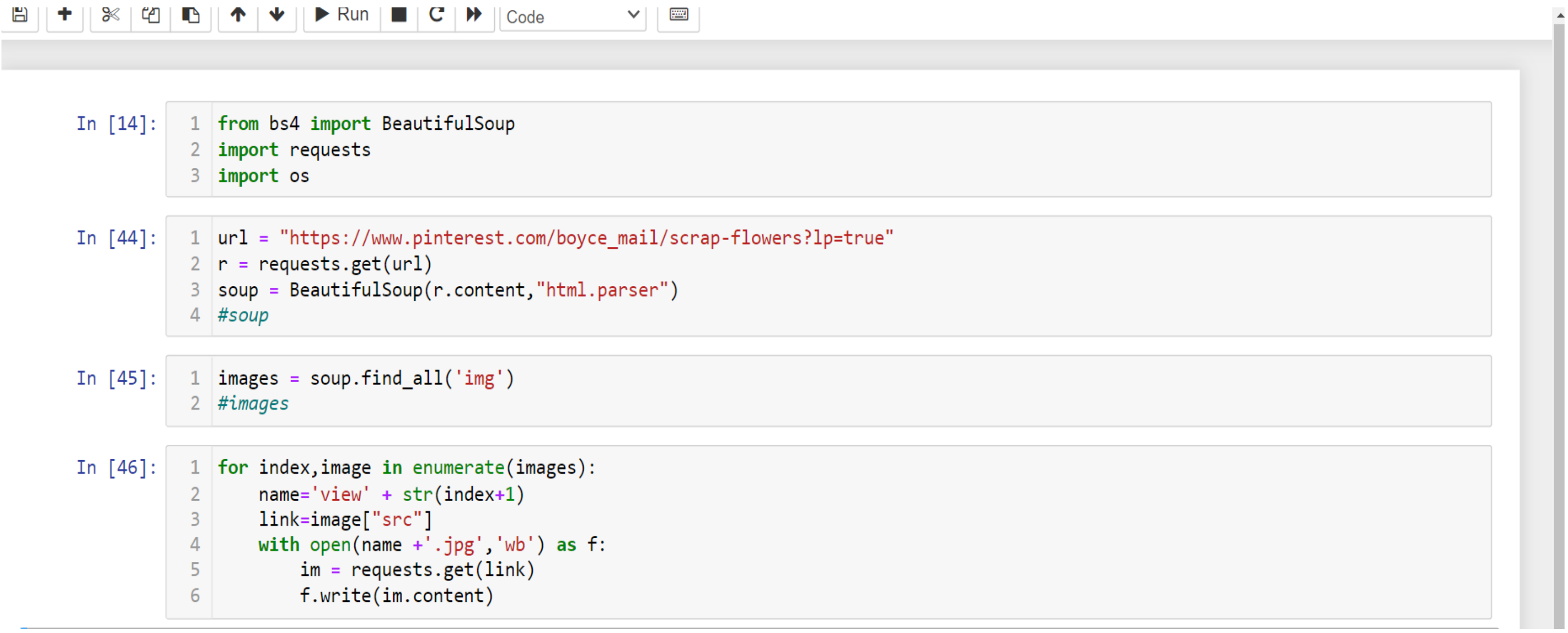
Images scraping

Code:



In [14]:

```python
from bs4 import BeautifulSoup
import requests
import os
```

In [44]:

```python
url = "https://www.pinterest.com/boyce_mail/scrap-flowers?lp=true"
r = requests.get(url)
soup = BeautifulSoup(r.content,"html.parser")
#soup
```

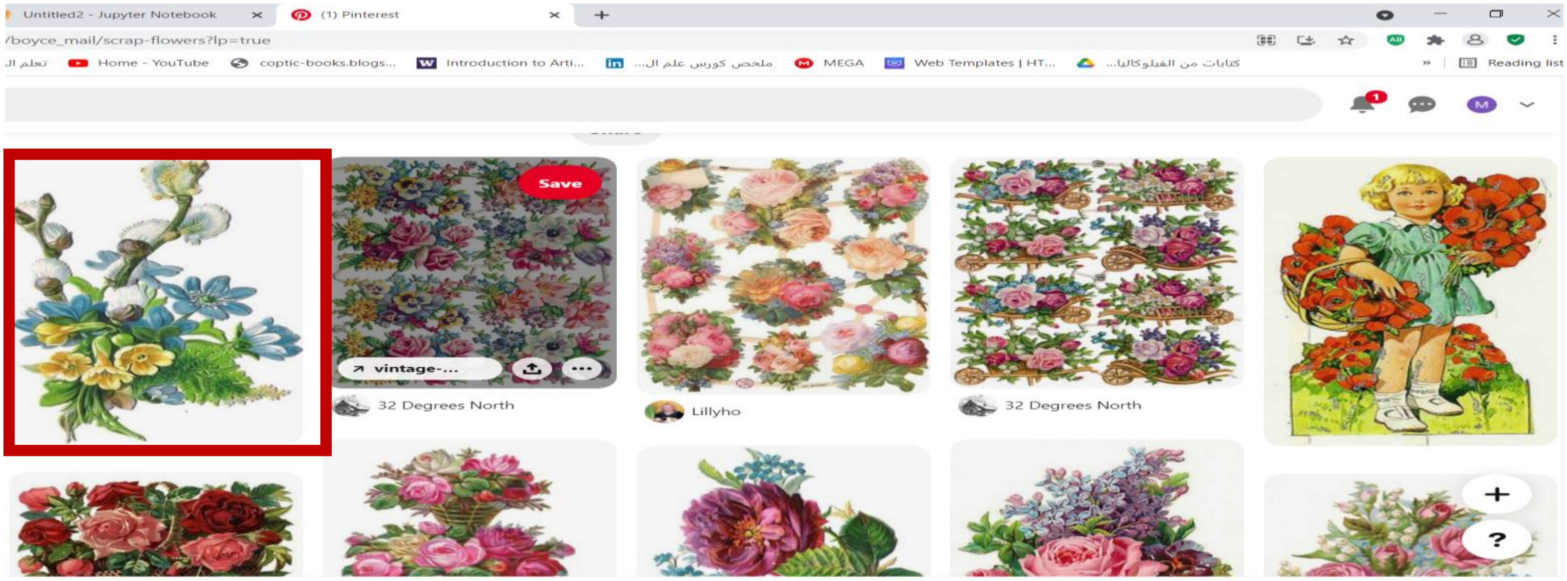
In [45]:

```python
images = soup.find_all('img')
#images
```

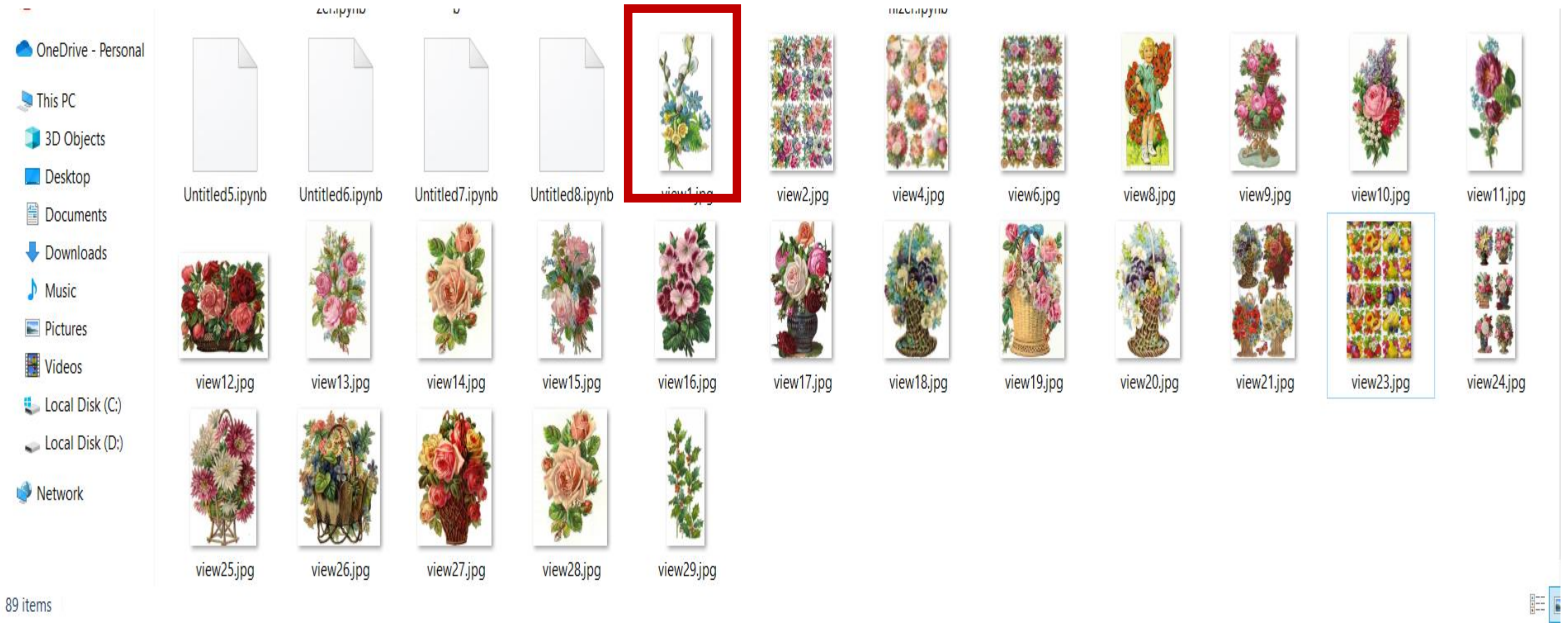
In [46]:

```python
for index,image in enumerate(images):
    name='view' + str(index+1)
    link=image["src"]
    with open(name +'.jpg','wb') as f:
        im = requests.get(link)
        f.write(im.content)
```

From website : (1st picture)

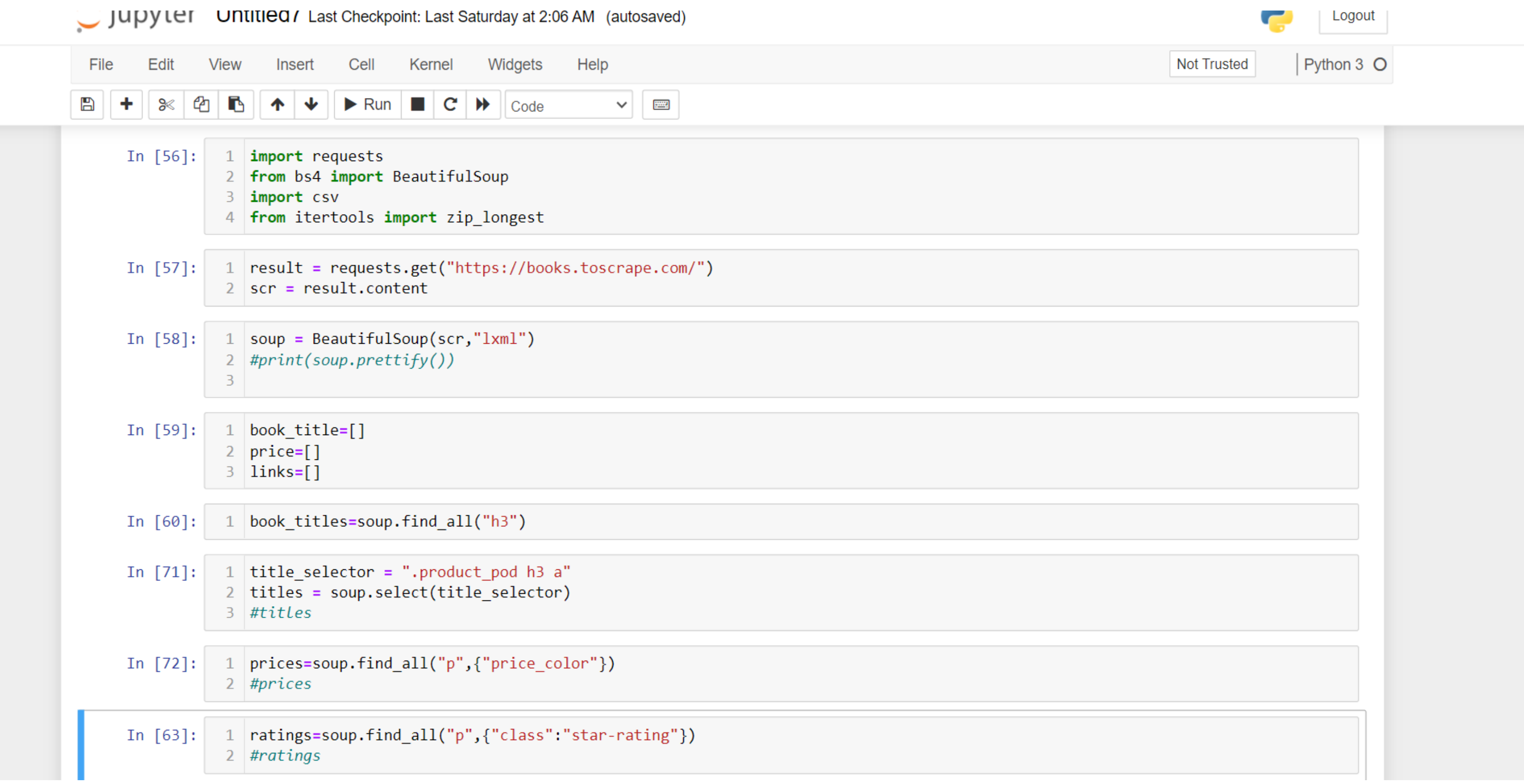


From my computer : Here same(1st picture)



Web scraping(html):

Code:



In [56]:

```
import requests
from bs4 import BeautifulSoup
import csv
from itertools import zip_longest
```

In [57]:

```
result = requests.get("https://books.toscrape.com/")
scr = result.content
```

In [58]:

```
soup = BeautifulSoup(scr,"lxml")
#print(soup.prettify())

```

In [59]:

```
book_title=[]
price=[]
links=[]
```

In [60]:

```
book_titles=soup.find_all("h3")
```

In [71]:

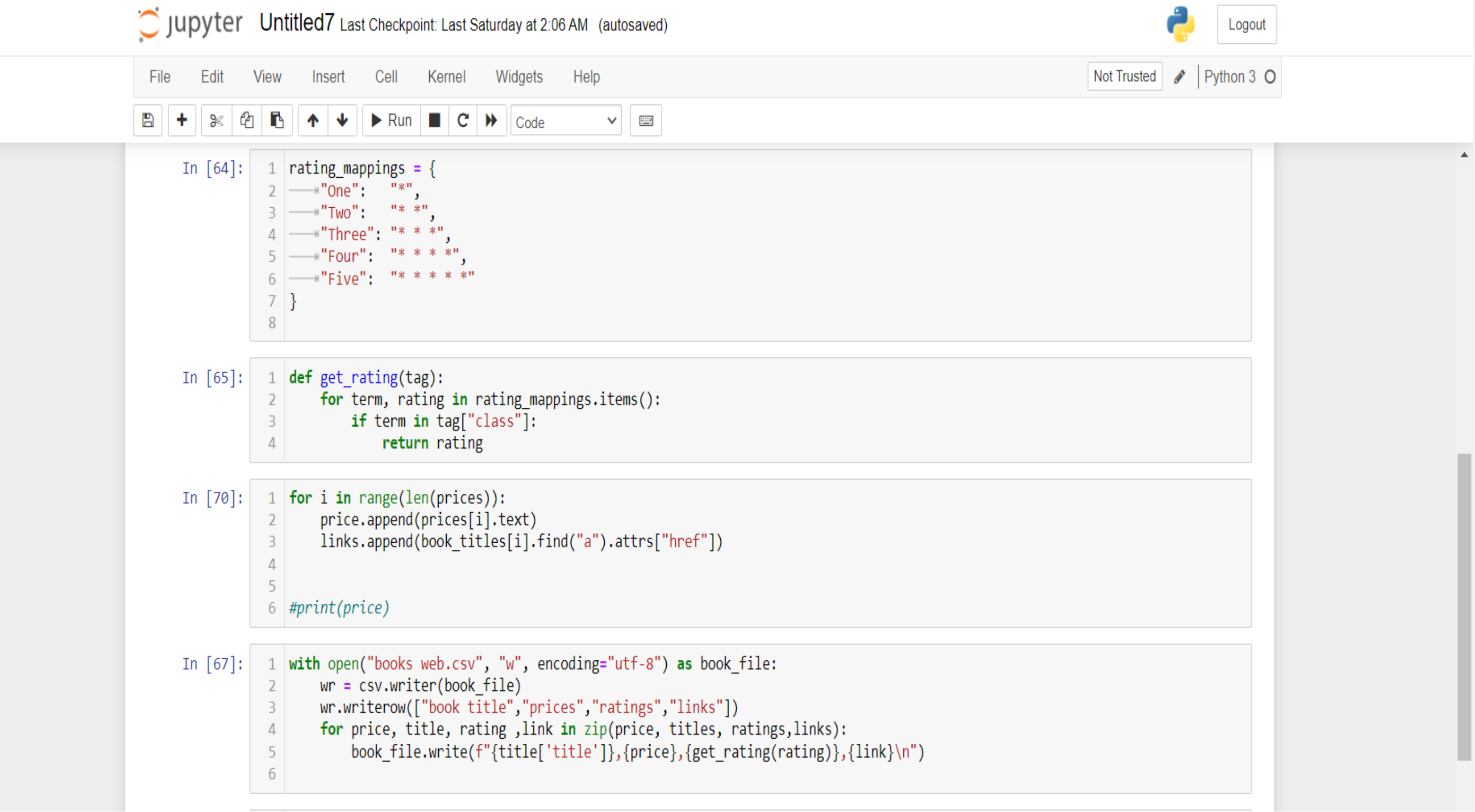
```
title_selector = ".product_pod h3 a"
titles = soup.select(title_selector)
#titles
```

In [72]:

```
prices=soup.find_all("p",{"price_color"})
#prices
```

In [63]:

```
ratings=soup.find_all("p",{"class":"star-rating"})
#ratings
```


jupyter Untitled7 Last Checkpoint: Last Saturday at 2:06 AM (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code

```
In [64]:
rating_mappings = {
    "One":   "*",
    "Two":   "* *",
    "Three": "* * *",
    "Four":  "* * * *",
    "Five":  "* * * * *"
}

```

```
In [65]:
def get_rating(tag):
    for term, rating in rating_mappings.items():
        if term in tag["class"]:
            return rating
```

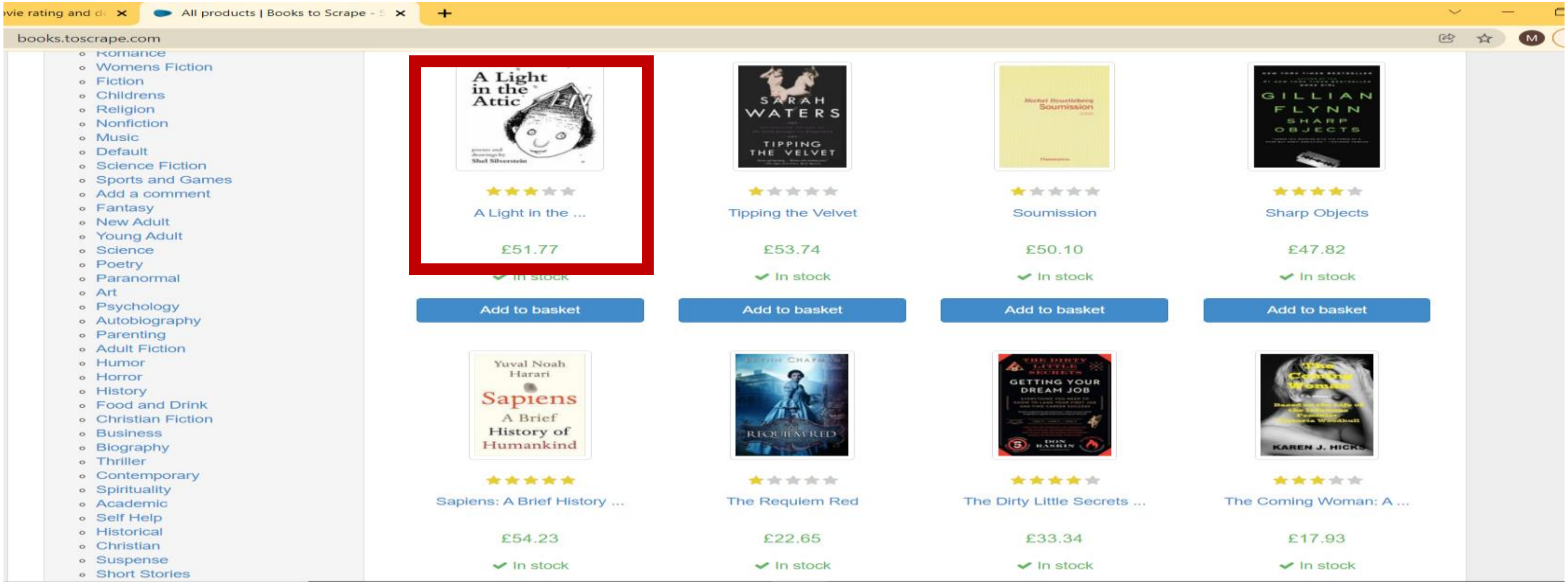
```
In [70]:
for i in range(len(prices)):
    price.append(prices[i].text)
    links.append(book_titles[i].find("a").attrs["href"])


#print(price)
```

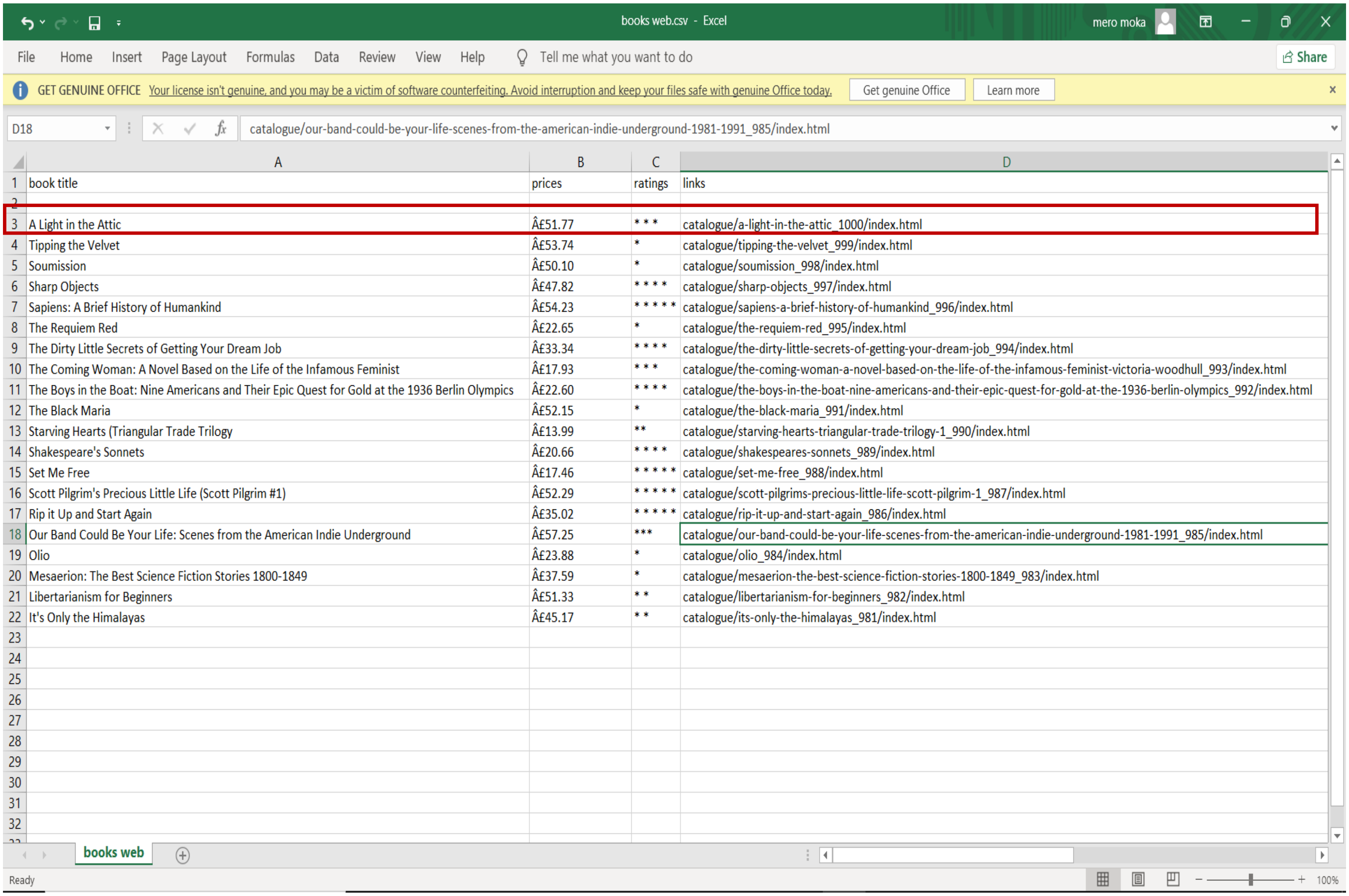
```
In [67]:
with open("books web.csv", "w", encoding="utf-8") as book_file:
    wr = csv.writer(book_file)
    wr.writerow(["book title","prices","ratings","links"])
    for price, title, rating ,link in zip(price, titles, ratings,links):
        book_file.write(f"{title['title']},{price},{get_rating(rating)},{link}\n")

```

Website:



Excel file:



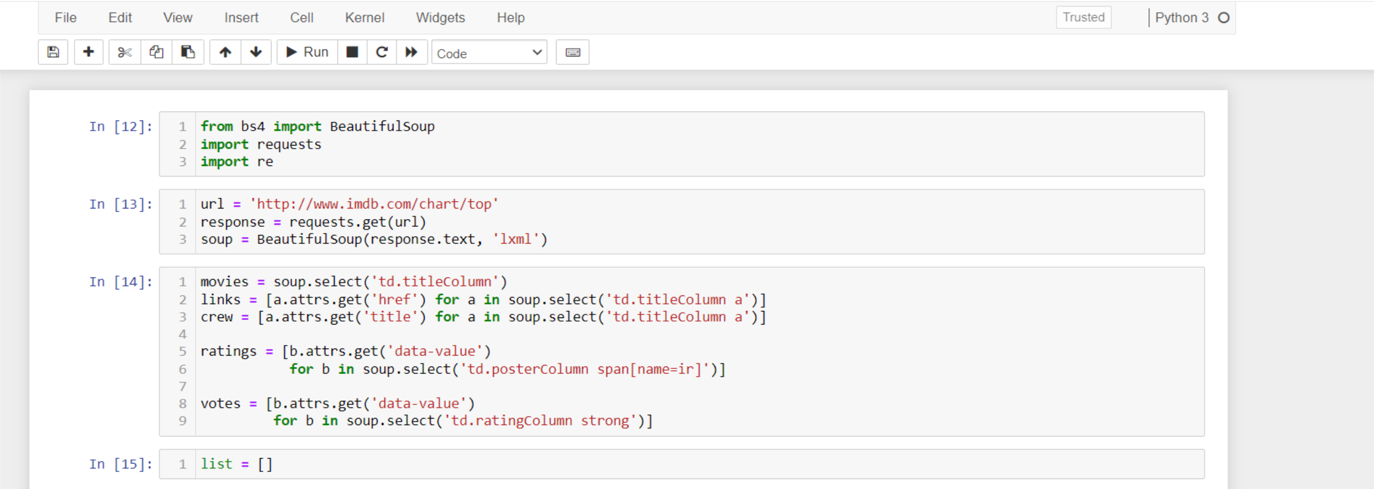
GET GENUINE OFFICE Your license isn't genuine, and you may be a victim of software counterfeiting. Avoid interruption and keep your files safe with genuine Office today.

D18 catalogue/our-band-could-be-your-life-scenes-from-the-american-indie-underground-1981-1991_985/index.html

	A	B	C	D
1	book title	prices	ratings	links
2				
3	A Light in the Attic	Â£51.77	* * *	catalogue/a-light-in-the-attic_1000/index.html
4	Tipping the Velvet	Â£53.74	*	catalogue/tipping-the-velvet_999/index.html
5	Soumission	Â£50.10	*	catalogue/soumission_998/index.html
6	Sharp Objects	Â£47.82	* * * *	catalogue/sharp-objects_997/index.html
7	Sapiens: A Brief History of Humankind	Â£54.23	* * * * *	catalogue/sapiens-a-brief-history-of-humankind_996/index.html
8	The Requiem Red	Â£22.65	*	catalogue/the-requiem-red_995/index.html
9	The Dirty Little Secrets of Getting Your Dream Job	Â£33.34	* * * *	catalogue/the-dirty-little-secrets-of-getting-your-dream-job_994/index.html
10	The Coming Woman: A Novel Based on the Life of the Infamous Feminist	Â£17.93	* * *	catalogue/the-coming-woman-a-novel-based-on-the-life-of-the-infamous-feminist-victoria-woodhull_993/index.html
11	The Boys in the Boat: Nine Americans and Their Epic Quest for Gold at the 1936 Berlin Olympics	Â£22.60	* * * *	catalogue/the-boys-in-the-boat-nine-americans-and-their-epic-quest-for-gold-at-the-1936-berlin-olympics_992/index.html
12	The Black Maria	Â£52.15	*	catalogue/the-black-maria_991/index.html
13	Starving Hearts (Triangular Trade Trilogy	Â£13.99	**	catalogue/starving-hearts-triangular-trade-trilogy-1_990/index.html
14	Shakespeare's Sonnets	Â£20.66	* * * *	catalogue/shakespeares-sonnets_989/index.html
15	Set Me Free	Â£17.46	* * * * *	catalogue/set-me-free_988/index.html
16	Scott Pilgrim's Precious Little Life (Scott Pilgrim #1)	Â£52.29	* * * * *	catalogue/scott-pilgrims-precious-little-life-scott-pilgrim-1_987/index.html
17	Rip it Up and Start Again	Â£35.02	* * * * *	catalogue/rip-it-up-and-start-again_986/index.html
18	Our Band Could Be Your Life: Scenes from the American Indie Underground	Â£57.25	***	catalogue/our-band-could-be-your-life-scenes-from-the-american-indie-underground-1981-1991_985/index.html
19	Olio	Â£23.88	*	catalogue/olio_984/index.html
20	Mesaerion: The Best Science Fiction Stories 1800-1849	Â£37.59	*	catalogue/mesaerion-the-best-science-fiction-stories-1800-1849_983/index.html
21	Libertarianism for Beginners	Â£51.33	* *	catalogue/libertarianism-for-beginners_982/index.html
22	It's Only the Himalayas	Â£45.17	* *	catalogue/its-only-the-himalayas_981/index.html
23				
24				
25				
26				
27				
28				
29				
30				
31				
32				

Implementing Imdb movie

Code:



In [12]:

```
from bs4 import BeautifulSoup
import requests
import re
```

In [13]:

```
url = 'http://www.imdb.com/chart/top'
response = requests.get(url)
soup = BeautifulSoup(response.text, 'lxml')
```

In [14]:

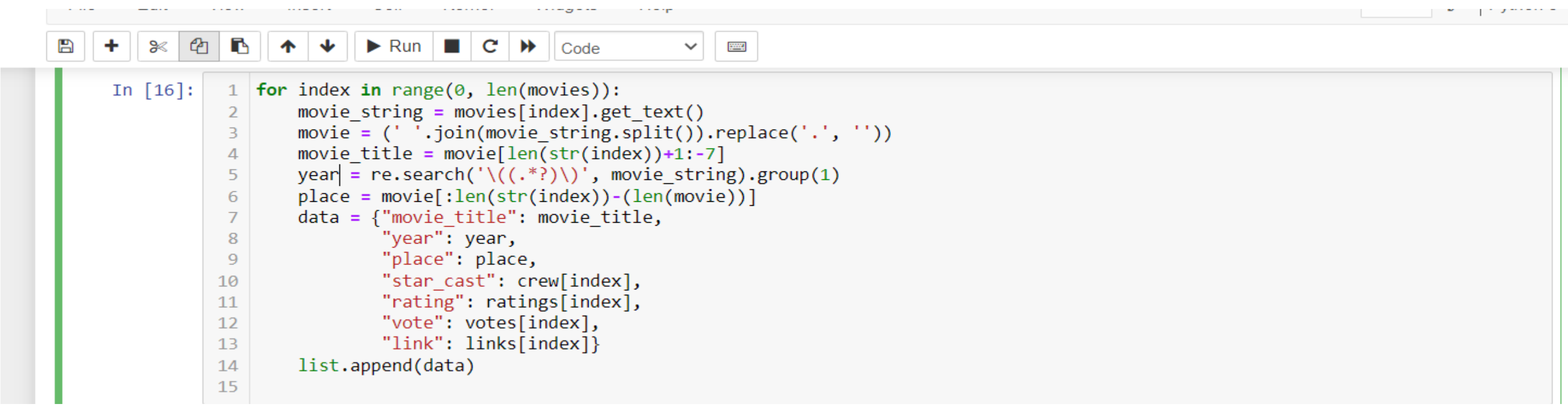
```
movies = soup.select('td.titleColumn')
links = [a.attrs.get('href') for a in soup.select('td.titleColumn a')]
crew = [a.attrs.get('title') for a in soup.select('td.titleColumn a')]

ratings = [b.attrs.get('data-value')
           for b in soup.select('td.posterColumn span[name=ir]')]

votes = [b.attrs.get('data-value')
         for b in soup.select('td.ratingColumn strong')]
```

In [15]:

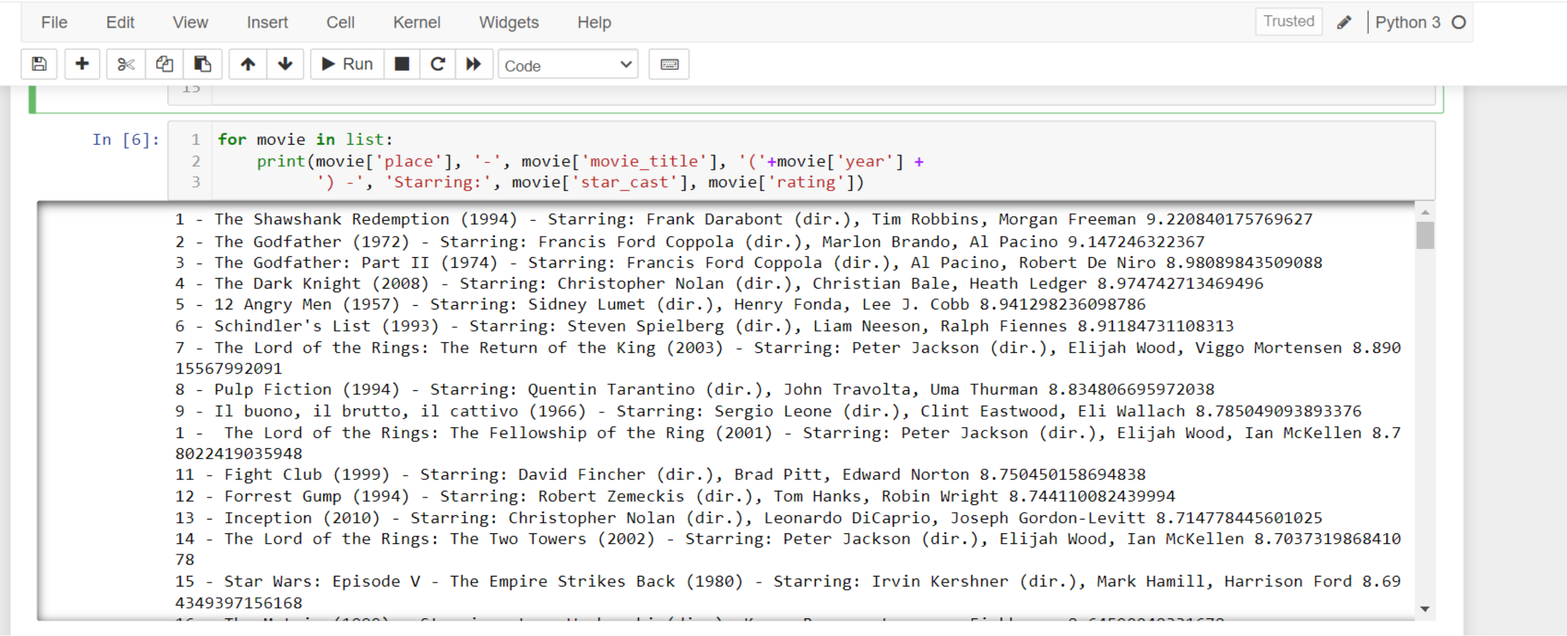
```
list = []
```



In [16]:

```
for index in range(0, len(movies)):
    movie_string = movies[index].get_text()
    movie = (' '.join(movie_string.split()).replace('.', ''))
    movie_title = movie[len(str(index))+1:-7]
    year = re.search('\((.*?)\)', movie_string).group(1)
    place = movie[:len(str(index))-(len(movie))]
    data = {"movie_title": movie_title,
            "year": year,
            "place": place,
            "star_cast": crew[index],
            "rating": ratings[index],
            "vote": votes[index],
            "link": links[index]}
    list.append(data)

```



In [6]:

```
for movie in list:
    print(movie['place'], '-', movie['movie_title'], '('+movie['year'] +
          ') -', 'Starring:', movie['star_cast'], movie['rating'])
```

```
1 - The Shawshank Redemption (1994) - Starring: Frank Darabont (dir.), Tim Robbins, Morgan Freeman 9.220840175769627
2 - The Godfather (1972) - Starring: Francis Ford Coppola (dir.), Marlon Brando, Al Pacino 9.147246322367
3 - The Godfather: Part II (1974) - Starring: Francis Ford Coppola (dir.), Al Pacino, Robert De Niro 8.98089843509088
4 - The Dark Knight (2008) - Starring: Christopher Nolan (dir.), Christian Bale, Heath Ledger 8.974742713469496
5 - 12 Angry Men (1957) - Starring: Sidney Lumet (dir.), Henry Fonda, Lee J. Cobb 8.941298236098786
6 - Schindler's List (1993) - Starring: Steven Spielberg (dir.), Liam Neeson, Ralph Fiennes 8.91184731108313
7 - The Lord of the Rings: The Return of the King (2003) - Starring: Peter Jackson (dir.), Elijah Wood, Viggo Mortensen 8.890
15567992091
8 - Pulp Fiction (1994) - Starring: Quentin Tarantino (dir.), John Travolta, Uma Thurman 8.834806695972038
9 - Il buono, il brutto, il cattivo (1966) - Starring: Sergio Leone (dir.), Clint Eastwood, Eli Wallach 8.785049093893376
1 -  The Lord of the Rings: The Fellowship of the Ring (2001) - Starring: Peter Jackson (dir.), Elijah Wood, Ian McKellen 8.7
8022419035948
11 - Fight Club (1999) - Starring: David Fincher (dir.), Brad Pitt, Edward Norton 8.750450158694838
12 - Forrest Gump (1994) - Starring: Robert Zemeckis (dir.), Tom Hanks, Robin Wright 8.744110082439994
13 - Inception (2010) - Starring: Christopher Nolan (dir.), Leonardo DiCaprio, Joseph Gordon-Levitt 8.714778445601025
14 - The Lord of the Rings: The Two Towers (2002) - Starring: Peter Jackson (dir.), Elijah Wood, Ian McKellen 8.7037319868410
78
15 - Star Wars: Episode V - The Empire Strikes Back (1980) - Starring: Irvin Kershner (dir.), Mark Hamill, Harrison Ford 8.69
4349397156168
```

Report of imdb code:

- Modules (**BeautifulSope** for scrapping , **Requests** for requesting an access to a link , re for searching using regular expression)
- We put the URL of website that we want to scrape the data and using requests to access to link, beautifulsope to scrape
- Then we make lists of all data we want to extract (movies' name , links , crew , ratings , votes) each of this lists contains of **html tags** that reach to the data that we want to extract
- For loop on all lists that contain html tags to covert them to text and stored details of each movie in a dictionary then added them to one list and print them