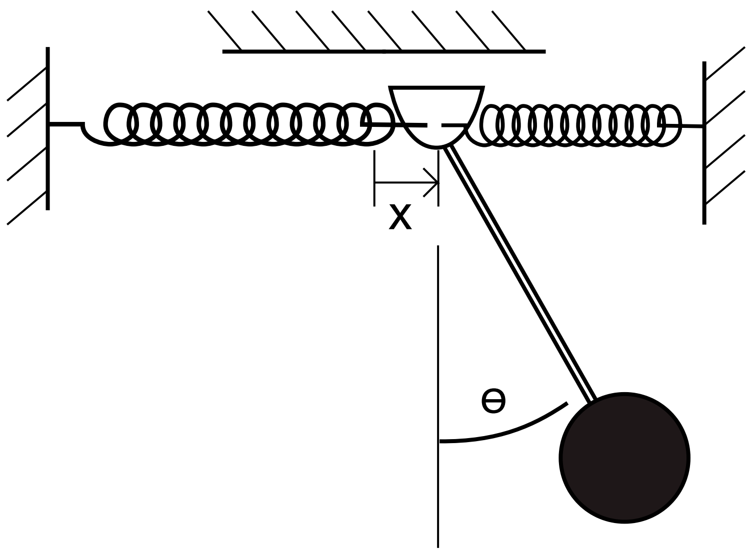
In [ ]:
```
import numpy as np
from numpy import sin,cos,pi
import matplotlib.pyplot as plt
import scipy
from scipy.integrate import solve_ivp
plt.style.use('fivethirtyeight')
```

In [ ]:
```
solve_ivp
```

Out[ ]:
```
<function scipy.integrate._ivp.ivp.solve_ivp(fun, t_span, y0, method='RK45', t_eva
l=None, dense_output=False, events=None, vectorized=False, args=None, **options)>
```

# Homework #4

## Problem 1




The pendulum bob of mass m, shown in the figure above, is suspended by an inextensible string from the point p. This point is free to move along a straight horizontal line under the action of the springs, each having a constant k. Assume that the mass is displaced only slightly from the equilibrium position and released. Neglecting the mass of the springs, show that the pendulum oscillates with a period of

$$P = 2\pi\sqrt{\frac{mg+2kr}{2kg}}$$

use a first-order Taylor series approximation for $\sin\theta \approx \theta$ and $\cos\theta \approx 1$

Solve for $\theta(t)$ if m=0.1 kg, r=1 m, $\theta(0)$=pi/6 rad, and $\dot{\theta}(0)$=0 rad/s for 2 cases:

a. k=20 N/m

b. k=$\infty$ N/m

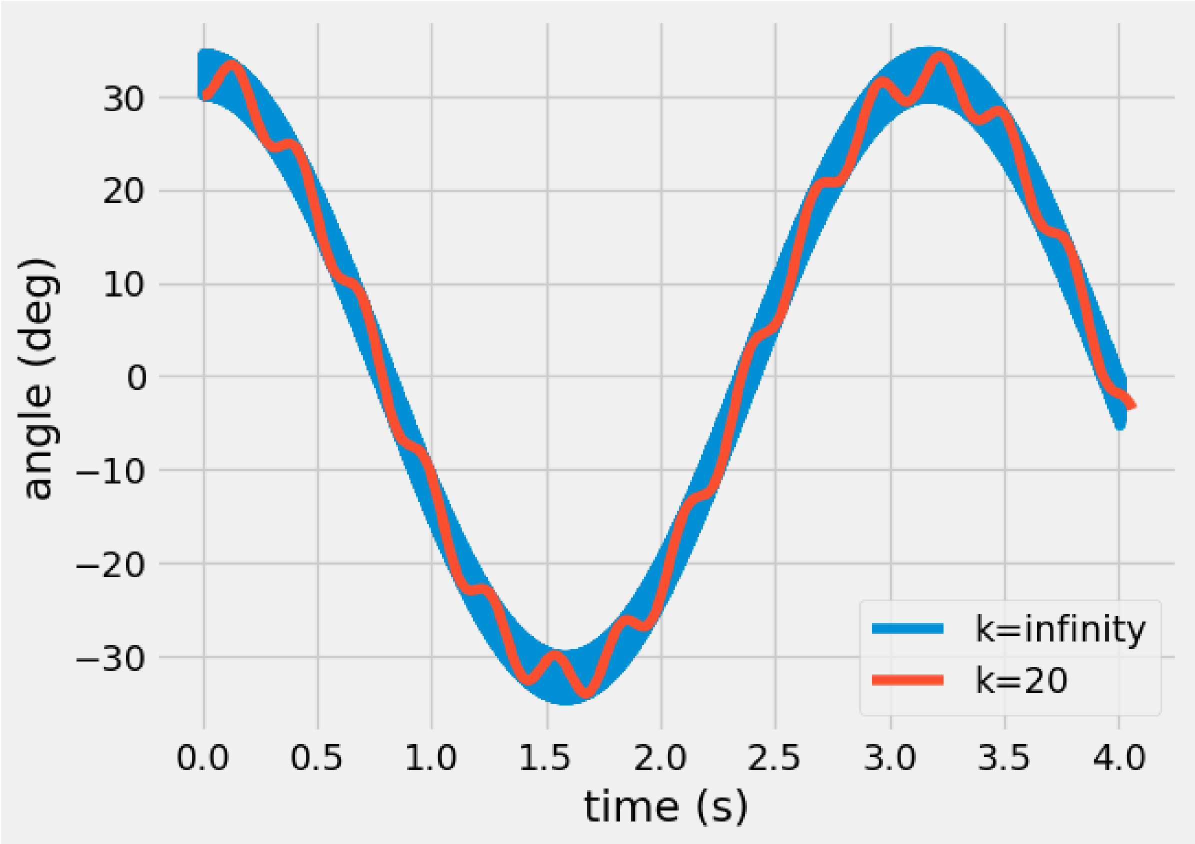
c. Plot the solutions of $\theta(t)$ for 2 periods on one figure

In [ ]:
```
l=1
m=0.1
k=20
g=9.81
P=2*pi/np.sqrt((2*k*g)/(2*k*l+m*g))
k=999999
P2=2*pi/np.sqrt((2*k*g)/(2*k*l+m*g))
print(P)
t=np.linspace(0,2*P,10000);
t2=np.linspace(0,2*P2,10000);
# your work
# your new solutions, convert rad to deg with 180/pi
def my_ode_1(t,r, k=20):
    """ Help documentation for "my_ode"
     input is time, t (s) and r=[position p (m), angle (rad), velocity p (m/s), ang
     output is dr=[velocity p (m/s), angle velocity (rad/s), accel p (m/s/s), angle
     the ODE is defined by:

     dr = f(t,r)"""
    l=1
    m=0.1
    g=9.81
    k=20
    dr=np.zeros(np.size(r))
    dr[0]=r[2]
    dr[1]=r[3]

    x, a, v, w = r

    M = np.array([[m, m*l/2],
                  [m*l/2, m*l**2/4*5]])
    rhs = np.array([m*l/2*w**2*a - 2*k*x,
                      -m*g*l/2*a])

    dr[2:] = np.linalg.solve(M, rhs)

    return dr

def my_ode_2(t,r, k=999999):
    """ Help documentation for "my_ode"
     input is time, t (s) and r=[position p (m), angle (rad), velocity p (m/s), ang
     output is dr=[velocity p (m/s), angle velocity (rad/s), accel p (m/s/s), angle
     the ODE is defined by:

     dr = f(t,r)"""
    l=1
    m=0.1
    g=9.81
    dr=np.zeros(np.size(r))
    dr[0]=r[2]
    dr[1]=r[3]
```

```
    x, a, v, w = r

    M = np.array([[m, m*l/2],
                  [m*l/2, m*l**2/4*5]])
    rhs = np.array([m*l/2*w**2*a- 2*k*x,
                    -m*g*l/2*a])

    dr[2:] = np.linalg.solve(M, rhs)

    return dr

# print(compound_pendulum(0, np.array([0.1, np.pi/6, 0, 0])))

sol = solve_ivp(my_ode_1, [0, 2*P], [0.1, np.pi/6, 0, 0], t_eval=t)
sol2 = solve_ivp(my_ode_2, [0, 2*P], [0.1, np.pi/6, 0, 0], t_eval=t2)

a_inf =  t# create solution for k=infty
a_20 = t # create solution for k=20 N/m

plt.plot(t2,sol2.y[1]*180/pi, label='k=infinity')
plt.plot(t,sol.y[1]*180/pi, label='k=20')
plt.xlabel('time (s)')
plt.ylabel('angle (deg)')
plt.legend()
```

```
2.0305170699770856
```

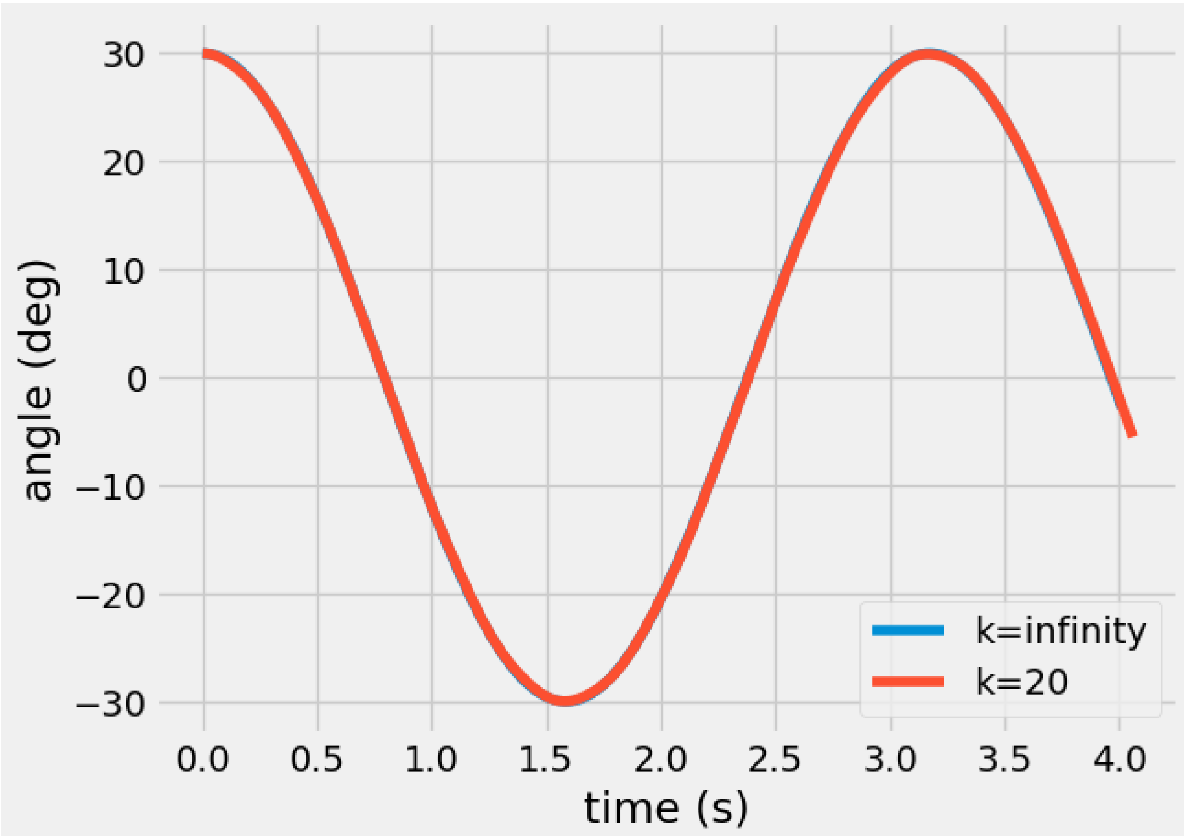
Out[ ]: <matplotlib.legend.Legend at 0x233d9f62220>




In [ ]:
```
sol = solve_ivp(my_ode_1, [0, 2*P], [0, np.pi/6, 0, 0], t_eval=t)
sol2 = solve_ivp(my_ode_2, [0, 2*P], [0, np.pi/6, 0, 0], t_eval=t2)
```

```
a_inf =  t# create solution for k=infty
a_20 = t # create solution for k=20 N/m

plt.plot(t2,sol2.y[1]*180/pi, label='k=infinity')
plt.plot(t,sol.y[1]*180/pi, label='k=20')
plt.xlabel('time (s)')
plt.ylabel('angle (deg)')
plt.legend()
```

Out[ ]: `<matplotlib.legend.Legend at 0x233db8f6fa0>`



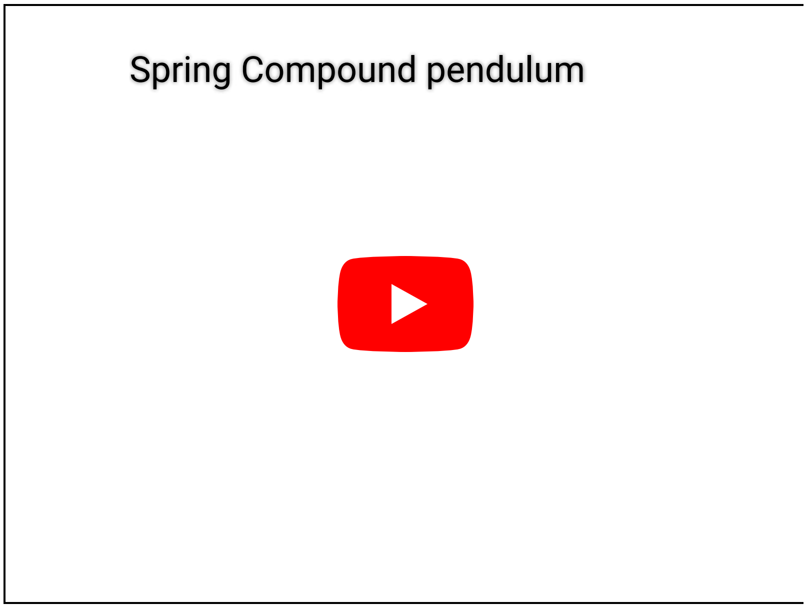

In [ ]:
```
from scipy.linalg import *
from scipy.optimize import fsolve,root
```

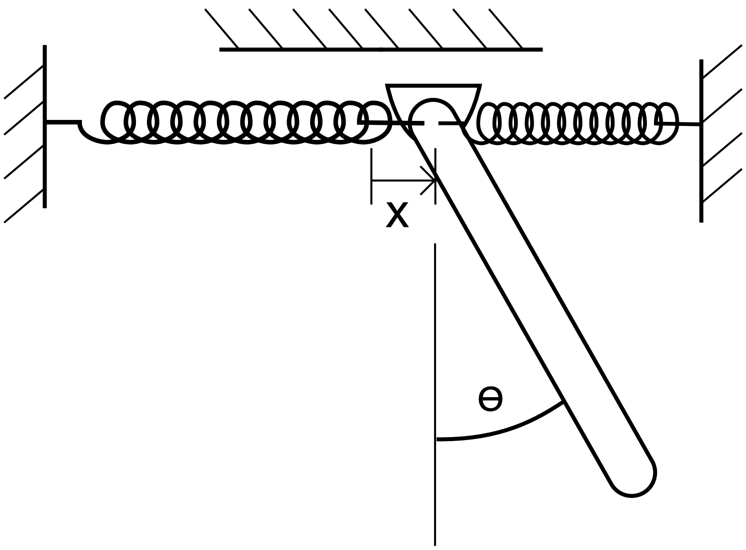
In [ ]:
```
from scipy.integrate import solve_ivp # import the ordinary differential equation i
```

# Problem 2

In [ ]:
```
from IPython.display import YouTubeVideo
YouTubeVideo('eOvwiYRroso')
```

Out[ ]:




The pendulum arm of mass m, shown in the figure above, is held in place by two springs. This point is free to move along a straight horizontal line under the action of the springs, each having a constant k. Assume that the mass is displaced only slightly from the equilibrium position and released. Neglecting the mass of the springs, solve for the nonlinear equations of motion and use the `solve_ivp` to determine $\theta(t)$

Solve for $\theta(t)$ if m=1 kg, L=1 m, $\theta(0)$=pi/6 rad, and $\dot{\theta}(0)$=0 rad/s for

k=20 N/m

Plot the nonlinear solutions of $\theta(t)$ for 2 periods on one figure

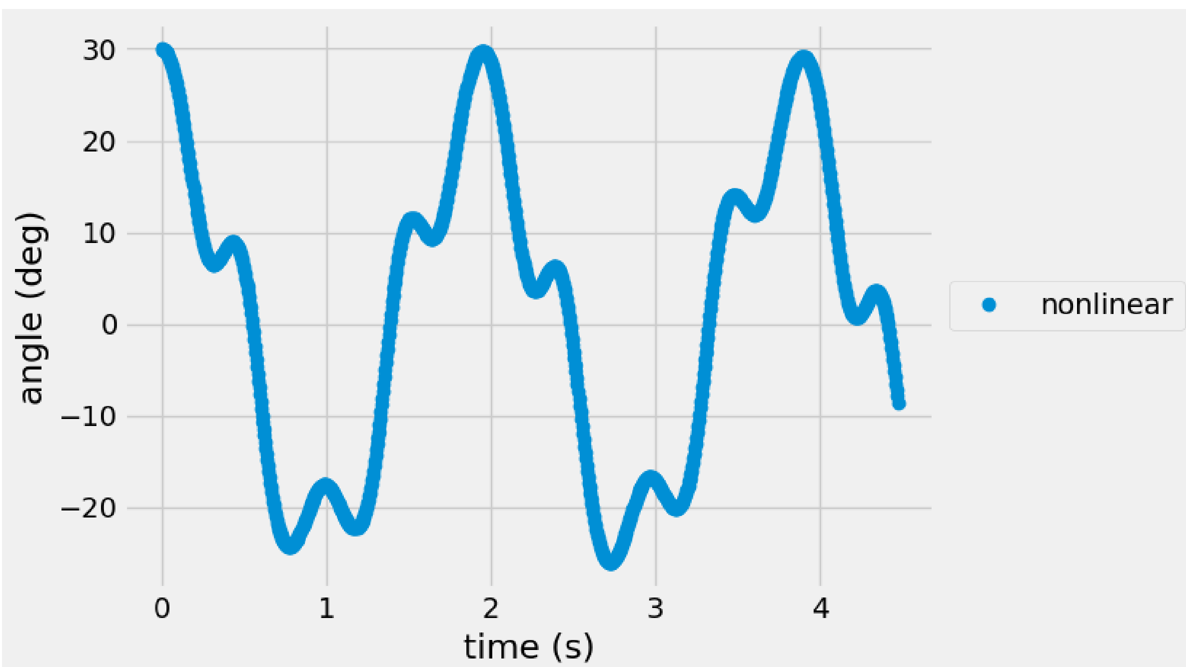
In [ ]:
```
def my_ode(t,r,):
    """ Help documentation for "my_ode"
     input is time, t (s) and r=[position p (m), angle (rad), velocity p (m/s), ang
     output is dr=[velocity p (m/s), angle velocity (rad/s), accel p (m/s/s), angle
```

```
    the ODE is defined by:

    dr = f(t,r)"""
    l=1
    m=1
    k=20
    g=9.81
    dr=np.zeros(np.size(r))
    dr[0]=r[2]
    dr[1]=r[3]

    x, a, v, w = r
    M = np.array([[m, m*l/2*np.cos(a)],
                  [m*l/2*np.cos(a), m*l**2/3]])
    rhs = np.array([m*l/2*w**2*np.sin(a) - 2*k*x,
                    -m*g*l/2*np.sin(a)])

    dr[2:] = np.linalg.solve(M, rhs)

    return dr
```

In [ ]:

```
l=1
m=1
k=20
g=9.81
P=2*pi/np.sqrt((2*k*g)/(2*k*l+m*g))
t = np.linspace(0, 2*P, 1000)
r=solve_ivp(my_ode,[0,2*P],[0, pi/6,0,0], t_eval=t); # default = 'RK45'
plt.plot(r.t, r.y[1]*180/pi,'o',label='nonlinear') # <-------------- your new plot,
plt.legend(loc='center left', bbox_to_anchor=(1, 0.5))
plt.xlabel('time (s)')
plt.ylabel('angle (deg)')
```

Out[ ]: Text(0, 0.5, 'angle (deg)')



In [ ]:

```

```