CompSci 101
Sorting, CSV

	A	B	C
1	**Rank**	**Song**	**Artist**
2	1	Like a Rolling Stone	Bob Dylan
3	2	Satisfaction	The Rolling Stones
4	3	Imagine	John Lennon
5	4	What's Going On	Marvin Gaye
6	5	Respect	Aretha Franklin
7	6	Good Vibrations	The Beach Boys
8	7	Johnny B. Goode	Chuck Berry
9	8	Hey Jude	The Beatles
10	9	Smells Like Teen Spirit	Nirvana
11	10	What'd I Say	Ray Charles

Susan Rodger

March 30, 2023

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**

APT Quiz 2

We take cheating seriously in this course!

- **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**

APT Quiz

There will be two APT Quizzes that are just like APTs but are your own work and are timed. Start the APT quiz on Sakai under quizzes, but not until you are ready to take the quiz.

APTs

See below for hints on what to do if your APT doesn't run.

For each problem in an APT set, complete these steps by the due date

- first click on the APT set below to go to the APT page.
- write the code, upload the file, select the problem, and click the **Submit** link
- **check your grade** on the grade code page by clicking on **check submissions**

In solving APTs, your program should work for all cases, not just the test cases we provide. We may test your program on additional data.

APT	Due Date
APT-1	January 26
APT-2	February 9
APT-3	February 23
PRACTICE FOR APT QUIZ 1	NOT FOR CREDIT
APT-4	March 9
REVIEW YOUR APT QUIZ 1 Problems	NOT FOR CREDIT
APT 5	March 30
PRACTICE for APT Quiz 2	NOT DUE

We may do some APTs partially in class or lab, but you still have to do them and submit them. There will usually be extra apts listed. You can do more than required to challenge yourself. We do notice if you do more APTs than those required. If you do extra APTs, they still have to be turned in on the due date.

Regrades

If you have concerns about an item that was graded (lab, apt or assignment), you have one week after the grade is posted to fill out the regrade form here.

Problems Running an APT? Some Tips!

APT Quiz

There will be two APT Quizzes that are just like APTs but are your own work and are timed. Start the APT quiz on Sakai under quizzes, but not until you are ready to take the quiz.

APTs

See below for hints on what to do if your APT doesn't run.

For each problem in an APT set, complete these steps by the due date

- first click on the APT set below to go to the APT page.
- write the code, upload the file, select the problem, and click the **Submit** link
- **check your grade** on the grade code page by clicking on **check submissions**

In solving APTs, your program should work for all cases, not just the test cases we provide. We may test your program on additional data.

APT	Due Date
APT-1	January 26
APT-2	February 9
APT-3	February 23
PRACTICE FOR APT QUIZ 1	NOT FOR CREDIT
APT-4	March 9
REVIEW YOUR APT QUIZ 1 Problems	NOT FOR CREDIT
APT 5	March 30
PRACTICE for APT Quiz 2	NOT DUE

We may do some APTs partially in class or lab, but you still have to do them and submit them. There will usually be ext listed. You can do more than required to challenge yourself. We do notice if you do more APTs than those require d do extra APTs, they still have to be turned in on the due date.

Regrades

If you have concerns about an item that was graded (lab, apt or assignment), you have one week after the grade is posted to fill out the regrade form here.

Problems Running an APT? Some Tips!

APT Quiz Info

Practice (old APT quiz)

Debugging Tips

Stuck! Use 7 steps!

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

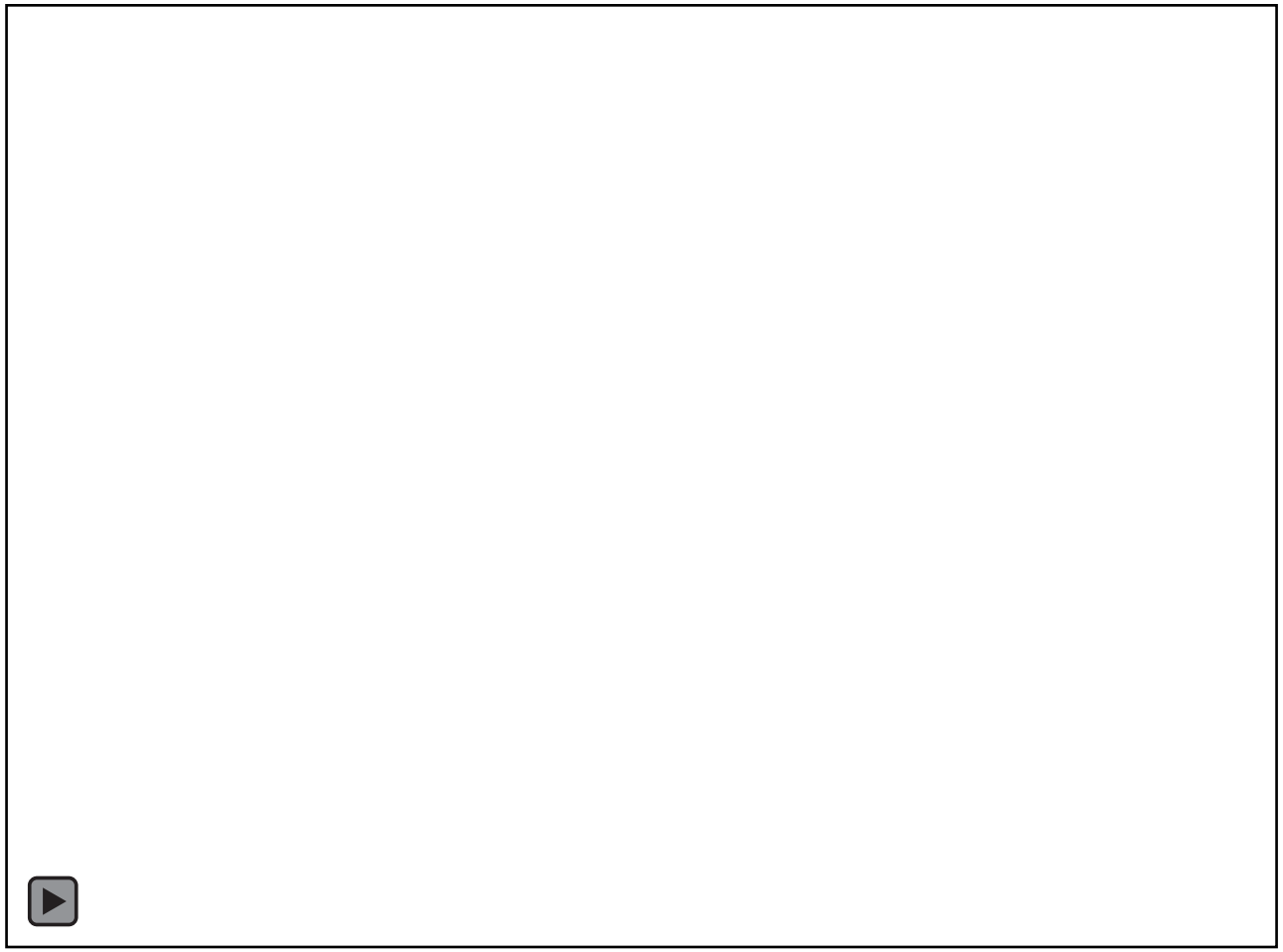
- **Write a helper function**
  - Code is shorter, easier to debug, test helper function
- **Change the format to something easier to work with**
  - "5:7-8-10:21" to [5, [7, 8, 10], 21]
  - Easier to get parts, work with ints instead of strings
- **Break the problem into several steps**
  - Print after each step, before going on
- **Follow Seven Steps!!!!**

PFTD

- **Sorting**
  - Sorting using standard Python APIs
- **CSV Library**
  - How to read data using standard Python APIs

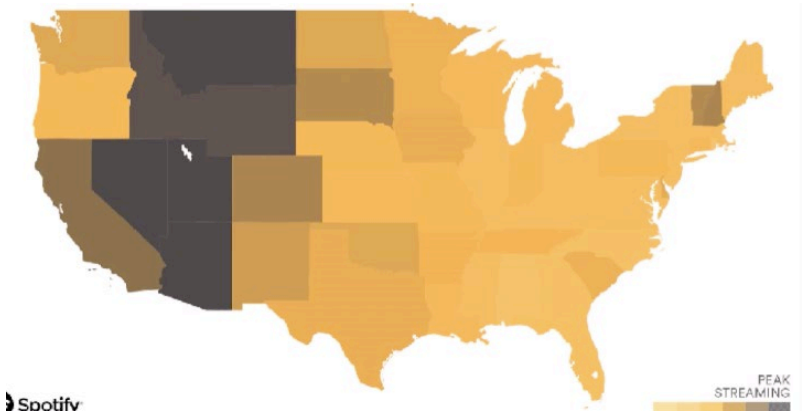
Song: Total Eclipse of the Heart, Bonnie Tyler

https://www.youtube.com/watch?v=lcOxhH8N3Bo



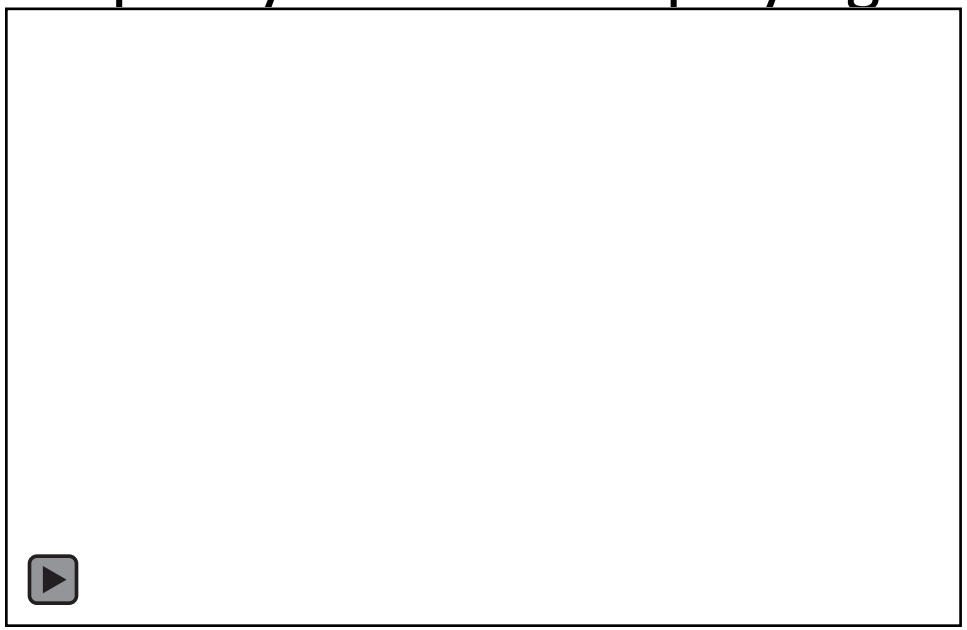
Why Sort Data?

- **Help understand data**
  - Great American Eclipse, August 21, 2017
  - http://bit.ly/spotify-eclipse-cnet
  - Spotify tracked the playing of the song



Why Sort Data?

- **Help understand data**
  - Great American Eclipse, August 21, 2017
  - http://bit.ly/spotify-eclipse-cnet
  - 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

https://www.esri.com/arcgis-blog/products/apps/local-government/how-your-gis-department-can-respond-to-covid-19/

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**
  - Bubblesort, Insertion sort, Selection sort
  - Quicksort, Mergesort, Timsort, Bogo sort
  - Python uses Timsort

WOTO-1 Popular Music
http://bit.ly/101s23-0330-1

- Make a copy of this spreadsheet:
  - http://bit.ly/101s23-0330-data
- Who are top two artists? Most Songs
- How did you do it?

	A	B	C
1	**Rank**	**Song**	**Artist**
2	1	Like a Rolling Stone	Bob Dylan
3	2	Satisfaction	The Rolling Stones
4	3	Imagine	John Lennon
5	4	What's Going On	Marvin Gaye
6	5	Respect	Aretha Franklin
7	6	Good Vibrations	The Beach Boys
8	7	Johnny B. Goode	Chuck Berry
9	8	Hey Jude	The Beatles
10	9	Smells Like Teen Spirit	Nirvana
11	10	What'd I Say	Ray Charles

WOTO-1 Popular Music
http://bit.ly/101s23-0330-1

	A	B	C
1	**Rank**	**Song**	**Artist**
2	1	Like a Rolling Stone	Bob Dylan
3	2	Satisfaction	The Rolling Stones
4	3	Imagine	John Lennon
5	4	What's Going On	Marvin Gaye
6	5	Respect	Aretha Franklin
7	6	Good Vibrations	The Beach Boys
8	7	Johnny B. Goode	Chuck Berry
9	8	Hey Jude	The Beatles
10	9	Smells Like Teen Spirit	Nirvana
11	10	What'd I Say	Ray Charles

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 …

```
 9  def countByArtist(name):
10      csvf = open(name, 'r', encoding='utf-8')
11      freader = csv.reader(csvf)
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
21      csvf.close()
22      return data
```

Hits by Artists: SongReader.py

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

```
 9  def countByArtist(name):
10      csvf = open(name, 'r', encoding='utf-8')
11      freader = csv.reader(csvf)
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
21      csvf.close()
22      return data
```

What is new?
What does it do?

WOTO-2 countByArtist
http://bit.ly/101s23-0330-2

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

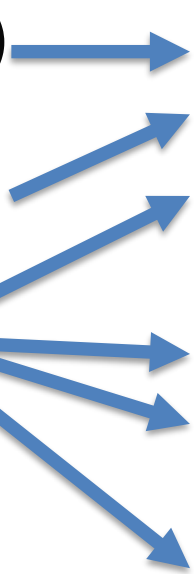
- **freader = csv.reader(file) – returns an iterable**
  - Every line from the file in a form ready for you
- **line = next(freader)**
  - Gives you next row as list of strings
- **for row in freader:**
  - Gives you the rest of rows as list of strings

What does this do? freader an iterable Where name is a filename

```
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

```
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)
```



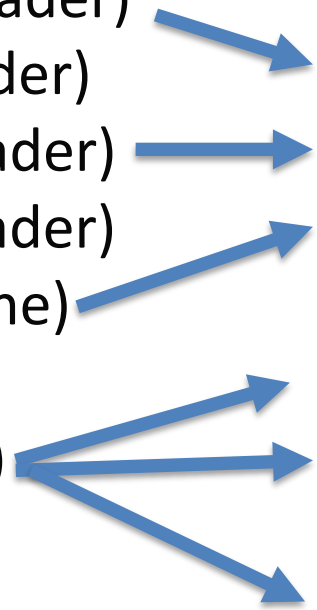
```
freader <csv.reader object at 034>
header [‘Rank’, ‘Song’, ‘Artist’]
row [‘1’, ‘Stairway to Heaven’, ‘Led Zepppelin’]
row [‘2’, ‘Hey Jude’, ‘Beatles’]
row [‘3’, ‘All along the Watchtower’, ‘Hendrix, Jimi’]
row [‘4’, ‘Satisfaction’, ‘Rolling Stones’]
…
```

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

```
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)
```

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

```
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)
```



freader <csv.reader object at 034>

header [‘Rank’, ‘Song’, ‘Artist’]

next [‘1’, ‘Stairway to Heaven’, ‘Led Zepppelin’]

row [‘2’, ‘Hey Jude’, ‘Beatles’]

row [‘3’, ‘All along the Watchtower’, ‘Hendrix, Jimi’]

row [‘4’, ‘Satisfaction’, ‘Rolling Stones’]

…

Sorting to Print/Visualize

- **Dictionary is tuples, (key,value) like ('Beatles', 51)**
  - But tuples not in order, so we must put in order…

```
24  if __name__ == '__main__':
25      counts = countByArtist("data/top1000.csv")
26
27      print('\nFirst 5 artists:')
28      for artist in sorted(counts.items())[:5]:
29          print(artist)
30
31      print('\nTop 5 artists:')
32      sortbycount = sorted([(a[1], a[0]) for a in counts.items()])
33      sortedArtists = [(a[1], a[0]) for a in sortbycount]
34      for artist in sortedArtists[-5:]:
35          print(artist)
```

Sorting to Print/Visualize

- **Dictionary is tuples, (key,value) like ('Beatles', 51)**
  - But tuples not in order, so we must put in order …

```
24 ▶ if __name__ == '__main__':
25       counts = countByArtist("data/top1000.csv")
26
27       print('\nFirst 5 artists:')
28       for artist in sorted(counts.items())[:5]:
29           print(artist)
30
31       print('\nTop 5 artists:')
32       sortbycount = sorted([(a[1], a[0]) for a in counts.items()])
33       sortedArtists = [(a[1], a[0]) for a in sortbycount]
34       for artist in sortedArtists[-5:]:
35           print(artist)
```

What is going on here?

Why more complicated than lines 28 & 29?

WOTO-3 Calling countByArtist
http://bit.ly/101s23-0330-3

Sorting API and Sorting Concepts

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

```
27      print('\nFirst 5 artists:')
28      for artist in sorted(counts.items())[:5]:
29          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?**

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

How does Python evaluate slice?

- **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

```
27      print('\nFirst 5 artists:')
28      for artist in sorted(counts.items())[:5]:
29          print(artist)
30
31      print('\nTop 5 artists:')
32      sortedArtists = sorted([(a[1], a[0]) for a in counts.items()])
33      sortedArtists = [(a[1], a[0]) for a in sortedArtists]
34      for artist in sortedArtists[-5:]:
35          print(artist)
```

Sorting by Number of Songs

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

```
27      print('\nFirst 5 artists:')
28      for artist in sorted(counts.items())[:5]:
29          print(artist)
30
31      print('\nTop 5 artists:')
32      sortedArtists = sorted([(a[1], a[0]) for a in counts.items()])
33      sortedArtists = [(a[1], a[0]) for a in sortedArtists]
34      for artist in sortedArtists[-5:]:
35          print(artist)
```

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