# Python Code:

import numpy as np

from numpy.linalg import matrix\_rank

import control as ctrl

# Define the system matrices

# System 1

A1 = np.array([[-4, 1], [0, 2]])

B1 = np.array([[1], [0]])

# System 2

A2 = np.array([[-3, 2], [4, 1]])

B2 = np.array([[0], [1]])

# Check stabilizability and find K for System 1

R1 = np.hstack([B1, A1@B1]) # Controllability matrix [B AB]

if matrix\_rank(R1) == A1.shape[0]: # Check if controllability matrix is full rank

    P1 = np.array([-3, -5]) # Define desired closed-loop poles

    K1 = ctrl.place(A1, B1, P1) # Compute gain K of stabilizing static state-feedback control law u = K\*xp

    print("System 1 is stabilizable by a static state-feedback control law u = K\*xp")

    print("Stabilizing gain matrix K for System 1 with desired poles at {} is:".format(P1))

    print(K1)

else:

    print("System 1 is not stabilizable")

# Check stabilizability and find K for System 2

R2 = np.hstack([B2, A2@B2]) # Controllability matrix [B AB]

if matrix\_rank(R2) == A2.shape[0]: # Check if controllability matrix is full rank

    P2 = np.array([-3, -5]) # Define desired closed-loop poles

    K2 = ctrl.place(A2, B2, P2) # Compute gain K of stabilizing static state-feedback control law u = K\*xp

    print("\nSystem 2 is stabilizable by a static state-feedback control law u = K\*xp")

    print("Stabilizing gain matrix K for System 2 with desired poles at {} is:".format(P2))

    print(K2)

else:

    print("\nSystem 2 is not stabilizable")

# Output:

System 1 is not stabilizable

System 2 is stabilizable by a static state-feedback control law u = K\*xp

Stabilizing gain matrix K for System 2 with desired poles at [-3 -5] is:

[[4. 6.]]

# Screenshot:

