S is for ...

- **Software**
  - Joy, sorrow, fun, changing the world

- **System and sys**
  - Connecting to the machine at different levels

- **Sorting**
  - From hat to timsort to more

---

### Barbara Liskov

- Among first women in US to earn Ph.D. in Computer Science: 1968
- Turing Award 2008, Software Engineering and Programming Languages
- Object-Oriented
  - CLU
- Liskov Substitution Principle

“Every time you exchange e-mail with a friend, check your bank statement online, or run a Google search, you are riding the momentum of her research” – MIT President Rafael Reif about Liskov

---

### Announcements

- APT 5 due today!
- Assignment 5 due Thurs, April 6
- No Lab this week
- Reading and Sakai Quizzes due next week
- APT Quiz 2 – today through Monday
APT Quiz 2 March 30-April 3

- Opens March 30, Thursday, 1:15pm
- Closes at 11pm Mon 4/3 – must finish all by this time
- There are two parts based on APTs 1-5
  - Each part has two APT problems
  - Each part is 3 hours – more if you get accommodations
  - Each part starts in Sakai under tests and quizzes
  - Sakai is a starting point with countdown timer that sends you to a new apt page just for each part
  - Could do each part on different day or same days
- Old APT Quiz so you can practice (not for credit) – on APT Page

**Practice (old APT quiz)**

APT Quiz 2

- Is your own work!
- No collaboration with others!
- Use your notes, lecture notes, your code, textbook
- DO NOT search for answers!
- Do not talk to others about the quiz until grades are posted
- **Post private questions on Ed Discussion**
  - We are not on between 9pm and 9am!
  - We are not on all the time, especially weekends
  - Will try to answer questions between 9am – 9pm
  - About typos, cannot help you in solving APTs

**See 101 APT page for tips on debugging APTs**
Don't go to Sakai to start APT Quiz until you are ready to start

If you click on it, you start it!

Other Tips for APT Quiz 2

• Write a helper function
  • Code is shorter, easier to debug, test helper function

• Change the format to something easier to work with
  • "5:7-8-10:21" to [5, [7, 8, 10], 21]
  • Easier to get parts, work with ints instead of strings

• Break the problem into several steps
  • Print after each step, before going on

• Follow Seven Steps!!!!

Song: Total Eclipse of the Heart, Bonnie Tyler
https://www.youtube.com/watch?v=lcOxhH8N3Bo

PFTD

• Sorting
  • Sorting using standard Python APIs

• CSV Library
  • How to read data using standard Python APIs
Why Sort Data?

• Help understand data
  • Great American Eclipse, August 21, 2017
  • Spotify tracked the playing of the song

---

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
WOTO-1 Popular Music

• Make a copy of this spreadsheet:
• Who are top two artists? Most Songs
• How did you do it?

<table>
<thead>
<tr>
<th>Rank</th>
<th>Song</th>
<th>Artist</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Like a Rolling Stone</td>
<td>Bob Dylan</td>
</tr>
<tr>
<td>2</td>
<td>Satisfaction</td>
<td>The Rolling Stones</td>
</tr>
<tr>
<td>3</td>
<td>Imagine</td>
<td>John Lennon</td>
</tr>
<tr>
<td>4</td>
<td>What's Going On</td>
<td>Marvin Gaye</td>
</tr>
<tr>
<td>5</td>
<td>Respect</td>
<td>Aretha Franklin</td>
</tr>
<tr>
<td>6</td>
<td>Good Vibrations</td>
<td>The Beach Boys</td>
</tr>
<tr>
<td>7</td>
<td>Johnny B. Goode</td>
<td>Chuck Berry</td>
</tr>
<tr>
<td>8</td>
<td>Hey Jude</td>
<td>The Beatles</td>
</tr>
<tr>
<td>9</td>
<td>Smells Like Teen Spirit</td>
<td>Nirvana</td>
</tr>
<tr>
<td>10</td>
<td>What'd I Say</td>
<td>Ray Charles</td>
</tr>
</tbody>
</table>

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?

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
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

Hits by Artists: SongReader.py

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

```python
9  def countByArtist(name):
10      csvfile = open(name, 'r', encoding='utf-8')
11      freader = csv.reader(csvfile)
12      header = next(freader)
13      print("header row labels", header)
14      data = {}
15      for row in freader:
16          artist = row[2]
17          if artist not in data:
18              data[artist] = 0
19              data[artist] += 1
20      csvfile.close()
21      return data
```

What is new?
What does it do?

WOTO-2 countByArtist
Two APIs: CSV and Sorting

• CSV Library to read and process data
  • Comma-separated, but can separate by ":", 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 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)
freader <csv.reader object at 034>
header ['Rank', 'Song', 'Artist']
row ['1', 'Stairway to Heaven', 'Led Zeppelin']
row ['2', 'Hey Jude', 'Beatles']
row ['3', 'All along the Watchtower', 'Hendrix, Jimi']
row ['4', 'Satisfaction', 'Rolling Stones']
...
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 API and Sorting Concepts

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

```python
print('First 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 by Number of Songs

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

```python
print('Top 5 artists:')
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)
```
Sorting by Number of Songs

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

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

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

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