Reminders

- Assignments
  - Assign 5 live

- APT Quiz 2
  - 11/12-11/15

- Final Exam
  - APT Quiz style
Key instructions

• Input ✔
• Output ✔
• Assignments* ✔
• Math/Logic ✔
• Conditionals ✔
• Repetition ✔

*not listed in book
Python Data Types

- int, float, bool ✔
- Collections
  - Strings ✔
  - Lists ✔
  - Tuples ✔
  - Sets ✔
  - Dictionaries ✔
PFTD

• Sorting
  • Sorting using standard Python APIs

• CSV Library
  • How to read data using standard Python APIs

• Lambda
  • Language construct to make sorting simpler
KISS Principle

• Think of the non-computing context for any word/terms

• KISS model
  • Work smarter, not harder!!

• “Good programmers are simply good designers.”
  • -Dr. Washington

• Design first and always!

• Importance of reusability

• USE PyCharm/PythonTutor IF YOU HAVE QUESTIONS!
People to Know: Taraneh BigBow

- Software Engineer (backend development)
  - TEKSystems at Apple
  - Jasco Products, Microsoft
- BigBow Technologies
- Oklahoma
  - Kiowa Tribe
TPS: Why Sort Data?

- Help understand data
  - https://www.youtube.com/watch?v=lcoxhH8N3Bo
Why Sort Data?

• Every field needs to visualize and understand data
  • Sorting helps with this from movies to policy to sports to location of infections to


How your GIS department can respond to COVID-19

Local Government
March 09, 2020

Mike Schoelen

A staggering wealth of geospatial information has emerged regarding the COVID-19 outbreak. Dashboards, near real-time services, and GitHub repositories have built the foundation for an extraordinarily transparent response effort.
How To Sort: Algorithms

- Does scale matter? It depends!

- You're playing Spades, Hearts, Bridge, Go-Fish
  - How you sort doesn't really matter, but whether you sort makes play more efficient? Better?

- Many ways to sort
  - Bubble, Insertion, Selection
  - Quick, Merge, Tim, Bogo
Activity 1: Popular Music

- Who are top two artists? Most Songs
- How did you do it?

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Rank</td>
<td>Song</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>Like a Rolling Stone</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>Satisfaction</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>Imagine</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>What's Going On</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>Respect</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>Good Vibrations</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>Johnny B. Goode</td>
</tr>
<tr>
<td>9</td>
<td>8</td>
<td>Hey Jude</td>
</tr>
<tr>
<td>10</td>
<td>9</td>
<td>Smells Like Teen Spirit</td>
</tr>
<tr>
<td>11</td>
<td>10</td>
<td>What'd I Say</td>
</tr>
</tbody>
</table>
Solve a Larger Problem

  - Top 1,000 songs, find top 10 artists
  - How many songs per artist?
As the size of the problem grows we want …
- The algorithm to still work and be fast!
- What to do?

Search example
- Google search results work
- SoundHound/Shazam results work
- ContentID on YouTube results work
Python to the Rescue

• Using `.sort(...)`, `sorted(...)`, and `lambda`

• Using CSV library and its API
  • CSV – Comma Separated Values

• Why use the CSV library?
  • How to handle the song “Hello, I Love You”?
Hits by Artists: SongReader.py

• What is returned by this function?
  • details of csv: `next` and no `split` and ...

```python
def countByArtist(name):
    csvf = open(name, 'r', encoding='utf-8')
    freader = csv.reader(csvf)
    header = next(freader)
    print("header row labels", header)
    data = {}
    for row in freader:
        artist = row[2]
        if artist not in data:
            data[artist] = 0
            data[artist] += 1
    csvf.close()
    return data
```
Sorting to Print/Visualize

- Dictionary is ('Beatles', 51) tuples
- But tuples not in order, so we must …

```python
if __name__ == '__main__':
    counts = countByArtist("data/top1000.csv")

    print('First 5 artists: ')
    for artist in sorted(counts.items())[:5]:
        print(artist)

print('Top 5 artists: ')
sortbycount = sorted([[a[1], a[0]] for a in counts.items()])
sortedArtists = [[a[1], a[0]] for a in sortbycount]
for artist in sortedArtists[-5:]:
    print(artist)
```

What is going on here?

Why more complicated than lines 28 & 29?
Two APIs: CSV and Sorting

• CSV Library to read and process data
  • Comma-separated, but can by ":" separated, or any character as we'll see later

• Similar to reading a file – returned by open
  • Iterable is returned by `csv.reader`
  • The `next` function advances iterable
  • Don't call `split`, we can access by index
    • Also by header-row label with `csv.dictreader`
Sorting API and Sorting Concepts

• What is `counts.items()` – how is it sorted?

• What does `sorted` return?
  • A list, you can slice a list, look for clues.
  • What can be sorted? A sequence
  • `sorted(counts.items())`
Sorting by Number of Songs

- Sort by first value vs sort by second value
- Need to put sequence back to original format

```python
print('\nFirst 5 artists:')
for artist in sorted(counts.items())[:5]:
    print(artist)

print('\nTop 5 artists:')
sortedArtists = sorted([(a[1], a[0]) for a in counts.items()])
for artist in sortedArtists[-5:]:
    print(artist)
```

If we comment out 33, what's printed? Why?
Python Sorting API

• We'll use both `sorted()` and `.sort()` API
  • How to call, what options are
  • How to sort on several criteria

• Creating a new list, modifying existing list
  • `sorted(..)` creates list from .. Iterable
  • `x.sort()` modifies the list x
API to change sorting

• In SongReader.py we changed order of tuples to change sorting order
  • Then we sliced the end to get "top" songs

• Can supply a function to compare elements
  • Function return value used to sort, key=function
  • Change order: reverse=True
Sorting Examples

• Use key=function argument and reverse=True
  • What if we want to write our own function?

In[2]: a = ["red", "orange", "green", "blue", "indigo", "violet"]
In[3]: sorted(a)
Out[3]: ['blue', 'green', 'indigo', 'orange', 'red', 'violet']
In[4]: sorted(a,key=len)
Out[4]: ['red', 'blue', 'green', 'orange', 'indigo', 'violet']
In[5]: sorted(a,key=len,reverse=True)
Out[5]: ['orange', 'indigo', 'violet', 'green', 'blue', 'red']

• What do you notice about 4 and 5?
The power of lambda

• We want to create a function "on-the-fly"
  • aka anonymous ("throw away") function
• \textit{lambda} \textit{parameter(s)}: \textit{expression}

\begin{verbatim}
In[7]: a
Out[7]: ['red', 'orange', 'green', 'blue', 'indigo', 'violet']
In[8]: sorted(a,key=lambda x : x.count("e"))
Out[8]: ['indigo', 'red', 'orange', 'blue', 'violet', 'green']
\end{verbatim}

• Why 'indigo' first and 'green' last?
  • What about order of ties? Next class! Stable
Anonymous Functions

• Useful when want “throw-away” function
  • Our case mainly sort

• Syntax: lambda PARAMETERS: EXPRESSION
  • PARAMETERS – 0 or more comma separated
  • EXPRESSION – evaluates to something
key=FUNCTION

- from operator import itemgetter
  - Can use
- \texttt{itemgetter}(N) same as \texttt{lambda x: x[N]}

- Past used itemgetter
- Now use lambda (more versatile)
Why is lambda used?

• It doesn't matter at all
  • [https://en.wikipedia.org/wiki/Alonzo_Church](https://en.wikipedia.org/wiki/Alonzo_Church)
    • ola aside: church->martin davis->don loveland->ola
  • Lisp and Scheme have lambda expressions
  • Guido, former BDFL, learned to live with lambda
What is a lambda expression?

• It's a function object, treat like expression/variable
  • Like list comprehensions, access variables

```python
>>> inc = lambda x : x + 1
>>> p = [1, 3, 5, 7]
>>> [inc(num) for num in p]
[2, 4, 6, 8]
```
Syntactic sugar
(makes the medicine go down)

• Syntactic sugar for a normal function definition

```python
def f(x):
    return x[1]

f = lambda x : x[1]
```

```python
sorted(lst, key=f)
```

```python
sorted(lst, key=lambda x : x[1])
```

```python
>>> d.items()
dict_items([('a', [1, 2, 3]), ('b', [4, 7]), ('c', [1, 1, 5, 8])])
```

```python
>>> sorted(d.items(), key=lambda x : len(x[1]))
[('b', [4, 7]), ('a', [1, 2, 3]), ('c', [1, 1, 5, 8])]
```

```python
>>> sorted(d.items(), key=lambda sparky : len(sparky[1]))
[('b', [4, 7]), ('a', [1, 2, 3]), ('c', [1, 1, 5, 8])]
```

Parameter name does not matter
Syntax and Semantics of Lambda

• Major use: single variable function as key

```python
>>> fruits = ["banana", "apple", "lemon", "kiwi", "pineapple"]
>>> sorted(fruits)
['apple', 'banana', 'kiwi', 'lemon', 'pineapple']
>>> min(fruits)
'apple'
>>> max(fruits)
'pineapple'
>>> min(fruits, key=lambda f: len(f))
'kiwi'
>>> max(fruits, key=lambda z: z.count("e"))
'pineapple'
>>> sorted(fruits, key=lambda z: z.count("e"))
['banana', 'kiwi', 'apple', 'lemon', 'pineapple']
```
Reminders

• Work smarter, not harder
• Design first
• Try to identify where you are stuck
  • Identify resources to help solve problem
• Leverage your design and PythonTutor to understand program flow of control
  • http://pythontutor.com