ECS 10

2/25

Announcements

- Lab hours:
  - Today: 11-1
  - Tomorrow: 12-2
- Mac users and anyone else who wants to:

```
open("drinkingWater.csv","r",encoding="latin-1")
```

- Friday in class - MIDTERM 2
- Sample midterm on Web site
- Bring Scantron 2000 form, book, notes, programs…

Using the dictionary

- Make it:

```
twDict = {}
```

- Add an entry:

```
twDict[tweeter] = 0
```

- Check before you get an entry:

```
if tweeter in twDict:
```

- Change an entry:

```
twDict[tweeter] = twDict[tweeter]+1
```

For loop on a dictionary

```
for x in M:
```

- If M is a dictionary, then x is each key in turn

```
for tweeter in twDict:
```

- **tweeter** will be each twitter handle that is a key in the dictionary (in pseudo-random order)

Sorting

- The sorted function produces a sorted list.

```
>>> L = [6,2,8,4,1,7]
>>> sorted(L)
[1, 2, 4, 6, 7, 8]
>>> sorted(L,reverse=True)
[8, 7, 6, 4, 2, 1]
```

- On a dictionary, the result is a sorted list of the keys. Not what we wanted….

List from a dictionary

- The list() function turns something into a list.
- On a dictionary…

```
>>> list(D)
['hay', 'corn', 'milo', 'oats']
>>> list(D.values())
[30, 5, 25, 15]
>>> list(D.items())
[('hay', 30), ('corn', 5), ('milo', 25), ('oats', 15)]
```

List of tuples

- Each item in the list is itself a tuple

```
>>> L
[('hay', 30), ('corn', 5), ('milo', 25), ('oats', 15)]
>>> L[0]
('hay', 30)
>>> L[0][1]
30
```

for loop on list of tuples

- Normal for loop on a list, except the list elements are tuples
- So the variable in the for is each tuple in turn

```
>>> for pair in L:
        crop = pair[0]
        tons = pair[1]
        print("I have",tons,"of",crop)
```

Sorting a list of tuples

- Sorts by position 0 first, then position 1, and so on (if the tuples are longer).
- Just like sorting a list of strings

```
>>> sorted( ["owl", "awk", "jay"] )
['awk', 'jay', 'owl']
>>> sorted(L)
[('corn', 5), ('hay', 30), ('milo', 25), ('oats', 15)]
```

- Still sorted by keys…

Make our own darn list of tuples.

- Use the for loop on the dictionary.

```
>>> L = []
>>> for grain in D:
        tons = D[grain]
        pair = (tons,grain)
        L.append(pair)
>>> L
[(30, 'hay'), (5, 'corn'), (25, 'milo'), (15, 'oats')]
```

Now we can sort by values

```
>>> L
[(30, 'hay'), (5, 'corn'), (25, 'milo'), (15, 'oats')]
>>> L = sorted(L,reverse=True)
>>> for pair in L:
        print("I have",pair[0],"tons of",pair[1])
```

More compressed versions

```
>>> for (tons,grain) in L:
        print(tons,grain)
>>> for (grain,tons) in D.items():
        print(grain,tons)
```

- The pair of variables in the for loop take on the values of the pairs in the list of tuples or dictionary.

List of tuples vs dictionary

- Both are ways of storing pairs of items.
- Both are ways of labeling lots of memory.
- Dictionary lets you store and look up things by key.
- List of tuples lets you sort by values (if values are in position zero).
- Move back and forth using for loops.