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

import sys
sys.path.append(r"/Users/robertbrothers/Desktop/Fall 2014/Fundamentals_of_Robotics/r
obo_git/python/")
import robotics_functions as rf, sympy as sy, numpy as np

[R, t1, I1, m1, q1, qdot1, qddot1, I2] = sy.symbols("R t1 I1 m1 q1 qdot1 qddot1 I2")

link_list_cm = [[
    [ R, 0, 0, q1],
    ],[
      sy.Matrix(
        [
        [0], [0], [0], [1]
        ])
      ]
    ]

[link_list, ocm] = link_list_cm
A0n = rf.sym_get_A0n(link_list)
J = rf.sym_pt_jacobian(link_list_cm)
qdot = sy.Matrix([
  [qdot1]
  ])

M = [
    sy.Matrix([
      [m1,0,0],
      [0,m1,0],
      [0,0,m1]
      ])
    ]

I = [
    sy.Matrix([
      [I1,0,0],
      [0,I1,0],
      [0,0,I1]
      ])
    ]

q = sy.Matrix([
  [q1],
  ])

tdv_vec = [
    (q1, qdot1),
    (qdot1, qddot1),
    ]
print "A matrices"
for A in A0n:
  sy.pprint(sy.trigsimp(A))
print "A Jacobian"
for j in J:
  sy.pprint(sy.trigsimp(j))
print "a) expression for the Lagrangian of the Particle: \n"
print sy.printing.latex(sy.trigsimp(rf.sym_lagrangian( link_list_cm, M, I, qdot)))
print "\n"
print "b) Find the Equations of Motion of the Particle: \n"
print sy.printing.latex(sy.simplify(sy.trigsimp(rf.sym_torque(link_list_cm, M, I, qd
ot, q, tdv_vec))))

[l1, l2, l3, t1, t2, t3, a1, a2, a3, d1, d2, d3] = sy.symbols("l1 l2 l3 t1 t2 t3 a1
a2 a3 d1 d2 d3")
[q1, q2, qdot1, qdot2, qddot1, qddot2, m1, m2, r1, r2] = sy.symbols("q1 q2 qdot1 qdo
t2 qddot1 qddot2 m1 m2 r1 r2")

link_list_cm = [[
    [0,np.pi/2, 0, q1],
    [0,0,q2,0],
    ],[
      sy.Matrix([
```

```
        [0],[0],[0],[1]]),
      sy.Matrix([
        [0],[0],[q2],[1]])
      ]
    ]

m = np.array([m1, m2])
l = np.array([l1, l2])
r = np.array([r1, r2])


M = [sy.Matrix([
  [m[i],0,0],
  [0,m[i],0],
  [0,0,m[i]]
  ]) for i in range(len(m))]

I = [sy.Matrix([
  [I1,0,0],
  [0,I1, 0],
  [0, 0,I1]
  ]),
  sy.Matrix([
    [I2,0,0],
    [0,I2,0],
    [0,0,I2]
    ])
  ]
q = sy.Matrix([
  [q1],
  [q2]
  ])
qdot = sy.Matrix([
  [qdot1],
  [qdot2]
  ])
tdv_vec = [
    (qdot1,qddot1),
    (qdot2,qddot2),
    (q1, qdot1),
    (q2, qdot2),
    ]
link_list = link_list_cm[0]
A0n = rf.sym_get_A0n(link_list)
J = rf.sym_pt_jacobian(link_list_cm)

print "A matrices"
for A in A0n:
  sy.pprint(sy.trigsimp(A))

print "A Jacobian"
for j in J:
  sy.pprint(sy.trigsimp(j))
print "jacobian"
sy.pprint(rf.sym_pt_jacobian(link_list_cm))
print "\nLagrangian of the Manipulator"
print sy.printing.latex(sy.simplify(sy.trigsimp(rf.sym_lagrangian(link_list_cm, M, I
, qdot)[0])))

print "\nEquations of Motion of the Two Link Manipulator"
print sy.printing.latex(sy.simplify(sy.trigsimp(rf.sym_torque(link_list_cm, M, I, qd
ot, q, tdv_vec))))


link_list_cm = [[
    [l1, 0, 0, q1],
    ],[sy.Matrix([
      [0],[0],[0],[1]])
      ]
    ]

q = sy.Matrix([
  [q1],
```

```
  ])

qdot = sy.Matrix([
  [qdot1],
  ])
tdv_vec = [
    (qdot1,qddot1),
    (q1, qdot1),
    ]

M = [sy.Matrix([
  [m[i],0,0],
  [0,m[i],0],
  [0,0,m[i]]
  ])
  ]

I = [sy.Matrix([
  [0,0,0],
  [0,0, 0],
  [0, 0,0]
  ])
  ]

print "\nLagrangian of the pendulum"
print sy.printing.latex(sy.simplify(sy.trigsimp(rf.sym_lagrangian(link_list_cm, M, I
, qdot)[0])))

print "\nEquations of Motion of the Two Link Manipulator"
print sy.printing.latex(sy.simplify(sy.trigsimp(rf.sym_torque(link_list_cm, M, I, qd
ot, q, tdv_vec))))
```