Data scraping

Data scraping, or webscraping, is the process of importing information froma website of multiple websites into a spreadsheet or a file. It's a common practice for people doing research, finding the best deals (think of travel websites like Kayak), etc.

Concrete example: Say you want to know how certain politician have voted throughout the years without having to click throught a website.
https://clerk.house.gov/Votes

doing it by hand: Click "view details" for each bill -> copy and paste

But we can do much better through coding and web scraping

Steps:

1. Manually inspect the data source (the website). The better you know the website, the easier it will be to scrape
2. Scrape HTML content from the page. HTML is a coding language to create websites.
3. Parse HTML code form the page.

Let's try this on a simple website first: https://realpython.github.io/fake-jobs/

In [ ]:
```
import sys

!{sys.executable} -m pip install requests             # requests: python library that allows you to access websites
 + resources
!{sys.executable} -m pip install beautifulsoup4       # bs4: python library for parsing structured data
```

In [ ]:
```
# Store websites content so we can use the content in Python
import requests
from bs4 import BeautifulSoup

URL = "https://realpython.github.io/fake-jobs/"
page = requests.get(URL)                              # get's website page, returns requests.Response object

#print(page.status_code)                              # 200 means okay, 404 means page not found
#print(page.text)                                     # could also print page.url, page.links, ...

soup = BeautifulSoup(page.content, "html.parser")           # creates beautiful soup object which stores the page's
                                                            #      information in a format that's easily usable
                                                            # pass page.content helps with character encoding
                                                            # html.parser makes sure you use the appropriate parser

results = soup.find(id="ResultsContainer")

#print all job titles on the page
job_elements = results.find_all("div", class_="card-content")
for job_element in job_elements:
    title_element = job_element.find("h2", class_="title")
    print(title_element.text.strip())
```

In [ ]:
```
# But let's say we only care about the jobs that explicitly mention python
# We can use a lambda function

python_jobs = results.find_all("h2", string=lambda text: "python" in text.lower())

for jobs in python_jobs:
    print(jobs.text.strip())
```

For more information on any of the above code, visit: https://realpython.com/beautiful-soup-web-scraper-python/

If you're interested in doing a project involving web scraping, try the following exercises:

1. From the same page we have been working on above, print out all roles that mention Engineer along with the company name and location
2. Write a function that takes in a string and outputs all jobs (along with the company name and location) that mention the string in their job title
3. Write a function that takes in a string and outputs all jobs (along with company name and location) that are in the state corresponding to the input string

Scraping a social media site

Most social media and other copmanies make their data easily accessible using an API (application programming interface). I won't go over APIs today but if we have a few people who think it would be useful for their project then I can go over this topic. In most cases though, you can easily use code that someone else wrote or the dataset already exists.

Note: If you cannot get this to work on your computer, let Dominic know and he will help you!

In [ ]:
```
import sys
!{sys.executable} -m pip install twint                  # Twitter intelligence tool, Twitter scraping tool

!{sys.executable} -m pip install nest_asyncio           # Allows Twint looping to work in Juptyer notebooks
```

In [ ]:
```
import nest_asyncio
nest_asyncio.apply()
```

In [ ]:
```
import twint

c = twint.Config()                              # configure twint object
c.Username = "kanyewest"                        # Choose tweets from a specific twitter user
#c.Search = "Stranger Things"                   # Choose tweets with specific key words
c.Limit = 50
c.Store_csv = True
c.Output = 'Kanye_tweet_data.csv'
twint.run.Search(c)
```