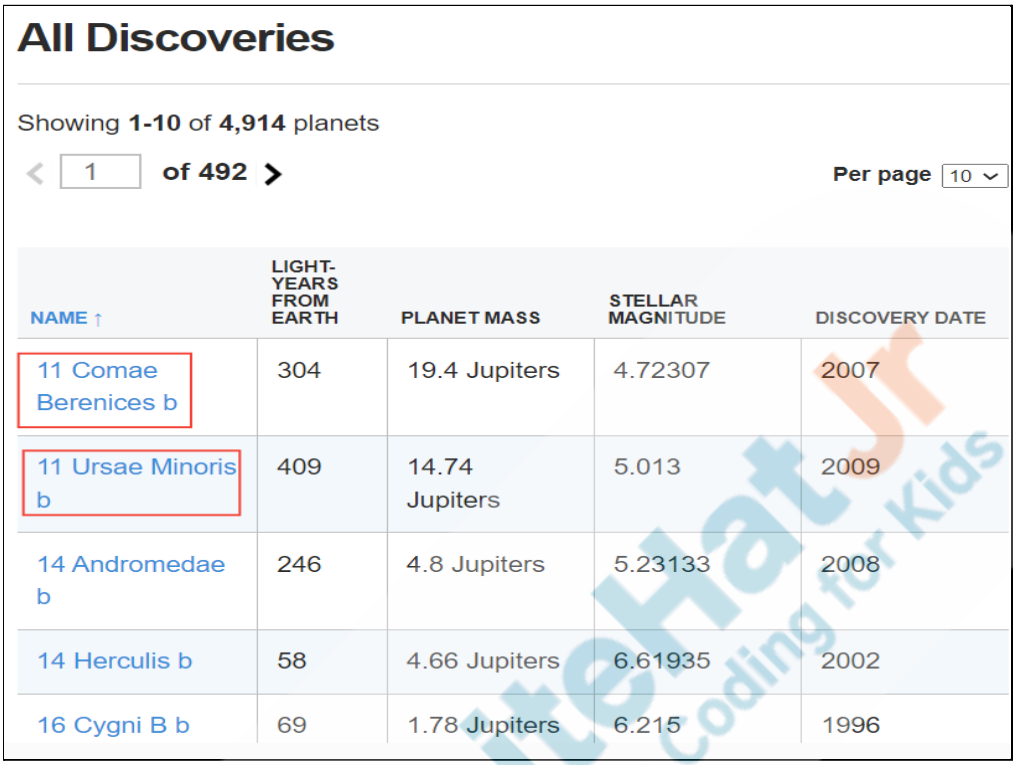
Topic	WEB SCRAPING - 2	
Class Description	**Students would be reworking the previously written code to scrape more data.**	
Class	**PRO C128**	
Class time	**45 mins**	
Goal	• Scrape more data about all the exoplanets • Create a CSV file to store data	

Resources Required	• Teacher Resources:
	  ○ Laptop with internet connectivity
	  ○ Earphones with mic
	  ○ Notebook and pen
	  ○ Smartphone
	• Student Resources:
	  ○ Laptop with internet connectivity
	  ○ Earphones with mic
	  ○ Notebook and pen	

Class structure	**Warm-Up** **Teacher-Led Activity 1** **Student-Led Activity 1** **Wrap-Up**	**10 mins** **10 mins** **20 mins** **05 mins**
Credit & Permissions:	**Exoplanet Exploration by NASA** **BeautifulSoup by Crummy (webspace of Leonard Richardson)** **Selenium under Apache 2.0 License**	
WARM-UP SESSION - 10 mins		
Teacher Starts Slideshow **Slide # to #** <**Note**: Only Applicable for Classes with VA>		

Refer to speaker notes and follow the instructions on each slide.	
Teacher Action	**Student Action**
Hey <student's name>. How are you? It's great to see you! Are you excited to learn something new today? **Following are the WARM-UP session deliverables:** • Greet the student. • Revision of previous class activities. • Quizzes.	**ESR**: Hi, thanks! Yes, I am excited about it! Click on the slide show tab and present the slides
WARM-UP QUIZ Click on In-Class Quiz	
Continue WARM-UP Session **Slide # to #** <**Note**: Only Applicable for Classes with VA>	
Activity Details **Following are the session deliverables:** • Appreciate the student. • Narrate the story by using hand gestures and voice modulation methods to bring in more interest in students.	
Teacher Action	**Student Action**
In the last class, we scraped the exoplanet data from NASA's website. Can you recall all the tools that we used in the last class? ***Note:** Encourage the student to give answers and connect the answer with today's topic.* Great! Now, in today's class, we will scrape some more data from the same website. We got some data like distance from earth, planet size, etc. but today we will	**ESR:** - Selenium - BeautifulSoup

scrape more data. This is important because we have to perform analysis later. Thus, we can better predict the planets, for instance, to see if they are likely habitable, etc. Are you excited?	**ESR:** Yes
Teacher Ends Slideshow 	
TEACHER-LED ACTIVITY - 10 mins	
Teacher Initiates Screen Share	
ACTIVITY • **Scraping more data from the website and letting students lead the development this time.**	
Teacher Action	**Student Action**
Open Teacher Activity 1 to show the website to the student. Do you remember this exoplanet website that we saw in the previous class? Great!! Here, if we look closely, we can see that the name of these exo-planets is a hyperlink. Let's click on the link and see what kind of data we can find?	**ESR:** We scraped the data from this website.



All Discoveries

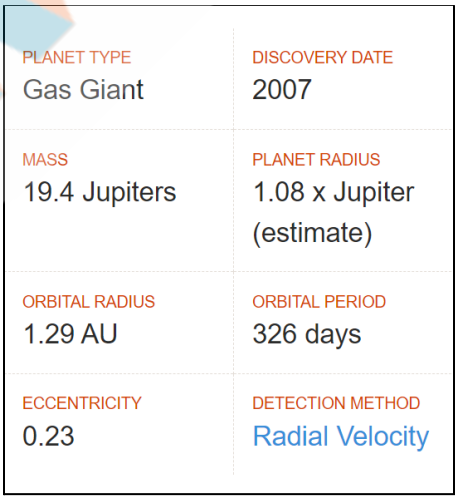
Showing **1-10** of **4,914** planets

1 **of 492** > Per page 10

NAME ↑	LIGHT-YEARS FROM EARTH	PLANET MASS	STELLAR MAGNITUDE	DISCOVERY DATE
11 Comae Berenices b	304	19.4 Jupiters	4.72307	2007
11 Ursae Minoris b	409	14.74 Jupiters	5.013	2009
14 Andromedae b	246	4.8 Jupiters	5.23133	2008
14 Herculis b	58	4.66 Jupiters	6.61935	2002
16 Cygni B b	69	1.78 Jupiters	6.215	1996

Note: *The total number of exoplanets may change due to the timely updation of data on the website. Open the link and mention the number accordingly.*

Let's click on the link and see what kind of data we can find?



PLANET TYPE	DISCOVERY DATE
Gas Giant	2007
MASS	**PLANET RADIUS**
19.4 Jupiters	1.08 x Jupiter (estimate)
ORBITAL RADIUS	**ORBITAL PERIOD**
1.29 AU	326 days
ECCENTRICITY	**DETECTION METHOD**
0.23	Radial Velocity

Great! Now, let's say we want to scrape this data as well. Can you tell me what's the first change that we'll have to make in our previous code? That's great! Let's get started. Open Teacher Activity 2 for boilerplate code.	**ESR:** We need to save the hyperlink's href in our CSV.
Let's make some changes to our **scrape()** function. We have to make sure that the data is being scraped in the same order as we have it on a web page. 1. Compare the **current page number** with counter **'i'** of the **for** loop. 2. To save the value of the current page number we are using the **get()** method. 3. Click the buttons to go to the current page number. 4. If the current page number is not the same as i, to click the button use their **XPath** of respective buttons.	

```
def scrape():
    for i in range(1,2):
        while True:
            time.sleep(2)

            soup = BeautifulSoup(browser.page_source, "html.parser")

            # Check page number
            current_page_num = int(soup.find_all("input", attrs={"class", "page_num"})[0].get("value"))

            if current_page_num < i:
                browser.find_element(By.XPATH, value='//*[@id="primary_column"]/footer/div/div/div/nav/span[2]/a').click()
            elif current_page_num > i:
                browser.find_element(By.XPATH, value='//*[@id="primary_column"]/footer/div/div/div/nav/span[1]/a').click()
            else:
                break
```

```
if current_page_num < i:
    browser.find_element(By.XPATH,
    value='//*[@id="primary_column"]/footer/div/div/div/nav/span[2]/a')
    .click()
elif current_page_num > i:
    browser.find_element(By.XPATH,
    value='//*[@id="primary_column"]/footer/div/div/div/nav/span[1]/a')
    .click()
```

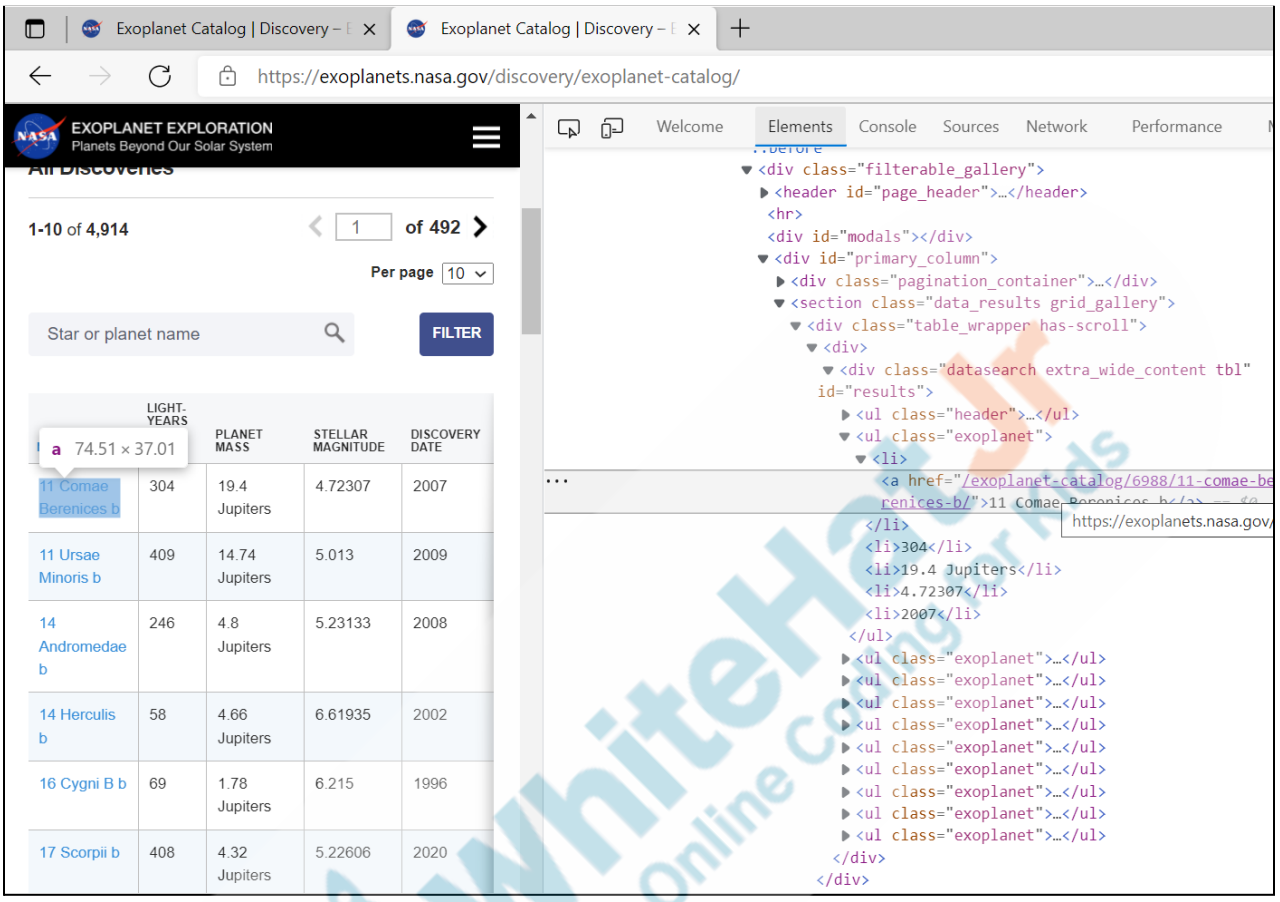
```
else:
    break
```

Now, we have the same code for accessing all the elements using **<ul>** and **<li>** tags. This will remain the same to get the data from the HTML page.

```
26          for ul_tag in soup.find_all("ul", attrs={"class", "exoplanet"}):
27              li_tags = ul_tag.find_all("li")
28              temp_list = []
29              for index, li_tag in enumerate(li_tags):
30                  if index == 0:
31                      temp_list.append(li_tag.find_all("a")[0].contents[0])
32                  else:
33                      try:
34                          temp_list.append(li_tag.contents[0])
35                      except:
36                          temp_list.append("")
```

We have added an extra hyperlink to our header list. Now, we also need to add this into the **temp_list** before we append it into the **planet_data.** Thus, we have to make a small change.

Before we do that, let's investigate the **href** for URL in these hyperlinks. Right-click on the hyperlink and click on inspect. Inside the <a> tag, a link is given which gives us more detail about exoplanet.



Here, we can see that these links do not have https://exoplanets.nasa.gov before them. We will have to add them.

Now to achieve this, we will do the following:

1. First, Create a variable **hyperlink_li_tag** and then we are using this variable to find all the **<a>** tags with **href**.
2. Take the first **<a>** tag (since we know there's only one **<a>** tag in all **<li>** tags) and then we are taking out the **href** from it.
3. Since we have to give the full URL of the page, we are adding **'https://exoplanets.nasa.gov'** to the hyperlink.
4. Then append it into **temp_list**.
5. This list is then appended in the **planet_data** list. Thus **planet_data** is a **list of lists.**

6. Lastly, we are using **f-string** to print the current page number which we have already finished scraping.

```
        for ul_tag in soup.find_all("ul", attrs={"class", "exoplanet"}):
            li_tags = ul_tag.find_all("li")
            temp_list = []
            for index, li_tag in enumerate(li_tags):
                if index == 0:
                    temp_list.append(li_tag.find_all("a")[0].contents[0])
                else:
                    try:
                        temp_list.append(li_tag.contents[0])
                    except:
                        temp_list.append("")

            # Get Hyperlink Tag
            hyperlink_li_tag = li_tags[0]

            temp_list.append("https://exoplanets.nasa.gov"+ hyperlink_li_tag.find_all("a", href=True)[0]["href"])

            planets_data.append(temp_list)

        browser.find_element(By.XPATH, value='//*[@id="primary_column"]/footer/div/div/div/nav/span[2]/a').click()

        print(f"Page {i} scraping completed")

```

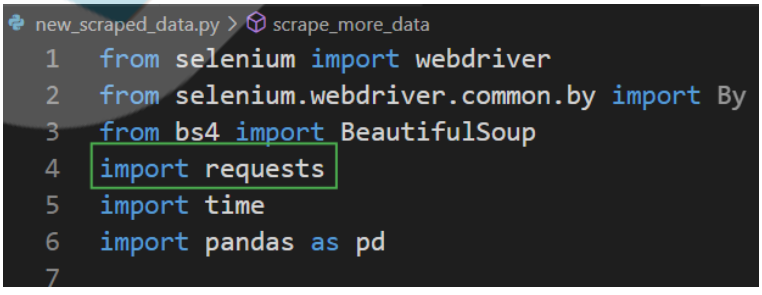
7. Since we are adding hyperlinks to our data, update the header with **hyperlink** and save the updated CSV file by name **updated_scraped_data.csv**

```
# Calling Method
scrape()

# Define Header
headers = ["name", "light_years_from_earth", "planet_mass", "stellar_magnitude", "discovery_date", "hyperlink"]

# Define pandas DataFrame
planet_df_1 = pd.DataFrame(planets_data, columns=headers)

# Convert to CSV
planet_df_1.to_csv('updated_scraped_data.csv',index=True, index_label="id")
```

Now that we have the links in **planet_data**, can you tell me what should be our next steps?
Perfect, we will create a new function called **scraper_more_data()** function that will take these hyperlinks one by one, get the HTML, and then we will scrape the data.

Next I'll help you to write the function to scrape data from

ESR: We'll scrape data by using these links!

these links. Earlier, we used selenium because we wanted to click a button on the page (next button) but this time, we do not want to interact with the browser, therefore we can do this without selenium.	
Teacher Stops Screen Share	
So now it's your turn. Please share your screen with me.	
STUDENT-LED ACTIVITY - 20 mins	
• **Ask the student to press the ESC key to come back to the panel.** • **Guide the student to start Screen Share.** • **The teacher gets into Full Screen.**	
Student Initiates Screen Share	
ACTIVITY • **Create a new function to use all the hyperlinks one by one and scrape data from there**	
Teacher Action	**Student Action**
Open Student Activity 1 to start coding. Download the files. Open **new_scraper.py.** As we'll be using the GET request method to get the data, the **requests** module is imported.	
	
In this file **scrape_more_data()** function is given to scrape data from **hyperlinks.**	

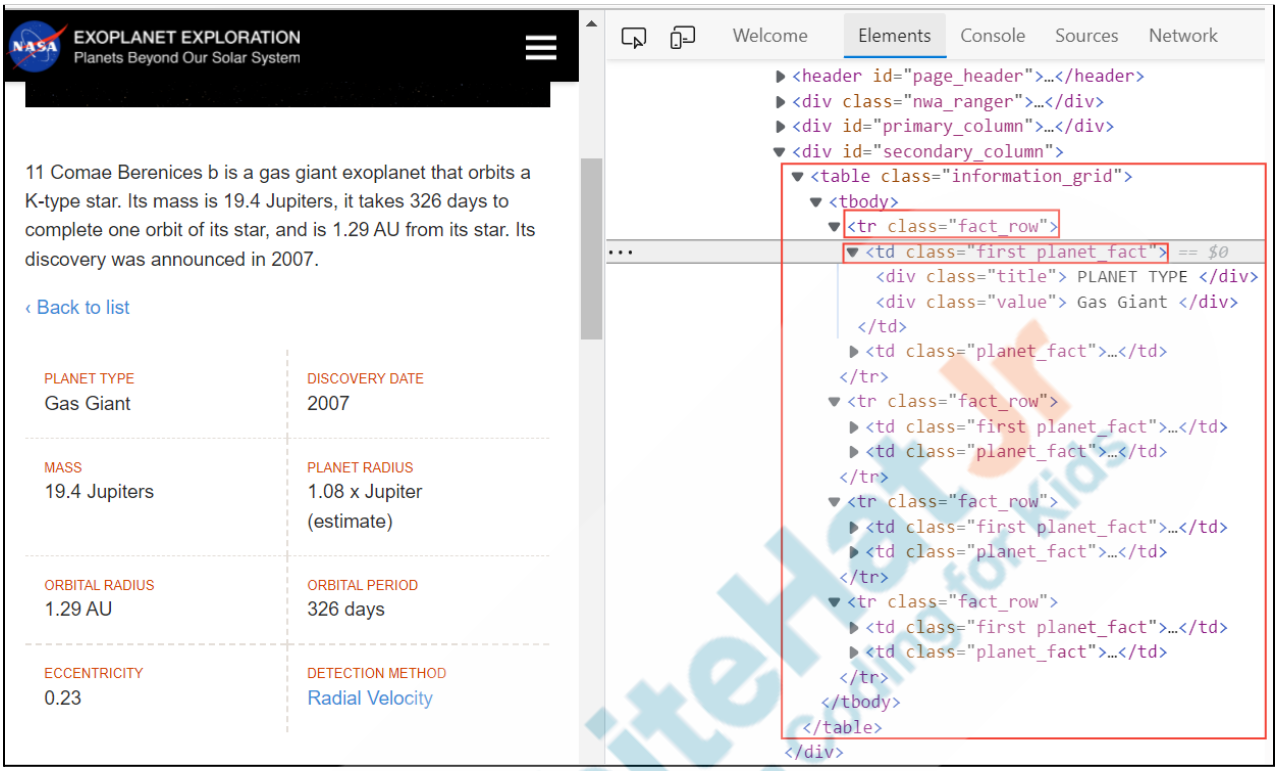
A new list called **new_planets_data** is created to store new data of planets from hyperlinks.

To get the data:

1. Create a variable **page** and get the content of the HTML page using the **hyperlink** using the **get()** method of the requests module.
2. Then, create a soup for parsing the HTML page.
3. Also, create an empty list called **temp_list** to store the data temporarily.

```
19  def scrape_more_data(hyperlink):
20      try:
21          page = requests.get(hyperlink)
22
23          soup = BeautifulSoup(page.content, "html.parser")
24
25          temp_list = []
26
```

4. Inspect the table and access the table row and table data i.e. **<tr>** and **<td>** tags one by one.



5. Here, you can see we have to access all the table rows with the class name **fact_row**. In all the rows **titles** and **values** are present as table data, we'll be accessing **values** only.

Note: *The teacher can refer to the previous class code as a reference code. Instead of* ***<ul>*** *and* ***<li>****, we are searching for <tr> and <td> tags here. Also, write all the code in except and try blocks.*

6. After getting the data into **temp_list**, we'll append it to the **new_planets_data** list**.**

```
def scrape_more_data(hyperlink):
    try:
        page = requests.get(hyperlink)
        soup = BeautifulSoup(page.content, "html.parser")
        temp_list = []
        for tr_tag in soup.find_all("tr", attrs={"class": "fact_row"}):
            td_tags = tr_tag.find_all("td")
            for td_tag in td_tags:
                try:
                    temp_list.append(td_tag.find_all("div", attrs={"class": "value"})[0].contents[0])
                except:
                    temp_list.append("")
        new_planet_data.append(temp_list)
```

7. Write the except block for handling exception as we did before to iterate it and go to the next link.

```
def scrape_more_data(hyperlink):
    try:
        page = requests.get(hyperlink)
        soup = BeautifulSoup(page.content, "html.parser")
        temp_list = []
        for tr_tag in soup.find_all("tr", attrs={"class": "fact_row"}):
            td_tags = tr_tag.find_all("td")
            for td_tag in td_tags:
                try:
                    temp_list.append(td_tag.find_all("div", attrs={"class": "value"})[0].contents[0])
                except:
                    temp_list.append("")
        new_planet_data.append(temp_list)
    except:
        time.sleep(1)
        scrape_more_data(hyperlink)
```

8. **Planet_df_1** is used to store the updated CSV file in the form of DataFrame.

9. Use for loop to get data from the hyperlink. Use the **iterrow()** method of pandas to iterate through the rows of the data frame and get the hyperlink.

10. Call **scrape_more_data()** and pass the hyperlink. It will scrape more data from the hyperlink.

11. To check the data, print 10 elements of **new_planets_data**.

```
planet_df_1 = pd.read_csv("updated_scraped_data.csv")

for index, row in planet_df_1.iterrows():
    print(row['hyperlink'])
    scrape_more_data(row['hyperlink'])
    print(f"Data Scraping at hyperlink {index+1} completed")

print(new_planets_data[0:10])
```

Save it and run the file using the command prompt.

```
C:\Whitehat_jr\PRO-C128-Reference-Code-main\PRO-C128-Reference-Code-main>pyth
C:\Whitehat_jr\PRO-C128-Reference-Code-main\PRO-C128-Reference-Code-main\new_
e object
  browser = webdriver.Edge("C:/Whitehat_jr/PRO-127-130/msedgedriver.exe")

DevTools listening on ws://127.0.0.1:51357/devtools/browser/b4354f5b-f623-4f2
[20516:3220:0415/130949.921:ERROR:fallback_task_provider.cc(124)] Every rende
 task is shown, it is a bug. If you have repro steps, please file a new bug a
https://exoplanets.nasa.gov/exoplanet-catalog/6988/11-comae-berenices-b/
Data Scraping at hyperlink 1 completed
https://exoplanets.nasa.gov/exoplanet-catalog/6989/11-ursae-minoris-b/
Data Scraping at hyperlink 2 completed
https://exoplanets.nasa.gov/exoplanet-catalog/6990/14-andromedae-b/
Data Scraping at hyperlink 3 completed
https://exoplanets.nasa.gov/exoplanet-catalog/6991/14-herculis-b/
Data Scraping at hyperlink 4 completed
https://exoplanets.nasa.gov/exoplanet-catalog/6992/16-cygni-b-b/
Data Scraping at hyperlink 5 completed
```

Also, check the list new_planets_data. As you can see the data has the '\n' character which is responsible to add new lines to the data.

```
[['\nGas Giant\n', '\n2007\n', '\n19.4 Jupiters\n', '\n1.08 x Jupiter', '\n1.29 AU\n', '\n326
, '\n1.09 x Jupiter', '\n1.53 AU\n', '\n1.4 years\n', '\n0.08\n', '\n'], ['\nGas Giant\n', '\n
n', '\n0.0\n', '\n'], ['\nGas Giant\n', '\n2002\n', '\n4.66 Jupiters\n', '\n1.15 x Jupiter', '
, '\n1.78 Jupiters\n', '\n1.2 x Jupiter', '\n1.66 AU\n', '\n2.2 years\n', '\n0.68\n', '\n'], [
AU\n', '\n1.6 years\n', '\n0.06\n', '\n'], ['\nGas Giant\n', '\n2008\n', '\n10.3 Jupiters\n',
Giant\n', '\n2008\n', '\n8 Jupiters\n', '\n1.664 x Jupiter\n', '\n330.0 AU\n', '\n6505.9 years
.24 x Jupiter', '\n0.19 AU\n', '\n30.4 days\n', '\n0.04\n', '\n'], ['\nGas Giant\n', '\n2010\n
'\n0.09\n', '\n']]
```

12. Create a new list called **scraped_data**. The data we get after scraping the hyperlink has the character **'\n'.**
13. Use the **replace()** method to remove the character.

```
52  scrapped_data = []
53
54  for row in new_planets_data:
55      replaced = []
56      for el in row:
57          el = el.replace("\n", "")
58          replaced.append(el)
59      scrapped_data.append(replaced)
60
61  print(scrapped_data)
62
```

14. Save this and run the file to check whether the character is removed or not.

```
[['Gas Giant', '2007', '19.4 Jupiters', '1.08 x Jupiter', '1.29 AU', '326 days', '0.23', ''],
s', '0.08', ''], ['Gas Giant', '2008', '4.8 Jupiters', '1.15 x Jupiter', '0.83 AU', '185.8 da
 AU', '4.9 years', '0.37', ''], ['Gas Giant', '1996', '1.78 Jupiters', '1.2 x Jupiter', '1.66
Jupiter', '1.45 AU', '1.6 years', '0.06', ''], ['Gas Giant', '2008', '10.3 Jupiters', '1.11 x
s', '1.664 x Jupiter', '330.0 AU', '6505.9 years', '0.0', ''], ['Gas Giant', '2018', '0.91 Ju
2010', '1.99 Jupiters', '1.19 x Jupiter', '1.333 AU', '452.8 days', '0.09', ''], ['Gas Giant'
], ['Gas Giant', '2021', '4 Jupiters', '1.16 x Jupiter', '118.0 AU', '3110.6 years', '0.0', '
ars', '0.0', ''], ['Gas Giant', '2013', '24.5 Jupiters', '1.0 x Jupiter', '52.0 AU', '593.2 y
6.0 AU', '5878.1 years', '0.0', ''], ['Gas Giant', '2010', '7.5 Jupiters', '1.13 x Jupiter',
 Jupiter', '55.0 AU', '2885.9 years', '0.0', ''], ['Gas Giant', '2015', '1.9 Jupiters', '1.2
iant', '2009', '20.95 Jupiters', '0.92 x Jupiter', 'Unknown', '20.1 years', '0.26', ''], ['Ga
 '0.0', ''], ['Gas Giant', '2009', '13.82 Jupiters', '1.1 x Jupiter', '0.99 AU', '335.1 days'
U', '479.1 days', '0.38', ''], ['Gas Giant', '1996', '2.53 Jupiters', '1.18 x Jupiter', '2.1
iter', '3.6 AU', '6.6 years', '0.1', ''], ['Gas Giant', '2009', '1.64 Jupiters', '1.2 x Jupit
```

Great job! Now we have **scrapped_data** without any special character.

15. A list called headers is created to save the data into a CSV file.
16. Using these headers, scrapped_data is converted into Dataframe.

17. This DataFrame is then saved into the **new_scraped_data.csv** file. The id will be the name of the first column with serial numbers.

```
63
64  headers = ["planet_type","discovery_date", "mass", "planet_radius", "orbital_radius",
65              "orbital_period", "eccentricity", "detection_method"]
66
67  new_planet_df_1 = pd.DataFrame(scrapped_data,columns = headers)
68  new_planet_df_1.to_csv('new_scraped_data.csv',index=True, index_label="id")
69
```

18. Save and run this file to check the CSV file. It opened the browser and start scraping the data and displaying the message each time we are scraping the data.

***Note:** Guide the student to add the webdriver to the current directory and copy the path as done in the previous class. Run the Python file using a virtual environment.*

new_scraped_data.csv

```
1   id,planet_type,discovery_date,mass,planet_radius,orbital_radius,orbital_period,eccentricity,detection_method
2   0,Gas Giant,2007,19.4 Jupiters,1.08 x Jupiter,1.29 AU,326 days,0.23,
3   1,Gas Giant,2009,14.74 Jupiters,1.09 x Jupiter,1.53 AU,1.4 years,0.08,
4   2,Gas Giant,2008,4.8 Jupiters,1.15 x Jupiter,0.83 AU,185.8 days,0.0,
5   3,Gas Giant,2002,4.66 Jupiters,1.15 x Jupiter,2.93 AU,4.9 years,0.37,
6   4,Gas Giant,1996,1.78 Jupiters,1.2 x Jupiter,1.66 AU,2.2 years,0.68,
7   5,Gas Giant,2020,4.32 Jupiters,1.15 x Jupiter,1.45 AU,1.6 years,0.06,
8   6,Gas Giant,2008,10.3 Jupiters,1.11 x Jupiter,2.6 AU,2.7 years,0.08,
9   7,Gas Giant,2008,8 Jupiters,1.664 x Jupiter,330.0 AU,6505.9 years,0.0,
10  8,Gas Giant,2018,0.91 Jupiters,1.24 x Jupiter,0.19 AU,30.4 days,0.04,
```

Also, check the saved file in the directory.

	A	B	C	D	E	F	G	H	I
1	id	planet_type	discovery_date	mass	planet_radius	orbital_radiu	orbital_period	eccentricity	detection_method
2	0	Gas Giant	2007	19.4 Jupiters	1.08 x Jupiter	1.29 AU	326 days	0.23	
3	1	Gas Giant	2009	14.74 Jupiter	1.09 x Jupiter	1.53 AU	1.4 years	0.08	
4	2	Gas Giant	2008	4.8 Jupiters	1.15 x Jupiter	0.83 AU	185.8 days	0	
5	3	Gas Giant	2002	4.66 Jupiters	1.15 x Jupiter	2.93 AU	4.9 years	0.37	
6	4	Gas Giant	1996	1.78 Jupiters	1.2 x Jupiter	1.66 AU	2.2 years	0.68	
7	5	Gas Giant	2020	4.32 Jupiters	1.15 x Jupiter	1.45 AU	1.6 years	0.06	
8	6	Gas Giant	2008	10.3 Jupiters	1.11 x Jupiter	2.6 AU	2.7 years	0.08	
9	7	Gas Giant	2008	8 Jupiters	1.664 x Jupiter	330.0 AU	6505.9 years	0	
10	8	Gas Giant	2018	0.91 Jupiters	1.24 x Jupiter	0.19 AU	30.4 days	0.04	

Great work!!

We scraped exoplanet data from NASA's website. This data can be used to find useful insights.

Teacher Guides Student to Stop Screen Share

WRAP-UP SESSION - 05 mins

Teacher Starts Slideshow
Slide # to #
<**Note**: Only Applicable for Classes with VA>

Activity details

Following are the WRAP-UP session deliverables:

- Appreciate the student.
- Revise the current class activities.
- Discuss the quizzes.

WRAP-UP QUIZ
Click on In-Class Quiz

Continue WRAP-UP Session
Slide # to #
<**Note**: Only Applicable for Classes with VA>

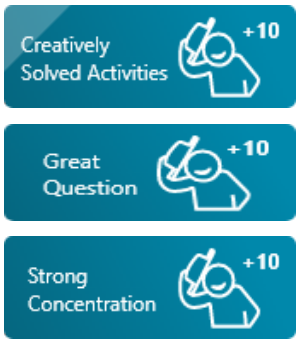
Activity Details

Following are the session deliverables:

- Explain the facts and trivia
- Next class challenge
- Project for the day
- Additional Activity (Optional)

FEEDBACK

- **Appreciate and compliment the student for trying to learn a difficult concept.**
- **Get to know how they are feeling after the session.**
- **Review and check their understanding.**

Teacher Action	Student Action
So, in this project class we revisited the concepts from the previous class and you did the majority of the scraping yourself! Congratulations! You get "hats-off" for your excellent work! In the next class, we will be downloading more data and preprocessing it for further analysis.	*Make sure you have given at least 2 hats-off during the class for:* 

PROJECT OVERVIEW DISCUSSION

Refer the document below in Activity Links Sections

Teacher Clicks ✖ End Class

ACTIVITY LINKS		
Activity Name	**Description**	**Links**
Teacher Activity 1	Exoplanet Exploration	https://exoplanets.nasa.gov/discovery/exoplanet-catalog/
Teacher Activity 2	Boilerplate Code	https://github.com/procodingclass/PRO-C128-Teacher-Boilerplate-Code
Teacher Activity 3	Reference Code	https://github.com/procodingclass/PRO-C128-Reference-Code
Teacher Reference 1	Project	https://s3-whjr-curriculum-uploads.whjr.online/def32133-27cf-4d33-b49b-cea2606ce399.pdf
Teacher Reference 2	Project Solution	https://github.com/procodingclass/PRO-C128-Project-Solution
Teacher Reference 3	Visual-Aid	Will be added after VA creation
Teacher Reference 4	In-Class Quiz	https://s3-whjr-curriculum-uploads.whjr.online/20562587-c9b9-4474-8394-0e41a33d1f69.pdf
Student Activity 1	Boilerplate Code	https://github.com/procodingclass/PRO-C128-Student-Boilerplate-Code