Agenda

• Announcements

• Snarf code for class today: “SortingANDItemgetter”

• Dictionary and type of keys

• Dictionary and sorting
Dictionary

For loop using the following functions:

- `.keys()` gives you all the keys in the dictionary.
- `.values()` gives you all the values in the dictionary.
- `.items()` gives you BOTH the keys and the values in the dictionary as a tuple.
Dictionary

What is printed by this code?

dictPerson= { ["Peter", "Lorensen"]: "919-659-8462", ["Owen", "Astrachan"]: "919-985-3689" }
dictPerson[["Susan", "Rodger"]]= dictPerson.get(["Susan", "Rodger"], "919-867-3515")
dictPerson[["Kate", "Moss"]]= dictPerson.get(["Peter", "Lorensen"], "919-659-8462")
print dictPerson.values()

A: [ ["Peter", "Lorensen"],["Owen", "Astrachan"],
     ["Susan", "Rodger"],["Kate", "Moss"] ]
B: [ "919-659-8462", "919-985-3689", "919-867-3515", "919-659-8462"]
C: Error
D: [ "919-659-8462", "919-985-3689", "919-867-3515" ]
Mary Shaw

- Software engineering and software architecture
  - Tools for constructing large software systems
  - Development is a small piece of total cost, maintenance is larger, depends on well-designed and developed techniques

- Interested in computer science, programming, curricula, and canoeing, health-care costs
- ACM Fellow, Alan Perlis Professor of Compsci at CMU
- “Outstanding Research Award” winner from ACM SIGSOFT
Dictionary sort()

```
ages = [18, 14, 27, 23, 12, 31]
print ages.sort()
```

A: [12, 14, 18, 23, 27, 31]
B: None
C: Error
D: [31, 27, 23, 18, 14, 12]
Dictionary sorted()

```python
ages = [18, 14, 27, 23, 12, 31]
print sorted(ages)
```

A: [12, 14, 18, 23, 27, 31]
B: None
C: Error
D: [31, 27, 23, 18, 14, 12]
## Sort() vs sorted()

<table>
<thead>
<tr>
<th>sort()</th>
<th>sorted()</th>
</tr>
</thead>
<tbody>
<tr>
<td>In-place</td>
<td>Return list</td>
</tr>
<tr>
<td>Return None</td>
<td></td>
</tr>
<tr>
<td>Function in list</td>
<td>Built-in function</td>
</tr>
<tr>
<td>Stable</td>
<td>Stable</td>
</tr>
<tr>
<td>Timsort</td>
<td>Timsort</td>
</tr>
</tbody>
</table>
Sorting need

• Given a list of tuples, `sorted(sequence)` always uses the first item in the list to sort on:

```python
ages = ["Bob", 37], ("Joe", 21), ("Linda", 27), ("Ant", 37)
print sorted(ages)
```

```bash
>>> [('Ant', 37), ('Bob', 37), ('Joe', 21), ('Linda', 27)]
```

We often want to sort via the second (or third item) in a list of tuples.
def topsong(songd):
    """songd is dictionary: key is email address corresponding value is list of song titles purchased by person with email address returns: song bought by most people''"
    mostDict = {}
    for songList in songd.values():  # songList=['IceBaby','Raise Glass']
        for song in songList:        # song = 'Ice Ice Baby'
            mostDict[song] = mostDict.get(song, 0) + 1  # Adding new song if not there AND updating existing song at the same time
    """mostDict now contains this:
    mostDict = {'song1':3,
                  'song2':9,
                  'song3':2,
                  'song4':4, }
    """
    halfList = mostDict.items()  # [(Ice Ice Baby', 1), ('The Cave', 1), ('Raise Your Glass', 3), ('Landfill', 2), ('Sleepyhead', 1)]
    """TODO: Go through list of tuples, sort on the second item""
    return halfList[0][0]  # [('Raise Your Glass', 3), .....]
operator.itemgetter(...)

sorted(list, key = func, reverse = True)

- sorted() takes 2 extra arguments.
- func (key in documentation) is a function that is applied on every element.

operator.itemgetter(itemNumber)

- The itemgetter() function simply gets the element number specified
Operator.itemgetter(...) example

- You must import the operator to access the function .itemgetter(..)
- The function is applied for each tuple and you simply specify which element you want

```python
import operator

ages = [('Bob', 37), ('Joe', 21), ('Linda', 27), ('Ant', 37)]
print sorted(ages, key = operator.itemgetter(1))

>>> [('Joe', 21), ('Linda', 27), ('Bob', 37), ('Ant', 37)]
```
Try the snarfed project
import operator
employees = {
    "Li":("Sale", 1900.00),
    "Joe":("Sale", 2000.00),
    "Dale":("Transp", 1950.00),
    "Lyn":("Sale", 2250.00),
    "Pete":("Transp", 2150.00)
}

lstSort = sorted(employees.values(),
                 key=operator.itemgetter(1), reverse=True)
print lstSort[0][0]

A: [("Sale", 2250.0), ("Transp", 2150.0), ("Sale", 2000.0), ("Transp", 1950.0), ("Sale", 1900.0)]
B: 1900.00
C: [("Sale", 1900.0), ("Transp", 1950.0), ("Sale", 2000.0), ("Transp", 2150.0), ("Sale", 2250.0)]
D: 2250.00