Python Basics

Data Pipeline
Jeopardy Problem
myfile = open( 'jeopardy.csv' )  # must be in same directory
lines = myfile.readlines()
d = {}  # accumulator
for line in lines:
    cat = line.split( ',', )[ 3 ]  # retrieve the field at index 3
    if cat in d:  # category in d, so increment it
        d[ cat ] += 1
    else:  # category not yet in d, so add it
        d[ cat ] = 1
def sortDictAsList( d ):
    items = list( d.items() )
    items.sort( key=lambda x:x[1] )
    return items

This is MAGIC. Don’t worry AT ALL about understanding it in 101.

d = { 'a':2, 'b':1, 'c':-1, 'd':14 }
sortDictAsList( d )
Indexing Redux
a = [
  [1,2,3],
  [4,5,6],
  [7,8,9]
]

How would you refer to the value 8?
A  a[2][3]
B  a[1][2]
C  a[2,3]
D  a[2][1]
a = [ [1,2,3],
      [4,5,6],
      [7,8,9] ]

How would you refer to the value 8?
A  a[2][3]
B  a[1][2]
C  a[2,3]
D  a[2][1] ✭
a = [ [1,2,3],
      [4,5,6],
      [7,8,9] ]

How would you refer to the value 8?

A  a[2][3]
B  a[1][2]
C  a[2,3]
D  a[2][1]  ✯Grab the first thing, then the second.
Recap: File I/O
myfile = open( 'odyssey.txt' )
text = myfile.read()
for l in text.split():
    c = text.count( l )
    print( l,c )

What does this code do?
A  Counts all of the lines in 'odyssey.txt'.
B  Counts all of the words in 'odyssey.txt'.
C  Counts all of the characters in 'odyssey.txt'.
myfile = open( 'odyssey.txt' )
text = myfile.read()
for l in text.split():
    c = text.count( l )
    print( l, c )

- What is type of text?
- What is type of text.split()?
- What is type of l?
- What is contents of c?
myfile = open( 'odyssey.txt' )

for l in text.split():
    c = text.count( l )
    print( l, c )

What does this code do?
A  Counts all of the lines in ’odyssey.txt’.
B  Counts all of the words in ’odyssey.txt’.*
C  Counts all of the characters in ’odyssey.txt’.

How can we improve this?
myfile = open( 'odyssey.txt' )
text = myfile.read()
for l in text.split():
    c = text.count( l )
    print( l,c )

What does this code do?
A  Counts all of the lines in ’odyssey.txt’.
B  Counts all of the words in ’odyssey.txt’. ★
C  Counts all of the characters in ’odyssey.txt’.

How can we improve this? (exclude punctuation, make all lower-case, close the file)
myfile = open( 'words.txt' )  #<= string!
for line in myfile:
    print( line.title() )
myfile.close()  # process responsibly
myfile = open( 'words.txt' )
data = myfile.read()
myfile.close()

for line in data.split():
    print( line.title() )
```
mymfile = open( 'words.txt' )
data = myfile.readlines()
myfile.close()

for line in data():
    print( line.title() )
```
File modes

- 'r'—read a file
- 'w'—write to a file
- 'a'—append to a file
- 'rb'—read a binary file
- 'wb'—write to a binary file

```python
myfile = open( 'words.txt','w' )
myfile.write( 'Hello, this is a test.' )
myfile.close() # important now!
```
Every program tells a story.

- Beginning = Input = Preprocessing
- Middle = Analysis = Processing
- End = Output = Postprocessing

A good way to write a program is to make this explicit!

Everything else we do in this class will follow this pattern.
This structure applies at every level.

- expressions
- statements
- blocks
- programs

This is one reason why return type is so critical!
Input Sources
The user:
input

The hard drive:
open (files)
plain text files
comma-separated value files (csv)

The Internet:
requests
Input sources

- The user: input
- The hard drive: open (files)
  - plain text files
  - comma-separated value files (csv)
- The Internet: requests
Review: User input

- input:
  - accepts as argument a message
  - *blocks* (pauses) for the user
  - returns a string
open:
  - accepts as argument a file name
  - returns a file data type

file has four useful methods:
  - read—returns a string
  -.readlines—returns a list
  - write
  - close
Files/csv

- csv files look like spreadsheets with columns separated by commas.

Year,Make,Model,Price
2007,Chevrolet,Camaro,5000.00
2010,Ford,F150,8000.00
Given a field report on plankton populations, determine the largest plankton and the most common (at any location and during any season).
csv files look like spreadsheets with columns separated by commas.

<table>
<thead>
<tr>
<th>Year</th>
<th>Make</th>
<th>Model</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>2007</td>
<td>Chevrolet</td>
<td>Camaro</td>
<td>5000.00</td>
</tr>
<tr>
<td>2010</td>
<td>Ford</td>
<td>F150</td>
<td>8000.00</td>
</tr>
</tbody>
</table>

There are two ways to interpret them:

- tokenize (split) the line into components
- use the csv.DictReader tool to access components

Commas can mess things up for split!
# assuming that we have a file autos.csv
myfile = open( 'autos.csv' )
rows = myfile.readlines()
for row in rows:
    print( row[ 0 ], row[ 1 ] )
# assuming that we have a file autos.csv
from csv import DictReader
reader = DictReader( open( 'autos.csv' ) )
for row in reader:
    print( row[ 'Make' ], row[ 'Price' ] )

- So how would our plankton.csv example look?
requests is a module to access server-based resources
  This is a complex process!
  get returns a Response data type (but you don’t need to know this)
  The ONLY thing you need is the text attribute (NOT method).
The text attribute is a str.

But websites are HTML!
- We will only access plain-text resources.
- HTML requires parsing, which we won’t cover.
- Another possible approach is to inspect the page for structure.
import requests
url = 'http://www.nws.noaa.gov/mdl/gfslamp/lavlamp.shtml'
website = requests.get( url )
offset = website.text.find( 'KCMI' )+169
temperature_string = website.text[ offset:offset+3 ]
temperature = float( temperature_string )

(See? This is possible but gets messy.)
This code should produce a list containing the comma-separated numbers at the URL. What should replace the `???

A `text.split(‘,’)`
B `page.text.split(‘,’)`
C `text().split(‘,’)`
D `page.text().split(‘,’)`
import requests
page = requests.get('mydataurl.com/data')
data = page.text.split(',,')

This code should produce a list containing the comma-separated numbers at the URL. What should replace the ????

A text.split(',,')
B page.text.split(',,') ★
C text().split(',,')
D page.text().split(',,')