```python
"""
Registration: xxxx;
Description: Heun's Method 2nd Order ODE : Damped SHM d2x/dt2 + lambda*dx/dt + kx = 0
             with IVP (t0, x0, y0), tn and dt.
Author: AKB
"""

import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")

# Choose first solve=1 to generate files and then plot=1 to plot them all
solve = 0;
plot  = 1;

# Enter parameters & initial conditions
#lam, x, t, tn, dt= input('Enter rate constant and initial value x, t, tn, dt: ')
k, lam, m = 1.0, 0.2, 1.0
t, tn, dt = 0.0, 50.0, 0.01
step = int(tn/dt)
x = 0; y = 1; # Initial condition

# Analytic position (Weak Damping) X(t) = Ae^(-lam*t/2m)cos(wprime*t-phi)
discr = k/m - lam**2/(4.0*m**2)
wprime = np.sqrt(discr)
phi = np.pi/2

if(solve):

   def dshm(x,y,t,k,lam):
       dxdt = y
       dydt = -(k*x + lam*y)/m
       return np.array([dxdt, dydt])

   # Open a file (in C, fp is file pointer)
   if  (dt==0.01):   fp = open("data/eulerPC_ode2_dt0.01.dat","w")
   elif(dt==0.005):  fp = open("data/eulerPC_ode2_dt0.005.dat","w")
   elif(dt==0.001):  fp = open("data/eulerPC_ode2_dt0.001.dat","w")
   elif(dt==0.0005): fp = open("data/eulerPC_ode2_dt0.0005.dat","w")

   # Predictor-Corrector iteration step
   for i in range(step):

       # Predict x using Euler method
       [dxdt, dydt] = dshm(x,y,t,k,lam)
       tp = t + dt
       xp = x + dt*dxdt
       yp = y + dt*dydt

       # Correct using Modified Euler
       [dxpdt, dypdt] = dshm(xp,yp,t,k,lam)
       xc = x + 0.5*dt*(dxdt+dxpdt)
       yc = y + 0.5*dt*(dydt+dypdt)
       t = tp
       x = xc
       y = yc

       # Print it using file pointer
       print >> fp,t,x,y

   # Close the file
   fp.close()
   print 'Final value at t = ',t,' is x = ', x,' and y = ', y
```

```
if(plot):

   # Read the datafiles
   fp1 = np.loadtxt('data/eulerPC_ode2_dt0.01.dat');   T1 = fp1[:,0]; X1 = fp1[:,1]; Y1 =
fp1[:,2]
   fp2 = np.loadtxt('data/eulerPC_ode2_dt0.005.dat');  T2 = fp2[:,0]; X2 = fp2[:,1]; Y2 =
fp2[:,2]
   fp3 = np.loadtxt('data/eulerPC_ode2_dt0.001.dat');  T3 = fp3[:,0]; X3 = fp3[:,1]; Y3 =
fp3[:,2]
   fp4 = np.loadtxt('data/eulerPC_ode2_dt0.0005.dat'); T4 = fp4[:,0]; X4 = fp4[:,1]; Y4 =
fp4[:,2]

   # Plot the data
   plt.figure(1)
   plt.subplot(3,1,1)
   plt.plot(T1, X1, 'd', lw=2, ms=4, c='olive', label=r'$dt=10^{-2}$')
   plt.plot(T2, X2, 'x', lw=2, ms=4,  c='b',     label=r'$dt=5\times 10^{-3}$')
   plt.plot(T3, X3, '>', lw=2, ms=4,  c='m',     label=r'$dt=10^{-3}$')
   plt.plot(T4, X4, '<', lw=2, ms=4,  c='c',     label=r'$dt=5\times 10^{-4}$')
   plt.legend(loc='right', prop={'size':24})
   plt.axis([0, tn, -1, 1])
   plt.axhline(lw=2) # draw a horizontal line
   plt.xticks(size = 20)
   plt.ylabel('Position', size = 26)
   plt.yticks(size = 20)
   plt.grid()
   if(discr > 0):  plt.title('Weakly Damped Harmonic Motion (Heun)',fontsize=26)
   elif(discr==0): plt.title('Critically Damped Harmonic Motion (Heun)',fontsize=26)
   else:           plt.title('Heavily Damped Harmonic Motion (Heun)',fontsize=26)

   plt.subplot(3,1,2)
   plt.plot(T1, X1-np.exp(-lam*T1/(2.0*m))*np.cos(wprime*T1-phi), 'd', lw=2, ms=2,
c='olive')
   plt.plot(T2, X2-np.exp(-lam*T2/(2.0*m))*np.cos(wprime*T2-phi), 'x', lw=2, ms=2,  c='b')
   plt.plot(T3, X3-np.exp(-lam*T3/(2.0*m))*np.cos(wprime*T3-phi), '>', lw=2, ms=2,  c='m')
   plt.plot(T4, X4-np.exp(-lam*T4/(2.0*m))*np.cos(wprime*T4-phi), '<', lw=2, ms=2,  c='c')
   plt.xlim([0, tn])
   plt.xticks(size = 20)
   plt.ylabel(r'$L_1$ norm (Position)', size = 26)
   plt.yticks(size = 20)
   plt.grid()

   plt.subplot(3,1,3)
   plt.plot(T1, Y1, 'd-', lw=2, ms=4, c='olive', label=r'$dt=10^{-2}$')
   plt.plot(T2, Y2, 'x-', lw=2, ms=4,  c='b',     label=r'$dt=5\times 10^{-3}$')
   plt.plot(T3, Y3, '>-', lw=2, ms=4,  c='m',     label=r'$dt=10^{-3}$')
   plt.plot(T4, Y4, '<-', lw=2, ms=4,  c='c',     label=r'$dt=5\times 10^{-4}$')
   plt.axis([0, tn, -1, 1])
   plt.axhline(lw=2) # draw a horizontal line
   plt.xlabel('Time', fontsize = 26)
   plt.xticks(size = 20)
   plt.ylabel('Velocity', size = 26)
   plt.yticks(size = 20)
   plt.grid()
   plt.text(35,-0.45,r'$k='+str(k)+', \lambda=' + str(lam)+', m=' + str(m)+'$',
fontsize=26)
   plt.text(35,-0.85,r'$m\frac{d^2x}{dt^2}+\lambda\frac{dx}{dt}+kx=0$', fontsize=30)
   plt.show()
```