Lecture 12: Comma Separated Values (CSV) Format

CSV

CSV data representing financial information from Apple Computer. This data might appear in a file called `aapl.csv`.

```
Date,Open,High,Low,Close,Volume,Adj Close
2009-12-31,213.13,213.35,210.56,210.73,88102700,28.40
2009-12-30,208.83,212.00,208.31,211.64,103021100,28.52
2009-12-29,212.63,212.72,208.73,209.10,111301400,28.18
2009-12-28,211.72,213.95,209.61,211.61,161141400,28.51
```

```python
import csv

with open('aapl.csv', 'r') as fin:
    print(list(csv.reader(fin)))
```

Highest Stock Price

Find the highest stock price

```
import csv
HIGHPRICECOL = 2

with open('aapl.csv', 'r') as fin:
    data = list(csv.reader(fin))
    prices = [float(row[HIGHPRICECOL]) for row in data[1:]]
    print(max(prices))
```

Highest Stock Price: Streaming

Find the highest stock price

```
import csv
COLNAME = "High"

with open('aapl.csv', 'r') as fin:
    maxprice = float('-inf')
    maxpricecol = None

    for rownum, row in enumerate(csv.reader(fin)):
        if rownum == 0:
            maxpricecol = row.index(COLNAME)
        else:
            maxprice = max(maxprice, float(row[maxpricecol]))

    print(maxprice)
```

Practice

Imagine you have a file called `realestate.csv` that contains real estate transactions in Sacramento over 5 days. The format of the data is

$$street, city, zip, state, beds, baths, sqft, type, sale\_date, price, lat, long.$$

Write a short script that finds that average sale price for every transaction in the file. You know that price occurs at index 9, that the `statistics.mean` function is available, and that the data can easily fit into memory.

Solution

street,city,zip,state,beds,baths,sqft,type,sale_date,price,lat,long.

```
import sys
import csv
import statistics
PRICECOL = 9

def mean_sale(filename):
    with open(filename, 'r') as fin:
        rows = list(csv.reader(fin))[1:]
        prices = [float(row[PRICECOL]) for row in rows]
        return(statistics.mean(prices))

if __name__ == '__main__':
    print(mean_sale(sys.argv[1]))
```

Practice

Write a function that returns the mean sale price of houses over 2000 square feet. Square footage is given by the column at index 6.

$$street, city, zip, state, beds, baths, sqft, type, sale\_date, price, lat, long.$$

Practice

```
import sys
import csv
import statistics

PRICECOL = 9
SQFTCOL = 6
SQFTMIN = 2000

def mean_sale_high(filename):
    with open(filename, 'r') as fin:
        rows = list(csv.reader(fin))[1:]
        prices = []
        for row in rows:
            if int(row[SQFTCOL]) > SQFTMIN:
                prices.append(float(row[PRICECOL]))
        return(statistics.mean(prices))

if __name__ == '__main__':
    print(mean_sale(sys.argv[1]))
```

io.StringIO

Suppose that the CSV data, however, is in a string `data`, instead of a file. In this case, one would use the `io.StringIO` type to wrap the string inside something that behaves like a *file object*. You can think of this as *buffering* the string.

```
data = 'purple,cow,moo\nhappy,moose,grunt'
reader = csv.reader(io.StringIO(data))
for row in reader:
    print("*".join(row))
```

Some CSV Reader Options

`delimiter`	A chracter used to separate fields. It defaults to ','.
`escapechar`	On reading, the `escapechar` removes any special meaning from the subsequent character. It defaults to `None`, which disables escaping.

CSV Writer

```
[['Williams', 'Ephs', 'Purple Cows'],
 ['Middlebury', 'Panthers', 'Panther']]
```

To write this to the file called `nescac.csv` we would use the following code

```
import csv
with open('nescac.csv', 'w', newline='') as csvfile:
    writer = csv.writer(csvfile, delimiter=',')
    writer.writerow(['School', 'Nickname', 'Mascot'])
    writer.writerows(data)
```

Practice Problem

Suppose you had a list of constellations and their galactic coordinates (right ascension and declination) in CSV format.

```
constellation, right ascension, declination
Sagittarius,19,-25
Taurus, 4.9, 19
Perseus, 3, 45
```

Write a function that takes a file in CSV format and returns a list of constellations. Suppose that you know one of the headers is labelled `constellation`, but not which one. Suppose further that you can easily fit all the data in memory.

Practice Problem

Suppose you had a list of constellations and their galactic coordinates (right ascension and declination) in CSV format.

```
constellation, right ascension, declination
Sagittarius,19,-25
Taurus, 4.9, 19
Perseus, 3, 45
```

Write a function that takes a file in CSV format and returns a list of constellations. Suppose that you know one of the headers is labelled `constellation`, but not which one. Suppose further that you can easily fit all the data in memory.

```
with open(file, newline='') as fp:
  data = [row for row in csv.reader(file)]
  col = data[0].index('constellation')
  return [row[col] for row in data[1:]]
```