import matplotlib.pyplot as plt

from matplotlib import animation

These lines import the necessary modules from the `matplotlib` library. `matplotlib.pyplot` is used for creating plots, and `animation` is used to create animations.

G = 6.67e-11

Mb = 4.0e30

Ms = 2.0e30

Me = 5.972e24

Mm = 6.39e23

Mc = 6.39e20

AU = 1.5e11

daysec = 24.0 \* 60 \* 60

These lines define various physical constants that will be used throughout the simulation. These constants include the gravitational constant (`G`), masses of different celestial bodies in kilograms (`Mb`, `Ms`, `Me`, `Mm`, `Mc`), the astronomical unit in meters (`AU`), and the number of seconds in a day (`daysec`).

e\_ap\_v = 29290

m\_ap\_v = 21970

commet\_v = 7000

These lines define initial velocities for Earth at its aphelion (farthest point from the Sun), Mars at its aphelion, and a comet.

gravconst\_e = G \* Me \* Ms

gravconst\_m = G \* Mm \* Ms

gravconst\_c = G \* Mc \* Ms

These lines calculate the gravitational constants specific to each celestial body by multiplying the gravitational constant `G` with the product of the relevant masses. These constants will be used to compute gravitational forces later.

xe, ye, ze = 1.0167 \* AU, 0, 0

xve, yve, zve = 0, e\_ap\_v, 0

xm, ym, zm = 1.666 \* AU, 0, 0

xvm, yvm, zvm = 0, m\_ap\_v, 0

xc, yc, zc = 2 \* AU, 0.3 \* AU, 0

xvc, yvc, zvc = 0, commet\_v, 0

xs, ys, zs = 0, 0, 0

xvs, yvs, zvs = 0, 0, 0

These lines set up the initial conditions for the positions and velocities of Earth (`xe`, `ye`, `ze`, `xve`, `yve`, `zve`), Mars (`xm`, `ym`, `zm`, `xvm`, `yvm`, `zvm`), the comet (`xc`, `yc`, `zc`, `xvc`, `yvc`, `zvc`), and the Sun (`xs`, `ys`, `zs`, `xvs`, `yvs`, `zvs`). These initial positions and velocities define the starting state of the simulation.

t = 0.0

dt = 1 \* daysec

These lines initialize the simulation's time (`t`) and the time step (`dt`). The time step determines how much time each iteration of the simulation represents. In this case, `dt` is set to one day.

xelist, yelist, zelist = [], [], []

xslist, yslist, zslist = [], [], []

xmlist, ymlist, zmlist = [], [], []

xclist, yclist, zclist = [], [], []

These lines initialize empty lists to store the positions of Earth, Mars, the comet, and the Sun over time. The lists will be populated as the simulation progresses.

while t < 5 \* 365 \* daysec:

This line starts a `while` loop that continues as long as the simulation time `t` is less than five years (in seconds).

rx, ry, rz = xe - xs, ye - ys, ze - zs

modr3\_e = (rx \*\* 2 + ry \*\* 2 + rz \*\* 2) \*\* 1.5

fx\_e = -gravconst\_e \* rx / modr3\_e

fy\_e = -gravconst\_e \* ry / modr3\_e

fz\_e = -gravconst\_e \* rz / modr3\_e

These lines calculate the gravitational forces acting on Earth due to the Sun's gravitational attraction. It calculates the components of the distance vector (`rx`, `ry`, `rz`) from the Sun to Earth, computes the cube of the distance magnitude (`modr3\_e`), and then calculates the individual force components (`fx\_e`, `fy\_e`, `fz\_e`) using Newton's law of universal gravitation.

xve += fx\_e \* dt / Me

yve += fy\_e \* dt / Me

zve += fz\_e \* dt / Me

xe += xve \* dt

ye += yve \* dt

ze += zve \* dt

xelist.append(xe)

yelist.append(ye)

zelist.append(ze)

These lines update Earth's velocity and position due to the gravitational force from the Sun. The velocities (`xve`, `yve`, `zve`) are updated using the acceleration (`fx\_e`, `fy\_e`, `fz\_e`) divided by Earth's mass (`Me`). Then, the positions (`xe`, `ye`, `ze`) are updated using the new velocities, and these positions are appended to the respective lists.

This same pattern of calculations and updates is repeated for Mars and the comet.

xvs += -(fx\_e + fx\_m) \* dt / Ms

yvs += -(fy\_e + fy\_m) \* dt / Ms

zvs += -(fz\_e + fz\_m) \* dt / Ms

xs += xvs \* dt

ys += yvs \* dt

zs += zvs \* dt

xslist.append(xs)

yslist.append(ys)

zslist.append(zs)

These lines update the position and velocity of the Sun due to the gravitational forces from

Earth and Mars. The Sun's velocities (`xvs`, `yvs`, `zvs`) are updated using the sum of the forces from Earth and Mars, divided by the Sun's mass (`Ms`). Then, the positions (`xs`, `ys`, `zs`) are updated using the new velocities, and these positions are appended to the respective lists.

t += dt

This line increments the simulation time `t` by the time step `dt`, effectively moving the simulation forward in time.

print('data ready')

This line prints a message to indicate that the data preparation phase of the simulation is complete.

The remaining lines of code set up the animation figure, define the `update` function to update the animation frames, and create the animation using the `animation.FuncAnimation` function. The animation is then displayed using `plt.show()`.