I will be doing research using this video. It is from the same group as the video I used for python research, so there should hopefully be some overlap in teachings and focus to avoid turbulence in the early-learning phase.

NOTES

Set up beautifulsoup just by installing it in pycharm and then using 'from bs4 import BeautifulSoup'

Open html file (that is in the project directory) by simply using
with open('file.html', 'action') as variable:
Do stuff

For example:

```
from bs4 import BeautifulSoup

with open('webscrape.html', 'r') as html_file:
   content = html_file.read()
   print(content)
```

^ this (from my example code) takes the html code from the html file I made and prints it to the console

You want to download parsers, lxml seems to work. Download it by clicking python packages (at the bottom) and then searching lxml (it should be in PyPI) and then installing it.

```
from bs4 import BeautifulSoup

with open('webscrape.html', 'r') as html_file:
   content = html_file.read()
   soup = BeautifulSoup(content, 'lxml')
   print(soup.prettify())
```

^ Use BeautifulSoup to prettify the code printed to console.

```
from bs4 import BeautifulSoup

with open('webscrape.html', 'r') as html_file:
   content = html_file.read()

   soup = BeautifulSoup(content, 'lxml')
   tags = soup.find('h1') #/findAll, find_all
   print(tags)
```

^ Use BeautifulSoup to print h1s' code. Find stops after first element, findAll (or find_all) prints them all.

```
from bs4 import BeautifulSoup

with open('webscrape.html', 'r') as html_file:
   content = html_file.read()
   soup = BeautifulSoup(content, 'lxml')
   tags = soup.find_all('h1')

   for tag in tags:
       print(tag.text)
```

^ Iterate over the retrieved information to cleanly show the text itself.

```
from bs4 import BeautifulSoup

with open('google.html', 'r') as html_file2:
   content = html_file2.read()

   soup2 = BeautifulSoup(content, 'lxml')

   tags2 = soup2.find_all('div', class_='FPdoLc lJ9FBc')

   for tag in tags2:
       tag_text = tag.text
       print(tag_text.split()[0:3])

       print(f'{tag_text} is the text version, {tag} is the html version')
```

^ Extract data from (downloaded) google homepage html file.

Using the website they name to avoid issues with differences in names/locaitons.

```
from bs4 import BeautifulSoup
import requests

html_text =
requests.get('https://www.timesjobs.com/candidate/job-search.html?searchType=pe
rsonalizedSearch&from=submit&searchTextSrc=&searchTextText=&txtKeywords=python&
txtLocation=').text #same website as video

soup = BeautifulSoup(html_text, 'lxml')
```

^ Use requests package in order to webscrape from existing websites

```
from bs4 import BeautifulSoup
import requests

html_text =
requests.get('https://www.timesjobs.com/candidate/job-search.html?searchType=pe
rsonalizedSearch&from=submit&searchTextSrc=&searchTextText=&txtKeywords=python&
txtLocation=').text #same website as video

soup = BeautifulSoup(html_text, 'lxml')

jobs = soup.find_all('li', class_ = 'clearfix job-bx wht-shd-bx')

for job in jobs:
   published_date = job.find('span', class_ = 'sim-posted').span.text

   if 'few' in published_date:
       company_name = job.find('h3', class_='joblist-comp-name').text.replace('
', '')
       skills = job.find('span', class_='srp-skills').text.replace(' ', '')
       more_info = job.header.h2.a['href']

       print("Company: " + company_name)
       print("Necessary Skills: " + skills)
       print("More Info: " + more_info)
```

^ Use requests to scrape data from the jobs website, naming the company, required skills and a link for more info on the listing for every listing on the website that was posted recently enough to have the date tag ‘a few days ago’

```
from bs4 import BeautifulSoup
import requests

print("List a skill that you aren't familiar with")
unfamiliar_skill = input(">")
print("Filtering out any jobs that require: " + unfamiliar_skill)


html_text =
requests.get('https://www.timesjobs.com/candidate/job-search.html?searchType=pe
rsonalizedSearch&from=submit&searchTextSrc=&searchTextText=&txtKeywords=python&
txtLocation=').text #same website as video

soup = BeautifulSoup(html_text, 'lxml')

jobs = soup.find_all('li', class_ = 'clearfix job-bx wht-shd-bx')

for job in jobs:
   published_date = job.find('span', class_ = 'sim-posted').span.text

   if 'few' in published_date:
       company_name = job.find('h3', class_='joblist-comp-name').text.replace('
', '')
       skills = job.find('span', class_='srp-skills').text.replace(' ', '')
       more_info = job.header.h2.a['href']

       if unfamiliar_skill not in skills:
           print("Company: " + company_name)
           print("Necessary Skills: " + skills)
           print("More Info: " + more_info)
```

^ Same program, but updated to allow the user to filter out listings that have a skill unfamiliar to them

```
from bs4 import BeautifulSoup
import requests
import time
import datetime

print("List a skill that you aren't familiar with")
unfamiliar_skill = input(">")
print("Filtering out any jobs that require: " + unfamiliar_skill)

def actions(index, company_name, skills, more_info):
   with open('job listings/listing ' + str(index) + ' as of ' +
datetime.datetime.now().strftime(
           "%B %d, %Y") + '.txt', 'w') as f:
       print("Company: " + company_name)
       print("Necessary Skills: " + skills)
       print("More Info: " + more_info)
       f.write("Company: " + company_name)
       f.write("Necessary Skills: " + skills)
       f.write("More Info: " + more_info)

def find_jobs():
   html_text =
requests.get('https://www.timesjobs.com/candidate/job-search.html?searchType=pe
rsonalizedSearch&from=submit&searchTextSrc=&searchTextText=&txtKeywords=python&
txtLocation=').text #same website as video


   soup = BeautifulSoup(html_text, 'lxml')


   jobs = soup.find_all('li', class_ = 'clearfix job-bx wht-shd-bx')


   for index, job in enumerate(jobs):
      published_date = job.find('span', class_ = 'sim-posted').span.text


      if 'few' in published_date:
          company_name = job.find('h3',
class_='joblist-comp-name').text.replace(' ', '')
          skills = job.find('span', class_='srp-skills').text.replace(' ', '')
          more_info = job.header.h2.a['href']

          if unfamiliar_skill != "":
              if unfamiliar_skill not in skills:
                  actions(index, company_name, skills, more_info)
          else:
              actions(index, company_name, skills, more_info)
```

```
if __name__ == '__main__':
    while True:
        find_jobs()

        time_wait = 10 #minutes to wait before refreshing data

        time.sleep(time_wait * 60)
```

^ Final version of the same program, now exporting the information to .txt files, and doing so every 10 minutes (so long as the program is still running)

Video is done, I feel confident in BeautifulSoup and in Requests.