Compsci 101
Sorting, CSV

Susan Rodger
March 24, 2022

R is for …

• Random
  • .choice, .shuffle, .seed, .randint
• R
  • Programming language of choice in stats
• Refactoring
  • A way to rename your variable, function name

Esther Brown

• Harvard MS Data Sci
• Starting PhD in CS at Harvard!
• At Duke, as Senior did I.S. creating five Apps
  • Covid tracker
  • Movie App

Announcements

• APT 5 due today!
• Assignment 5 due Tue, March 29
• APT-6 out today, due Thur, March 31
• Lab 9 Friday
  • There is NO prelab!
• Reading and Sakai Quizzes due next week
PFTD

- **Sorting**
  - Sorting using standard Python APIs
- **CSV Library**
  - How to read data using standard Python APIs
- **Lambda**
  - Language construct to make sorting simpler (next week)

### Why Sort Data?
- **Help understand data**
  - Great American Eclipse, August 21, 2017
  - Spotify tracked the playing of the song

### Song: Total Eclipse of the Heart, Bonnie Tyler
[https://www.youtube.com/watch?v=lcOxhH8N3Bo](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 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
  • Bubblesort, Insertion sort, Selection sort
  • Quicksort, Mergesort, Timsort, Bogosort
  • Python uses Timsort

WOTO-1 Popular Music


• Make a copy of this spreadsheet:

WOTO-1 Popular Music


• Make a copy of this spreadsheet:

• Who are top two artists? Most Songs
• How did you do it?
Solve a Larger Problem

• Suppose I were to give you the top 1000 artists
  • Top 1,000 songs, find top 10 artists
  • How many songs per artist?

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”?
  • Row 166 in spreadsheet

Scale

• 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

Hits by Artists: SongReader.py

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

```
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
```
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
```

What is new? What does it do?

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`

 CSV API

- `freader = csv.reader(file)` – returns an iterable
  - Every line from the file in a form ready for you
- `line = next(freader)`
  - Gives you next row as list of strings
- `for row in freader`:
  - Gives you the rest of rows as list of strings
What does this do? `freader` an iterable
Where name is a filename

```python
csvf = open(name, 'r', encoding='utf-8')
freader = csv.reader(csvf)
print("freader", freader)
header = next(freader)
print("header", header)
for row in freader:
    print("row", row)
```

What if you call `next` one extra time?
Where name is a filename

```python
csvf = open(name, 'r', encoding='utf-8')
freader = csv.reader(csvf)
print("freader", freader)
header = next(freader)
print("header", header)
nextline = next(freader)
print("next", nextline)
for row in freader:
    print("row", row)
```
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('\nFirst 5 artists:')
    for artist in sorted(counts.items())[:5]:
        print(artist)
    print('\nTop 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)
```

Why more complicated than lines 28 & 29?

WOTO-3 Calling countByArtist

Sorting API and Sorting Concepts

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

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

- What does `sorted` return?
  - A list, you can slice a list, look for clues!
  - What can be sorted? A sequence
    - `sorted(counts.items())`
Sorting API and Sorting Concepts

- What is `counts.items()` – how is it sorted?
  ```python
  print('\nFirst 5 artists:\n')
  for artist in sorted(counts.items())[:5]:
      print(artist)
  ```

- What does `sorted()` return?
  - A list, you can slice a list, look for clues!
  - What can be sorted? A sequence
    ```python
    sorted(counts.items())
    ```

- How does Python evaluate slice?

Sorting by Number of Songs

- Sort by first value vs sort by second value
  ```python
  print('\nFirst 5 artists:\n')
  for artist in sorted(counts.items())[:5]:
      print(artist)
  ```

- Need to put sequence back to original format

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

```python
a = ['red', 'orange', 'green', 'blue', 'indigo', 'violet']
print(sorted(a))
print(sorted(a, key=len))
print(sorted(a, key=len, reverse=True))
```

WOTO-4 Sorting