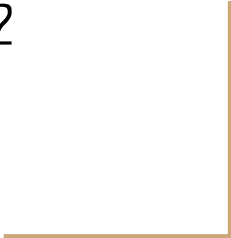


# Programming, Problem Solving, and Algorithms

CPSC203, 2023 W2



# Announcements

- **Reminder: I respond to Q4 of Learning Logs!**
  - Comments/feedback have been released
  - Remember to look at our sample answers for Q1
- **Test 2 results are released**
- **Lab4 is this week**
  - More on Lists and Dictionaries
- **Problem of the Week 5 is due this week**
  - Functions and dictionaries

# Today's Plan...

1. Announcements! (5 mins)
2. Weekly Videos Review/Questions (10 mins)
3. Billboard 100 Demo (65 mins)

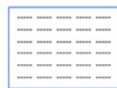


# Slides from the Assigned Videos



# 103 to 203

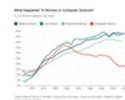
## Typical Introductory Data Flow:



.csv file

```
if pivot != right:
    array[right], array[pivot] = array[pivot], array[right]
    return partition_right(array, left, right, pivot)
def partition_right(array, left, right, pivot):
    mid = array[right]
    mid = left
```

Python problem solution using simple data types and elementary list iteration.



Matplotlib bar or line graph or other summative output illustrating results of computation.

## CPSC103++ Data Flow:



Diverse data sources

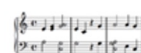
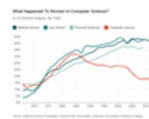
```
23 if pivot != right:
24     array[right], array[pivot] = array[pivot], array[right]
25     return partition_right(array, left, right, pivot)
26 def partition_right(array, left, right, pivot):
27     mid = array[right]
28     mid = left
```

```
23 if pivot != right:
24     array[right], array[pivot] = array[pivot], array[right]
25     return partition_right(array, left, right, pivot)
26 def partition_right(array, left, right, pivot):
27     mid = array[right]
28     mid = left
```

data synthesis

```
23 if pivot != right:
24     array[right], array[pivot] = array[pivot], array[right]
25     return partition_right(array, left, right, pivot)
26 def partition_right(array, left, right, pivot):
27     mid = array[right]
28     mid = left
```

Analysis and data assembly

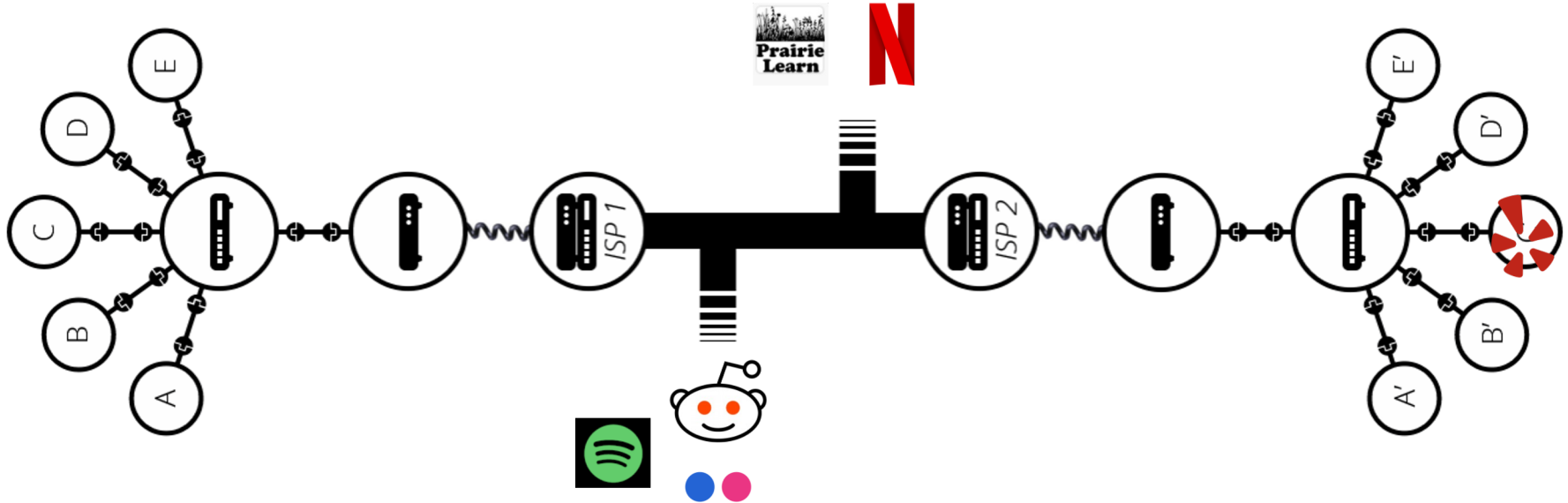


mpg	hp	wt
18.700	123.000	3.125
20.091	146.688	3.217
1.055	12.120	0.173
2.173	24.720	0.153
34.134	490.007	0.157
6.827	68.563	0.978
8.308	8.497	0.104



Diverse outputs

# The internet...



# Billboard Hot 100...

Navigate to <https://www.billboard.com/charts/hot-100>

What happens to the URL if you load a past week? \_\_\_\_\_

What happens to the page if you substitute a different date into the URL?

\_\_\_\_\_

Write one question you would like to ask of this data: \_\_\_\_\_

\_\_\_\_\_

# Anatomy of html...

```
<!DOCTYPE html>
```

```
<html><head><title>The Dormouse's story</title></head>
```

```
<body><p class="title"><b>The Dormouse's story</b></p>
```

```
<p class="story">Once upon a time there were two little sisters.  
Their names were <a href="http://example.com/elsie" class="sister"  
id="link1">Elsie</a>, and <a href="http://example.com/lacie"  
class="sister" id="link2">Lacie</a>, and they lived at the bottom of  
a well.</p>
```

```
</body>
```

```
</html>
```



# Billboard Hot 100... page source

```
<div class="chart-list-item piano-content-overlay__gated-item" data-rank="49" data-artist="Taylor Swift" data-title="Lover" data-has-content="true"> <div
class="chart-list-item__first-row chart-list-item__cursor-pointer"> <div class="chart-list-item__position chart-list-item__position--centered"> <div
class="chart-list-item__rank "> 49 </div> <div class="chart-list-item__award"> </div> </div> <div class="chart-list-item__image-wrapper"> <div class="chart-list-item__trend-icon">
src="https://assets.billboard.com/assets/1568911107/images/charts/arrow-down.svg?df89925e3b37f64521bd"
srcset="https://assets.billboard.com/assets/1568911107/images/charts/arrow-down-mobile.svg?df89925e3b37f64521bd 30w,
https://assets.billboard.com/assets/1568911107/images/charts/arrow-down.svg?df89925e3b37f64521bd 38w" sizes="(min-width: 768px) 38px,
30px"></div>

</div>

<div class="chart-list-item__text-wrapper"> <div class="chart-list-item__text "> <div class="chart-list-item__title">
<span class="chart-list-item__title-text">
Lover
```

Lover

chart-list-item\_\_artist">[Taylor Swift](/music/taylor-swift)

chart-list-item\_\_lyrics">[Song Lyrics](https://www.billboard.com/articles/news/lyrics/7950218/ready-for-it-taylor-swift-lyrics)

chart-list-item\_\_chevron-wrapper<i class="fa fa-chevron-down"></i></div></div>

chart-list-item\_\_extra-info<div class="chart-list-item\_\_extra-info-shadow"></div>

chart-list-item\_\_stats<div class="chart-list-item\_\_stats-cell basic-user chart-list-item\_\_stats-cell--first-cell"> <div class="chart-list-item\_\_stats-icon fa fa-arrow-up fa-rotate-45"></div>

chart-list-item\_\_last-week>23</div>

LAST WEEK </div>

chart-list-item\_\_stats-cell basic-user "> <div class="chart-list-item\_\_stats-icon fa fa-arrow-up fa-rotate-45"></div>

chart-list-item\_\_last-week>10</div>

TWO WEEKS AGO</div>

chart-list-item\_\_stats-cell basic-user "> <div class="chart-list-item\_\_stats-icon fa fa-line-chart"></div>

chart-list-item\_\_weeks-at-one">10</div>

PEAK POSITION </div>

chart-list-item\_\_stats-cell basic-user chart-list-item\_\_stats-cell--no-border-right"><div class="chart-list-item\_\_stats-icon fa fa-clock-o"></div>

chart-list-item\_\_weeks-on-chart">4</div>

WEEKS ON CHART</div></div></div></div>

# Beautiful Soup

Reads the html source into a data structure that's easy to query!

<https://www.crummy.com/software/BeautifulSoup/bs4/doc/>

```
html = simple_get("https://www.billboard.com/charts/hot-100" + '/' + date)
mydivs = html.findAll("div", {"class": "chart-list-item"}) // all the data is here!!

for div in mydivs:
    s = Song(div.attrs['data-title'], div.attrs['data-artist'], int(div.attrs['data-rank']))
```

Still too messy for us! Remedy? <https://github.com/guoguo12/billboard-charts>

demo...

# References

POTD: Continue every weekday! Submit to PL.

References:

[https://pandas.pydata.org/Pandas\\_Cheat\\_Sheet.pdf](https://pandas.pydata.org/Pandas_Cheat_Sheet.pdf)

<https://www.crummy.com/software/BeautifulSoup/bs4/doc/>

# Billboard 100 Demo