```python
import numpy as np
import Getting_EOM.linearizeEOM as eoms

# one of these two forms for importing control library functions may be
helpful.
import control as cnt
#from control import tf, bode, etc.

# physical parameters
g = 9.8
theta = 45.0*np.pi/180.0   #this variable just defines the slope of the
block, and may not be used explicitly
m = 0.5
k1 = 0.05
k2 = 0.02
F_max = 5.0
b = 0.1

# simulation parameters
t_start = 0.0
t_end = 40.0
Ts = 0.01
t_plot = 0.1
sigma = 0.05

# ---------------- THINGS ADDED ----------------
# --- Part 2 ---
F_e = -(np.sqrt(2)/2) * m*g

# --- Part 3 ---
print(" ")
print("----- PID Design -----")
# kp = u_max / e_max
e_max = 1.0
kp = (F_max + F_e) / e_max
print("kp: ", kp)
zeta = 0.707
wn = np.sqrt((k1 + kp)/m)
kd = (2*zeta*wn*m) - b
print("wn: ", wn)
print("kd: ", kd)

ki = 0.4
print("ki: ", ki)

# --- Part 4 ---
print(" ")
print("----- Observer Design -----")
int_pole = -5
A = np.array(eoms.A_lin).astype(np.float64)
```

```
B = np.array(eoms.B_lin).astype(np.float64)
C = np.array(eoms.C).astype(np.float64)

des_char_state = [1, 4, 8]
des_char_int = [1, -int_pole]
des_char = np.convolve(des_char_state, des_char_int)
des_poles = np.roots(des_char)

A1 = np.vstack((np.hstack((A, np.zeros((np.size(A, 1), 1)))),
                np.hstack((-C, np.array([[0.0]])))))
B1 = np.vstack((B, 0))

if np.linalg.matrix_rank(cnt.ctrb(A1, B1)) != 3:
    print("The system is not controllable")
else:
    K1 = cnt.place(A1, B1, des_poles)
    K = K1[0][0:2]
    ki2 = K1[0][2]

print("K: ", K)
print("ki: ", ki2)

des_obsv_poles = des_poles[0:2]*5
if np.linalg.matrix_rank(cnt.ctrb(A.T, C.T)) != 2:
    print("The system is not observable")
else:
    L = cnt.acker(A.T, C.T, des_obsv_poles).T

print("L^T: ", L.T)

# --- Part 5 ---
print(" ")
print("----- Loopshape Design -----")
```