Now that we have NIRSPEC data, let's play with it! For now we will just do some basic operations but will do more science with data in later lessons.

```
# Again, we will cover reading FITS data in more detail in lesson 2
data = fits.getdata(fitsFile)
```

```
# See, the data is just a NumPy array. Not so scary.
type(data)
```

```
numpy.ndarray
```

Print the array shape and verify that it matches the values for the NAXIS1 and NAXIS2 headers.

```
print(data.shape)
print(header['NAXIS1'], header['NAXIS2'])

print(data.shape[0] == header['NAXIS1'])
print(data.shape[1] == header['NAXIS2'])
```

```
(1024, 1024)
(1024, 1024)
True
True
```

Print the array size and verify that it matches the product of NAXIS1 and NAXIS2

```
print(data.size)
dataSize = header['NAXIS1'] * header['NAXIS2']

print(data.size == dataSize)
```

```
1048576
True
```

Find the index of the maximum value. **Hint:** run the next code cell to see help on .argmax()

```
np.argmax?
```

```
np.argmax(data)
```

```
15321
```

```
np.argmax(data, axis=0)
```

```
array([ 201,  136,  693, ...,  355, 1008, 1023])
```

```
np.argmax(data, axis=1)
```

```
array([   0,   12,  325, ...,  513,   10, 1023])
```