Names of Group Members

Marcy Guo

Jialong Feng

Q1.) Web Crawling Tables

Q1.A.) Create a list of links for all the wikipedia pages for NYSE traded companies A-Z and 0-9

In [ ]:
```
# URL = "https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(A)"

base_url = "https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_("
links = []

for letter in range(ord('A'), ord('Z')+1):
    link = base_url + chr(letter) + ")"
    links.append(link)

# Add link with (0–9) as the end
link_09 = base_url + "0–9)"
links.append(link_09)

for i in links:
    print(i)

len(links)
```

```
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(A)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(B)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(C)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(D)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(E)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(F)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(G)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(H)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(I)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(J)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(K)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(L)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(M)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(N)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(O)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(P)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(Q)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(R)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(S)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(T)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(U)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(V)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(W)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(X)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(Y)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(Z)
https://en.wikipedia.org/wiki/Companies_listed_on_the_New_York_Stock_Exchange_(0–9)
```

Out[ ]: 27

Q1.B.) Crawl through all the URLs and make 1 DF with all the NYSE publically traded companies

In [ ]:
```
import pandas as pd
```

```
import requests
```

In [ ]:
```
all_data = []

for url in links:
    html = requests.get(url).content
    df_list = pd.read_html(html)
    all_data.append(df_list)

all_tables = [df for sublist in all_data for df in sublist]

df = pd.concat(all_tables, ignore_index=True)
```

In [ ]:
```
df.info()
print(df)
df
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2731 entries, 0 to 2730
Data columns (total 3 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   Stock name         2731 non-null   object
 1   Symbol             2731 non-null   object
 2   Country of origin  2731 non-null   object
dtypes: object(3)
memory usage: 64.1+ KB
                                  Stock name  Symbol Country of origin
0                       A. O. Smith Corporation     AOS               US
1                            A10 Networks, Inc.    ATEN               US
2                             AAC Holdings Inc.     AAC               US
3                               AAR Corporation     AIR               US
4                                  Aaron's Inc.     AAN               US
...                                         ...     ...              ...
2726     Zurn Elkay Water Solutions Corporation     ZWS     United States
2727  10X Capital Venture Acquisition Corp. III    VCXB     United States
2728  10X Capital Venture Acquisition Corp. III  VCXB.U     United States
2729                      3D Systems Corporation     DDD     United States
2730                                  3M Company     MMM     United States

[2731 rows x 3 columns]
```

Out[ ]:

	Stock name	Symbol	Country of origin
0	A. O. Smith Corporation	AOS	US
1	A10 Networks, Inc.	ATEN	US
2	AAC Holdings Inc.	AAC	US
3	AAR Corporation	AIR	US
4	Aaron's Inc.	AAN	US
...	...	...	...
2726	Zurn Elkay Water Solutions Corporation	ZWS	United States
2727	10X Capital Venture Acquisition Corp. III	VCXB	United States
2728	10X Capital Venture Acquisition Corp. III	VCXB.U	United States
2729	3D Systems Corporation	DDD	United States
2730	3M Company	MMM	United States

2731 rows × 3 columns

Q1.C.) What is the percetages of companies that contain 1 letter, 2 letters, 3 letters, 4 letters, 5 letters,... in the ticker (drop punctuation)?

In [ ]:
```
def clean_symbol(symbol):
    return ''.join(filter(str.isalpha, symbol))

df['Cleaned Symbol'] = df['Symbol'].apply(clean_symbol)

df['Symbol Length'] = df['Cleaned Symbol'].apply(len)

symbol_length_counts = df['Symbol Length'].value_counts(normalize=True) * 100

symbol_length_percentages = symbol_length_counts.sort_index().reset_index()
symbol_length_percentages.columns = ['Symbol Length', 'Percentage(%)']

print(df['Symbol Length'].value_counts())
print(symbol_length_percentages)
```

```
Symbol Length
3    1728
4     452
6     249
2     186
5      50
7      42
1      23
8       1
Name: count, dtype: int64
   Symbol Length  Percentage(%)
0              1       0.842182
1              2       6.810692
2              3      63.273526
3              4      16.550714
4              5       1.830831
5              6       9.117539
6              7       1.537898
7              8       0.036617
```

Q2.) Web Scraping Using Beautiful Soup

Q2.A.) Using Beautiful soup .findAll method you will webscrape the front page of Reddit. Get a list of all of the "timestamps"

In [ ]:
```
from selenium import webdriver
from bs4 import BeautifulSoup
import time
```

In [ ]:
```
driver = webdriver.Chrome()

url = 'https://www.reddit.com'
driver.get(url)

time.sleep(20)

page_source = driver.page_source
driver.quit()
```

In [ ]:
```
soup = BeautifulSoup(page_source, 'html.parser')
timestamps = soup.findAll('time', attrs={'datetime': True})

for time in timestamps:
    print(time['datetime'])
```

```
2024-04-27T23:26:38.435Z
2024-04-28T00:16:45.758Z
2024-04-28T00:34:40.974Z
2024-04-28T01:44:35.390Z
2024-04-27T21:59:41.564Z
2024-04-27T21:25:06.389Z
2024-04-27T23:16:29.168Z
2024-04-27T15:33:18.173Z
2024-04-27T10:48:40.715Z
2024-04-27T14:45:17.055Z
2024-04-27T18:52:25.282Z
2024-04-27T20:58:03.685Z
2024-04-27T21:15:57.154Z
2024-04-27T15:17:08.277Z
2024-04-27T22:26:28.201Z
2024-04-27T13:59:56.733Z
2024-04-27T22:29:43.358Z
2024-04-27T12:22:55.600Z
2024-04-27T18:38:38.678Z
2024-04-27T21:13:23.906Z
2024-04-28T02:34:02.258Z
2024-04-27T17:05:04.765Z
2024-04-27T16:05:15.410Z
2024-04-27T12:27:13.925Z
2024-04-27T23:19:11.558Z
2024-04-28T01:09:47.795Z
2024-04-27T20:52:08.474Z
2024-04-27T16:12:40.906Z
```

Q2.B.) Using the functions findChild, descendents, etc. locate the post title, text and post time into a dataframe.

In [ ]:
```
posts_data = []
articles = soup.findAll('article')

#articles
```

In [ ]:
```
for article in articles:
    post_dict = {}

    post_dict['Title'] = article.get('aria-label', 'No Title Found')
    p_tag = article.findChild('p', class_='px-md pt-md m-0 whitespace-normal text-14 leading-5 font

    if p_tag:
        post_dict['Text'] = p_tag.text.strip()
    else:
        post_dict['Text'] = "Text Not Found"

    # Timestamp extraction from 'created-timestamp' attribute
    post_dict['Timestamp'] = article.get('created-timestamp', 'Timestamp Not Found')

    time_tag = article.findChild('time')
    if time_tag and 'datetime' in time_tag.attrs:
        post_dict['Timestamp'] = time_tag['datetime']

    posts_data.append(post_dict)

# Create a DataFrame to neatly organize data
df = pd.DataFrame(posts_data)
print(df.head())
```

	Title	Text	Timestamp
0	As scary as they can be, alligators just don’t...	Pictures, gifs and videos of animals being derps	2024-04-27T23:26:38.435Z
1	Noticed my pupils are two different sizes.	Aww, cripes. I didn't know I'd have to write a...	2024-04-28T00:16:45.758Z
2	An elderly Lion in his final hours. \nPhotogra...	A place for photographs, pictures, and other i...	2024-04-28T00:34:40.974Z
3	Madlad behavior	Reddit's largest humor depository	2024-04-28T01:44:35.390Z
4	A picture is worth one sound	Screenshots of Black people being hilarious or...	2024-04-27T21:59:41.564Z

In [ ]:
```
df
```

Out[ ]:

	Title	Text	Timestamp
0	As scary as they can be, alligators just don't...	Pictures, gifs and videos of animals being derps	2024-04-27T23:26:38.435Z
1	Noticed my pupils are two different sizes.	Aww, cripes. I didn't know I'd have to write a...	2024-04-28T00:16:45.758Z
2	An elderly Lion in his final hours. \nPhotogra...	A place for photographs, pictures, and other i...	2024-04-28T00:34:40.974Z
3	Madlad behavior	Reddit's largest humor depository	2024-04-28T01:44:35.390Z
4	A picture is worth one sound	Screenshots of Black people being hilarious or...	2024-04-27T21:59:41.564Z
5	Supposed to be watching my parent's dog this w...	Things that make you go AWW! -- like puppies, ...	2024-04-27T21:25:06.389Z
6	Old man only has 25 views. Best fiddle I've ev...	Welcome! /r/MadeMeSmile is a place to share th...	2024-04-27T23:16:29.168Z
7	What's something that women say to men that th...	r/AskReddit is the place to ask and answer tho...	2024-04-27T15:33:18.173Z
8	TikTok will not be sold, Chinese parent ByteDa...	The place for news articles about current even...	2024-04-27T10:48:40.715Z
9	AITAH for separating from my husband because h...	this is a community like r/AmITheAsshole excep...	2024-04-27T14:45:17.055Z
10	Friend in college asked me to review her job a...	/r/facepalm - please sir can I have some more?	2024-04-27T18:52:25.282Z
11	Day three of snipers at Indiana University	A place for photographs, pictures, and other i...	2024-04-27T20:58:03.685Z
12	GOP caters to extremists for decades, surprise...	'I never thought leopards would eat MY face,' ...	2024-04-27T21:15:57.154Z
13	Never letting my bf stock the tp again...	jugkfmghgug	2024-04-27T15:17:08.277Z
14	LPT: If you rent a tool from Home Depot, and y...	Tips that improve your life in one way or anot...	2024-04-27T22:26:28.201Z
15	Whos boomer parents will be voting for trump j...	BoomersBeingFools is for images, videos, and s...	2024-04-27T13:59:56.733Z
16	Sent from my friend who says he's "Enlightened...	The subreddit for the weird, strange, odd and ...	2024-04-27T22:29:43.358Z
17	Your kids' mispronunciations of classmates names?	A community for those interested in names. You...	2024-04-27T12:22:55.600Z
18	Images of Apollo 11 and 12 taken my indias moo...	For the most interesting things on the internet	2024-04-27T18:38:38.678Z
19	"You want to go home? Why?! You only did CPR f...	BoomersBeingFools is for images, videos, and s...	2024-04-27T21:13:23.906Z
20	[Highlight] LeBron is extremely angry after Da...	A community for NBA discussion.	2024-04-28T02:34:02.258Z
21	Former beauty Queen, Miss Wyoming winner Joyce...	For anything truly interesting as fuck	2024-04-27T17:05:04.765Z
22	Is it just me or do girls do way better in sch...	Ask away!	2024-04-27T16:05:15.410Z
23	Bernie Sanders to Netanyahu: 'It Is Not Antise...	/r/Politics is for news and discussion about U...	2024-04-27T12:27:13.925Z
24	'Like a war zone': Emory University grapples w...	The place for news articles about current even...	2024-04-27T23:19:11.558Z
25	Grigori Perelman, mathematician who refused to...	A place for photographs, pictures, and other i...	2024-04-28T01:09:47.795Z

	Title	Text	Timestamp
26	Functional fake plants	Funny and interesting viral videos from around...	2024-04-27T20:52:08.474Z
27	Kristi Noem Faces Backlash Over Killing Her Ow...	For true stories that you could have sworn wer...	2024-04-27T16:12:40.906Z

Q3.) RegEx

Q3.A.) Using RegEx, get all the urls of ladder faculty profiles for UCLA Economics

In [ ]:
```
import re

URL = "https://economics.ucla.edu/faculty/ladder"


response = requests.get(URL)
webpage = response.text


if response.status_code != 200:
    print("Failed to load the page")
```

In [ ]:
```
soup = BeautifulSoup(webpage, 'html.parser')

pattern = re.compile(r'https://economics.ucla.edu/person/[^"]+')


faculty_urls = set()
for link in soup.find_all('a', href=True):
    if re.match(pattern, link['href']):
        faculty_urls.add(link['href'])

for url in sorted(faculty_urls):
    print(url)

len(faculty_urls)
```

```
https://economics.ucla.edu/person/aaron-tornell/
https://economics.ucla.edu/person/adriana-lleras-muney/
https://economics.ucla.edu/person/alexander-bloedel/
https://economics.ucla.edu/person/andres-santos/
https://economics.ucla.edu/person/andrew-atkeson/
https://economics.ucla.edu/person/ariel-burstein/
https://economics.ucla.edu/person/bernardo-s-silveira/
https://economics.ucla.edu/person/daniel-clark/
https://economics.ucla.edu/person/daniel-haanwinckel/
https://economics.ucla.edu/person/david-baqaee/
https://economics.ucla.edu/person/denis-chetverikov/
https://economics.ucla.edu/person/dora-costa/
https://economics.ucla.edu/person/felipe-goncalves/
https://economics.ucla.edu/person/gary-d-hansen/
https://economics.ucla.edu/person/hugo-hopenhayn/
https://economics.ucla.edu/person/ichiro-obara/
https://economics.ucla.edu/person/jay-lu/
https://economics.ucla.edu/person/jinyong-hahn/
https://economics.ucla.edu/person/joao-guerreiro/
https://economics.ucla.edu/person/john-asker/
https://economics.ucla.edu/person/jonathan-vogel/
https://economics.ucla.edu/person/juliana-londono-velez/
https://economics.ucla.edu/person/kathleen-mcgarry/
https://economics.ucla.edu/person/lee-e-ohanian/
https://economics.ucla.edu/person/martha-bailey/
https://economics.ucla.edu/person/martin-b-hackmann/
https://economics.ucla.edu/person/maurizio-mazzocco/
https://economics.ucla.edu/person/michael-rubens/
https://economics.ucla.edu/person/michela-giorcelli/
https://economics.ucla.edu/person/moritz-meyer-ter-vehn/
https://economics.ucla.edu/person/natalie-bau/
https://economics.ucla.edu/person/oleg-itskhoki/
https://economics.ucla.edu/person/pablo-fajgelbaum/
https://economics.ucla.edu/person/pierre-olivier-weill/
https://economics.ucla.edu/person/rodrigo-pinto/
https://economics.ucla.edu/person/rosa-liliana-matzkin/
https://economics.ucla.edu/person/saki-bigio/
https://economics.ucla.edu/person/shuyang-sheng/
https://economics.ucla.edu/person/simon-board/
https://economics.ucla.edu/person/sule-ozler/
https://economics.ucla.edu/person/till-von-wachter/
https://economics.ucla.edu/person/tomasz-sadzik/
https://economics.ucla.edu/person/will-rafey/
https://economics.ucla.edu/person/yotam-shem-tov/
https://economics.ucla.edu/person/zhipeng-liao/
```

Out[ ]: 45

Q3.B.) Webcrawl the links from A and use RegEx to get all the emails and phone numbers of ladder faculty profiles

In [ ]:
```
email_pattern = re.compile(r'\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,}\b')
phone_pattern = re.compile(r'\(?\d{3}\)?[-.\s]?\d{3}[-.\s]?\d{4}')

faculty_contact_info = {}

for url in faculty_urls:
    try:
        response = requests.get(url)
        if response.status_code == 200:
            soup = BeautifulSoup(response.text, 'html.parser')

            prof_name_tag = soup.find('h1', class_='name')
            prof_name = prof_name_tag.text.strip() if prof_name_tag else 'Name not found'

            prof_title_tag = soup.find('h2', class_='title')
            prof_title = prof_title_tag.text.strip() if prof_title_tag else 'Title not found'

            emails = set(email_pattern.findall(soup.get_text()))
```

```
            phones = set(phone_pattern.findall(soup.get_text()))

            if not emails:
                emails = 'No email found'

            if not phones:
                phones = 'No phone number found'

            faculty_contact_info[url] = {
                'Professor Name': prof_name,
                'Professor Title': prof_title,
                'Emails': emails,
                'Phone Numbers': phones
            }
        else:
            print(f"Failed to fetch {url} with status code {response.status_code}")
    except Exception as e:
        print(f"An error occurred while fetching {url}: {e}")
```

In [ ]:
```
faculty_df = pd.DataFrame.from_dict(faculty_contact_info, orient='index').reset_index()
faculty_df.columns = ['URL', 'Professor Name', 'Professor Title', 'Emails', 'Phone Numbers']
faculty_df['Emails'] = faculty_df['Emails'].apply(lambda x: ', '.join(x) if isinstance(x, set) else
faculty_df['Phone Numbers'] = faculty_df['Phone Numbers'].apply(lambda x: ', '.join(x) if isinstanc
faculty_df
```

Out[ ]:

	URL	Professor Name	Professor Title	Emails	Phone Numbers
0	https://economics.ucla.edu/person/shuyang-sheng/	Shuyang Sheng	Assistant Professor	ssheng@econ.ucla.edu	(310) 825-8018
1	https://economics.ucla.edu/person/andrew-atkeson/	Andrew Atkeson	Stanley M. Zimmerman Professor of Economics an...	andy@atkeson.net	No phone number found
2	https://economics.ucla.edu/person/david-baqaee/	David Baqaee	Associate Professor	baqaee@econ.ucla.edu	No phone number found
3	https://economics.ucla.edu/person/daniel-clark/	Daniel Clark	Assistant Professor	dclark@econ.ucla.edu	No phone number found
4	https://economics.ucla.edu/person/kathleen-mcg...	Kathleen McGarry	Professor	mcgarry@ucla.edu	(310) 825-1011
5	https://economics.ucla.edu/person/jinyong-hahn/	Jinyong Hahn	Department Chair	chair@econ.ucla.edu, hahn@econ.ucla.edu	(310) 825-1011
6	https://economics.ucla.edu/person/jay-lu/	Jay Lu	Associate Professor	jay@econ.ucla.edu	(310) 825-7380
7	https://economics.ucla.edu/person/natalie-bau/	Natalie Bau	Associate Professor	nbau@g.ucla.edu	No phone number found
8	https://economics.ucla.edu/person/felipe-gonca...	Felipe Goncalves	Assistant Professor	fgoncalves@econ.ucla.edu	No phone number found
9	https://economics.ucla.edu/person/moritz-meyer...	Moritz Meyer-ter-Vehn	Professor	mtv@econ.ucla.edu	No phone number found
10	https://economics.ucla.edu/person/pablo-fajgel...	Pablo Fajgelbaum	Dubchansky Chair in Economics	pfajgelbaum@econ.ucla.edu	(310) 794-7241
11	https://economics.ucla.edu/person/oleg-itskhoki/	Oleg Itskhoki	Venu and Ana Kotamraju Endowed Chair in Economics	itskhoki@econ.ucla.edu	No phone number found
12	https://economics.ucla.edu/person/till-von-wac...	Till Von Wachter	Professor	tvwachter@econ.ucla.edu	(310) 825-5665
13	https://economics.ucla.edu/person/zhipeng-liao/	Zhipeng Liao	Professor	zhipeng.liao@econ.ucla.edu	No phone number found
14	https://economics.ucla.edu/person/joao-guerreiro/	Joao Guerreiro	Assistant Professor	jGuerreiro@econ.ucla.edu	No phone number found
15	https://economics.ucla.edu/person/adriana-ller...	Adriana Lleras-Muney	Professor	alleras@econ.ucla.edu	(310) 825-3925
16	https://economics.ucla.edu/person/martin-b-hac...	Martin B. Hackmann	Associate Professor	hackmann@econ.ucla.edu	(310) 825-1011
17	https://economics.ucla.edu/person/juliana-lond...	Juliana Londoño-Vélez	Assistant Professor	j.londonovelez@econ.ucla.edu	No phone number found

	URL	Professor Name	Professor Title	Emails	Phone Numbers
18	https://economics.ucla.edu/person/michael-rubens/	Michael Rubens	Assistant Professor	No email found	No phone number found
19	https://economics.ucla.edu/person/denis-chetve...	Denis Chetverikov	Professor	chetverikov@econ.ucla.edu	No phone number found
20	https://economics.ucla.edu/person/lee-e-ohanian/	Lee E. Ohanian	Distinguished Professor of Economics	ohanian@econ.ucla.edu	No phone number found
21	https://economics.ucla.edu/person/bernardo-s-s...	Bernardo S. Silveira	Assistant Professor	silveira@econ.ucla.edu	No phone number found
22	https://economics.ucla.edu/person/sule-ozler/	Sule Ozler	Associate Professor	ozler@econ.ucla.edu	No phone number found
23	https://economics.ucla.edu/person/pierre-olivi...	Pierre-Olivier Weill	Professor	poweill@econ.ucla.edu	(310) 794-6495
24	https://economics.ucla.edu/person/gary-d-hansen/	Gary D. Hansen	Professor	ghansen@econ.ucla.edu	No phone number found
25	https://economics.ucla.edu/person/ariel-burstein/	Ariel Burstein	Professor	arielb@econ.ucla.edu	(310) 206-6732
26	https://economics.ucla.edu/person/saki-bigio/	Saki Bigio	Associate Professor	sbigio@econ.ucla.edu	No phone number found
27	https://economics.ucla.edu/person/michela-gior...	Michela Giorcelli	Associate Professor	mgiorcelli@econ.ucla.edu	No phone number found
28	https://economics.ucla.edu/person/john-asker/	John Asker	Armen Alchian Professor of Economics	johnasker@econ.ucla.edu	No phone number found
29	https://economics.ucla.edu/person/rodrigo-pinto/	Rodrigo Pinto	Assistant Professor	rodrig@econ.ucla.edu	(310) 825-0849
30	https://economics.ucla.edu/person/simon-board/	Simon Board	Benjamin Graham Centennial Chair in Value Inve...	sboard@econ.ucla.edu	No phone number found
31	https://economics.ucla.edu/person/aaron-tornell/	Aaron Tornell	Professor	tornell@econ.ucla.edu	No phone number found
32	https://economics.ucla.edu/person/will-rafey/	Will Rafey	Assistant Professor	rafey@econ.ucla.edu	No phone number found
33	https://economics.ucla.edu/person/alexander-bl...	Alexander Bloedel	Assistant Professor	abloedel@econ.ucla.edu	No phone number found
34	https://economics.ucla.edu/person/rosa-liliana...	Rosa Matzkin	Charles E. Davidson Distinguished Professor of...	matzkin@econ.ucla.edu	(310) 825-7371

	URL	Professor Name	Professor Title	Emails	Phone Numbers
35	https://economics.ucla.edu/person/tomasz-sadzik/	Tomasz Sadzik	Assistant Professor	tsadzik@econ.ucla.edu	(310) 206-2833
36	https://economics.ucla.edu/person/yotam-shem-tov/	Yotam Shem-Tov	Assistant Professor	shemtov@econ.ucla.edu	No phone number found
37	https://economics.ucla.edu/person/jonathan-vogel/	Jonathan Vogel	Professor	jvogel@econ.ucla.edu	No phone number found
38	https://economics.ucla.edu/person/martha-bailey/	Martha Bailey	Professor	marthabailey@ucla.edu	No phone number found
39	https://economics.ucla.edu/person/maurizio-maz...	Maurizio Mazzocco	Professor	mmazzocc@econ.ucla.edu	(310) 825-6682
40	https://economics.ucla.edu/person/hugo-hopenhayn/	Hugo Hopenhayn	Professor	hopen@econ.ucla.edu	(310) 206-8896
41	https://economics.ucla.edu/person/andres-santos/	Andres Santos	Professor	andres@econ.ucla.edu	No phone number found
42	https://economics.ucla.edu/person/dora-costa/	Dora Costa	Kenneth L. Sokoloff Chair in Economic History	costa@econ.ucla.edu	No phone number found
43	https://economics.ucla.edu/person/daniel-haanw...	Daniel Haanwinckel	Assistant Professor	haanwinckel@econ.ucla.edu	No phone number found
44	https://economics.ucla.edu/person/ichiro-obara/	Ichiro Obara	Professor	iobara@econ.ucla.edu	(310) 794-7098

Q4.) Selenium

Q4.A.) Pick a website that has useful data to a business or economic question. Put your website you plan to scrape here : https://docs.google.com/spreadsheets/d/1PJ2DOTCVCh51fn0ry1yB7qTyccR33_IXFpk usp=sharing

You must have use website that no other group has. First come first serve

In [ ]:
```
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.common.exceptions import NoSuchElementException
from selenium.common.exceptions import TimeoutException
import pandas as pd
import re
import json
```

The website we choose is https://www.careeronestop.org/Toolkit/Wages/highest-paying-careers.aspx.

It is a rank for the high paying jobs in the United States, as the list is long, we scraped the first 10 subpages of the website as below.

In [ ]:
```
# This is a test for the link extraction part of the code

base_url = 'https://www.careeronestop.org/Toolkit/Wages/highest-paying-careers.aspx?&curPage='
links = []

driver = webdriver.Chrome()
wait = WebDriverWait(driver, 30)

for i in range(1, 11):  # Loop from page 1 to 10

    try:
        driver.get(base_url + str(i))
        print(f"Page {i} loaded successfully.")

    except TimeoutException:
        print(f"Page {i} took too long to load and has been skipped.")

driver.quit()
```

```
Page 1 loaded successfully.
Page 2 loaded successfully.
Page 3 loaded successfully.
Page 4 loaded successfully.
Page 5 loaded successfully.
Page 6 loaded successfully.
Page 7 loaded successfully.
Page 8 loaded successfully.
Page 9 loaded successfully.
Page 10 loaded successfully.
```

Q4.B.) Use Selenium to scrape valuable information from your website and store in a dataframe.

In [ ]:
```
driver = webdriver.Chrome()

all_job_data = []

for i in range(1, 11):
    try:
        driver.get(base_url + str(i))

        WebDriverWait(driver, 10).until(EC.presence_of_element_located((By.TAG_NAME, 'tbody')))

        job_rows = driver.find_elements(By.CSS_SELECTOR, 'tbody tr')

        for row in job_rows:
            rank = row.find_element(By.CSS_SELECTOR, 'td[headers="thRank"] div').text
            occupation = row.find_element(By.CSS_SELECTOR, 'td[headers="thName"] a').text
            hourly_wage = row.find_element(By.CSS_SELECTOR, 'td[headers="thHour"] div.orNumber').te
            annual_wage = row.find_element(By.CSS_SELECTOR, 'td[headers="thAnnual"] div.orNumber').
            typical_education = row.find_element(By.CSS_SELECTOR, 'td[headers="thTypEdu"] div').tex

            # Append each job's data as a dictionary to the list
            all_job_data.append({
                'Rank': rank,
                'Occupation': occupation,
                'Hourly Wage': hourly_wage,
                'Annual Wage': annual_wage,
                'Typical Education': typical_education
            })

    except TimeoutException:
        print(f"Page {i} took too long to load and was skipped.")

driver.quit()
```

In [ ]:
```
jobs_df = pd.DataFrame(all_job_data)

jobs_df
```

Out[ ]:

	Rank	Occupation	Hourly Wage	Annual Wage	Typical Education
0	1	Anesthesiologists	$115.00+	$239,200+	Doctoral or professional degree
1	1	Cardiologists	$115.00+	$239,200+	Doctoral or professional degree
2	1	Dermatologists	$115.00+	$239,200+	Doctoral or professional degree
3	1	Emergency Medicine Physicians	$115.00+	$239,200+	Doctoral or professional degree
4	1	Obstetricians and Gynecologists	$115.00+	$239,200+	Doctoral or professional degree
...	...	...	...	...	...
95	95	Education Administrators, Postsecondary	$48.05	$99,900	Master's degree
96	95	Medical Scientists, Except Epidemiologists	$48.04	$99,900	Doctoral or professional degree
97	98	Bioengineers and Biomedical Engineers	$47.86	$99,600	Bachelor's degree
98	98	Software Quality Assurance Analysts and Testers	$47.89	$99,600	Bachelor's degree
99	100	Elevator and Escalator Installers and Repairers	$47.60	$99,000	High school diploma or equivalent

100 rows × 5 columns

Q4.C.) Write a short paragraph about the businesses or research that would use the data you scraped. Describe it's value and what it can be used for.

Schools, universities, and career counseling services can use this data to advise students and job seekers about potential career paths. Understanding which occupations offer the highest wages and the educational requirements for these positions can help guide curriculum development and career advice tailored to align with lucrative job markets.

Companies, especially in sectors with high-paying roles such as healthcare and engineering, can use this data to benchmark salary offerings and ensure competitive compensation packages that attract top talent. HR departments can also analyze the typical education levels required for these roles to aid in recruitment strategies and job postings.

Market analysts and investors might find this data useful for identifying growth industries and sectors with high-paying jobs, which can signal robust economic sectors worth investing in or starting businesses within.