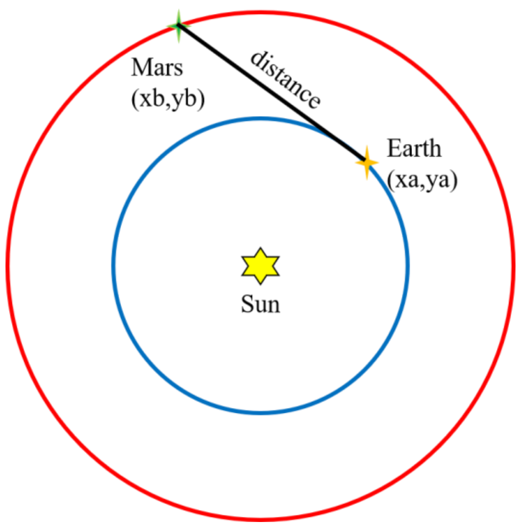
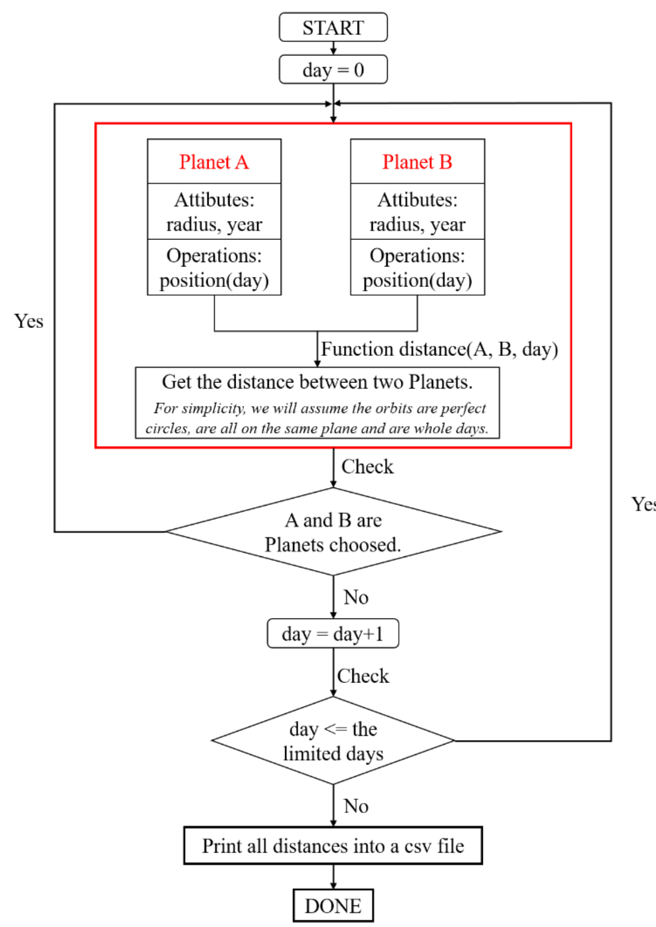
Solar System Project Report

This Solar system/Planet project is designed to calculate the distances between planets based on specific days and simulates the distances for pairs of planets in solar system. The core function built is *position* and *distance*. With given planet radius and period of revolution as earth days (year), this program is capable of calculating position of a planet at a given day and the distance between itself and another planet in the given eight planets.

The simplified understanding to solve the *position* function is to imagine it as a math problem. That is assuming planets' orbits as concentric circles and each circle has their radius and revolution unit. Thus, using trigonometry function *sin* and *cos* is considered as the core of constructing *position* function. By defining radian as 2*pi*day/days_in_a_year, it helps to calculate the radian for a given day. By using *radius*cos(radian)* and *radius*sin(radian)*, it generates out corresponding x and y coordinates. Then, the distance function becomes relatively easier to construct which is basically computing the Euclidean distance between two points. This is simply applying the formula $sqrt((xb-xa)^2+(yb-ya)^2)$.



After having the *Planet* class, position and distance function set up, simulations also becomes relatively straight-forward but requires some time and strategies to simulate through a relatively large number. In the main simulation, a total of 365000 earth days are simulated and relevant results are generated as csv files. The design general structure chart is presented as follows.



As we see from the top-down design chart, it presents a structure that takes attributes from two planets and compute the distance based on given days. The general condition is by checking if two planets are selected, if they yes, then continuously compute their distance with the *distance* function, so that pairs of distances are generated between planets. The iteration continues by adding days for planets, then iterate through the rest days on them until the days reach the set limit, then it generates out the csv file with format as: "distance_for_1000_earth_years_{start_day}-{end_day-1}."

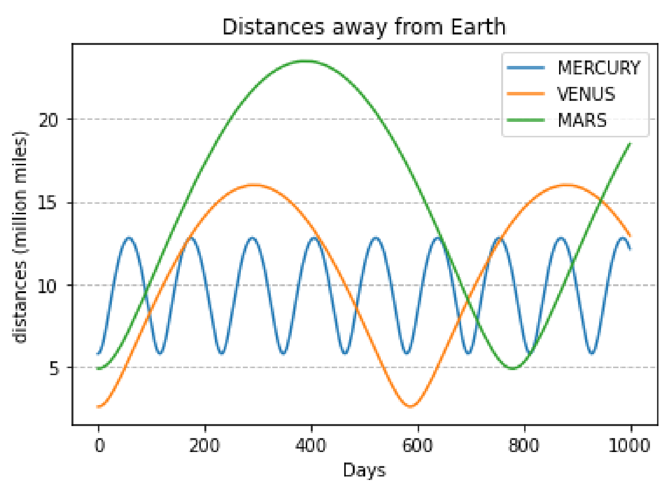
As it has been mentioned in the script file, it takes a certain amount of time to calculate out all the results. It takes a total about 24.6 hours to generate out all simulation results. The relevant output files have been uploaded to my *GitHub* repository. The strategy of solving this problem is by splitting the result to sections as mentioned. By doing so, it does improve the computing efficiency, meanwhile, it neither shows any negative effects such as losing data or outputting incorrect results. The method is applied as splitting 365000 days of simulation to 11 identical parts but with different day ranges. The first part with day range (0-164000) takes 20.7 hours which is 85% of total time. Therefore, by setting day interval of 20,000 days, a total of 10 sectional simulations are generated (164001-365000). After finishing all simulations and generating 11 csv output files, results are combined through pandas module as the complete data frame.

	Unnamed: 0	day	Planets	mercury	venus	earth	mars	jupiter	saturn	uranus	neptune
0	0	0	mercury	0.00	3.20	5.80	10.70	44.90	85.40	175.50	284.50
1	1	0	venus	3.20	0.00	2.60	7.50	41.70	82.20	172.30	281.30
2	2	0	earth	5.80	2.60	0.00	4.90	39.10	79.60	169.70	278.70
3	3	0	mars	10.70	7.50	4.90	0.00	34.20	74.70	164.80	273.80
4	4	0	jupiter	44.90	41.70	39.10	34.20	0.00	40.50	130.60	239.60
...	...	...	...	...	...	...	...	...	...	...	...
120003	2920003	365000	mars	17.43	8.71	19.04	0.00	35.50	99.18	190.73	286.68
120004	2920004	365000	jupiter	51.88	41.73	48.54	35.50	0.00	116.28	209.62	268.96
120005	2920005	365000	saturn	87.84	91.08	80.72	99.18	116.28	0.00	93.34	240.77
120006	2920006	365000	uranus	177.29	182.25	171.77	190.73	209.62	93.34	0.00	255.38
120007	2920007	365000	neptune	289.83	284.40	279.46	286.68	268.96	240.77	255.38	0.00

2920024 rows × 11 columns

It takes about 13.5 minutes to compute the mean distances and export as the csv file. The output matrix shows reasonable result on the right. As we see that even though that Mercury is on average the closest planet to earth. This generally proves that even though the orbits of Venus and Mars are the closest to the earth, due to the reason that Mercury has higher revolution period (88 days) which produces it much more frequent chances to have the closest distance to earth on average. In comparison, Venus has revolution period of 225 days and Mars has even 687 days, it is possible to see they produce the closest distance instead of average closest distances computationally. As for further information, the matplotlib plot is also created as we see that Mercury is indeed on average the "closest" planet to earth based on 1000-day time series.

	mercury	venus	earth	mars	jupiter	saturn	uranus	neptune
mercury	0.000000	7.164883	9.632364	14.416315	48.463270	88.934557	179.017217	288.010716
venus	7.164883	0.000000	10.550810	15.002387	48.632192	89.025683	179.062136	288.038397
earth	9.632364	10.550810	0.000000	15.771669	48.849467	89.142644	179.119818	288.075643
mars	14.416315	15.002387	15.771669	0.000000	49.445328	89.464797	179.279246	288.170727
jupiter	48.463270	48.632192	48.849467	49.445328	0.000000	95.491055	182.202339	289.948236
saturn	88.934557	89.025683	89.142644	89.464797	95.491055	0.000000	190.080733	295.306346
uranus	179.017217	179.062136	179.119818	179.279246	182.202339	190.080733	0.000000	320.885517
neptune	288.010716	288.038397	288.075643	288.170727	289.948236	295.306346	320.885517	0.000000



As for some further discussions regarding this project, some concerns and findings are worth considering. Efficiency appears to be one of the problems, as mentioned, the way to encounter such problem is to compute as sectionals. As the days accumulate, the runtime decreases dramatically.

For instance, it only takes about 22 seconds to run for 1000 days under the limit of 100000 days, but it takes more than 10 minutes to run 1000 days for days over 100000 day. Some more efficient methods could be applied to simulate it faster such as only calculating of the simulation since the distance matrix is symmetrical. It might also be helpful by applying the technique of parallel computing. Even by applying the idea of computing by pieces, it also saves a lot of time in general. As for the rest 200000 days, it only takes a total about three hours to complete.

When computing the average distance, is might be an idea of randomly sampling out 1000 days but it might not be better or more accurate than simulate all 365000 days. As for reference, the matrix of random 1000 days is also generated as comparison. It only takes 23 second to generate and shows similar results as the complete 1000-year version. But the argument remains that it might be ambiguous to conclude the random simulation is better since such result depends on days selected. There might always be chance that it generates unreasonable results.

	mercury	venus	earth	mars	jupiter	saturn	uranus	neptune
mercury	0.00000	7.11953	9.70920	14.43251	48.38348	88.91974	179.12208	288.01487
venus	7.11953	0.00000	10.53301	14.90608	48.60414	89.16094	179.02247	287.92745
earth	9.70920	10.53301	0.00000	15.88746	49.11912	89.15760	179.36755	288.14756
mars	14.43251	14.90608	15.88746	0.00000	49.48863	89.23732	179.08412	288.47863
jupiter	48.38348	48.60414	49.11912	49.48863	0.00000	96.94949	182.77070	287.73106
saturn	88.91974	89.16094	89.15760	89.23732	96.94949	0.00000	189.38963	294.91496
uranus	179.12208	179.02247	179.36755	179.08412	182.77070	189.38963	0.00000	317.31512
neptune	288.01487	287.92745	288.14756	288.47863	287.73106	294.91496	317.31512	0.00000

As we look through the whole simulation, when we review the definition of "closest" again, we are able to conclude that "closest" matters. The main reason is because in order to find the closest distance between two planets, we must find the closest Euclidean distance between the two. But we should always keep in mind that what this project has done is just a general simulation. It assumes all planets and orbits are in a 2-D system without considering gravity or any acceleration in the universe.

In reality, simulation and computation takes place in a 3-D space with other important factors and the most difference is that the orbits are not perfect circles, and they have angles in 3-D space. As mentioned in the precious content, the distance we have been focusing on is the average distance, it would be different if focuses on "closest" distance on specific days such that Venus is the closest to earth around day 580 with less than 25 million miles and that is a lot closer than the calculated average "closest" distance of Mercury.

Links:

Please use the following link to access all full csv outputs through my Google drive if interested.

https://drive.google.com/drive/folders/1uRGaKht4wU9DFOoqU3A7fLGOVJKwXGIe?usp=sharing

Please use the following link to access my other Python projects through GitHub if interested.

https://github.com/WenxuanGu/Boot_Camp_Challenges/tree/main/Challenges/Python_Modules