```
#!/usr/bin/env python

import numpy as np

def cross_product( vec1, vec2):
  return [ (vec1[1]*vec2[2] - vec2[1]*vec1[2]),
          -(vec1[0]*vec2[2] - vec2[0]*vec1[2]),
           (vec1[0]*vec2[1] - vec2[0]*vec1[1])
           ]

def dot_product( vec1, vec2):
  return sum([ vec1[i]*vec2[i] for i in range(len(vec1)) ])

def magnitude(vec):
  val = 0
  for i in vec:
    val = val + i**2
  return val**(1.0/2.0)

def unit( vec):
  '''
  returns unit vector (direction) of the input vector
  '''
  return np.divide(vec, float(np.linalg.norm(vec)))

def moment_from_weight( weight, radius_to_cg):
  '''
  this function accepts 2 vectors of weight and radius of cg to determine
  moment about the cg
  [ Wx, Wy, Wz] x [ rx, ry, rz]
  '''
  return np.cross( weight, radius_to_cg)

def rotation( theta, axis_of_rotation='z'):
  if axis_of_rotation == 'x':
    return np.matrix(
        (1, 0, 0),
        (0, np.cos(theta), -np.sin(theta)),
        (0, np.sin(theta),  np.cos(theta))
        )
  elif(axis_of_rotation == 'y'):
    return np.matrix(
        (np.cos(theta), 0, -np.sin(theta)),
        (0, 1, 0),
        (np.sin(theta), 0,  np.cos(theta))
        )
  elif(axis_of_rotation == 'z'):
    return np.matrix(
        (np.cos(theta), -np.sin(theta), 0),
        (np.sin(theta),  np.cos(theta), 0),
        (0, 0, 1)
        )
  else:
    return 0

def acceleration_necessary( resistance_vec, pvec1, pvec2, time):
  '''
  this function uses equations of motion to determine the necessary acceleration
  to get the leg to a certain point at a specific time
  currently this function will be just ideal acceleration with no resistances
  '''
  return np.subtract(
      np.divide(np.subtract(pvec2, pvec1), np.divide(time**2, 2)),
      resistance_vec)

def output_torque( inertial_moment, radial_acceleration):
  return np.multipy( inertial_moment, radial_acceleration)



def Angular_Momentum_l2_about_hip():
  '''
  this function will use equations of motion in a cylindrical path to determine
```

```
    the angular momentum about of link 2 about the hip
    '''

def angular_momentum_l1_about_hp():
    '''
    this function will use equations of motion in a cylindrical path to determine
    the angular momentum about of link 1 about the hip
    '''
```