Lab 5

Student: Kyler Halat-Shafer

UserID: uxt5qb

Problem 0

In [1]:

```
import numpy as np
import pandas as pd
import requests
from bs4 import BeautifulSoup
import sys
sys.tracebacklimit = 0 # turn off the error tracebacks
```

Problem 1

In [4]:

```
url = 'https://books.toscrape.com/'
```

In [5]:

```
useragent = 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537
```

In [117…

```
r = requests.get(url, headers = {'User-agent' :'Kyler Halat-Shafer, a studen
r
```

Out[117]:

```
<Response [200]>
```

In [126…

```
soup = BeautifulSoup(r.text,"html.parser")
#soup
```

Problem 2

In [118…

```
#First understanding where to find the book title
soup.find_all('img')[0]['alt']
```

Out[118]:

```
'A Light in the Attic'
```

In [119…

```
#Extracting the 20 books by using a list comprehension
books = [x['alt'] for x in soup.find_all('img')]
books
```

```
Out[119]: ['A Light in the Attic',
           'Tipping the Velvet',
           'Soumission',
           'Sharp Objects',
           'Sapiens: A Brief History of Humankind',
           'The Requiem Red',
           'The Dirty Little Secrets of Getting Your Dream Job',
           'The Coming Woman: A Novel Based on the Life of the Infamous Feminist, Vic
          toria Woodhull',
           'The Boys in the Boat: Nine Americans and Their Epic Quest for Gold at the
          1936 Berlin Olympics',
           'The Black Maria',
           'Starving Hearts (Triangular Trade Trilogy, #1)',
           "Shakespeare's Sonnets",
           'Set Me Free',
           "Scott Pilgrim's Precious Little Life (Scott Pilgrim #1)",
           'Rip it Up and Start Again',
           'Our Band Could Be Your Life: Scenes from the American Indie Underground,
          1981-1991',
           'Olio',
           'Mesaerion: The Best Science Fiction Stories 1800-1849',
           'Libertarianism for Beginners',
           "It's Only the Himalayas"]
```

Problem 3

```
In [52]: # the 'p' is the first thing that we see in the HTML, then we are looking at
         soup.find_all('p',{'class':'price_color'})[0].string
```

```
Out[52]: 'Â£51.77'
```

```
In [51]: # This is using a list comprehension, x as a string for the output, instead
         price = [x.string for x in soup.find_all('p',{'class':'price_color'})]
         price
```

Out[51]:

```
['Â£51.77',
 'Â£53.74',
 'Â£50.10',
 'Â£47.82',
 'Â£54.23',
 'Â£22.65',
 'Â£33.34',
 'Â£17.93',
 'Â£22.60',
 'Â£52.15',
 'Â£13.99',
 'Â£20.66',
 'Â£17.46',
 'Â£52.29',
 'Â£35.02',
 'Â£57.25',
 'Â£23.88',
 'Â£37.59',
 'Â£51.33',
 'Â£45.17']
```

In [53]:

```
# From there we are able to replace the symbols with spaces to make them num
price = [s.replace('Â£', '') for s in price]
price
```

Out[53]:

```
['51.77',
 '53.74',
 '50.10',
 '47.82',
 '54.23',
 '22.65',
 '33.34',
 '17.93',
 '22.60',
 '52.15',
 '13.99',
 '20.66',
 '17.46',
 '52.29',
 '35.02',
 '57.25',
 '23.88',
 '37.59',
 '51.33',
 '45.17']
```

Problem 4

In [121…

```
# This was after a couple of attempts to figure out how to get down to what
soup.find_all('p',{'class':'star-rating'})[0]
```

Out[121]:
```
<p class="star-rating Three">
<i class="icon-star"></i>
<i class="icon-star"></i>
<i class="icon-star"></i>
<i class="icon-star"></i>
<i class="icon-star"></i>
</p>
```

In [120…
```
#This is searching for p and is then looking at 'star-rating' for the first
#then calling the class list and the second element in that list, which is t

soup.find_all('p','star-rating')[0]['class'][1]
```

Out[120]:
```
'Three'
```

In [122…
```
# x['class'][1] is the output | for x (the object) in the list that is defir
ratings = [x['class'][1] for x in soup.find_all('p','star-rating')]
ratings
```

Out[122]:
```
['Three',
 'One',
 'One',
 'Four',
 'Five',
 'One',
 'Four',
 'Three',
 'Four',
 'One',
 'Two',
 'Four',
 'Five',
 'Five',
 'Five',
 'Three',
 'One',
 'One',
 'Two',
 'Two']
```

Problem 5

In [127…
```
#Using a similar format to what was done in Problem 2, but instead of using
soup.find_all('img')[0]['src']
```

Out[127]:
```
'../media/cache/00/25/0025515e987a1ebd648773f9ac70bfe6.jpg'
```

In [96]:
```
images = [x['src'] for x in soup.find_all('img')]
images
```

Out[96]:

```
['media/cache/2c/da/2cdad67c44b002e7ead0cc35693c0e8b.jpg',
 'media/cache/26/0c/260c6ae16bce31c8f8c95daddd9f4a1c.jpg',
 'media/cache/3e/ef/3eef99c9d9adef34639f510662022830.jpg',
 'media/cache/32/51/3251cf3a3412f53f339e42cac2134093.jpg',
 'media/cache/be/a5/bea5697f2534a2f86a3ef27b5a8c12a6.jpg',
 'media/cache/68/33/68339b4c9bc034267e1da611ab3b34f8.jpg',
 'media/cache/92/27/92274a95b7c251fea59a2b8a78275ab4.jpg',
 'media/cache/3d/54/3d54940e57e662c4dd1f3ff00c78cc64.jpg',
 'media/cache/66/88/66883b91f6804b2323c8369331cb7dd1.jpg',
 'media/cache/58/46/5846057e28022268153beff6d352b06c.jpg',
 'media/cache/be/f4/bef44da28c98f905a3ebec0b87be8530.jpg',
 'media/cache/10/48/1048f63d3b5061cd2f424d20b3f9b666.jpg',
 'media/cache/5b/88/5b88c52633f53cacf162c15f4f823153.jpg',
 'media/cache/94/b1/94b1b8b244bce9677c2f29ccc890d4d2.jpg',
 'media/cache/81/c4/81c4a973364e17d01f217e1188253d5e.jpg',
 'media/cache/54/60/54607fe8945897cdcced0044103b10b6.jpg',
 'media/cache/55/33/553310a7162dfbc2c6d19a84da0df9e1.jpg',
 'media/cache/09/a3/09a3aef48557576e1a85ba7efea8ecb7.jpg',
 'media/cache/0b/bc/0bbcd0a6f4bcd81ccb1049a52736406e.jpg',
 'media/cache/27/a5/27a53d0bb95bdd88288eaf66c9230d7e.jpg']
```

Problem 6

In [128…

```
#First I create a dictionary by naming the keys accoridng to what is in the

mydict = {'title':books,
          'price': price,
          'ratings':ratings,
         'JPEG URL':images}

mydf = pd.DataFrame(mydict)
mydf
```

Out[128]:

	title	price	ratings	JPEG URL
0	A Light in the Attic	51.77	Three	media/cache/2c/da/2cdad67c44b002e7ead0cc35693c...
1	Tipping the Velvet	53.74	One	media/cache/26/0c/260c6ae16bce31c8f8c95daddd9f...
2	Soumission	50.10	One	media/cache/3e/ef/3eef99c9d9adef34639f51066202...
3	Sharp Objects	47.82	Four	media/cache/32/51/3251cf3a3412f53f339e42cac213...
4	Sapiens: A Brief History of Humankind	54.23	Five	media/cache/be/a5/bea5697f2534a2f86a3ef27b5a8c...
5	The Requiem Red	22.65	One	media/cache/68/33/68339b4c9bc034267e1da611ab3b...
	The Dirty Little Secrets of			

6	Getting Your Dream...	33.34	Four	media/cache/92/27/92274a95b7c251fea59a2b8a7827...
7	The Coming Woman: A Novel Based on the Life of...	17.93	Three	media/cache/3d/54/3d54940e57e662c4dd1f3ff00c78...
8	The Boys in the Boat: Nine Americans and Their...	22.60	Four	media/cache/66/88/66883b91f6804b2323c8369331cb...
9	The Black Maria	52.15	One	media/cache/58/46/5846057e28022268153beff6d352...
10	Starving Hearts (Triangular Trade Trilogy, #1)	13.99	Two	media/cache/be/f4/bef44da28c98f905a3ebec0b87be...
11	Shakespeare's Sonnets	20.66	Four	media/cache/10/48/1048f63d3b5061cd2f424d20b3f9...
12	Set Me Free	17.46	Five	media/cache/5b/88/5b88c52633f53cacf162c15f4f82...
13	Scott Pilgrim's Precious Little Life (Scott Pi...	52.29	Five	media/cache/94/b1/94b1b8b244bce9677c2f29ccc890...
14	Rip it Up and Start Again	35.02	Five	media/cache/81/c4/81c4a973364e17d01f217e118825...
15	Our Band Could Be Your Life: Scenes from the A...	57.25	Three	media/cache/54/60/54607fe8945897cdcced0044103b...
16	Olio	23.88	One	media/cache/55/33/553310a7162dfbc2c6d19a84da0d...
17	Mesaerion: The Best Science Fiction Stories 18...	37.59	One	media/cache/09/a3/09a3aef48557576e1a85ba7efea8...
18	Libertarianism for Beginners	51.33	Two	media/cache/0b/bc/0bbcd0a6f4bcd81ccb1049a52736...
19	It's Only the Himalayas	45.17	Two	media/cache/27/a5/27a53d0bb95bdd88288eaf66c923...

Problem 7

In [129…

```
#By combining everything under one roof of a function it can then be called

def scraplyfe(url):
    r = requests.get(url, headers = {'User-agent' :useragent})
    soup = BeautifulSoup(r.text,"html.parser")

    books = [x['alt'] for x in soup.find_all('img')]
    price = [x.string for x in soup.find_all('p',{'class':'price_color'})]
    price = [s.replace('Â£', '') for s in price]
    ratings = [x['class'][1] for x in soup.find_all('p','star-rating')]
    images = [x['src'] for x in soup.find_all('img')]

    mydict = {'title':books,
          'price': price,
          'ratings':ratings,
         'JPEG URL':images}
    mydf = pd.DataFrame(mydict)
    return mydf
```

In [106…

```
scraplyfe(url)
```

Out[106]:

	title	price	ratings	JPEG URL
0	A Light in the Attic	51.77	Three	media/cache/2c/da/2cdad67c44b002e7ead0cc35693c...
1	Tipping the Velvet	53.74	One	media/cache/26/0c/260c6ae16bce31c8f8c95daddd9f...
2	Soumission	50.10	One	media/cache/3e/ef/3eef99c9d9adef34639f51066202...
3	Sharp Objects	47.82	Four	media/cache/32/51/3251cf3a3412f53f339e42cac213...
4	Sapiens: A Brief History of Humankind	54.23	Five	media/cache/be/a5/bea5697f2534a2f86a3ef27b5a8c...
5	The Requiem Red	22.65	One	media/cache/68/33/68339b4c9bc034267e1da611ab3b...
6	The Dirty Little Secrets of Getting Your Dream...	33.34	Four	media/cache/92/27/92274a95b7c251fea59a2b8a7827...
7	The Coming Woman: A Novel Based on the Life of...	17.93	Three	media/cache/3d/54/3d54940e57e662c4dd1f3ff00c78...
8	The Boys in the Boat: Nine Americans and Their...	22.60	Four	media/cache/66/88/66883b91f6804b2323c8369331cb...
9	The Black Maria	52.15	One	media/cache/58/46/5846057e28022268153beff6d352...

10	Starving Hearts (Triangular Trade Trilogy, #1)	13.99	Two	media/cache/be/f4/bef44da28c98f905a3ebec0b87be...
11	Shakespeare's Sonnets	20.66	Four	media/cache/10/48/1048f63d3b5061cd2f424d20b3f9...
12	Set Me Free	17.46	Five	media/cache/5b/88/5b88c52633f53cacf162c15f4f82...
13	Scott Pilgrim's Precious Little Life (Scott Pi...	52.29	Five	media/cache/94/b1/94b1b8b244bce9677c2f29ccc890...
14	Rip it Up and Start Again	35.02	Five	media/cache/81/c4/81c4a973364e17d01f217e118825...
15	Our Band Could Be Your Life: Scenes from the A...	57.25	Three	media/cache/54/60/54607fe8945897cdcced0044103b...
16	Olio	23.88	One	media/cache/55/33/553310a7162dfbc2c6d19a84da0d...
17	Mesaerion: The Best Science Fiction Stories 18...	37.59	One	media/cache/09/a3/09a3aef48557576e1a85ba7efea8...
18	Libertarianism for Beginners	51.33	Two	media/cache/0b/bc/0bbcd0a6f4bcd81ccb1049a52736...
19	It's Only the Himalayas	45.17	Two	media/cache/27/a5/27a53d0bb95bdd88288eaf66c923...

Problem 8

In [130…

```
#By seperating out the url into 3 pieces, I was then able to write a for loo
#Because the function is then called in the for loop, it is then iterated or

new_df = pd.DataFrame()

for i in range(1,51):
    url = 'http://books.toscrape.com/catalogue/page-' + str(i) + '.html'
    one_df = scraplyfe(url)
    new_df = pd.concat([new_df,one_df])

new_df
```

Out[130]:

	title	price	ratings	JPEG URL
0	A Light in the Attic	51.77	Three	../media/cache/2c/da/2cdad67c44b002e7ead0cc356...
1	Tipping the Velvet	53.74	One	../media/cache/26/0c/260c6ae16bce31c8f8c95dadd...
2	Soumission	50.10	One	../media/cache/3e/ef/3eef99c9d9adef34639f51066...
3	Sharp Objects	47.82	Four	../media/cache/32/51/3251cf3a3412f53f339e42cac...
4	Sapiens: A Brief History of Humankind	54.23	Five	../media/cache/be/a5/bea5697f2534a2f86a3ef27b5...
...	...	...	...	...
15	Alice in Wonderland (Alice's Adventures in Won...	55.53	One	../media/cache/96/ee/96ee77d71a31b7694dac6855f...
16	Ajin: Demi-Human, Volume 1 (Ajin: Demi-Human #1)	57.06	Four	../media/cache/09/7c/097cb5ecc6fb3fbe1690cf0cb...
17	A Spy's Devotion (The Regency Spies of London #1)	16.97	Five	../media/cache/1b/5f/1b5ff86f3c75e51e24c573d3f...
18	1st to Die (Women's Murder Club #1)	53.98	One	../media/cache/2b/41/2b4161c5b72a4ae386b644682...
19	1,000 Places to See Before You Die	26.08	Five	../media/cache/d7/0f/d70f7edd92705c45a82118c3f...

1000 rows × 4 columns

In [ ]: