




Topic	DATA SCIENCE - 1	
Class Description	The student will be applying statistics to the exoplanet data and plotting charts to find interesting insights.	
Class	PRO C131	
Class time	45 mins	
Goal	<ul style="list-style-type: none"> Plot various charts from our data Find interesting insights about the exo-planets 	
Resources Required	<ul style="list-style-type: none"> Teacher Resources: <ul style="list-style-type: none"> Laptop with internet connectivity Earphones with mic Notebook and pen Smartphone Student Resources: <ul style="list-style-type: none"> Laptop with internet connectivity Earphones with mic Notebook and pen 	
Class structure	Warm-Up Teacher-Led Activity 1 Student-Led Activity 1 Wrap-Up	5mins 15mins 20 mins 05 mins
Credit & Permissions:	Exoplanet Exploration by NASA Plotly Express (Open-source MIT license)	
WARM-UP SESSION - 10 mins		
<div>  </div> <p>Teacher Starts Slideshow</p> <p>Slide # to #</p> <p><Note: Only Applicable for Classes with VA></p> <p>Refer to speaker notes and follow the instructions on each slide.</p>		

Teacher Action	Student Action
<p>Hey <student's name>. How are you? It's great to see you! Are you excited to learn something new today?</p> <p>Following are the WARM-UP session deliverables:</p> <ul style="list-style-type: none"> • Greet the student. • Revision of previous class activities. • Quizzes. 	<p>ESR: Hi, thanks! Yes, I am excited about it!</p>
<p align="center">WARM-UP QUIZ Click on In-Class Quiz</p>	
<p align="center">  Continue WARM-UP Session Slide # to # <Note: Only Applicable for Classes with VA> </p>	
<p>Activity Details</p> <p>Following are the session deliverables:</p> <ul style="list-style-type: none"> • 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
<p>In the last class, we understood and cleaned our data. We first understood the meaning of the fields, determined which ones we needed and removed the rest. We then re-named all the columns to make our data more readable for us.</p> <p>In today's class, we will jump to the exciting stuff! We will be plotting different graphs and applying statistics to our data to see if we can find any interesting insights. We'll be diving into the very first step of Data Science.</p>	

<p>Data Science is a field that is used to extract useful information from data. It follows a statistical approach to retrieve the information and make decisions.</p> <p>Doesn't that sound interesting?</p>	<p>ESR: Yes</p>
<p style="text-align: center;">Teacher Ends Slideshow </p>	
<p style="text-align: center;">TEACHER-LED ACTIVITY - 10 mins</p>	
<p style="text-align: center;">Teacher Initiates Screen Share</p>	
<p style="text-align: center;"><u>ACTIVITY</u></p> <ul style="list-style-type: none"> • Discussing with the student on what can we plot • Plotting the charts and applying statistics • Finding interesting insights 	
<p style="text-align: center;">Teacher Action</p>	<p style="text-align: center;">Student Action</p>
<p><i>Note: Open Teacher Activity 1 and download main.csv. This is the same file that we got after preprocessing the data.</i></p> <p><i>Note: Open Teacher Activity 2 and run all the cells to get started with the code.</i></p> <p>Open Google Colab and create a new project. Use The link to upload the dataset.</p>	
<pre>!git clone https://github.com/procodingclass/PRO-NASA-Exoplanet-Processed-Data.git fatal: destination path 'PRO-NASA-Exoplanet-Processed-Data' already exists and is not an empty directory.</pre>	
<p>Let's just start by recalling the meaning of all the columns we have left in our CSV.</p> <p><i><Discuss all the columns one by one with the student></i></p>	

<p>name - the name of the planet.</p> <p>light_years_from_earth - Distance of the exo-planet from the earth in light-years.</p> <p>planet_mass - Mass of the planet.</p> <p>stellar_magnitude - This is the brightness of the host star of the planet when observed from Earth (just as the sun is our host star).</p> <p>discovery_date - This is the year of discovery for the exo-planet.</p> <p>planet_type - This is the type of planet (Gas Giant, Super-Earth, etc.).</p> <p>planet_radius - This is the radius of the exoplanet with respect to Earth or Jupiter.</p> <p>orbital_radius - This is the average distance of this exo-planet from its sun. Just like our solar system has 1 sun, multiple solar systems contain many planets and sun(s).</p> <p>orbital_period - This is the time it takes to complete one orbit of its sun.</p> <p>eccentricity - This denotes how circular the orbit is. It might be oval in shape too. The lower the eccentricity, the more circular is the orbit.</p> <p>solar_system_name - The name of the host solar system.</p> <p>planet_discovery_method - This is the discovery method that was used to find this exo-planet.</p> <p>planet_orbital_inclination - This is the orbital inclination, which means that it is the tilt of the exoplanet's orbit when it revolves around its sun.</p> <p>planet_density - This is the density of the planet.</p> <p>right_ascension - This is the right ascension of the planetary system, which is the east-west coordinate by which the position of this planet is measured.</p> <p>declination - This is the north-south coordinate by which the position of the planet is measured.</p> <p>host_temperature - This is the temperature of the host star in Kelvin.</p> <p>host_mass - This is the amount of mass contained in the host star.</p> <p>host_radius - This is the radius of the host star.</p>	
<p>Now, in our Solar System, we have about 8 planets. Can you name them?</p>	<p>ESR: - Mercury</p>

	<ul style="list-style-type: none"> - Venus - Earth - Mars - Jupiter - Saturn - Neptune - Uranus
<p>Awesome. Just like we have 8 planets in our solar system, let's try to find out how many planets have we found in other solar systems!</p> <p>For this, we have a column known as solar_system_name in our CSV. Using this, we can create a dictionary to maintain the count of planets each solar system has! Before we do that, if we look at the CSV we just downloaded, we can see that there is an index mentioned in the rows as the first element, but the header's first element is empty -</p> <p>Here, we are just reading the CSV file and segregating the two into headers and planet_data.</p>	
<pre>import csv rows = [] with open("/content/PRO-NASA-Exoplanet-Processed-Data/main.csv", "r") as f: csvreader = csv.reader(f) for row in csvreader: rows.append(row) headers = rows[0] planet_data_rows = rows[1:] print(headers) print(planet_data_rows[0])</pre>	
<p>After we run this code we get the headers and the first row of the exoplanet's data.</p>	

```
[', 'name', 'light_years_from_earth', 'planet_mass', 'stellar_magnitude',  
['0', '11 Comae Berenices b', '305.0', '19.4 Jupiters', '4.74', '2007', 'G
```

Here, we can see that there is an extra field added to the CSV, denoting the row count but the first header is an empty string. Let's fix that and then proceed with finding the number of planets in each of the solar systems!

Here, we are first adding the header, to make our data consistent. Then, we are creating an empty dictionary where we can store the **number of planets** in each **solar system**. We are iterating over all the planets and we are checking the solar system name of the planet at index number 11, if we already have an entry for that solar system in our dictionary or not.

If we do, we are increasing the count by 1 else we are adding this solar system into our dictionary with a count of 1. Finally, we are just finding the name of the solar system (key in our dictionary) with the maximum number of planets (values in our dictionary). The **format()** method inserts the values inside the string's placeholders. Thus, it takes the string and variables to give the output.

```
headers[0] = "row_num"
```

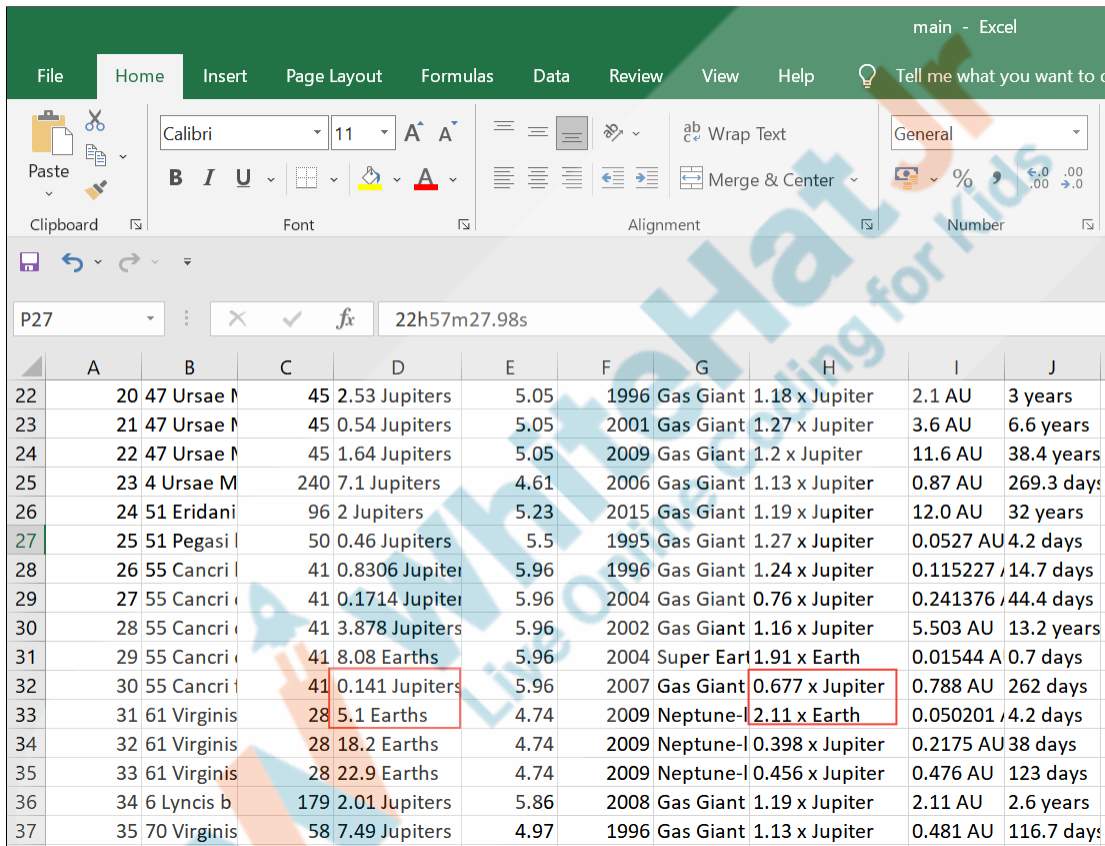
```
solar_system_planet_count = {}  
for planet_data in planet_data_rows:  
    if solar_system_planet_count.get(planet_data[11]):  
        solar_system_planet_count[planet_data[11]] += 1  
    else:  
        solar_system_planet_count[planet_data[11]] = 1
```

```
max_solar_system = max(solar_system_planet_count, key=solar_system_planet_count.get)  
print("Solar system {} has maximum planets {} out of all the solar systems we have discovered so far!".format(
```

```
Solar system KOI-351 has maximum planets 8 out of all the solar systems we have discovered so far!
```

Here, we can see that there is a solar system known as **KOI-351** where there are **8** planets! That's awesome!! Could this be our next home? Let's get the list of all the planets from this solar system!

Before we proceed, we need to make sure that our data is uniform! We have 2 columns, **planet_mass** & **planet_radius** where in some rows, the values are given with reference to **Jupiter** while in some places, it's given with the reference of **Earth**.



	A	B	C	D	E	F	G	H	I	J
22	20	47 Ursae M	45	2.53 Jupiters	5.05	1996	Gas Giant	1.18 x Jupiter	2.1 AU	3 years
23	21	47 Ursae M	45	0.54 Jupiters	5.05	2001	Gas Giant	1.27 x Jupiter	3.6 AU	6.6 years
24	22	47 Ursae M	45	1.64 Jupiters	5.05	2009	Gas Giant	1.2 x Jupiter	11.6 AU	38.4 years
25	23	4 Ursae M	240	7.1 Jupiters	4.61	2006	Gas Giant	1.13 x Jupiter	0.87 AU	269.3 days
26	24	51 Eridani	96	2 Jupiters	5.23	2015	Gas Giant	1.19 x Jupiter	12.0 AU	32 years
27	25	51 Pegasi	50	0.46 Jupiters	5.5	1995	Gas Giant	1.27 x Jupiter	0.0527 AU	4.2 days
28	26	55 Cancri	41	0.8306 Jupiters	5.96	1996	Gas Giant	1.24 x Jupiter	0.115227 AU	14.7 days
29	27	55 Cancri	41	0.1714 Jupiters	5.96	2004	Gas Giant	0.76 x Jupiter	0.241376 AU	44.4 days
30	28	55 Cancri	41	3.878 Jupiters	5.96	2002	Gas Giant	1.16 x Jupiter	5.503 AU	13.2 years
31	29	55 Cancri	41	8.08 Earths	5.96	2004	Super Earth	1.91 x Earth	0.01544 AU	0.7 days
32	30	55 Cancri	41	0.141 Jupiters	5.96	2007	Gas Giant	0.677 x Jupiter	0.788 AU	262 days
33	31	61 Virginis	28	5.1 Earths	4.74	2009	Neptune-like	2.11 x Earth	0.050201 AU	4.2 days
34	32	61 Virginis	28	18.2 Earths	4.74	2009	Neptune-like	0.398 x Jupiter	0.2175 AU	38 days
35	33	61 Virginis	28	22.9 Earths	4.74	2009	Neptune-like	0.456 x Jupiter	0.476 AU	123 days
36	34	6 Lynx b	179	2.01 Jupiters	5.86	2008	Gas Giant	1.19 x Jupiter	2.11 AU	2.6 years
37	35	70 Virginis	58	7.49 Jupiters	4.97	1996	Gas Giant	1.13 x Jupiter	0.481 AU	116.7 days

We have to make it uniform in our CSV. Let's make it **Earth** for all since it would be easier for us to compare that way!

For this, we need to look at our 3rd and 7th element and see if they are measured in **Jupiter** or **Earth**. If they are measured in Jupiter, we need to change it. There are also a few rows with values as **Unknown**. We need to remove these rows as well!

To make this data uniform, at first, we created a temporary planet list (using **list()** function) and iterated over it. Then, we just checked if the third element of the list (**planet_mass**) is **unknown** or not.

If it's not, we take out the value and reference (**Earth or Jupiter**) for the planet. If it's Jupiter, we replace the values and then we change the value of the element in the list.

If in case the value was **unknown**, we are removing the **planet_data** from our main list **planet_data_rows**.

We did the same for the 7th element (**planet_radius**) as well!

1 Jupiter Mass = 317.8 Earth Mass

1 Jupiter Radius = 11.2 Earth Radius

Can you tell why we created a temporary list of all planet rows?

Yes, so let's remove the planets with unknown mass and create a list of planets with known mass. Also converting the mass according to the mass of earth so that it'll be easier to compare and data will become uniform.

ESR:

Because we are removing the element from the list inside the **for** loop. If we iterate over the same list and remove it from the same list too, then our code will perform unexpectedly.


```
temp_planet_data_rows = list(planet_data_rows)
for planet_data in temp_planet_data_rows:
    planet_mass = planet_data[3]
    if planet_mass.lower() == "unknown":
        planet_data_rows.remove(planet_data)
        continue
    else:
        planet_mass_value = planet_mass.split(" ")[0]
        planet_mass_ref = planet_mass.split(" ")[1]
        if planet_mass_ref == "Jupiters":
            planet_mass_value = float(planet_mass_value) * 317.8
        planet_data[3] = planet_mass_value
```

The same operation will be performed on the planet radius and changing it w.r.t to the radius of the earth.

```
temp_planet_data_rows = list(planet_data_rows)
for planet_data in temp_planet_data_rows:
    planet_mass = planet_data[3]
    if planet_mass.lower() == "unknown":
        planet_data_rows.remove(planet_data)
        continue
    else:
        planet_mass_value = planet_mass.split(" ")[0]
        planet_mass_ref = planet_mass.split(" ")[1]
        if planet_mass_ref == "Jupiters":
            planet_mass_value = float(planet_mass_value) * 317.8
        planet_data[3] = planet_mass_value

    planet_radius = planet_data[7]
    if planet_radius.lower() == "unknown":
        planet_data_rows.remove(planet_data)
        continue
    else:
        planet_radius_value = planet_radius.split(" ")[0]
        planet_radius_ref = planet_radius.split(" ")[2]
        if planet_radius_ref == "Jupiter":
            planet_radius_value = float(planet_radius_value) * 11.2
        planet_data[7] = planet_radius_value

print(len(planet_data_rows))
```

4251

Next, we would simply check if a given planet's solar system is equal to the solar system with maximum planets or not. If it is the same, we are adding the details of that planet into a list.

```
koi_351_planets = []
for planet_data in planet_data_rows:
    if max_solar_system == planet_data[11]:
        koi_351_planets.append(planet_data)
```

```
print(len(koi_351_planets))
print(koi_351_planets)
```

```
7
[['3665', 'Kepler-903 b', '2704.0', '4.7', '14.615', '2016', 'Super Earth', '2.01', 'Unknown',
```

Now, let's plot a bar chart on the planet mass -
We are creating separate lists of planet masses and names, and then we are adding the details of our planet Earth as well, just to compare.

We are finally plotting a bar chart with names on the X-coordinate and masses on the Y-coordinate.

We can see that there are 2 planets - **Kepler-903b** and **KOI-1599.01** who are close to earth in terms of mass (**Earth's** mass would be 1 since we converted these metrics with reference to Earth!)

Here, we are using **Plotly Express**. It is a part of plotly library that can be used for creating graphs and charts.

[Teacher Activity 3: Plotly Express](#)

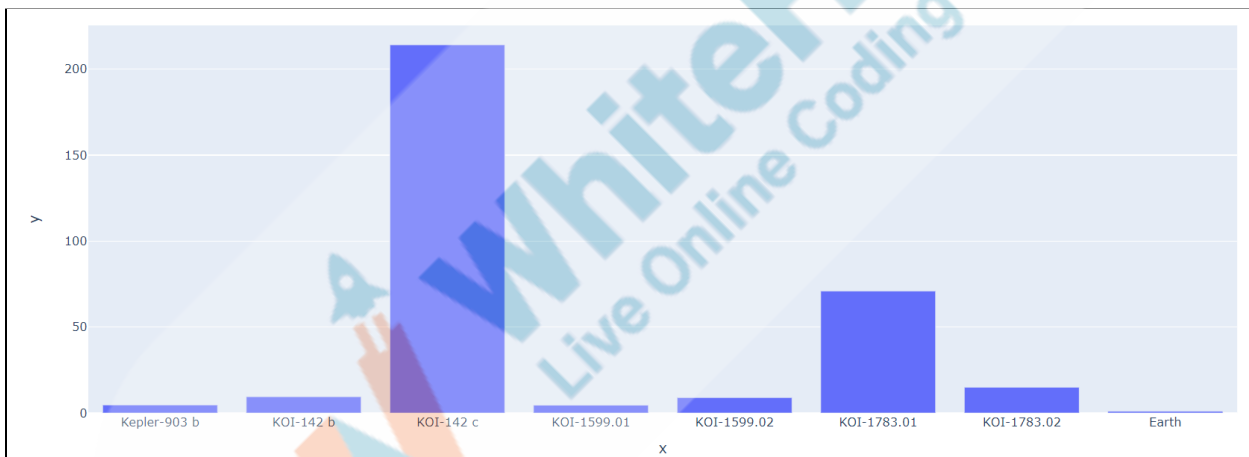
```
import plotly.express as px
```

```
koi_351_planet_masses = []  
koi_351_planet_names = []  
for planet_data in koi_351_planets:  
    koi_351_planet_masses.append(planet_data[3])  
    koi_351_planet_names.append(planet_data[1])
```

```
koi_351_planet_masses.append(1)  
koi_351_planet_names.append("Earth")
```

```
fig = px.bar(x=koi_351_planet_names, y=koi_351_planet_masses)  
fig.show()
```

Run the above cell to check the bar graph.



Okay, now let's try to have this with all the planets we have discovered so far and see if we can find more Earth-like planets.

Before we do that, let's understand one more thing.

Great Scientist Albert Einstein gave us a formula with which we can calculate the gravity of any planet.

The formula is this -

$$g = \frac{G * M_{\text{earth}}}{d^2}$$

Here, G is a gravitational constant, which means that it will always be the same.

M(earth) is the mass of Earth (or any other planet if we are calculating it for another planet)

d is the radius of the planet!

Here, we can see an inverse relationship between the radius of the planet and gravity. The more the radius (and bigger the planet), the lesser would be Gravity.

But then, we also see a direct relationship between the mass of the planet and gravity. The more the mass of the planet, the more will be the gravity.

Can you tell me what is the value of the earth's gravitational force?

Yes!! Our Earth's gravity is **9.8 m/s²** and we as humans are accustomed to it.



In order for us to exist on any other planet, gravity should be close to what we have here.

Mars has a gravity of **3.711 m/s** and the **Moon** has a gravity of **1.62 m/s**.

Next, we'll be plotting the graph of the radius versus the mass of these planets.

ESR: g=9.8 m/s²

Teacher Stops Screen Share

So now it's your turn. Please share your screen with me.	
<div style="text-align: center;">  Teacher Starts Slideshow Slide # to # <Note: Only Applicable for Classes with VA> Refer to speaker notes and follow the instructions on each slide. </div>	
We have one more class challenge for you. Can you solve it? Let's try. I will guide you through it.	
<div style="text-align: center;">  Teacher Ends Slideshow </div>	
STUDENT-LED ACTIVITY - 20 mins	
<ul style="list-style-type: none"> • 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	
<div style="text-align: center;"> <u>ACTIVITY</u> </div> <ul style="list-style-type: none"> • Student Code to plot the scatter plot • Finding low gravity planet 	
Teacher Action	Student Action
Given what we have just learned, let's try to plot a scatter plot for all the planets, where we will keep the mass of the planet as the Y-coordinate , the Radius of the planet as the X-coordinate , and the size of the blob as the gravity of it. Let's see if we can find anything interesting! Open Student Activity 1 for boilerplate code.	

<Help the student in writing the code for this>

The value of G (Gravitational Constant) is 6.674e-11

Since we have the **planet_mass** and **planet_radius** with reference to Earth, don't forget to multiply the mass of the earth and the radius of the Earth with these values.

Mass of Earth = 5.972×10^{24}

Radius of Earth = 6371000

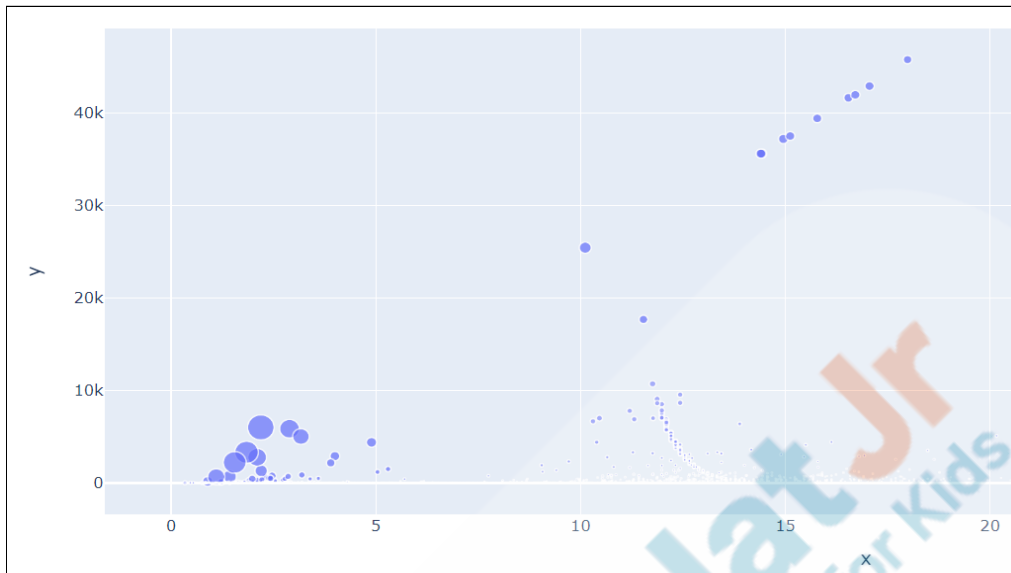
Let's understand the code:

1. Segregate the names, masses, and radiuses of the planets using different lists.
2. Applying the formula by filling the values we know and creating a list of gravities for all the planets.
3. Finally, we are plotting the graph with X as **planet_radiuses**, Y as **planet_masses**, size as **planet_gravity** and **hover_data** as the name of the planet!

```
planet_masses = []
planet_radiuses = []
planet_names = []
for planet_data in planet_data_rows:
    planet_masses.append(planet_data[3])
    planet_radiuses.append(planet_data[7])
    planet_names.append(planet_data[1])
planet_gravity = []
for index, name in enumerate(planet_names):
    gravity = (float(planet_masses[index])*5.972e+24) / (float(planet_radiuses[index])*
                                                         float(planet_radiuses[index])*6371000*6371000) * 6.674e-11
    planet_gravity.append(gravity)

fig = px.scatter(x=planet_radiuses, y=planet_masses, size=planet_gravity, hover_data=[planet_names])
fig.show()
```

4. Run the above cell to check the scatter graph.



Fun Fact - Our standing human bodies can withstand a gravitational force 90 times stronger than the Earth!

Although that is going to be a bit extreme, we can still survive at 10 times the gravity we have on Earth.

5. Create an empty list. Iterate through the list `planet_gravity`.
6. Iterate over the gravities of the planet with `enumerate()` function (to get the index of the planet as well) and then if the planet's gravity is less than 100.
7. Add the planet data into a list.
8. Run the cell to check the output

```
low_gravity_planets = []
for index, gravity in enumerate(planet_gravity):
    if gravity < 100:
        low_gravity_planets.append(planet_data_rows[index])

print(len(low_gravity_planets))
```

3951

We got **3,951 Planets!** That's insane! Let's see how many are there with gravity less than 10?

9. Repeat the same steps for the condition less than 10.

```
low_gravity_planets = []
for index, gravity in enumerate(planet_gravity):
    if gravity < 10:
        low_gravity_planets.append(planet_data_rows[index])

print(len(low_gravity_planets))
```

1012

So, We got 1,012 Planets!

Wow! We have a lot of potential options to survive on different planets!

Great! That's it!

So, in this class, we converted all the planet's mass and radius with reference to Earth and understood how gravity is calculated. We also found a list of 3,951 planets (1,012 heavily favorable) that could be our next home! Our human bodies are amazing, but they have limitations too. We will try to understand what we need to survive as humans in the next class!

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:

© 2021 - WhiteHat Education Technology Private Limited.

Note: This document is the original copyright of WhiteHat Education Technology Private Limited.

Please don't share, download or copy this file without permission.

- 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
<p>You get “hats-off” for your excellent work!</p> <p>Next class, we will dive more deeply into our data and filter out some other planets!</p>	<p><i>Make sure you have given at least 2 hats-off during the class for:</i></p> <div> Creatively Solved Activities  +10 </div> <div> Great Question  +10 </div>

	<div>Strong Concentration </div>
<p align="center">PROJECT OVERVIEW DISCUSSION Refer the document below in Activity Links Sections</p>	
<p align="center">Teacher Clicks</p>	<div>✕ End Class</div>



ACTIVITY LINKS		
Activity Name	Description	Links
Teacher Activity 1	Previous Class Code	https://github.com/procodingclass/PRO-C130-Reference-Code
Teacher Activity 2	Boilerplate Code	https://colab.research.google.com/drive/1LmN7l4-rQLkDltF6pssJVhzzZxnOim7?usp=sharing
Teacher Activity 3	Plotly Express	https://plotly.com/python/plotly-express/
Teacher Activity 4	Reference Code	https://colab.research.google.com/drive/1gKOosWvdQESXDNgmLePaLtq0G6wG4hIk?usp=sharing
Teacher Reference 1	Project	https://s3-whjr-curriculum-uploads.whjr.online/da976980-e91e-4576-8169-e3e9b6079a85.pdf
Teacher Reference 2	Project Solution	https://colab.research.google.com/drive/1kpYtwuZ6qTjdgIHBrJMKCKiPeyW1dJuF?usp=sharing
Teacher Reference 3	Visual-Aid	Will be added after VA creation
Teacher Reference 4	In-Class Quiz	https://s3-whjr-curriculum-uploads.whjr.online/33b7a86f-e33c-4799-bd4e-336414318a78.pdf
Student Activity 1	Boilerplate Code	https://colab.research.google.com/drive/1Vf1twLNEoh4h1ypIHU0EZQ0rmEiD88Td?usp=sharing