Announcements

• No Reading or RQ until after exam
• APT 7 due Tuesday, March 28
• Assignment 6 due Thursday, March 30

Today:
– Finish example from last time
– Dictionaries – a way to organize data for fast lookup

LAST TIME:
Problem: Popular Name
• Given a list of names, determine the most popular first name and print that name with all of its last names.
• Input: Names are always two words, names are in a file. If multiple names are on the same line they are separated by a “:”
• Output: Most popular first name, followed by a “:”, followed by corresponding last names separated by a blank

Example Input File with 5 lines
Susan Smith:Jackie Long:Mary White
Susan Brandt
Jackie Johnson:Susan Rodger:Mary Rodger
Eric Long:Susan Crackers:Mary Velios
Jack Frost:Eric Lund

Corresponding Output
Susan: Smith Brandt Rodger Crackers
Example – two lists

Now can we solve the problem?

- Compute those two lists that are associated with each other
  - List of unique first names
  - List of corresponding last names
- Compute the max list of last names
- Now easy to print the answer.
- See popular.py

This function generates the list of lists of corresponding last names

```python
def correspondingLastNames(data, firstNames):
    lastNames = []
    for name in firstNames:
        lastNames.append(allLastNames(data, name))
    return lastNames
```

Finish

```python
maxnum = max([len(item) for item in lastNames])
print(maxnum)
lastIndex = [index for (index, v) in enumerate(lastNames) if len(v) == maxnum]
print("first name with most last names is:")
```
Expanding the Problem

• Suppose we want to read from multiple data files
  names1.txt, names2.txt, names3.txt

See processFiles in popular.py

Another way – list of lists

First word in each list is a first name
The rest are last names.

Dictionaries/Maps

• Dictionaries/maps are another way of organizing data
• Keys and Values
  – Each key maps to a value
  – Some keys can map to the same value
  – Can change the value a key maps to
Example
• Each student could be mapped to their favorite ice cream flavor

<table>
<thead>
<tr>
<th>Students</th>
<th>Ice Cream Flavors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Astrachan</td>
<td>Chocolate</td>
</tr>
<tr>
<td>Sun</td>
<td>Chocolate Chip</td>
</tr>
<tr>
<td>Rodger</td>
<td>Strawberry</td>
</tr>
<tr>
<td>Forbes</td>
<td></td>
</tr>
</tbody>
</table>

How is dictionary different than a list?
• List – have to search for name first
• Dictionary – each key maps to a value
• getting name (or key) is automatic! Fast!

<table>
<thead>
<tr>
<th>Students</th>
<th>Ice Cream Flavors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Astrachan</td>
<td>Chocolate</td>
</tr>
<tr>
<td>Sun</td>
<td>Chocolate Chip</td>
</tr>
<tr>
<td>Rodger</td>
<td>Strawberry</td>
</tr>
<tr>
<td>Forbes</td>
<td></td>
</tr>
</tbody>
</table>

Implementing a Dictionary/Map
Keys map to values
• Create Empty dictionary
  somemap = {}
• Put in a key and its value
  somemap[“Forbes”] = “Strawberry”
• Get a value for a dictionary
  value = somemap[“Forbes”]
  OR value = somemap.get(“Forbes”, “default”)
• Change a value for a dictionary
  somemap[“Forbes’] = “Chocolate”

More on using a Dictionary/Map
• Get all the keys (as a list)
  listKeys = somemap.keys()
• Get all the values (as a list)
  listValues = somemap.values()
• Other methods
  - clear – empty dictionary
  - items – return (key,value) pairs
  - iteritems – return (key,value) pairs more efficiently, iterator – must use with for
  - update – update with another dictionary
Change Astrachan’s value
somemap[“Astrachan”] = Coffee Mocha

Value could be a set or list

Simple dictionary
bit.ly/101s17-0323-1
More simple dictionaries
bit.ly/101s17-0323-2

Back to Popular Name Problem:

- Given a list of names, determine the most popular first name and print that name with all of its last names.
- Input: Names are always two words, names are in a file. If multiple names are on the same line they are separated by a “:”
- Output: Most popular first name, followed by a “:”, followed by corresponding last names separated by a blank

Example Input File with 5 lines

Susan Smith:Jackie Long:Mary White
Susan Brandt
Jackie Johnson:Susan Rodger:Mary Rodger
Eric Long:Susan Crackers:Mary Velios
Jack Frost:Eric Lund

Corresponding Output

Susan: Smith Brandt Rodger Crackers

Use a dictionary/map

- Map first names to count of corresponding last names

```python
def mapNameToNumberLastNames(data):
    # Use a dictionary/map
    popularMap.py
```
Trace example with Python Tutor
see popularMapSolnSmall.py

Use a dictionary/map
www.bit.ly/101s17-0323-4

• Map first name to list of corresponding last names

```
def mapNameToLastNames(data):
```

Trace through example with
Python Tutor

• See the small example
popularMapSolnSmall.py

Use dictionary of first names mapped to corresponding last names

• How do you find the most popular first name?
Use a dictionary/map

• Map first name to set of corresponding last names

```python
def mapNameToSetLastNames(data):
```

Compare

• Using two parallel lists?
• Using one dictionary/map
• Which dictionary is most useful to solve the most popular name problem?