Documentation - Windows to the universe python project

Overview of projects goal, motivation, and structure

Goals

- The goal of our project was to create a program for people who are interested in space and would like to learn more about our solar system and about planets in general. The project aims to be accessible for both those who only have basic foundational knowledge about planets and the solar system and would like to learn about it in more depth but also for those who only have a deeper understanding and want to further their knowledge by accessing more complex details and how they relate to each other.
- Our project also provides insight into other possibly habitable planets in space, whether within our solar system or in galaxies far far away. We provide information, through our own calculation, on how habitable known exoplanets are and how they match up against others and some planets within our own solar system.
- We aim to make information on celestial bodies accessible and easy to compare.

Motivation

- We were motivated by our general interest in space and planets and the possibility of other habitable planets existing was very intriguing to us. We felt like we wanted to understand what information was important for planets and make it accessible for those who were similarly interested in it. With this information we could do our own calculations on habitability of planets and exoplanets and understand better which attributes were important for this calculation and what requirements needed to be met. Overall, we were motivated by the opportunity to explore the complexities of planets and their various aspects.

Structure

Web scraping

The script ”ourSolarSystem.py” performs web scraping to extract data from an HTML website, converts the data into a pandas DataFrame, and then saves the DataFrame as a CSV file. The code utilizes the Requests library to get the HTML code from the website which is fed into a BeautifulSoup object for the sake of having a nicer structure to work with. Pandas we need for data manipulation.
The script defines a function named scrape_planets_table(url) that takes a URL of a website that provides a table of information of planets from our solar system as input and scrapes that specific table of our interest from the webpage. The function makes an HTTP request to the URL using requests.get() and fetches the HTML content. It then parses the HTML content using BeautifulSoup with the "lxml" parser.
The target table is identified within the HTML content using soup.find('table'), and its headers (column names) are extracted from the table header elements (th tags) using a loop. The extracted headers are stored in a list and converted into a numpy array, which is reshaped into a 2D array with a shape of (17, 9) to match the table structure. The 2D numpy array is divided into columns, indices, and values. The first row and column of the 2D array correspond to the headers and indices, respectively. The data values of the table are extracted from the

remaining elements of the array. The extracted data is then used to create a pandas DataFrame (df_table) with columns as the extracted headers, indices as the extracted row indices, and values as the extracted data. The DataFrame is transposed to swap rows and columns to align the data correctly.
In the if __name__ == "__main__": block, the script sets a specific URL for scraping (url) and calls the scrape_planets_table() function with this URL to fetch the data and transform it into the DataFrame df_planets. The DataFrame is then saved as a CSV file named "solarPlanets.csv" in the "data" directory using df_planets.to_csv(). The script concludes by printing the DataFrame to the console using print(df_planets).

- Reading, displaying, and plotting from CSV (user prompted)
    - The second python program is user focussed and aims at providing planetary information from our solar system. It is structure in the following way:
    - The imported libraries are *Numpy*, *Matplotlib.pyplot*, *Plotnine* and *Pandas*.
    - First, we access the CSV dataset that we scraped from our URL in the previous program and saved in our data folder. We prepare and clean said dataset so that it is readable and malleable for our displaying and plotting needs. This includes changing some data types and renaming columns. We also defined a custom class to raise an exception when the user wants to exit the system under *ExitProgramException*.
    - Next, we wrote our functions, firstly *display_info* where the user is prompted to choose if they want to view all information regarding one specific planet or view and compare certain similar information across all planets (we divide this into spatial attributes, movement attributes and general planetary attributes). The input from the user is controlled through the entering of numbers that correspond to specific options. The *display_info()* function also includes checks for invalid inputs and reminds the user of their options. The function runs on a while loop, ensuring that the user can access as much data as they desire, as often as they want. This function also allows for the user to exit the system at any time or return back to the beginning where they may choose a different path to take, the return function works recursively with the *run_program()* function which I will define later. Depending on what is selected, the information is displayed in a readable and clear way.
    - Our second function, *plot_general()*, works in a similar way to the *display_info* function in that the user is able to enter numeric inputs to decide which specific aspect of the planets they would like to compare in a graphical plot. The function first sets the correct ordering of the planets and reorders the data frame, it also applies colors to each planet. Another while loop is used meaning the user can view as many different plots as they like, it also includes checks for invalid inputs. The plots are created using *ggplot* and displayed according to the same design language. This function includes the options to return or exit the system in the same way as the *display_info()* function.

    - The *run_program()* function initializes the whole system, it welcomes the user to the system and asks them which function of the program they would like to access. It works on a while loop within a try except structure which is the basis for *ExitProgramException* which is triggered anytime 'x' is entered as an input. Depending on the choice of the user, either the *display_info* or *plot_general* function will be triggered otherwise an invalid input message will be displayed. As mentioned above, this function can be called recursively in the other functions if the user chooses to change their choice of functionality of the program.

- Merging dataframes, calculating habitability and plotting results
    - The third Python script analyzes the habitability of planets and exoplanets and creates a plot of the most habitable ones. The code uses the *plotnine* library for data visualization and *pandas* for data manipulation. An overview of the code is as follows:
    - We import the necessary libraries, namely: *numpy*, *plotnine*, *pandas*, *math*, *os*, *sys*, and *ourSolarSystem*.
    - We read in our CSV files into pandas DataFrames: *exoplanet.eu_catalog.csv*, *solarPlanets.csv*.
    - We define several functions to clean and analyze the data: *clean_exo_dataset(d)*: Cleans the exoplanet dataset by extracting relevant columns and removing NaN values, *habitility_parameters(planet)*: Returns the mass, distance to the habitable zone, and orbital eccentricity of a planet, *calculate_habitable_zone(planet)*: Calculates the distance of a planet to the habitable zone based on the luminosities (calculated by the mass of the star to the power of 3) of the star and the planet's distance from the star, *habits_earth(df_planets)*: Prepares and cleans the solar system planets dataset for habitability analysis.
    - The following functions were used to calculate the habitability of the planets and plot the twenty most notable results.
      make_habit_df(): Creates a DataFrame with habitability values for both exoplanets and solar system planets by combining the cleaned exoplanet dataset and the solar system dataset. It calls the *habitility_parameters()* function for each planet, The script then calls the *make_habit_df()* function to create the habits DataFrame with habitability values and calculates the habitability formula while replacing infinite and large values with suitable values it proceeds to select the top 20 most habitable planets and exoplanets and creates a plot using the *plot_habitability(habitable_plot)*: Creates a scatter plot using *plotnine* to visualize the habitability values of the most habitable planets and exoplanets. It normalizes the formula values for exoplanets meaning planets in our solar system can have higher values. To combat this, we divided the solar system planet values by 10 so that they make sense in the context of the graph. We also set the Earth to value 50 as it is of course infinitely habitable and this value is also divided by 10 for the plot, this information is described in a caption for the plot.

- The scripts main aim is to calculate the habitability of the planets and to plot it but it does have some other functionalities for the user, for example:
- Sorts the DataFrame by habitability and prints the top 30 most habitable planets and exoplanets and saves the sorted DataFrame to a CSV file named *habitability.csv*.

Instructions for setting up project (local machine)

- Installation
  - The requirements of the program are stated clearly in the README.md document where we state which libraries are required and how to install them. The libraries required are Numpy, Pandas, Plotnine, Matplotlib, Requests and BS4 which we instruct the user to install using ‘pip’ package manager and tell them how to check that their version is correct. We then tell the user in which order to run the programs and how to use them.
  - The user will also be directed to run the programs in the correct order, specifically that the program *ourSolarSystem.py* should be executed first, either *plottingAndAllTheFun.py* or depending on what interests them the most, *exoplanets_formula.py*
- Links for external resources
  - https://phl.upr.edu/hec - Source for our original exoplanets dataset
  - https://www.windows2universe.org/?page=/our_solar_system/planets_table.html - website of where we scraped the solar system dataset
- Git link: https://github.com/grniemeyer/WindowsToTheUniverse

Instructions for intended usage of project

- Commands to run project.
  - To run the project, the user must simply navigate to the directory in which the project is saved using the cd command, they must then enter ‘python <filename>’ to run the program.
- Information on selecting inputs.
  - Inputs are selected only in the second python script name ‘*plottingAndAllTheFun.py’* where the user is prompted to enter numerical values in different ranges to indicate which data they would like to display and additionally either the letter x or b if they want to exit the system or return to the start, respectively.
- Information of selecting settings
  - There are some settings to be selected in the ‘*plottingAndAllTheFun.py’* script in which the user can decide in which way the data is presented, either in text or graphical form. This is decided as well by numerical inputs provided by the user.
- Examples for interpreting possible outputs.
  - IMG1 - Output of the first python script *ourSolarSystem.py* in the terminal once the information has been scrapped from the webpage, in the example some columns are hidden due to space restrictions.

- IMG2 - An example of an output when the user selects to see a specific planet's attributes.
- IMG3 - In this instance the user opted to display the spatial attributes of all planets in the solar system.
- IMG4 - Shows the various options of plots the user can request.
- IMG5&6 - an example of what the request plot looks like, IMG6 has labels to show the exact values of the points to make it clearer for the user.
- IMG7 - is the output of the combined solar system and exoplanets dataset with the habitability value included.
- IMG8 - The plot of the 20 most habitable planets and exoplanets based on our habitability calculations. The y-axis is in log scale to make the difference between values more readable and we added a disclaimer explaining that the calculations are normalized for exoplanets and therefore the values for the planets in our solar system have been edited to make sense within the graph.

```
(scipy) jordan@JordanMBA WindowsToTheUniverse % /opt/miniconda3/envs/scipy/bin/python "/Users/jordan/Desktop/Uniwork/5th semester/SciPy/Project/WindowsToTheUniverse/ou
arSystem.py"
         diameter (Earth=1) diameter (km) mass (Earth=1) mean distance from Sun (AU)  ... mean density (water=1) atmospheric composition number of moons rings?
Mercury               0.382         4,878          0.055                        0.39  ...                   5.43                     none              \n      no
Venus                 0.949        12,104          0.815                        0.72  ...                   5.25                      CO2              \n      no
Earth                     1        12,756              1                           1  ...                   5.52                  N2 + O2              \n      no
Mars                  0.532         6,787          0.107                        1.52  ...                   3.93                      CO2              \n      no
Jupiter              11.209       142,800            318                        5.20  ...                   1.33                    H2+He              \n     yes
Saturn                 9.44       120,000             95                        9.54  ...                   0.71                    H2+He              \n     yes
Uranus                4.007        51,118             15                       19.18  ...                   1.24                    H2+He              \n     yes
Neptune               3.883        49,528             17                       30.06  ...                   1.67                    H2+He              \n     yes
```

IMG1 - Output of the first python script *ourSolarSystem.py* in the terminal once the information has been scrapped from the webpage, in the example some columns are hidden due to space restrictions.

```
Type (1) for information on a specific planet and (2) to see attributes about all planets
Type (x) to exit and (b) to go back
1
Press:
(0) - Mercury
(1) - Venus
(2) - Earth
(3) - Mars
(4) - Jupiter
(5) - Saturn
(6) - Uranus
(7) - Neptune
5
                                      5
Planet                           Saturn
diameter (Earth=1)                 9.44
diameter (km)                   120,000
mass (Earth=1)                     95.0
mean distance from Sun (AU)        9.54
orbital period (Earth years)      29.46
orbital eccentricity              0.056
mean orbital velocity (km/sec)     9.64
rotation period (in Earth days)    0.44
inclination of axis (degrees)     26.73
mean temperature at surface (C)    -170
gravity at equator (Earth=1)       0.93
escape velocity (km/sec)          35.49
mean density (water=1)             0.71
atmospheric composition           H2+He
number of moons                      \n
rings?                              yes
Type (1) for information on a specific planet and (2) to see attributes about all planets
Type (x) to exit and (b) to go back
```

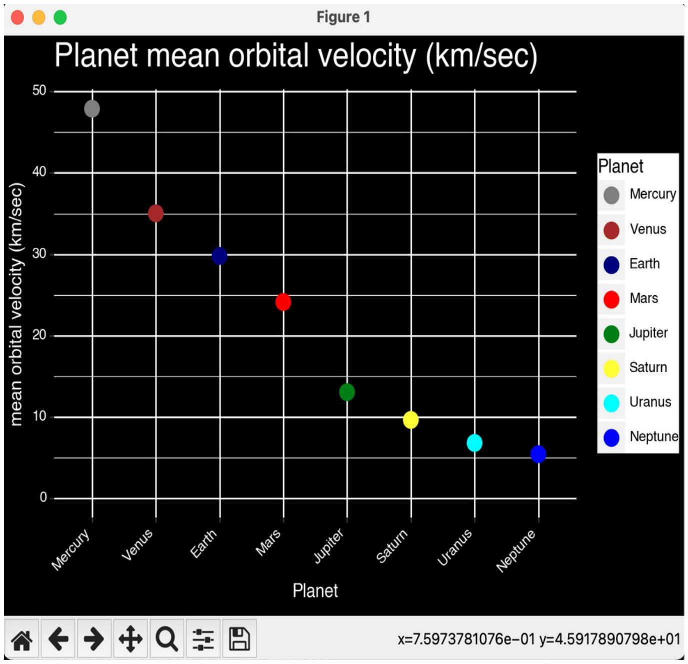
IMG2 - An example of an output when the user selects to see a specific planet's attributes.

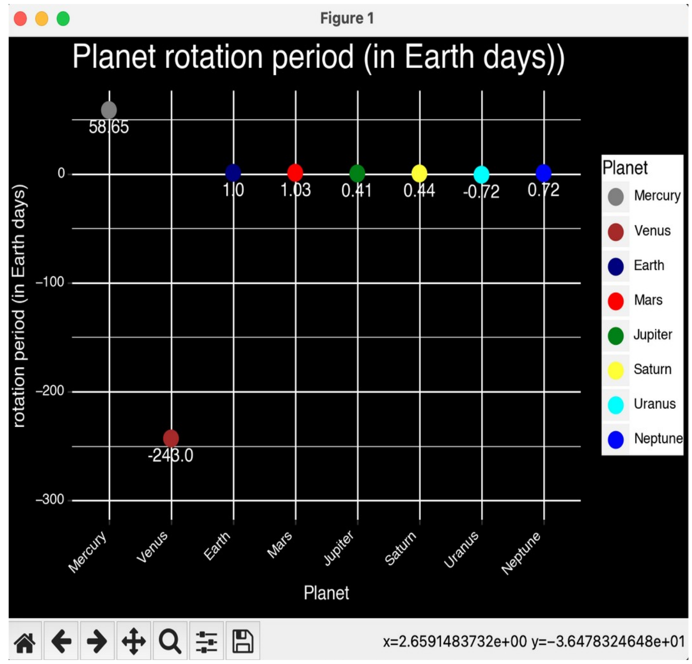
```
Welcome to the Windows to the Universe
What would you like to do?
View general information on planets in the solar system (Press 1)
Compare an attribute across all planets in graphical form (Press 2)
1
Type (1) for information on a specific planet and (2) to see attributes about all planets
Type (x) to exit and (b) to go back
2
Press:
(0) to compare the spatial attributes
(1) to compare movement attributes
(2) to compare planetary attributes
(x) to exit
0
    Planet diameter (km)  mass (Earth=1)  mean distance from Sun (AU) rings?
0  Mercury        4,878          0.055                         0.39     no
1    Venus       12,104          0.815                         0.72     no
2    Earth       12,756          1.000                         1.00     no
3     Mars        6,787          0.107                         1.52     no
4  Jupiter      142,800        318.000                         5.20    yes
5   Saturn      120,000         95.000                         9.54    yes
6   Uranus       51,118         15.000                        19.18    yes
7  Neptune       49,528         17.000                        30.06    yes
Type (1) for information on a specific planet and (2) to see attributes about all planets
Type (x) to exit and (b) to go back
```

IMG3 - In this instance the user opted to display the spatial attributes of all planets in the solar system.

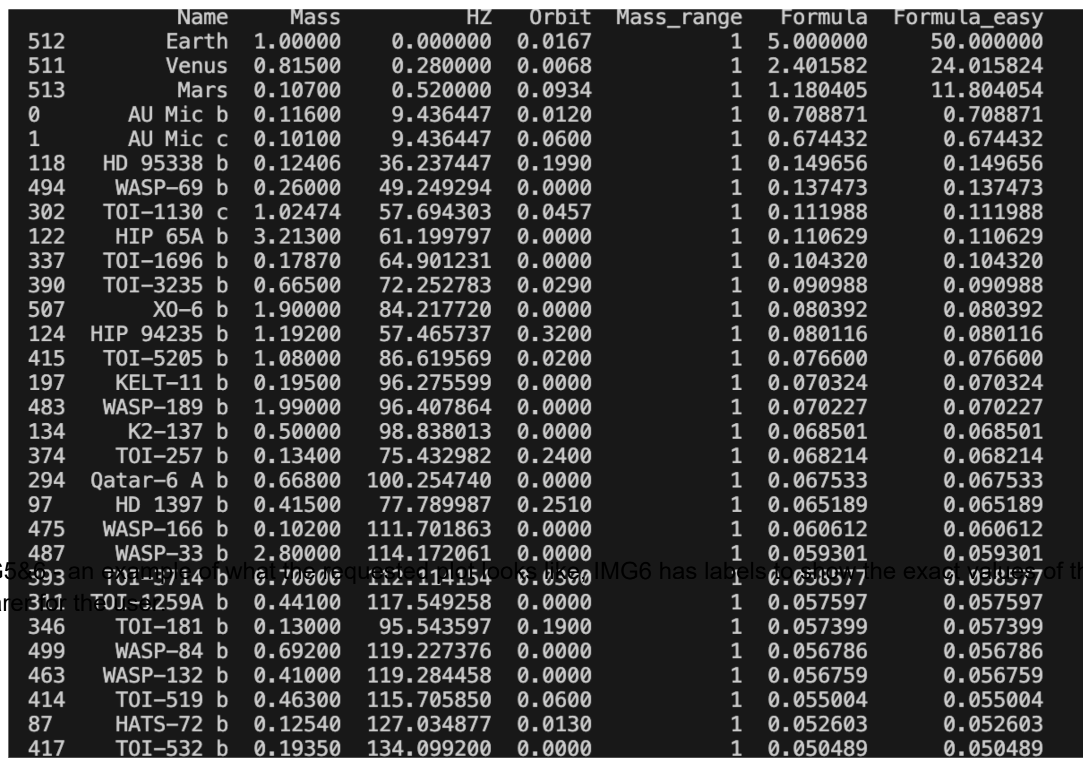
```
Welcome to the Windows to the Universe
What would you like to do?
View general information on planets in the solar system (Press 1)
Compare an attribute across all planets in graphical form (Press 2)
2
Press:
(0) to see a plot of the diameters
(1) to see a plot of the mass
(2) to see a plot of the mean distances from the sun
(3) to see a plot of the orbital periods in earth years
(4) to see a plot of the orbital eccentricities
(5) to see a plot of the mean orbital velocities
(6) to see a plot of the rotation periods in earth days
(7) to see a plot of the inclination of axes
(8) to see a plot of the gravities at equators
(9) to see a plot of the escape velocities
(10) to see a plot of the mean densities
... of all planets
(x) to exit or (b) to go back
5
```

IMG4 - Shows the various options of plots the user can request.



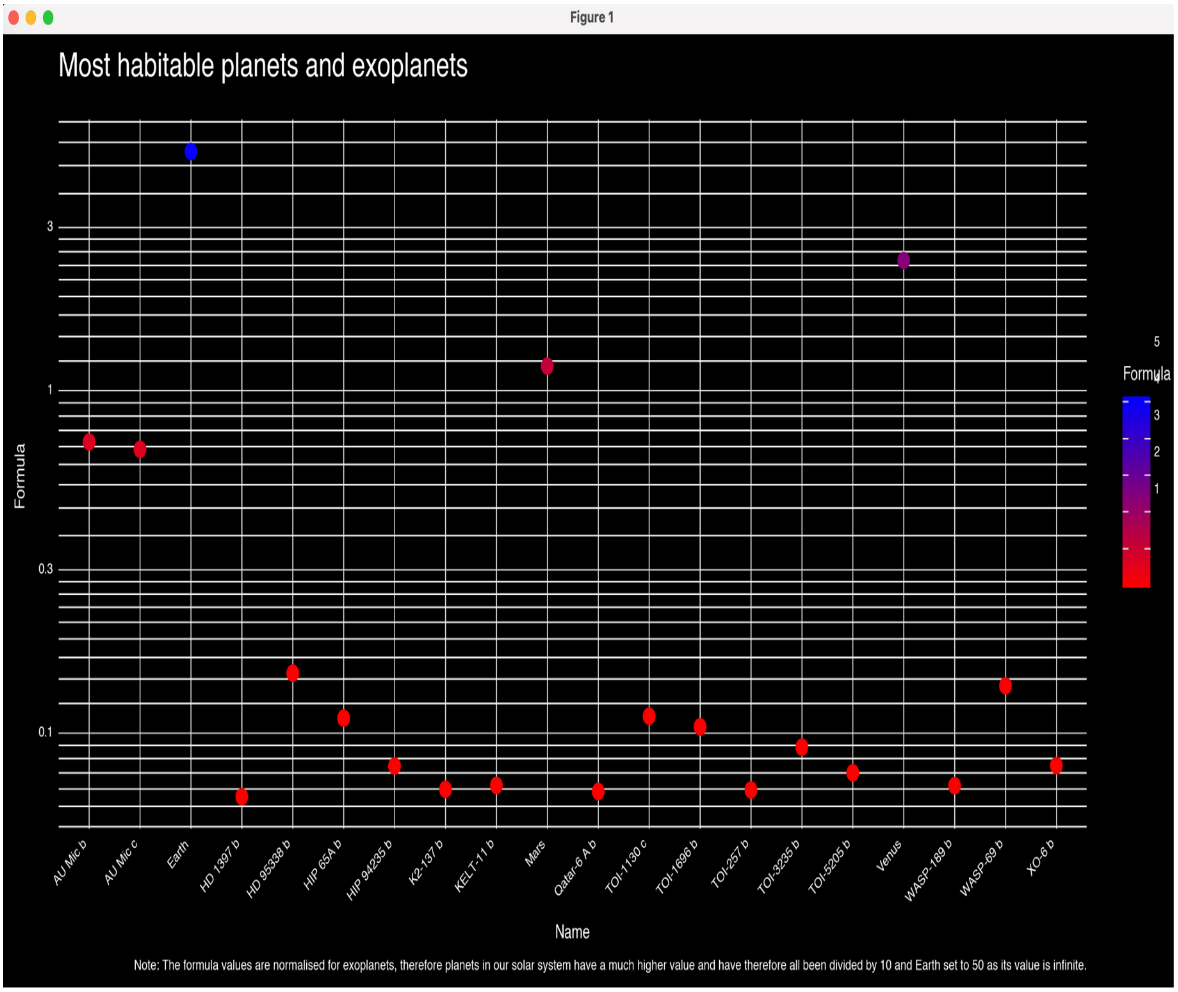


IMG5&6 - an example of what the requested plot looks like, IMG6 has labels to show the exact values of the points to make it clearer for the user.



	Name	Mass	HZ	Orbit	Mass_range	Formula	Formula_easy
512	Earth	1.00000	0.000000	0.0167	1	5.000000	50.000000
511	Venus	0.81500	0.280000	0.0068	1	2.401582	24.015824
513	Mars	0.10700	0.520000	0.0934	1	1.180405	11.804054
0	AU Mic b	0.11600	9.436447	0.0120	1	0.708871	0.708871
1	AU Mic c	0.10100	9.436447	0.0600	1	0.674432	0.674432
118	HD 95338 b	0.12406	36.237447	0.1990	1	0.149656	0.149656
494	WASP-69 b	0.26000	49.249294	0.0000	1	0.137473	0.137473
302	TOI-1130 c	1.02474	57.694303	0.0457	1	0.111988	0.111988
122	HIP 65A b	3.21300	61.199797	0.0000	1	0.110629	0.110629
337	TOI-1696 b	0.17870	64.901231	0.0000	1	0.104320	0.104320
390	TOI-3235 b	0.66500	72.252783	0.0290	1	0.090988	0.090988
507	XO-6 b	1.90000	84.217720	0.0000	1	0.080392	0.080392
124	HIP 94235 b	1.19200	57.465737	0.3200	1	0.080116	0.080116
415	TOI-5205 b	1.08000	86.619569	0.0200	1	0.076600	0.076600
197	KELT-11 b	0.19500	96.275599	0.0000	1	0.070324	0.070324
483	WASP-189 b	1.99000	96.407864	0.0000	1	0.070227	0.070227
134	K2-137 b	0.50000	98.838013	0.0000	1	0.068501	0.068501
374	TOI-257 b	0.13400	75.432982	0.2400	1	0.068214	0.068214
294	Qatar-6 A b	0.66800	100.254740	0.0000	1	0.067533	0.067533
97	HD 1397 b	0.41500	77.789987	0.2510	1	0.065189	0.065189
475	WASP-166 b	0.10200	111.701863	0.0000	1	0.060612	0.060612
487	WASP-33 b	2.80000	114.172061	0.0000	1	0.059301	0.059301
393	TOI-3714 b	0.70000	112.114154	0.0360	1	0.058577	0.058577
311	TOI-1259A b	0.44100	117.549258	0.0000	1	0.057597	0.057597
346	TOI-181 b	0.13000	95.543597	0.1900	1	0.057399	0.057399
499	WASP-84 b	0.69200	119.227376	0.0000	1	0.056786	0.056786
463	WASP-132 b	0.41000	119.284458	0.0000	1	0.056759	0.056759
414	TOI-519 b	0.46300	115.705850	0.0600	1	0.055004	0.055004
87	HATS-72 b	0.12540	127.034877	0.0130	1	0.052603	0.052603
417	TOI-532 b	0.19350	134.099200	0.0000	1	0.050489	0.050489

IMG7 - is the output of the combined solar system and exoplanets dataset with the habitability value included.



IMG8 - The plot of the 20 most habitable planets and exoplanets based on our habitability calculations. The y-axis is in log scale to make the difference between values more readable and we added a disclaimer explaining that the calculations are normalized for exoplanets and therefore the values for the planets in our solar system have been edited to make sense within the graph.