```python
#!/bin/anaconda3/bin/python

import math
import sympy as sym
import math
import numpy as np
from numpy import linalg as la
import scipy
from scipy import integrate
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

from ode_2 import ode_solve

print("Lets do the 2-body problem")

#Initial conditions
m_1 = 1.0
m_2 = 1000.0
#G = 6.67408e-11 #m^3 kg^-1 s^-2
G = 1
x_1 = np.array([0.0, 0.0])
x_2 = np.array([0.0, 10.0])
v_1 = np.array([9.0, 9.0])
v_2 = np.array([0.5, 0.0])

#Lets make a vector y that will give is a 1st order ode
#2D
#Y = np.array([x_1,y_1,x_2,x_2,dx_1,dy_1,dx_2,dy_2])

def func(Y, t):

    #Positions of the two bodies
    y_1 = np.array([ Y[0],Y[1] ])
    y_2 = np.array([ Y[2],Y[3] ])

    #Velocites
    dy_1 = np.array([ Y[4],Y[5] ])
    dy_2 = np.array([ Y[6],Y[7] ])

    #Accelerations
    dy_3 = -G*m_2*(y_1 - y_2)/(la.norm(y_1-y_2)**3)
    dy_4 = -G*m_1*(y_2 - y_1)/(la.norm(y_2-y_1)**3)

    #We will have our 8 ode's
    dy = np.array([dy_1[0],dy_1[1],dy_2[0],dy_2[1],dy_3[0],dy_3[1],dy_4[0],dy_4[1]])

    return dy
#End function

#Need a 1D vector for the input
Y=np.concatenate((x_1,x_2,v_1,v_2))

#Evenly spaced time
t = np.arange(0,100.0,0.001)


#Scipy's in built ode
sol = integrate.odeint(func,Y,t)
plt.plot(sol[:,0],sol[:,1])
plt.plot(sol[:,2],sol[:,3])
#plt.show()
```

```
#Our RK ode solver
sol_my = ode_solve(func,t,Y)

plt.plot(sol_my[:,0],sol_my[:,1])
plt.plot(sol_my[:,2],sol_my[:,3])
plt.ylabel('y')
plt.xlabel('x')
plt.show()


#error Between the two solvers
print(sol_my[:,0] - sol[:,0])
```