S-109A Introduction to Data Science

Homework 1

Harvard University
Summer 2019
Instructors: Pavlos Protopapas and Kevin Rader

Main Theme: Data Collection - Web Scraping - Data Parsing

Learning Objectives

In this homework, your goal is to learn how to acquire, parse, clean, and analyze data. Initially you read the data from a file, then you scrape them directly from a website. You look for specific pieces of information by parsing the data, you clean the data to prepare them for analysis, and finally, you answer some questions.

Instructions

- To submit your assignment follow the instructions given in Classroom.
- The deliverables in Classroom are: a) This python notebook with your code and answers, b) a .pdf version of this notebook, c) The BibTex file you created. d) The JSON file you created.
- Exercise **responsible scraping**. Web servers can become slow or unresponsive if they receive too many requests from the same source in a short amount of time. Use a delay of 10 seconds between requests in your code. This helps not to get blocked by the target website. Run the webpage fetching part of the homework only once and do not re-run after you have saved the results in the JSON file (details below).
- Web scraping requests can take several minutes. This is another reason why you should not wait until the last minute to do this homework.

Name: Lucas Machado Moschen

In [1]:

```
# import the necessary libraries
%matplotlib inline
import numpy as np
import scipy as sp
import matplotlib as mpl
import matplotlib.cm as cm
import matplotlib.pyplot as plt
import pandas as pd
import time
pd.set_option('display.width', 500)
pd.set_option('display.max_columns', 100)
pd.set_option('display.notebook_repr_html', True)
```

Part A [50 pts]: Help a professor convert his publications to bibTex

Overview

In Part 1 your goal is to parse the HTML page of a Professor containing some of his publications, and answer some questions. This page is provided to you in the file `data/publist_super_clean.html`. There are 44 publications in descending order from No. 244 to No. 200.

You are to use python's **regular expressions**, a powerful way of parsing text. You may **not** use any parsing tool such as Beautiful Soup yet. In doing so you will get more familiar with three of the common file formats for storing and transferring data, which are:

- CSV, a text-based file format used for storing tabular data that are separated by some delimiter, usually comma or space.
- HTML/XML, the stuff the web is made of.
- JavaScript Object Notation(JSON), a text-based open standard designed for transmitting structured data over the web.

Question 1: Parsing using Regular Expressions

1.1 Write a function called `get_pubs` that takes a .html filename as an input and returns a string containing the HTML page in this file (see definition below). Call this function using `data/publist_super_clean.html` as input and name the returned string `prof_pubs`.

1.2 Calculate how many times the author named '`C.M. Friend`' appears in the list of publications.

1.3 Find all unique journals and copy them in a variable named `journals`.

1.4 Create a list named `pub_authors` whose elements are strings containing the authors' names for each paper.

Hints

- Look for patterns in the HTML tags that reveal where each piece of information such as the title of the paper, the names of the authors, the journal name, is stored. For example, you might notice that the journal name(s) is contained between the <I> HTML tag.
- Each publication has multiple authors.
- `C.M. Friend` also shows up as `Cynthia M. Friend` in the file. Count just `C. M. Friend`.
- There is a comma at the end of the string of authors. You can choose to keep it in the string or remove it and put it back when you write the string as a BibTex entry.
- You want to remove duplicates from the list of journals.

Resources

- **Regular expressions:** a) https://docs.python.org/3.3/library/re.html, b) https://regexone.com, and c) https://docs.python.org/3/howto/regex.html.
- **HTML:** if you are not familiar with HTML see https://www.w3schools.com/html/ or one of the many tutorials on the internet.
- **Document Object Model (DOM):** for more on this programming interface for HTML and XML documents see https://www.w3schools.com/js/js_htmldom.asp.

1.1

In [2]:

```
# import the regular expressions library
import re
```

In [3]:

```
# use this file
pub_filename = 'data/publist_super_clean.html'
```

In [4]:

```
# definition of get_pubs
def get_pubs(filename: str) -> str:
    '''Open the file using the filename.

       Args:
           filename: A string name of the file.

       Returns:
           A string containing the HTML page ready to be parsed.
    '''
    with open(filename, 'r') as f:
        line = f.readline()
        html = ""
        while line != "":
            html += line
            line = f.readline()
    return html
```

In [5]:

```
page = get_pubs(pub_filename)
limit_print = 800
```

```
print(page[0:limit_print])
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
   "http://www.w3.org/TR/html4/loose.dtd">

<TITLE>Kaxiras E journal publications</TITLE>
<HEAD>
<meta http-equiv="Content-Type" content="text/html;charset=UTF-8">
<LINK REL="stylesheet" TYPE="text/css" HREF="../styles/style_pubs.css">
<META NAME="description" CONTENT="">
<META NAME="keywords" CONTENT="Kaxiras E, Multiscale Methods, Computational Materials" >
</HEAD>

<BODY>

<OL START=244>
<LI>
<A HREF="Papers/2011/PhysRevB_84_125411_2011.pdf" target="paper244">
&amp;quot;Approaching the intrinsic band gap in suspended high-mobility graphene nanoribbons&amp;quot;</A>
<BR>Ming-Wei Lin, Cheng Ling, Luis A. Agapito, Nicholas Kioussis, Yiyang Zhang, Mark Ming-Cheng Ch
eng,
<I>PHYSICAL REVIEW B </I> <b>84</b>,  125411 (2011)
<BR>
</LI>
</OL
```

1.2

In [6]:

```
# I am looking for this unique variation, as the exercise says.
cmfriend_counts = re.findall(r'C.M. Friend', page)
print("The author C.M. Friend appeared {} times.".format(len(cmfriend_counts)))
```

```
The author C.M. Friend appeared 5 times.
```

1.3

I did not find Biophysical journal in the original file.

In [7]:

```
journals = re.findall(r'<I>(.*) </I>', page)
#journals = np.unique(journals)
```

In [8]:

```
for j in journals:
    print(j)
```

```
PHYSICAL REVIEW B
PHYSICAL REVIEW B
PHYSICAL REVIEW B
PHYSICAL REVIEW B
Phil. Trans. R. Soc. A
New Journal of Physics
Nano Lett.
Langmuir
J. Phys. Chem. Lett.
J. Phys. Chem. C
J. Phys. Chem. C
J. Chem. Phys.
Chem. Eur. J.
Catal. Sci. Technol.
ACSNano.
Acta Mater.
New J. Phys.
Phys. Rev. B
2010 ACM/IEEE International Conference for High Performance
Molec. Phys.
```

```
Top. Catal.
Phys. Rev. Lett.
NanoLett.
Phys. Rev. B
J. Chem. Theory Comput.
Comp. Phys. Comm.
Concurrency Computat.: Pract. Exper.
Sol. St. Comm.
Phys. Rev. Lett.
Energy & Environmental Sci.
Comp. Phys. Comm.
J. Phys. Chem. C
Int. J. Cardiovasc. Imaging
Phys. Rev. B
J. Stat. Mech: Th. and Exper.
Phys. Rev. E - Rap. Comm.
J. Phys. Chem. B
Phys. Rev. Lett.
Phys. Rev. Lett.
Phys. Rev. E - Rap. Comm.
Phys. Rev. Lett.
J. Chem. Phys.
J. Phys. Chem. C
Sci. Model. Simul.
Phys. Rev. B
```

1.4 Create a list named pub_authors whose elements are strings containing the authors' names for each paper.

In [9]:

```
pub_authors = re.findall(r'<BR> *(.*)\n<I>', page)
# observe the number of authors is equal the number of publications.
print(len(pub_authors))
```

```
45
```

In [10]:

```
# check your code: print the list of strings containing the author(s)' names
for item in pub_authors:
    print (item)
```

```
Ming-Wei Lin, Cheng Ling, Luis A. Agapito, Nicholas Kioussis, Yiyang Zhang, Mark Ming-Cheng Cheng,
JAdam Gali, Efthimios Kaxiras, Gergely T. Zimanyi, Sheng Meng,
Jan M. Knaup, Han Li, Joost J. Vlassak, and Efthimios Kaxiras,
Martin Heiss, Sonia Conesa-Boj, Jun Ren, Hsiang-Han Tseng, Adam Gali,
Simone Melchionna, Efthimios Kaxiras, Massimo Bernaschi and Sauro Succi,
J R Maze, A Gali, E Togan, Y Chu, A Trifonov,
Kejie Zhao, Wei L. Wang, John Gregoire, Matt Pharr, Zhigang Suo,
Masataka Katono, Takeru Bessho, Sheng Meng, Robin Humphry-Baker, Guido Rothenberger,
Thomas D. Kuhne, Tod A. Pascal, Efthimios Kaxiras, and Yousung Jung,
Sheng Meng, Efthimios Kaxiras, Md. K. Nazeeruddin, and Michael Gratzel,
Bingjun Xu, Jan Haubrich, Thomas A. Baker, Efthimios Kaxiras, and Cynthia M. Friend,
Jun Ren, Sheng Meng, Yi-Lin Wang, Xu-Cun Ma, Qi-Kun Xue, Efthimios Kaxiras,
Jan Haubrich, Efthimios Kaxiras, and Cynthia M. Friend,
Thomas A. Baker, Bingjun Xu, Stephen C. Jensen, Cynthia M. Friend and Efthimios Kaxiras,
Youdong Mao, Wei L. Wang, Dongguang Wei, Efthimios Kaxiras, and Joseph G. Sodroski,
H. Li, J.M. Knaup, E. Kaxiras and J.J. Vlassak,
W.L. Wang and E. Kaxiras,
L.A. Agapito, N. Kioussis and E. Kaxiras,
A. Peters, S. Melchionna, E. Kaxiras, J. Latt, J. Sircar, S. Succi,
J. Ren, E. Kaxiras and S. Meng,
T.A. Baker, E. Kaxiras and C.M. Friend,
H.P. Chen, R.K. Kalia, E. Kaxiras, G. Lu, A. Nakano, K. Nomura,
S. Meng and E. Kaxiras,
C.L. Chang, S.K.R.S. Sankaranarayanan, D. Ruzmetov, M.H. Engelhard, E. Kaxiras and S. Ramanathan,
T.A. Baker, C.M. Friend and E. Kaxiras,
S. Melchionna, M. Bernaschi, S. Succi, E. Kaxiras, F.J. Rybicki, D. Mitsouras, A.U. Coskun and C.L
. Feldman,
M. Bernaschi, M. Fatica, S. Melchionna, S. Succi and E. Kaxiras,
E. Manousakis, J. Ren, S. Meng and E. Kaxiras,
A. Gali, E. Janzen, P. Deak, G. Kresse and E. Kaxiras,
S.K.R.S. Sankaranarayanan, E. Kaxiras and S. Ramanathan,
```

```
M. Bernaschi, S. Melchionna, S. Succi, M. Fyta, E. Kaxiras
T.A. Baker, B.J. Xu, X.Y. Liu, E. Kaxiras and C.M. Friend,
F.J. Rybicki, S. Melchionna, D. Mitsouras, A.U. Coskun, A.G. Whitmore, E. Kaxiras, S. Succi, P.H.
Stone and C.L. Feldman,
H. Chen, W.G. Zhu, E. Kaxiras, and Z.Y. Zhang,
M. Fyta, S. Melchionna, M. Bernaschi, E. Kaxiras and S. Succi,
E.M. Kotsalis, J.H. Walther, E. Kaxiras and P. Koumoutsakos,
C.E. Lekka, J. Ren, S. Meng and E. Kaxiras,
W.L. Wang, O.V. Yazyev, S. Meng and E. Kaxiras,
A. Gali and E. Kaxiras,
S. Melchionna, M. Bernaschi, M. Fyta, E. Kaxiras and S. Succi,
S.K.R.S. Sankaranarayanan, E. Kaxiras, S. Ramanathan,
T.A. Baker, C.M. Friend and E. Kaxiras,
T.A. Baker, C.M. Friend and E. Kaxiras,
E. Kaxiras and S. Succi,
E. Manousakis, J. Ren, S. Meng and E. Kaxiras,
```

Your output should look like this (a line for each paper's author(s) string, with or without the comma)

S. Meng and E. Kaxiras,
G. Lu and E. Kaxiras,
E. Kaxiras and S. Yip,
...
Simone Melchionna, Efthimios Kaxiras, Massimo Bernaschi and Sauro Succi,
J R Maze, A Gali, E Togan, Y Chu, A Trifonov,
E Kaxiras, and M D Lukin,

Question 2: Parsing and Converting to bibTex using Beautiful Soup

A lot of the bibliographic and publication information is displayed in various websites in a not-so-structured HTML files. Some publishers prefer to store and transmit this information in a .bibTex file which has the following format:

```
@article { _number_
     author = John Doyle
     title = Interaction between atoms
     URL = Papers/PhysRevB_81_085406_2010.pdf
     journal = Phys. Rev. B
     volume = 81
}


@article
{    author = Ming-Wei Lin, Cheng Ling, Luis A. Agapito, Nicholas Kioussis, Yiyang Zhang,
Mark Ming-Cheng Cheng
     title = "Approaching the intrinsic band gap in suspended high-mobility graphene
nanoribbons"
     URL = Papers/2011/PhysRevB_84_125411_2011.pdf
     journal = PHYSICAL REVIEW B
     volume = 84
}
```

About the bibTex format.

In Question 2 you are given an .html file containing a list of papers scraped from the author's website and you are to write the information into .bibTex format. We used regular expressions for parsing HTML in the previous question but just regular expressions are hard to use in parsing real-life websites. A useful tool is [BeautifulSoup] (http://www.crummy.com/software/BeautifulSoup/) (BS). You will parse the same file, this time using BS, which makes parsing HTML a lot easier.

2.1 Write a function called `make_soup` that accepts a filename for an HTML file and returns a BS object.

2.2 Write a function that reads in the BS object, parses it, converts it into the .bibTex format using python string manipulation and regular expressions, and writes the data into `publist.bib`. You will need to create that file in your folder.

HINT

- Inspect the HTML code for tags that indicate information chunks such as `title` of the paper. You had already done this in Part

- Inspect the HTML code for tags that indicate information chunks such as `title` of the paper. You had already done this in Part 1 when you figured out how to get the name of the journal from the HTML code. The `find_all` method of BeautifulSoup might be useful.
- Question 2.2 is better handled if you break the code into functions, each performing a small task such as finding the author(s) for each paper.
- Make sure you catch exceptions when needed.
- Regular expressions are a great tool for string manipulation.

Resources

- BeautifulSoup Tutorial.
- More about the BibTex format.

In [11]:

```
# import the necessary libraries
from bs4 import BeautifulSoup
from sys import argv
from urllib.request import urlopen
from urllib.error import HTTPError
```

2.1

In [12]:

```
# your code here

# definition of make_soup
def make_soup(filename: str) -> BeautifulSoup:
    '''Open the file and convert into a BS object.

       Args:
           filename: A string name of the file.

       Returns:
           A BS object containing the HTML page.
    '''
    raw_html = get_pubs(filename)
    soup = BeautifulSoup(raw_html, 'html5lib')
    return soup
```

In [13]:

```
# check your code: print the Beautiful Soup object, you should see an HTML page
soup = make_soup(pub_filename)
print(soup)
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html><head><title>Kaxiras E journal publications</title>

<meta content="text/html;charset=utf-8" http-equiv="Content-Type"/>
<link href="../styles/style_pubs.css" rel="stylesheet" type="text/css"/>
<meta content="" name="description"/>
<meta content="Kaxiras E, Multiscale Methods, Computational Materials" name="keywords"/>
</head>

<body>

<ol start="244">
<li>
<a href="Papers/2011/PhysRevB_84_125411_2011.pdf" target="paper244">
"Approaching the intrinsic band gap in suspended high-mobility graphene nanoribbons"</a>
<br/>Ming-Wei Lin, Cheng Ling, Luis A. Agapito, Nicholas Kioussis, Yiyang Zhang, Mark Ming-Cheng C
heng,
<i>PHYSICAL REVIEW B </i> <b>84</b>,  125411 (2011)
<br/>
</li>
</ol>

<ol start="243">
```


"Effect of symmetry breaking on the optical absorption of semiconductor nanoparticles"

JAdam Gali, Efthimios Kaxiras, Gergely T. Zimanyi, Sheng Meng,
PHYSICAL REVIEW B 84, 035325 (2011)

<ol start="242">

"Influence of CH2 content and network defects on the elastic properties of organosilicate glasses"

Jan M. Knaup, Han Li, Joost J. Vlassak, and Efthimios Kaxiras,
PHYSICAL REVIEW B 83, 054204 (2011)

<ol start="241">

"Direct correlation of crystal structure and optical properties in wurtzite/zinc-blende
GaAs nanowire heterostructures"

Martin Heiss, Sonia Conesa-Boj, Jun Ren, Hsiang-Han Tseng, Adam Gali,
PHYSICAL REVIEW B 83, 045303 (2011)

<ol start="240">

"Endothelial shear stress from large-scale blood flow simulations"

Simone Melchionna, Efthimios Kaxiras, Massimo Bernaschi and Sauro Succi,
Phil. Trans. R. Soc. A 369, 2354-2361 (2011)

<ol start="239">

"Properties of nitrogen-vacancy centers in diamond:
the group theoretic approach"

J R Maze, A Gali, E Togan, Y Chu, A Trifonov,
New Journal of Physics 13, 025025 (2011)

<ol start="238">

"Lithium-Assisted Plastic Deformation of Silicon Electrodes in
Lithium-Ion Batteries: A First-Principles Theoretical Study"

Kejie Zhao, Wei L. Wang, John Gregoire, Matt Pharr, Zhigang Suo,
Nano Lett. 11, 2962-2967 (2011)

<ol start="237">

"D-π-A Dye System Containing Cyano-Benzoic Acid as Anchoring
Group for Dye-Sensitized Solar Cells"

 Masataka Katono, Takeru Bessho, Sheng Meng, Robin Humphry-Baker, Guido Rothenberger,
Langmuir 27, 14248-14252 (2011)

<ol start="236">

"New Insights into the Structure of the Vapor/Water
Interface from Large-Scale First-Principles Simulations"

 Thomas D. Kuhne, Tod A. Pascal, Efthimios Kaxiras, and Yousung Jung,
J. Phys. Chem. Lett. 2, 105-113 (2011)

<ol start="235">

"Design of Dye Acceptors for Photovoltaics from First-Principles
Calculations"

 Sheng Meng, Efthimios Kaxiras, Md. K. Nazeeruddin, and Michael Gratzel,
J. Phys. Chem. C 115, 9276-9282 (2011)

<ol start="234">

"Theoretical Study of O-Assisted Selective Coupling of Methanol on
Au(111)"

 Bingjun Xu, Jan Haubrich, Thomas A. Baker, Efthimios Kaxiras, and Cynthia M. Friend,
J. Phys. Chem. C 115, 3703-3708 (2011)

<ol start="233">

"Properties of copper (fluoro-)phthalocyanine layers deposited
on epitaxial graphene"

 Jun Ren, Sheng Meng, Yi-Lin Wang, Xu-Cun Ma, Qi-Kun Xue, Efthimios Kaxiras,
J. Chem. Phys. 134, 194706 (2011)

<ol start="232">

"The Role of Surface and Subsurface Point Defects for Chemical Model
Studies on TiO2: A First-Principles Theoretical Study of Formaldehyde
Bonding on Rutile TiO2(110)"

 Jan Haubrich, Efthimios Kaxiras, and Cynthia M. Friend,
Chem. Eur. J. 17, 4496-4506(2011)

<ol start="231">

"Role of defects in propene adsorption and reaction on a partially
O-covered Au(111) surface"

 Thomas A. Baker, Bingjun Xu, Stephen C. Jensen, Cynthia M. Friend and Efthimios Kaxiras,
Catal. Sci. Technol. 1, 1166-1174 (2011)

<ol start="230">

"Graphene Structures at an Extreme
Degree of Buckling"

 Youdong Mao, Wei L. Wang, Dongguang Wei, Efthimios Kaxiras, and Joseph G. Sodroski,
ACSNano. ,5, 1395-1400 (2011)

<ol start="229">

"Stiffening of organosilicate glasses by organic cross-linking"

 H. Li, J.M. Knaup, E. Kaxiras and J.J. Vlassak,
Acta Mater. ,
59, 44-52 (2011).

<ol start="228">

"Graphene hydrate: theoretical prediction of a new insulating
form of graphene"

 W.L. Wang and E. Kaxiras,
New J. Phys. ,
12, 125012 (2010).

<ol start="227">

"Electric-field control of magnetism in graphene quantum dots:
Ab initio calculations"

 L.A. Agapito, N. Kioussis and E. Kaxiras,
Phys. Rev. B ,
82, 201411 (2010).

<ol start="226">

"Multiscale simulation of cardiovascular flows on the IBM Bluegene/P:
full heart-circulation system at near red-blood cell resolution"

 A. Peters, S. Melchionna, E. Kaxiras, J. Latt, J. Sircar, S. Succi,
2010 ACM/IEEE International Conference for High Performance ,
doi: 10.1109/SC.2010.33 (2010).

<ol start="225">

"Optical properties of clusters and molecules from real-time time-dependent
density functional theory using a self-consistent field"

 J. Ren, E. Kaxiras and S. Meng,
Molec. Phys. 108, 1829-1844 (2010).

<ol start="224">

"Insights from Theory on the Relationship Between Surface Reactivity
and Gold Atom Release"

 T.A. Baker, E. Kaxiras and C.M. Friend,
Top. Catal. 53, 365-377 (2010).

<ol start="223">

"Embrittlement of Metal by Solute Segregation-Induced Amorphization"

 H.P. Chen, R.K. Kalia, E. Kaxiras, G. Lu, A. Nakano, K. Nomura,
Phys. Rev. Lett. 104, 155502 (2010).

<ol start="222">

"Electron and Hole Dynamics in Dye-Sensitized Solar Cells:
Influencing Factors and Systematic Trends"

 S. Meng and E. Kaxiras,
NanoLett. 10, 1238-1247 (2010).

<ol start="221">

"Compositional tuning of ultrathin surface oxides on metal and alloy
substrates using photons: Dynamic simulations and experiments"

 C.L. Chang, S.K.R.S. Sankaranarayanan, D. Ruzmetov, M.H. Engelhard, E. Kaxiras and S.
Ramanathan,
Phys. Rev. B 81, 085406 (2010).

<ol start="220">

"Local Bonding Effects in the Oxidation of CO on Oxygen-Covered
Au(111) from Ab Initio Molecular Dynamics Simulations"

 T.A. Baker, C.M. Friend and E. Kaxiras,
J. Chem. Theory Comput. 6, 279-287 (2010).

<ol start="219">

"Hydrokinetic approach to large-scale cardiovascular blood flow"

 S. Melchionna, M. Bernaschi, S. Succi, E. Kaxiras, F.J. Rybicki, D. Mitsouras, A.U. Coskun a
nd C.L. Feldman,
Comp. Phys. Comm. 181, 462-472 (2010).

<ol start="218">

"A flexible high-performance Lattice Boltzmann GPU code for the
simulations of fluid flows in complex geometries"

 M. Bernaschi, M. Fatica, S. Melchionna, S. Succi and E. Kaxiras,
Concurrency Computat.: Pract. Exper.
22, 1-14 (2010).

<ol start="217">

"Is the nature of magnetic order in copper-oxides and iron-pnictides
different?"

 E. Manousakis, J. Ren, S. Meng and E. Kaxiras,
Sol. St. Comm. 150, 62-65 (2010).

<ol start="216">

"Theory of Spin-Conserving Excitation of the N-V Center in Diamond"

 A. Gali, E. Janzen, P. Deak, G. Kresse and E. Kaxiras,
Phys. Rev. Lett. 103, 186404 (2009).

<ol start="215">

"Electric field tuning of oxygen stoichiometry at oxide surfaces:
molecular dynamics isimulations studies iof zirconia"

 S.K.R.S. Sankaranarayanan, E. Kaxiras and S. Ramanathan,
Energy & Environmental Sci. 2, 1196-1204 (2009).

<ol start="214">

"MUPHY: A parallel MUlti PHYsics/scale code for high performance
bio-fluidic simulations"

 M. Bernaschi, S. Melchionna, S. Succi, M. Fyta, E. Kaxiras
Comp. Phys. Comm. 180, 1495-1502 (2009).

<ol start="213">

"Nature of Oxidation of the Au(111) Surface: Experiment and
Theoretical Investigation"

 T.A. Baker, B.J. Xu, X.Y. Liu, E. Kaxiras and C.M. Friend,
J. Phys. Chem. C 113, 16561-16564 (2009).

<ol start="212">

"Prediction of coronary artery plaque progression and potential rupture
from 320-detector row prospectively ECG-gated single heart beat CT angiography:
Lattice Boltzmann evaluation of endothelial shear stress"

 F.J. Rybicki, S. Melchionna, D. Mitsouras, A.U. Coskun, A.G. Whitmore, E. Kaxiras, S. Succi,
P.H. Stone and C.L. Feldman,
Int. J. Cardiovasc. Imaging 25, 289-299 (2009).

<ol start="211">

"Optimization of Mn doping in group-IV-based dilute magnetic semiconductors
by electronic codopants"

 H. Chen, W.G. Zhu, E. Kaxiras, and Z.Y. Zhang,
Phys. Rev. B 79, 235202 (2009).

<ol start="210">

"Numerical simulation of conformational variability in biopolymer translocation
through wide nanopores"

 M. Fyta, S. Melchionna, M. Bernaschi, E. Kaxiras and S. Succi,
J. Stat. Mech: Th. and Exper. 06, P06009 (2009).

<ol start="209">

"Control algorithm for multiscale flow simulations of water"

 E.M. Kotsalis, J.H. Walther, E. Kaxiras and P. Koumoutsakos,
Phys. Rev. E - Rap. Comm. 79, 045701 (2009).

<ol start="208">

"Structural, Electronic, and Optical Properties of Representative Cu-Flavonoid Complexes"

C.E. Lekka, J. Ren, S. Meng and E. Kaxiras,
J. Phys. Chem. B 113, 6478-6483 (2009).

<ol start="207">

"Topological Frustration in Graphene Nanoflakes: Magnetic Order and Spin Logic Devices"

 W.L. Wang, O.V. Yazyev, S. Meng and E. Kaxiras,
Phys. Rev. Lett. 102, 157201 (2009).

<ol start="206">

"Comment on 'Ab initio Electronic and Optical Properties of the N-V-Center in Diamond'"

 A. Gali and E. Kaxiras,
Phys. Rev. Lett. 102, 149703 (2009).

<ol start="205">

"Quantized biopolymer translocation through nanopores: Departure from simple scaling"

 S. Melchionna, M. Bernaschi, M. Fyta, E. Kaxiras and S. Succi,
Phys. Rev. E - Rap. Comm. 79, 030901 (2009).


```
<ol start="204">
<li>
<a href="Papers/PhysRevLett_102_095504_2009.pdf" target="paper204">
"Atomistic Simulation of Field Enhanced Oxidation of Al(1000) Beyond the Mott Potential"
</a>
<br/>S.K.R.S. Sankaranarayanan, E. Kaxiras, S. Ramanathan,
<i>Phys. Rev. Lett. </i> <b>102</b>, 095504 (2009).
<br/>
</li>
</ol>

<ol start="203">
<li>
<a href="Papers/JChemPhys_130_084701_2009.pdf" target="paper203">
"Effects of chlorine and oxygen coverage on the structure of the Au(111) surface"
</a>
<br/>T.A. Baker, C.M. Friend and E. Kaxiras,
<i>J. Chem. Phys. </i> <b>130</b>, 084701 (2009).
<br/>
</li>
</ol>

<ol start="202">
<li>
<a href="Papers/JPhysChemC_113_3232_2009.pdf" target="paper202">
"Atomic Oxygen Adsorption on Au(111) Surfaces with Defects"
</a>
<br/>T.A. Baker, C.M. Friend and E. Kaxiras,
<i>J. Phys. Chem. C </i> <b>113</b>, 3232-3238 (2009).
<br/>
</li>
</ol>

<ol start="201">
<li>
<a href="Papers/SciModSim_15_59_2008.pdf" target="paper201">
"Multiscale simulations of complex systems: computation meets reality"
</a>
<br/>E. Kaxiras and S. Succi,
<i>Sci. Model. Simul. </i> <b>15</b>, 59-65 (2008).
<br/>
</li>
</ol>

<ol start="200">
<li>
<a href="Papers/PhysRevB_78_205112_2008.pdf" target="paper200">
"Effective Hamiltonian for FeAs-based superconductors"
</a>
<br/>E. Manousakis, J. Ren, S. Meng and E. Kaxiras,
<i>Phys. Rev. B </i> <b>78</b>, 205112 (2008).
<br/>
</li>
</ol>


</body></html>
```

2.2

In [14]:

```
def get_author(ol) -> str:
    if ol.li:
        bs_item = ol.li.encode(formatter='html5')
        author = re.findall(b'<br>(.*)', bs_item)[0].decode('utf-8').strip(",")
        return author
    return ''

def get_title(ol) -> str:
    if ol.li.a:
        title = ol.li.a.text.strip('"\n')
        return title
    return ''
```

```
def get_url(ol) -> str:
    if ol.li.a.attrs['href']:
        url = ol.li.a.attrs['href']
        return url
    return ''

def get_journal(ol) -> str:
    if ol.li.i:
        journal = ol.li.i.text.strip()
        return journal
    return ''

def get_volume(ol) -> str:
    if ol.li.b:
        volume = ol.li.b.text.strip()
        return volume
    return ''

def write_bibtex(name: str):

    with open('publist.bib', 'w') as f:

        for ol in soup.body.find_all('ol'):
            f.write("@article")
            f.write("\n{")
            f.write("    author = {}".format(get_author(ol)))
            f.write("\n")
            f.write("     title = {}".format(get_title(ol)))
            f.write("\n")
            f.write("     URL = {}".format(get_url(ol)))
            f.write("\n")
            f.write("     journal = {}".format(get_journal(ol)))
            f.write("\n")
            f.write("     volume = {}".format(get_volume(ol)))
            f.write("\n}\n\n")

    print("DONE")

write_bibtex('publist.bib')
```

```
DONE
```

In [15]:

```
# check your code: print the BibTex file
f = open('publist.bib','r')
print (f.read())
```

```
@article
{    author = Ming-Wei Lin, Cheng Ling, Luis A. Agapito, Nicholas Kioussis, Yiyang Zhang, Mark
Ming-Cheng Cheng
     title = Approaching the intrinsic band gap in suspended high-mobility graphene nanoribbons
     URL = Papers/2011/PhysRevB_84_125411_2011.pdf
     journal = PHYSICAL REVIEW B
     volume = 84
}

@article
{    author = JAdam Gali, Efthimios Kaxiras, Gergely T. Zimanyi, Sheng Meng
     title = Effect of symmetry breaking on the optical absorption of semiconductor nanoparticles
     URL = Papers/2011/PhysRevB_84_035325_2011.pdf
     journal = PHYSICAL REVIEW B
     volume = 84
}

@article
{    author = Jan M. Knaup, Han Li, Joost J. Vlassak, and Efthimios Kaxiras
     title = Influence of CH2 content and network defects on the elastic properties of
organosilicate glasses
     URL = Papers/2011/PhysRevB_83_054204_2011.pdf
     journal = PHYSICAL REVIEW B
     volume = 83
}
```

```
@article
{     author = Martin Heiss, Sonia Conesa-Boj, Jun Ren, Hsiang-Han Tseng, Adam Gali
      title = Direct correlation of crystal structure and optical properties in wurtzite/zinc-
blende
GaAs nanowire heterostructures
      URL = Papers/2011/PhysRevB_83_045303_2011.pdf
      journal = PHYSICAL REVIEW B
      volume = 83
}

@article
{     author = Simone Melchionna, Efthimios Kaxiras, Massimo Bernaschi and Sauro Succi,
      title = Endothelial shear stress from large-scale blood flow simulations
      URL = Papers/2011/PhilTransRSocA_369_2354_2011.pdf
      journal = Phil. Trans. R. Soc. A
      volume = 369
}

@article
{     author = J R Maze, A Gali, E Togan, Y Chu, A Trifonov
      title = Properties of nitrogen-vacancy centers in diamond:
the group theoretic approach
      URL = Papers/2011/NewJPhys_13_025025_2011.pdf
      journal = New Journal of Physics
      volume = 13
}

@article
{     author = Kejie Zhao, Wei L. Wang, John Gregoire, Matt Pharr, Zhigang Suo
      title = Lithium-Assisted Plastic Deformation of Silicon Electrodes in
Lithium-Ion Batteries: A First-Principles Theoretical Study
      URL = Papers/2011/NanoLett_11_2962-2967_2011.pdf
      journal = Nano Lett.
      volume = 11
}

@article
{     author =  Masataka Katono, Takeru Bessho, Sheng Meng, Robin Humphry-Baker, Guido Rothenberger
      title = D-π-A Dye System Containing Cyano-Benzoic Acid as Anchoring
Group for Dye-Sensitized Solar Cells
      URL = Papers/2011/Langmuir_27_14248_2011.pdf
      journal = Langmuir
      volume = 27
}

@article
{     author =  Thomas D. Kuhne, Tod A. Pascal, Efthimios Kaxiras, and Yousung Jung
      title = New Insights into the Structure of the Vapor/Water
Interface from Large-Scale First-Principles Simulations
      URL = Papers/2011/JPhysChemLett_2_105-113_2011.pdf
      journal = J. Phys. Chem. Lett.
      volume = 2
}

@article
{     author =  Sheng Meng, Efthimios Kaxiras, Md. K. Nazeeruddin, and Michael Gratzel
      title = Design of Dye Acceptors for Photovoltaics from First-Principles
Calculations
      URL = Papers/2011/JPhysChemC_115_9276-9282_2011.pdf
      journal = J. Phys. Chem. C
      volume = 115
}

@article
{     author =  Bingjun Xu, Jan Haubrich, Thomas A. Baker, Efthimios Kaxiras, and Cynthia M. Friend
      title = Theoretical Study of O-Assisted Selective Coupling of Methanol on
Au(111)
      URL = Papers/2011/JPhysChemC_115_3703-3708_2011.pdf
      journal = J. Phys. Chem. C
      volume = 115
}

@article
{     author =  Jun Ren, Sheng Meng, Yi-Lin Wang, Xu-Cun Ma, Qi-Kun Xue, Efthimios Kaxiras
      title = Properties of copper (fluoro-)phthalocyanine layers deposited
on epitaxial graphene
      URL = Papers/2011/JChemPhys_134_194706_2011.pdf
```

journal = J. Chem. Phys.
volume = 134
}

@article
{ author = Jan Haubrich, Efthimios Kaxiras, and Cynthia M. Friend
title = The Role of Surface and Subsurface Point Defects for Chemical Model
Studies on TiO2: A First-Principles Theoretical Study of Formaldehyde
Bonding on Rutile TiO2(110)
URL = Papers/2011/Chemistry_17_4496-4506_2011.pdf
journal = Chem. Eur. J.
volume = 17
}

@article
{ author = Thomas A. Baker, Bingjun Xu, Stephen C. Jensen, Cynthia M. Friend and Efthimios
Kaxiras
title = Role of defects in propene adsorption and reaction on a partially
O-covered Au(111) surface
URL = Papers/2011/CatalSciTechnol_1_1166_2011.pdf
journal = Catal. Sci. Technol.
volume = 1
}

@article
{ author = Youdong Mao, Wei L. Wang, Dongguang Wei, Efthimios Kaxiras, and Joseph G. Sodroski
title = Graphene Structures at an Extreme
Degree of Buckling
URL = Papers/2011/ACSNano_5_1395-1400_2011.pdf
journal = ACSNano.
volume = 5
}

@article
{ author = H. Li, J.M. Knaup, E. Kaxiras and J.J. Vlassak
title = Stiffening of organosilicate glasses by organic cross-linking
URL = Papers/ActaMater_59_44-52_2011.pdf
journal = Acta Mater.
volume = 59
}

@article
{ author = W.L. Wang and E. Kaxiras
title = Graphene hydrate: theoretical prediction of a new insulating
form of graphene
URL = Papers/NewJPhys_12_125012_2010.pdf
journal = New J. Phys.
volume = 12
}

@article
{ author = L.A. Agapito, N. Kioussis and E. Kaxiras
title = Electric-field control of magnetism in graphene quantum dots:
Ab initio calculations
URL = Papers/PhysRevB_82_201411_2010.pdf
journal = Phys. Rev. B
volume = 82
}

@article
{ author = A. Peters, S. Melchionna, E. Kaxiras, J. Latt, J. Sircar, S. Succi,
title = Multiscale simulation of cardiovascular flows on the IBM Bluegene/P:
full heart-circulation system at near red-blood cell resolution
URL = Papers/IEEE-SC10_2010.pdf
journal = 2010 ACM/IEEE International Conference for High Performance
volume =
}

@article
{ author = J. Ren, E. Kaxiras and S. Meng,
title = Optical properties of clusters and molecules from real-time time-dependent
density functional theory using a self-consistent field
URL = Papers/MolPhys_108_1829-1844_2010.pdf
journal = Molec. Phys.
volume = 108
}

```
@article
{    author =  T.A. Baker, E. Kaxiras and C.M. Friend,
     title = Insights from Theory on the Relationship Between Surface Reactivity
and Gold Atom Release
     URL = Papers/TopicsCatal_53_365-377_2010.pdf
     journal = Top. Catal.
     volume = 53
}

@article
{    author =  H.P. Chen, R.K. Kalia, E. Kaxiras, G. Lu, A. Nakano, K. Nomura
     title = Embrittlement of Metal by Solute Segregation-Induced Amorphization
     URL = Papers/PhysRevLett_104_155502_2010.pdf
     journal = Phys. Rev. Lett.
     volume = 104
}

@article
{    author =  S. Meng and E. Kaxiras
     title = Electron and Hole Dynamics in Dye-Sensitized Solar Cells:
Influencing Factors and Systematic Trends
     URL = Papers/NanoLett_10_1238-1247_2010.pdf
     journal = NanoLett.
     volume = 10
}

@article
{    author =  C.L. Chang, S.K.R.S. Sankaranarayanan, D. Ruzmetov, M.H. Engelhard, E. Kaxiras and
S. Ramanathan,
     title = Compositional tuning of ultrathin surface oxides on metal and alloy
substrates using photons: Dynamic simulations and experiments
     URL = Papers/PhysRevB_81_085406_2010.pdf
     journal = Phys. Rev. B
     volume = 81
}

@article
{    author =  T.A. Baker, C.M. Friend and E. Kaxiras,
     title = Local Bonding Effects in the Oxidation of CO on Oxygen-Covered
Au(111) from Ab Initio Molecular Dynamics Simulations
     URL = Papers/JChemTheComp_6_279-287_2010.pdf
     journal = J. Chem. Theory Comput.
     volume = 6
}

@article
{    author =  S. Melchionna, M. Bernaschi, S. Succi, E. Kaxiras, F.J. Rybicki, D. Mitsouras, A.U.
Coskun and C.L. Feldman,
     title = Hydrokinetic approach to large-scale cardiovascular blood flow
     URL = Papers/CompPhysComm_181_462-472_2010.pdf
     journal = Comp. Phys. Comm.
     volume = 181
}

@article
{    author =  M. Bernaschi, M. Fatica, S. Melchionna, S. Succi and E. Kaxiras,
     title = A flexible high-performance Lattice Boltzmann GPU code for the
simulations of fluid flows in complex geometries
     URL = Papers/ConcComp_22_1-14_2010.pdf
     journal = Concurrency Computat.: Pract. Exper.
     volume = 22
}

@article
{    author =  E. Manousakis, J. Ren, S. Meng and E. Kaxiras,
     title = Is the nature of magnetic order in copper-oxides and iron-pnictides
different?
     URL = Papers/SolStComm_150_62-65_2010.pdf
     journal = Sol. St. Comm.
     volume = 150
}

@article
{    author =  A. Gali, E. Janzen, P. Deak, G. Kresse and E. Kaxiras,
     title = Theory of Spin-Conserving Excitation of the N-V Center in Diamond
     URL = Papers/PhysRevLett_103_186404_2009.pdf
     journal = Phys. Rev. Lett.
```

```
     volume = 103
}

@article
{    author =  S.K.R.S. Sankaranarayanan, E. Kaxiras and S. Ramanathan,
     title = Electric field tuning of oxygen stoichiometry at oxide surfaces:
molecular dynamics isimulations studies iof zirconia
     URL = Papers/EnEnviSci_2_1196-1204_2009.pdf
     journal = Energy & Environmental Sci.
     volume = 2
}

@article
{    author =  M. Bernaschi, S. Melchionna, S. Succi, M. Fyta, E. Kaxiras
     title = MUPHY: A parallel MUlti PHYsics/scale code for high performance
bio-fluidic simulations
     URL = Papers/CompPhysComm_180_1495-1502_2009.pdf
     journal = Comp. Phys. Comm.
     volume = 180
}

@article
{    author =  T.A. Baker, B.J. Xu, X.Y. Liu, E. Kaxiras and C.M. Friend,
     title = Nature of Oxidation of the Au(111) Surface: Experiment and
Theoretical Investigation
     URL = Papers/JPhysChemC_113_16561-16564_2009.pdf
     journal = J. Phys. Chem. C
     volume = 113
}

@article
{    author =  F.J. Rybicki, S. Melchionna, D. Mitsouras, A.U. Coskun, A.G. Whitmore, E. Kaxiras,
S. Succi, P.H. Stone and C.L. Feldman,
     title = Prediction of coronary artery plaque progression and potential rupture
from 320-detector row prospectively ECG-gated single heart beat CT angiography:
Lattice Boltzmann evaluation of endothelial shear stress
     URL = Papers/IntJCardImag_25_289-299_2009.pdf
     journal = Int. J. Cardiovasc. Imaging
     volume = 25
}

@article
{    author =  H. Chen, W.G. Zhu, E. Kaxiras, and Z.Y. Zhang
     title = Optimization of Mn doping in group-IV-based dilute magnetic semiconductors
by electronic codopants
     URL = Papers/PhysRevB_79_235202_2009.pdf
     journal = Phys. Rev. B
     volume = 79
}

@article
{    author =  M. Fyta, S. Melchionna, M. Bernaschi, E. Kaxiras and S. Succi
     title = Numerical simulation of conformational variability in biopolymer translocation
through wide nanopores
     URL = Papers/JStatMech_2009.pdf
     journal = J. Stat. Mech: Th. and Exper.
     volume = 06
}

@article
{    author =  E.M. Kotsalis, J.H. Walther, E. Kaxiras and P. Koumoutsakos,
     title = Control algorithm for multiscale flow simulations of water
     URL = Papers/PhysRevE_79_045701RC_2009.pdf
     journal = Phys. Rev. E - Rap. Comm.
     volume = 79
}

@article
{    author = C.E. Lekka, J. Ren, S. Meng and E. Kaxiras
     title = Structural, Electronic, and Optical Properties of Representative Cu-Flavonoid
Complexes
     URL = Papers/JPhysChemB_113_6478_2009.pdf
     journal = J. Phys. Chem. B
     volume = 113
}

@article
```

```
{     author =  W.L. Wang, O.V. Yazyev, S. Meng and E. Kaxiras,
      title = Topological Frustration in Graphene Nanoflakes: Magnetic Order and Spin Logic Devices
      URL = Papers/PhysRevLett_102_157201_2009.pdf
      journal = Phys. Rev. Lett.
      volume = 102
}

@article
{     author =  A. Gali and E. Kaxiras,
      title = Comment on 'Ab initio Electronic and Optical Properties of the N-V-Center in Diamond'
      URL = Papers/PhysRevLett_102_149703_2009.pdf
      journal = Ab initio
      volume = 102
}

@article
{     author =  S. Melchionna, M. Bernaschi, M. Fyta, E. Kaxiras and S. Succi,
      title = Quantized biopolymer translocation through nanopores: Departure from simple scaling
      URL = Papers/PhysRevE_79_030901RC_2009.pdf
      journal = Phys. Rev. E - Rap. Comm.
      volume = 79
}

@article
{     author = S.K.R.S. Sankaranarayanan, E. Kaxiras, S. Ramanathan
      title = Atomistic Simulation of Field Enhanced Oxidation of Al(1000) Beyond the Mott
Potential
      URL = Papers/PhysRevLett_102_095504_2009.pdf
      journal = Phys. Rev. Lett.
      volume = 102
}

@article
{     author = T.A. Baker, C.M. Friend and E. Kaxiras
      title = Effects of chlorine and oxygen coverage on the structure of the Au(111) surface
      URL = Papers/JChemPhys_130_084701_2009.pdf
      journal = J. Chem. Phys.
      volume = 130
}

@article
{     author = T.A. Baker, C.M. Friend and E. Kaxiras
      title = Atomic Oxygen Adsorption on Au(111) Surfaces with Defects
      URL = Papers/JPhysChemC_113_3232_2009.pdf
      journal = J. Phys. Chem. C
      volume = 113
}

@article
{     author = E. Kaxiras and S. Succi,
      title = Multiscale simulations of complex systems: computation meets reality
      URL = Papers/SciModSim_15_59_2008.pdf
      journal = Sci. Model. Simul.
      volume = 15
}

@article
{     author = E. Manousakis, J. Ren, S. Meng and E. Kaxiras,
      title = Effective Hamiltonian for FeAs-based superconductors
      URL = Papers/PhysRevB_78_205112_2008.pdf
      journal = Phys. Rev. B
      volume = 78
}
```

Part B [50 pts]: Follow the stars in IMDb's list of "The Top 100 Stars for 2017"

Overview

Overview

In Part 3 your goal is to extract information from IMDb's Top 100 Stars for 2017 (https://www.imdb.com/list/ls025814950/) and perform some analysis on each star in the list. In particular we are interested to know: a) how many performers made their first movie at 17? b) how many performers started as child actors? c) who is the most proliferate actress or actor in IMDb's list of the Top 100 Stars for 2017? . These questions are addressed in more details in the Questions below.

When data is **not** given to us in a file, we need to fetch them using one of the following ways:

- download a file from a source URL
- query a database
- query a web API
- scrape data from the web page

Question 1: Web Scraping Using Beautiful Soup

1.1 Download the webpage of the "Top 100 Stars for 2017" (https://www.imdb.com/list/ls025814950/) into a `requests` object and name it `my_page`. Explain what the following attributes are:

- `my_page.text`,
- `my_page.status_code`,
- `my_page.content`.

1.2 Create a Beautiful Soup object named `star_soup` giving `my_page` as input.

1.3 Write a function called `parse_stars` that accepts `star_soup` as its input and generates a list of dictionaries named `starlist` (see definition below). One of the fields of this dictionary is the `url` of each star's individual page, which you need to scrape and save the contents in the `page` field. Note that there is a ton of information about each star on these webpages.

1.4 Write a function called `create_star_table` to extract information about each star (see function definition for the exact information to extract). **Only extract information from the first box on each star's page. If the first box is acting, consider only acting credits and the star's acting debut, if the first box is Directing, consider only directing credits and directorial debut.**

1.5 Now that you have scraped all the info you need, it's a good practice to save the last data structure you created to disk. That way if you need to re-run from here, you don't need to redo all these requests and parsing. Save this information to a JSON file and **submit** this JSON file in Canvas with your notebook.

1.6 Import the contents of the teaching staff's JSON file (`data/staff_starinfo.json`) into a pandas dataframe. Check the types of variables in each column and clean these variables if needed. Add a new column to your dataframe with the age of each actor when they made first movie (name this column `age_at_first_movie`).

1.7 You are now ready to answer the following intriguing questions:

- How many performers made their first movie at 17?
- How many performers started as child actors? Define child actor as a person less than 12 years old.
- Who is the most prolific actress or actor in IMDb's list of the Top 100 Stars for 2017?

1.8 Make a plot of the number of credits versus the name of actor/actress.

Hints

- Create a variable that groups actors/actresses by the age of their first movie. Use pandas' `.groupby` to divide the dataframe into groups of performers that for example started performing as children (age $<$ 12). The grouped variable is a `GroupBy` pandas object and this object has all of the information needed to then apply some operation to each of the groups.
- When cleaning the data make sure the variables with which you are performing calculations are in numerical format.
- The column with the year has some values that are double, e.g. **'2000-2001'** and the column with age has some empty cells. You need to deal with these before performing calculations on the data!
- You should include both movies and TV shows.

Resources

- The `requests` library makes working with HTTP requests powerful and easy. For more on the `requests` library see http://docs.python-requests.org/

In [16]:

```
import requests
import time
```

1.1

In [17]:

```
# your code here
page = "https://www.imdb.com/list/ls025814950/"
my_page = requests.get(page)
```

Your answers here

---

1. `my_page.text`

Answer: It's the text of the entire page encoded by request package.

1. `my_page.status_code`

Answer: It carries the response status code. It will show what happened while the connection was made. Several number codes can be raised, like 200 (good one) or 404 (bad one).

1. `my_page.content`

Answer: It's the response of the page as bytes.

1.2

In [18]:

```
# your code here
star_soup = BeautifulSoup(my_page.content, 'html.parser')
```

In [19]:

```
# check your code - you should see an HTML page
print (star_soup.prettify()[:400])
```

```
<!DOCTYPE html>
<html xmlns:fb="http://www.facebook.com/2008/fbml" xmlns:og="http://ogp.me/ns#">
 <head>
  <meta charset="utf-8"/>
  <meta content="IE=edge" http-equiv="X-UA-Compatible"/>
  <meta content="app-id=342792525, app-argument=imdb:///list/ls025814950?src=mdot" name="apple-
itunes-app"/>
  <script type="text/javascript">
   var IMDbTimer={starttime: new Date().getTime(),pt:'java'};
  </scr
```

1.3

```
    Function
    --------
    parse_stars

    Input
    ------
    star_soup: the soup object with the scraped page

    Returns
    -------
    a list of dictionaries; each dictionary corresponds to a star profile and has the following
    data:
```

```
        name: the name of the actor/actress as it appears at the top
        gender: 0 or 1: translate the word 'actress' into 1 and 'actor' into '0'
        url: the url of the link under their name that leads to a page with details
        page: the string containing the soup of the text in their individual info page (from
    url)

    Example:
    --------
    {'name': Tom Hardy,
      'gender': 0,
      'url': https://www.imdb.com/name/nm0362766/?ref_=nmls_hd,
      'page': BS object with 'html text acquired by scraping the 'url' page'
    }
```

In [27]:

```
# your code here
def parse_stars(soup) -> list:
    stars_list = []
    act_list = star_soup.findAll('div', {'class': 'lister-item mode-detail'})
    gender_bool = lambda find: int(find == -1)
    for act in act_list:
        dict_stars = {}
        name = act.find('div', {'class': 'lister-item-content'}).a.text.strip()
        print("INFO - {} - DONE".format(name))
        url = act.find('div', {'class': 'lister-item-content'}).a['href']
        url = "https://www.imdb.com" + url
        gender = gender_bool(act.find('div', {'class': 'lister-item-content'}).p.text.find("Actor")
)
        try:
            page = requests.get(url)
            page_soup = BeautifulSoup(page.content, 'html.parser')
        except requests.exceptions.ConnectionError:
            page_soup = ''
        time.sleep(5)
        dict_stars['name'] = name
        dict_stars['gender'] = gender
        dict_stars['url'] = url
        dict_stars['page'] = page_soup
        stars_list.append(dict_stars)
    return stars_list
starlist = parse_stars(star_soup)
```

```
INFO - Gal Gadot - DONE
INFO - Tom Hardy - DONE
INFO - Emilia Clarke - DONE
INFO - Alexandra Daddario - DONE
INFO - Bill Skarsgård - DONE
INFO - Pom Klementieff - DONE
INFO - Ana de Armas - DONE
INFO - Dan Stevens - DONE
INFO - Sofia Boutella - DONE
INFO - Katherine Langford - DONE
INFO - Karen Gillan - DONE
INFO - Margot Robbie - DONE
INFO - Felicity Jones - DONE
INFO - Emma Stone - DONE
INFO - Dylan Minnette - DONE
INFO - Jennifer Lawrence - DONE
INFO - Alicia Vikander - DONE
INFO - Britt Robertson - DONE
INFO - Ruby Rose - DONE
INFO - Brie Larson - DONE
INFO - Keanu Reeves - DONE
INFO - Sophia Lillis - DONE
INFO - Jessica Henwick - DONE
INFO - Cara Delevingne - DONE
INFO - Haley Bennett - DONE
INFO - Luke Evans - DONE
INFO - Teresa Palmer - DONE
INFO - Tom Holland - DONE
INFO - Alison Brie - DONE
```

```
INFO - Robin Wright - DONE
INFO - Zendaya - DONE
INFO - Emma Watson - DONE
INFO - Scarlett Johansson - DONE
INFO - Dafne Keen - DONE
INFO - Kelly Rohrbach - DONE
INFO - Eiza González - DONE
INFO - Laura Haddock - DONE
INFO - Mary Elizabeth Winstead - DONE
INFO - Taron Egerton - DONE
INFO - Anya Taylor-Joy - DONE
INFO - Elizabeth Debicki - DONE
INFO - Katheryn Winnick - DONE
INFO - Sean Young - DONE
INFO - Bill Paxton - DONE
INFO - Charlie Hunnam - DONE
INFO - Yvonne Strahovski - DONE
INFO - Jason Momoa - DONE
INFO - Lily James - DONE
INFO - Jodie Whittaker - DONE
INFO - Ryan Gosling - DONE
INFO - Adrianne Palicki - DONE
INFO - Millie Bobby Brown - DONE
INFO - Allison Williams - DONE
INFO - Chris Pratt - DONE
INFO - Katherine Waterston - DONE
INFO - Tom Cruise - DONE
INFO - Johnny Depp - DONE
INFO - James McAvoy - DONE
INFO - Travis Fimmel - DONE
INFO - Charlize Theron - DONE
INFO - Cole Sprouse - DONE
INFO - Kaya Scodelario - DONE
INFO - Abigail Breslin - DONE
INFO - Daisy Ridley - DONE
INFO - Emily Browning - DONE
INFO - Christopher Nolan - DONE
INFO - Zoe Saldana - DONE
INFO - Lena Headey - DONE
INFO - Hugh Jackman - DONE
INFO - Kit Harington - DONE
INFO - Leonardo DiCaprio - DONE
INFO - Malina Weissman - DONE
INFO - Finn Jones - DONE
INFO - Chloë Grace Moretz - DONE
INFO - Alexander Skarsgård - DONE
INFO - Amy Adams - DONE
INFO - Bella Thorne - DONE
INFO - Rebecca Ferguson - DONE
INFO - Julia Garner - DONE
INFO - Joan Crawford - DONE
INFO - Kate Mara - DONE
INFO - Chris Pine - DONE
INFO - Bryce Dallas Howard - DONE
INFO - Halston Sage - DONE
INFO - Kate Beckinsale - DONE
INFO - Connie Nielsen - DONE
INFO - Auli'i Cravalho - DONE
INFO - Mädchen Amick - DONE
INFO - Serinda Swan - DONE
INFO - Dave Bautista - DONE
INFO - Rose Leslie - DONE
INFO - Annabelle Wallis - DONE
INFO - Zoey Deutch - DONE
INFO - Sophie Turner - DONE
INFO - Dakota Johnson - DONE
INFO - Rosamund Pike - DONE
INFO - Elodie Yung - DONE
INFO - Shailene Woodley - DONE
INFO - Nina Dobrev - DONE
INFO - Christian Navarro - DONE
```

In [36]:

```
# Testing if some has a problem
for star in starlist:
```

```
    if star['page'] == '':
        url = star['url']
        page = requests.get(url)
        page_soup = BeautifulSoup(page.content, 'html.parser')
        star['page'] = page_soup
        time.sleep(5)
```

In [37]:

```
# this list is large because of the html code into the `page` field
# to get a better picture, print only the first element
print(starlist[0]['name'])
print(starlist[0]['gender'])
print(starlist[0]['url'])
```

```
Gal Gadot
1
https://www.imdb.com/name/nm2933757
```

1.4

```
    Function
    --------
    create_star_table

    Input
    ------
    the starlist

    Returns
    -------

    a list of dictionaries; each dictionary corresponds to a star profile and has the following
    data:

        star_name: the name of the actor/actress as it appears at the top
        gender: 0 or 1 (1 for 'actress' and 0 for 'actor')
        year_born : year they were born
        first_movie: title of their first movie or TV show
        year_first_movie: the year they made their first movie or TV show
        credits: number of movies or TV shows they have made in their career.

    --------
    Example:

    {'star_name': Tom Hardy,
      'gender': 0,
      'year_born': 1997,
      'first_movie' : 'Batman',
      'year_first_movie' : 2017,
      'credits' : 24}
```

In [38]:

```
# your code here
def create_star_table(starlist: list) -> list:
    star_table = []
    for star in starlist:
        star_dict = {}
        try:
            year = star['page'].find('div', {'id': "name-born-info"}).time['datetime'][0:4]
        except AttributeError:
            year = np.nan
        filmography = star['page'].find('div',{'id': "filmography"})

        first_movie = filmography.find_all('div')[1].find_all('div', {'class': "filmo-row"})[-1]
```

```
        movie = first_movie.b.a.text.strip()
        year_movie = first_movie.span.text.strip()
        credit = re.findall(r'([0-9]*) credit', filmography.div.text)[0]

        star_dict['star_name'] = star['name']
        star_dict['gender'] = star['gender']
        star_dict['year_born'] = year
        star_dict['first_movie'] = movie
        star_dict['year_first_movie'] = year_movie
        star_dict['credits'] = int(credit)

        star_table.append(star_dict)
    return star_table
```

In [39]:

```
# RUN THIS CELL ONLY ONCE - IT WILL TAKE SOME TIME TO RUN
star_table = []
star_table = create_star_table(starlist)
```

In [43]:

```
# check your code
print(star_table)
```

```
[{'star_name': 'Gal Gadot', 'gender': 1, 'year_born': '1985', 'first_movie': 'Shemesh',
'year_first_movie': '1999', 'credits': 32}, {'star_name': 'Tom Hardy', 'gender': 0, 'year_born':
'1977', 'first_movie': 'Tommaso', 'year_first_movie': '2001', 'credits': 56}, {'star_name':
'Emilia Clarke', 'gender': 1, 'year_born': '1986', 'first_movie': 'Doctors', 'year_first_movie': '
2009', 'credits': 20}, {'star_name': 'Alexandra Daddario', 'gender': 1, 'year_born': '1986',
'first_movie': 'All My Children', 'year_first_movie': '2002-2003', 'credits': 59}, {'star_name':
'Bill Skarsgård', 'gender': 0, 'year_born': '1990', 'first_movie': 'Järngänget',
'year_first_movie': '2000', 'credits': 36}, {'star_name': 'Pom Klementieff', 'gender': 1,
'year_born': '1986', 'first_movie': 'Perigosa Obsessão', 'year_first_movie': '2007', 'credits': 37
}, {'star_name': 'Ana de Armas', 'gender': 1, 'year_born': '1988', 'first_movie': 'Una rosa de
Francia', 'year_first_movie': '2006', 'credits': 31}, {'star_name': 'Dan Stevens', 'gender': 0, 'y
ear_born': '1982', 'first_movie': 'Frankenstein', 'year_first_movie': '2004', 'credits': 44},
{'star_name': 'Sofia Boutella', 'gender': 1, 'year_born': '1982', 'first_movie': 'Le défi',
'year_first_movie': '2002', 'credits': 27}, {'star_name': 'Katherine Langford', 'gender': 1,
'year_born': '1996', 'first_movie': 'Story of Miss Oxygen', 'year_first_movie': '2015', 'credits':
11}, {'star_name': 'Karen Gillan', 'gender': 1, 'year_born': '1987', 'first_movie': 'Rebus', 'year
_first_movie': '2006', 'credits': 63}, {'star_name': 'Margot Robbie', 'gender': 1, 'year_born':
'1990', 'first_movie': 'City Homicide', 'year_first_movie': '2008', 'credits': 40}, {'star_name':
'Felicity Jones', 'gender': 1, 'year_born': '1983', 'first_movie': 'The Treasure Seekers',
'year_first_movie': '1996', 'credits': 41}, {'star_name': 'Emma Stone', 'gender': 1, 'year_born':
'1988', 'first_movie': 'The New Partridge Family', 'year_first_movie': '2005', 'credits': 48}, {'s
tar_name': 'Dylan Minnette', 'gender': 0, 'year_born': '1996', 'first_movie': 'Dois Homens e
Meio', 'year_first_movie': '2005', 'credits': 57}, {'star_name': 'Jennifer Lawrence', 'gender': 1,
'year_born': '1990', 'first_movie': 'Monk: Um Detetive Diferente', 'year_first_movie': '2006', 'cr
edits': 34}, {'star_name': 'Alicia Vikander', 'gender': 1, 'year_born': '1988', 'first_movie':
'Min balsamerade mor', 'year_first_movie': '2002', 'credits': 45}, {'star_name': 'Britt
Robertson', 'gender': 1, 'year_born': '1990', 'first_movie': 'Sheena', 'year_first_movie': '2000',
'credits': 55}, {'star_name': 'Ruby Rose', 'gender': 1, 'year_born': '1986', 'first_movie': 'Boys
Like You', 'year_first_movie': '2011', 'credits': 22}, {'star_name': 'Brie Larson', 'gender': 1, '
year_born': '1989', 'first_movie': 'The Tonight Show with Jay Leno', 'year_first_movie': '1998', '
credits': 63}, {'star_name': 'Keanu Reeves', 'gender': 0, 'year_born': '1964', 'first_movie': "Han
gin' In", 'year_first_movie': '1984', 'credits': 106}, {'star_name': 'Sophia Lillis', 'gender': 1,
'year_born': '2002', 'first_movie': 'The Lipstick Stain', 'year_first_movie': '2013', 'credits': 1
7}, {'star_name': 'Jessica Henwick', 'gender': 1, 'year_born': '1992', 'first_movie': 'Escola para
Garotas Bonitas e Piradas 2', 'year_first_movie': '2009', 'credits': 31}, {'star_name': 'Cara
Delevingne', 'gender': 1, 'year_born': '1992', 'first_movie': 'Anna Karenina', 'year_first_movie':
'2012/I', 'credits': 25}, {'star_name': 'Haley Bennett', 'gender': 1, 'year_born': '1988',
'first_movie': 'Letra e Música', 'year_first_movie': '2007', 'credits': 28}, {'star_name': 'Luke E
vans', 'gender': 0, 'year_born': '1979', 'first_movie': 'Taboo', 'year_first_movie': '2003',
'credits': 42}, {'star_name': 'Teresa Palmer', 'gender': 1, 'year_born': '1986', 'first_movie': 'O
rientation', 'year_first_movie': '2004', 'credits': 34}, {'star_name': 'Tom Holland', 'gender': 0,
'year_born': '1996', 'first_movie': 'O Mundo dos Pequeninos', 'year_first_movie': '2010',
'credits': 31}, {'star_name': 'Alison Brie', 'gender': 1, 'year_born': '1982', 'first_movie':
'Stolen Poem', 'year_first_movie': '2004', 'credits': 67}, {'star_name': 'Robin Wright', 'gender':
1, 'year_born': '1966', 'first_movie': 'The Yellow Rose', 'year_first_movie': '1983-1984',
'credits': 57}, {'star_name': 'Zendaya', 'gender': 1, 'year_born': '1996', 'first_movie': 'Bella T
horne & Zendaya: Watch Me', 'year_first_movie': '2011', 'credits': 44}, {'star_name': 'Emma
Watson', 'gender': 1, 'year_born': '1990', 'first_movie': 'Harry Potter e a Pedra Filosofal', 'yea
r_first_movie': '2001', 'credits': 23}, {'star_name': 'Scarlett Johansson', 'gender': 1,
```

'year_born': '1984', 'first_movie': 'O Anjo da Guarda', 'year_first_movie': '1994', 'credits': 67}
, {'star_name': 'Dafne Keen', 'gender': 1, 'year_born': '2005', 'first_movie': 'The Refugees',
'year_first_movie': '2014-2015', 'credits': 4}, {'star_name': 'Kelly Rohrbach', 'gender': 1,
'year_born': '1990', 'first_movie': 'The New Normal', 'year_first_movie': '2013', 'credits': 16},
{'star_name': 'Eiza González', 'gender': 1, 'year_born': '1990', 'first_movie': 'Lola: Érase una v
ez', 'year_first_movie': '2007', 'credits': 21}, {'star_name': 'Laura Haddock', 'gender': 1,
'year_born': '1985', 'first_movie': 'My Family', 'year_first_movie': '2007', 'credits': 37}, {'sta
r_name': 'Mary Elizabeth Winstead', 'gender': 1, 'year_born': '1984', 'first_movie': 'O Toque de
um Anjo', 'year_first_movie': '1997', 'credits': 56}, {'star_name': 'Taron Egerton', 'gender': 0,
'year_born': '1989', 'first_movie': 'The Last of the Haussmans', 'year_first_movie': '2012', 'cred
its': 25}, {'star_name': 'Anya Taylor-Joy', 'gender': 1, 'year_born': '1996', 'first_movie':
'Academia de Vampiros: O Beijo das Sombras', 'year_first_movie': '2014', 'credits': 27},
{'star_name': 'Elizabeth Debicki', 'gender': 1, 'year_born': '1990', 'first_movie': 'Depois dos 30
', 'year_first_movie': '2011', 'credits': 24}, {'star_name': 'Katheryn Winnick', 'gender': 1,
'year_born': '1977', 'first_movie': 'PSI Factor: Chronicles of the Paranormal',
'year_first_movie': '1999', 'credits': 66}, {'star_name': 'Sean Young', 'gender': 1, 'year_born':
'1959', 'first_movie': 'Jane Austen in Manhattan', 'year_first_movie': '1980', 'credits': 125}, {'
star_name': 'Bill Paxton', 'gender': 0, 'year_born': '1955', 'first_movie': 'Loucura da Mamãe', 'y
ear_first_movie': '1975', 'credits': 96}, {'star_name': 'Charlie Hunnam', 'gender': 0,
'year_born': '1980', 'first_movie': 'My Wonderful Life', 'year_first_movie': '1996', 'credits': 29
}, {'star_name': 'Yvonne Strahovski', 'gender': 1, 'year_born': '1982', 'first_movie': 'Double the
Fist', 'year_first_movie': '2004', 'credits': 37}, {'star_name': 'Jason Momoa', 'gender': 0,
'year_born': '1979', 'first_movie': 'S.O.S. Malibu', 'year_first_movie': '1999-2001', 'credits': 3
5}, {'star_name': 'Lily James', 'gender': 1, 'year_born': '1989', 'first_movie': 'Just William',
'year_first_movie': '2010', 'credits': 31}, {'star_name': 'Jodie Whittaker', 'gender': 1,
'year_born': '1982', 'first_movie': 'The Afternoon Play', 'year_first_movie': '2006', 'credits': 5
5}, {'star_name': 'Ryan Gosling', 'gender': 0, 'year_born': '1980', 'first_movie': 'Clube do
Terror', 'year_first_movie': '1995', 'credits': 45}, {'star_name': 'Adrianne Palicki', 'gender': 1
, 'year_born': '1983', 'first_movie': 'Rewrite', 'year_first_movie': '2003', 'credits': 44},
{'star_name': 'Millie Bobby Brown', 'gender': 1, 'year_born': '2004', 'first_movie': 'Era Uma Vez
no País das Maravilhas', 'year_first_movie': '2013', 'credits': 14}, {'star_name': 'Allison
Williams', 'gender': 1, 'year_born': '1988', 'first_movie': 'American Dreams', 'year_first_movie':
'2004', 'credits': 15}, {'star_name': 'Chris Pratt', 'gender': 0, 'year_born': '1979',
'first_movie': 'Cursed Part 3', 'year_first_movie': '2000', 'credits': 58}, {'star_name':
'Katherine Waterston', 'gender': 1, 'year_born': '1980', 'first_movie': 'Americana',
'year_first_movie': '2004', 'credits': 42}, {'star_name': 'Tom Cruise', 'gender': 0, 'year_born':
'1962', 'first_movie': 'Amor sem Fim', 'year_first_movie': '1981', 'credits': 50}, {'star_name': '
Johnny Depp', 'gender': 0, 'year_born': '1963', 'first_movie': 'A Hora do Pesadelo',
'year_first_movie': '1984', 'credits': 90}, {'star_name': 'James McAvoy', 'gender': 0,
'year_born': '1979', 'first_movie': 'The Near Room', 'year_first_movie': '1995', 'credits': 56}, {
'star_name': 'Travis Fimmel', 'gender': 0, 'year_born': '1979', 'first_movie': "Jennifer Lopez: I'
m Real", 'year_first_movie': '2001', 'credits': 33}, {'star_name': 'Charlize Theron', 'gender': 1,
'year_born': '1975', 'first_movie': 'Monster: Desejo Assassino', 'year_first_movie': '2003', 'cred
its': 21}, {'star_name': 'Cole Sprouse', 'gender': 0, 'year_born': '1992', 'first_movie': 'Grace U
nder Fire', 'year_first_movie': '1993-1998', 'credits': 37}, {'star_name': 'Kaya Scodelario',
'gender': 1, 'year_born': '1992', 'first_movie': 'Lunar', 'year_first_movie': '2009', 'credits': 2
4}, {'star_name': 'Abigail Breslin', 'gender': 1, 'year_born': '1996', 'first_movie': 'Toys R Us:
1999 Commercial', 'year_first_movie': '1999', 'credits': 48}, {'star_name': 'Daisy Ridley',
'gender': 1, 'year_born': '1992', 'first_movie': 'Memórias de Ontem', 'year_first_movie': '1991',
'credits': 33}, {'star_name': 'Emily Browning', 'gender': 1, 'year_born': '1988', 'first_movie': '
The Echo of Thunder', 'year_first_movie': '1998', 'credits': 30}, {'star_name': 'Christopher
Nolan', 'gender': 1, 'year_born': '1970', 'first_movie': 'Tarantella', 'year_first_movie': '1989',
'credits': 18}, {'star_name': 'Zoe Saldana', 'gender': 1, 'year_born': '1978', 'first_movie': 'Lei
& Ordem', 'year_first_movie': '1999', 'credits': 65}, {'star_name': 'Lena Headey', 'gender': 1,
'year_born': '1973', 'first_movie': "Terra d'Água", 'year_first_movie': '1992', 'credits': 85}, {'
star_name': 'Hugh Jackman', 'gender': 0, 'year_born': '1968', 'first_movie': 'Law of the Land', 'y
ear_first_movie': '1994', 'credits': 60}, {'star_name': 'Kit Harington', 'gender': 0, 'year_born':
'1986', 'first_movie': 'Silent Hill: Revelação', 'year_first_movie': '2012', 'credits': 16},
{'star_name': 'Leonardo DiCaprio', 'gender': 0, 'year_born': '1974', 'first_movie': 'Romper Room',
'year_first_movie': '1979', 'credits': 56}, {'star_name': 'Malina Weissman', 'gender': 1,
'year_born': '2003', 'first_movie': 'As Tartarugas Ninja', 'year_first_movie': '2014', 'credits':
6}, {'star_name': 'Finn Jones', 'gender': 0, 'year_born': '1988', 'first_movie': 'Hollyoaks
Later', 'year_first_movie': '2009', 'credits': 17}, {'star_name': 'Chloë Grace Moretz', 'gender':
1, 'year_born': '1997', 'first_movie': 'The Guardian', 'year_first_movie': '2004', 'credits': 71},
{'star_name': 'Alexander Skarsgård', 'gender': 0, 'year_born': '1976', 'first_movie': 'Åke och han
s värld', 'year_first_movie': '1984', 'credits': 64}, {'star_name': 'Amy Adams', 'gender': 1,
'year_born': '1974', 'first_movie': 'Lindas de Morrer', 'year_first_movie': '1999', 'credits': 62}
, {'star_name': 'Bella Thorne', 'gender': 1, 'year_born': '1997', 'first_movie': 'Ligado em Você',
'year_first_movie': '2003', 'credits': 105}, {'star_name': 'Rebecca Ferguson', 'gender': 1,
'year_born': '1983', 'first_movie': 'Nya tider', 'year_first_movie': '1999-2000', 'credits': 32},
{'star_name': 'Julia Garner', 'gender': 1, 'year_born': '1994', 'first_movie': 'The Dreamer', 'yea
r_first_movie': '2010/II', 'credits': 31}, {'star_name': 'Joan Crawford', 'gender': 1,
'year_born': '1904', 'first_movie': 'Lady of the Night', 'year_first_movie': '1925', 'credits': 10
7}, {'star_name': 'Kate Mara', 'gender': 1, 'year_born': '1983', 'first_movie': 'Lei & Ordem',
'year_first_movie': '1997', 'credits': 65}, {'star_name': 'Chris Pine', 'gender': 0, 'year_born':
'1980', 'first_movie': 'Plantão Médico', 'year_first_movie': '2003', 'credits': 57}, {'star_name':
'Bryce Dallas Howard', 'gender': 1, 'year_born': '1981', 'first_movie': 'O Tiro que não Saiu pela

Culatra', 'year_first_movie': '1989', 'credits': 38}, {'star_name': 'Halston Sage', 'gender': 1, '
year_born': '1993', 'first_movie': 'Victorious', 'year_first_movie': '2011', 'credits': 23},
{'star_name': 'Kate Beckinsale', 'gender': 1, 'year_born': '1973', 'first_movie': 'Couples',
'year_first_movie': '1975', 'credits': 51}, {'star_name': 'Connie Nielsen', 'gender': 1,
'year_born': '1965', 'first_movie': "Par où t'es rentré? On t'a pas vu sortir",
'year_first_movie': '1984', 'credits': 58}, {'star_name': "Auli'i Cravalho", 'gender': 1,
'year_born': '2000', 'first_movie': 'Moana: Um Mar de Aventuras', 'year_first_movie': '2016/I', 'c
redits': 11}, {'star_name': 'Mädchen Amick', 'gender': 1, 'year_born': '1970', 'first_movie':
'Days of Our Lives', 'year_first_movie': '1988', 'credits': 77}, {'star_name': 'Serinda Swan', 'ge
nder': 1, 'year_born': '1984', 'first_movie': "Neal 'N' Nikki", 'year_first_movie': '2005',
'credits': 43}, {'star_name': 'Dave Bautista', 'gender': 0, 'year_born': '1969', 'first_movie':
'OVW: Christmas Chaos', 'year_first_movie': '2001', 'credits': 71}, {'star_name': 'Rose Leslie', '
gender': 1, 'year_born': '1987', 'first_movie': 'Banged Up Abroad', 'year_first_movie': '2008', 'c
redits': 20}, {'star_name': 'Annabelle Wallis', 'gender': 1, 'year_born': '1984', 'first_movie': '
Dil Jo Bhi Kahey...', 'year_first_movie': '2005', 'credits': 35}, {'star_name': 'Zoey Deutch',
'gender': 1, 'year_born': '1994', 'first_movie': 'NCIS: Investigações Criminais',
'year_first_movie': '2011', 'credits': 27}, {'star_name': 'Sophie Turner', 'gender': 1,
'year_born': '1996', 'first_movie': 'Meu Outro Eu', 'year_first_movie': '2013', 'credits': 16}, {'
star_name': 'Dakota Johnson', 'gender': 1, 'year_born': '1989', 'first_movie': 'Loucos do
Alabama', 'year_first_movie': '1999', 'credits': 34}, {'star_name': 'Rosamund Pike', 'gender': 1,
'year_born': '1979', 'first_movie': 'A Rather English Marriage', 'year_first_movie': '1998',
'credits': 52}, {'star_name': 'Elodie Yung', 'gender': 1, 'year_born': '1981', 'first_movie': 'La
vie devant nous', 'year_first_movie': '2002-2003', 'credits': 27}, {'star_name': 'Shailene
Woodley', 'gender': 1, 'year_born': '1991', 'first_movie': 'Replacing Dad', 'year_first_movie': '1
999', 'credits': 38}, {'star_name': 'Nina Dobrev', 'gender': 1, 'year_born': '1989',
'first_movie': 'De Repente Grávida', 'year_first_movie': '2006', 'credits': 44}, {'star_name':
'Christian Navarro', 'gender': 0, 'year_born': nan, 'first_movie': 'Lei & Ordem: Crimes
Premeditados', 'year_first_movie': '2007', 'credits': 14}]

1.5

In [44]:

```
# your code here
import json
```

In [45]:

```
with open('starinfo.json', 'w', encoding='latin-1') as f:
    json.dump(star_table, f)
```

In [46]:

```
with open('starinfo.json', 'r', encoding='latin-1') as f:
    data = json.load(f)
```

1.6

In [48]:

```
# your code here
with open('data/staff_starinfo.json', 'r', encoding = 'latin-1') as f:
    data = json.load(f)
starinfo = pd.DataFrame(data)
starinfo.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 6 columns):
name                100 non-null object
gender              100 non-null int64
year_born           100 non-null object
first_movie         100 non-null object
year_first_movie    100 non-null object
credits             100 non-null object
dtypes: int64(1), object(5)
memory usage: 4.8+ KB
```

In [52]:

In [52]:

```
# cleaning Christian Navarro (I searched his year_born)
starinfo.set_value(99, 'year_born', 1991)
# The year of Daysy first movie is wrong in the website. I'm fixing it
starinfo.set_value(63, 'year_first_movie', '2012')
starinfo.year_born = starinfo.year_born.astype('int')
starinfo.gender = starinfo.gender.astype('bool')
# Taking the first year as the first year movie
starinfo.year_first_movie = [int(i[0:4]) for i in starinfo.year_first_movie]
starinfo.credits = starinfo.credits.astype('int')
starinfo.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 6 columns):
name                100 non-null object
gender              100 non-null bool
year_born           100 non-null int64
first_movie         100 non-null object
year_first_movie    100 non-null int64
credits             100 non-null int64
dtypes: bool(1), int64(3), object(2)
memory usage: 4.1+ KB
```

In [53]:

```
starinfo['age_at_first_movie'] = starinfo.year_first_movie - starinfo.year_born
```

1.7.1

In [54]:

```
# your code here
age = 17
performers_17 = starinfo[starinfo.age_at_first_movie == age]
print("{} performers made their first movie at {}". format(len(performers_17), age))
```

```
8 performers made their first movie at 17
```

1.7.2

In [55]:

```
# your code here. I considerer 'less' as < signal. Daysi is not here, because it was -1!
child = 12
performers_child = starinfo[starinfo.age_at_first_movie < child]
print("{} performers made their first movie when child (less than {} years)". format(len(performers
_child), child))
```

```
19 performers made their first movie when child (less than 12 years)
```

1.7.3

In [59]:

```
# your code here
act = starinfo[starinfo['credits'] == starinfo['credits'].max()][['name','credits']]
print("The most prolific actress in IMDb's list is ...")
time.sleep(1)
print(act)
```

```
The most prolific actress in IMDb's list is ...
          name  credits
42  Sean Young      122
```

1.8

In [60]:

```
plt.figure(figsize = (20,10))
plt.bar(list(range(len(starinfo.name))), starinfo.credits, color = 'green', alpha = 0.6)
plt.title("Number of Credits per Actor/Actress")
plt.xticks(list(range(len(starinfo.name))), starinfo.name,rotation=90)
plt.show()
```

Your answer here

In [1]:

```
from IPython.core.display import HTML
def css_styling(): styles = open("styles/cs109.css", "r").read(); return HTML(styles)
css_styling()
```

In [ ]:

```

```