ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

# ОТЧЕТ

# О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ

# «ДИНАМИКА СИСТЕМЫ»

# ПО ДИСЦИПЛИНЕ «ТЕОРЕТИЧЕСКАЯ МЕХАНИКА И ОСНОВЫ КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ»

# ВАРИАНТ ЗАДАНИЯ № 28

Выполнила студентка группы М8О-208Б-21

Шевлякова С. С.______________________________

подпись, дата

Проверил и принял

Зав. каф. 802, Бардин Б.С.______________________________

подпись, дата

с оценкой ______________________________

Москва, 2022

*Задание:* проинтегрировать систему дифференциальных уравнений движения системы с двумя степенями свободы с помощью средств Python. Построить анимацию движения системы, а также графики законов движения системы и указанных в задании реакций для разных случаев системы. Исследовать на устойчивость.

## Схема программы

Для решения поставленных задач требуется сделать следующие шаги:

- Отдельно от основной программы с помощью уравнений движения системы требуется сформировать функцию, которая будет принимать в себя значения (q1, q2, q1`, q2`), а на выход вернёт значения (q1`, q2`, q1``, q2``)
- В основной программе требуется задать значения всех параметров, начальное положение системы, и запустить процедуру численного интегрирования системы.
- Результаты численного интегрирования системы далее следует использовать при построении анимации движения системы.

## Уравнение движения

$$(J + ms^2 \sin^2 \alpha)\ddot{\varphi} + 2ms\dot{s}\dot{\varphi}\sin^2 \alpha + c\varphi = 0,$$

$$m\ddot{s} - ms\dot{\varphi}^2 \sin^2 \alpha + mg\cos\alpha + k\dot{s} = 0.$$

## Код программы

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation
from scipy.integrate import odeint
import math

Steps = 300
t = np.linspace(0, 7, Steps)

PlateHeight = 4
g = 9.81
m = 1
J = 3
alpha = np.pi / 6
PlateWidth = PlateHeight / np.tan(alpha)
k = 10
c = 10

def odesys(y, t, g, m, J, alpha, k, c):
```

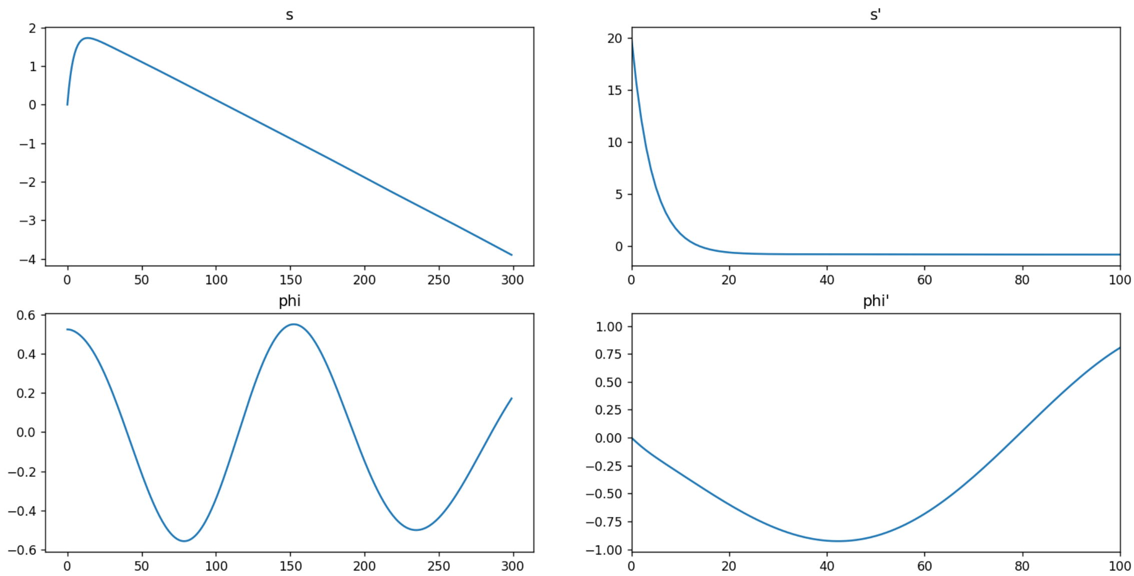
```
    dy = np.zeros(4)
    dy[0] = y[2]
    dy[1] = y[3]

    a11 = J + m * y[0] ** 2 * np.sin(alpha) ** 2
    a12 = 0
    a21 = 0
    a22 = m

    b1 = -c * y[1] - 2 * m * y[0] * y[2] * y[3] * np.sin(alpha) ** 2
    b2 = -k * y[2] - m * g * np.cos(alpha) + m * y[0] * y[3] ** 2 * np.sin(alpha) ** 2
    dy[2] = (b2 * a11 - b1 * a21) / (a11 * a22 - a12 * a21)
    dy[3] = (b1 * a22 - b2 * a12) / (a11 * a22 - a12 * a21)
    return dy

s0 = 0
phi0 = math.pi / 6
ds0 = 20
dphi0 = 0
y0 = [s0, phi0, ds0, dphi0]

Y = odeint(odesys, y0, t, (g, m, J, alpha, k, c))

d = Y[:, 0]
phi = Y[:, 1]

StandZ = 1

AX = PlateWidth / 2 * np.cos(phi)
AY = PlateWidth / 2 * np.sin(phi)
AZ = StandZ

BX = -PlateWidth / 2 * np.cos(phi)
BY = -PlateWidth / 2 * np.sin(phi)
BZ = StandZ

CX = BX
CY = BY
CZ = BZ + PlateHeight

DX = AX
DY = AY
DZ = AZ + PlateHeight

PathWidth = d * np.cos(alpha)

pointZ = StandZ + PlateHeight / 2 + d*np.sin(alpha)
pointX = PathWidth * np.cos(phi)
pointY = PathWidth * np.sin(phi)
```

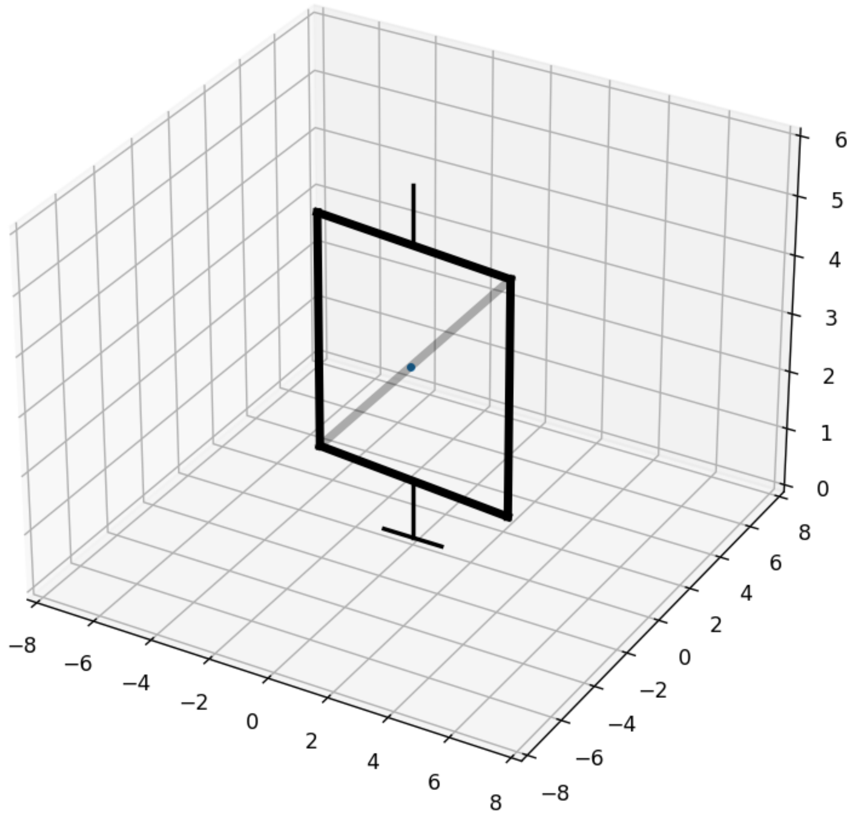
```
fig2 = plt.figure()
ax2 = fig2.add_subplot(2, 2, 1)
ax2.plot(Y[:, 0])
ax2.set_title("s")

ax4 = fig2.add_subplot(2, 2, 2)
ax4.plot(Y[:, 2])
ax4.set_xlim(left=0, right=100)
ax4.set_title("s'")

ax3 = fig2.add_subplot(2, 2, 3)
ax3.plot(Y[:, 1])
ax3.set_title("phi")

ax5 = fig2.add_subplot(2, 2, 4)
ax5.plot(Y[:, 3])
ax5.set_title("phi'")
ax5.set_xlim(left=0, right=100)

fig = plt.figure()
ax = fig.add_subplot(projection='3d')
ax.set(xlim=[-8, 8], ylim=[-8, 8], zlim=[0, 6])

pointPlot, = ax.plot(pointX[0], pointY[0], pointZ[0], marker='o', markersize='3')
lineABPLOT, = ax.plot([AX[0], BX[0]], [AY[0], BY[0]], [AZ, BZ], color='black', linewidth='4')
lineCDPLOT, = ax.plot([CX[0], DX[0]], [CY[0], DY[0]], [CZ, DZ], color='black', linewidth='4')
lineADPLOT, = ax.plot([AX[0], DX[0]], [AY[0], DY[0]], [AZ, DZ], color='black', linewidth='4')
lineBCPLOT, = ax.plot([BX[0], CX[0]], [BY[0], CY[0]], [BZ, CZ], color='black', linewidth='4')
lineBDPLOT, = ax.plot([BX[0], DX[0]], [BY[0], DY[0]], [BZ, DZ], color='black', linewidth='4', alpha=0.3)

axis = ax.plot([0, 0], [0, 0], [0, StandZ], color='black', linewidth='2')
axis1 = ax.plot([0, 0], [0, 0], [StandZ + PlateHeight, StandZ + PlateHeight + 1], color='black', linewidth='2')
axis2 = ax.plot([-1, 1], [0, 0], [0, 0], color='black', linewidth='2')

def Anima(i):
    pointPlot.set_data_3d(pointX[i], pointY[i], pointZ[i])
    lineABPLOT.set_data_3d([AX[i], BX[i]], [AY[i], BY[i]], [AZ, BZ])
    lineCDPLOT.set_data_3d([CX[i], DX[i]], [CY[i], DY[i]], [CZ, DZ])
    lineADPLOT.set_data_3d([AX[i], DX[i]], [AY[i], DY[i]], [AZ, DZ])
    lineBCPLOT.set_data_3d([BX[i], CX[i]], [BY[i], CY[i]], [BZ, CZ])
    lineBDPLOT.set_data_3d([BX[i], DX[i]], [BY[i], DY[i]], [BZ, DZ])
    return [pointPlot, lineABPLOT, lineCDPLOT, lineBCPLOT, lineADPLOT, lineBDPLOT]


anima = FuncAnimation(fig, Anima, frames=Steps, interval=24)
plt.show()
```

**Результат работы программы**

**1.** $m = 1$, $J = 3$, $alpha = pi/6$, $k = 10$, $c = 10$, $phi_0 = pi/6$, $s_0 = 0$, $ds_0 = 20$, $dphi_0 = 0$

В этом эксперименте я проверяю, как работает система при начальных условиях, взятых из условия задачи.

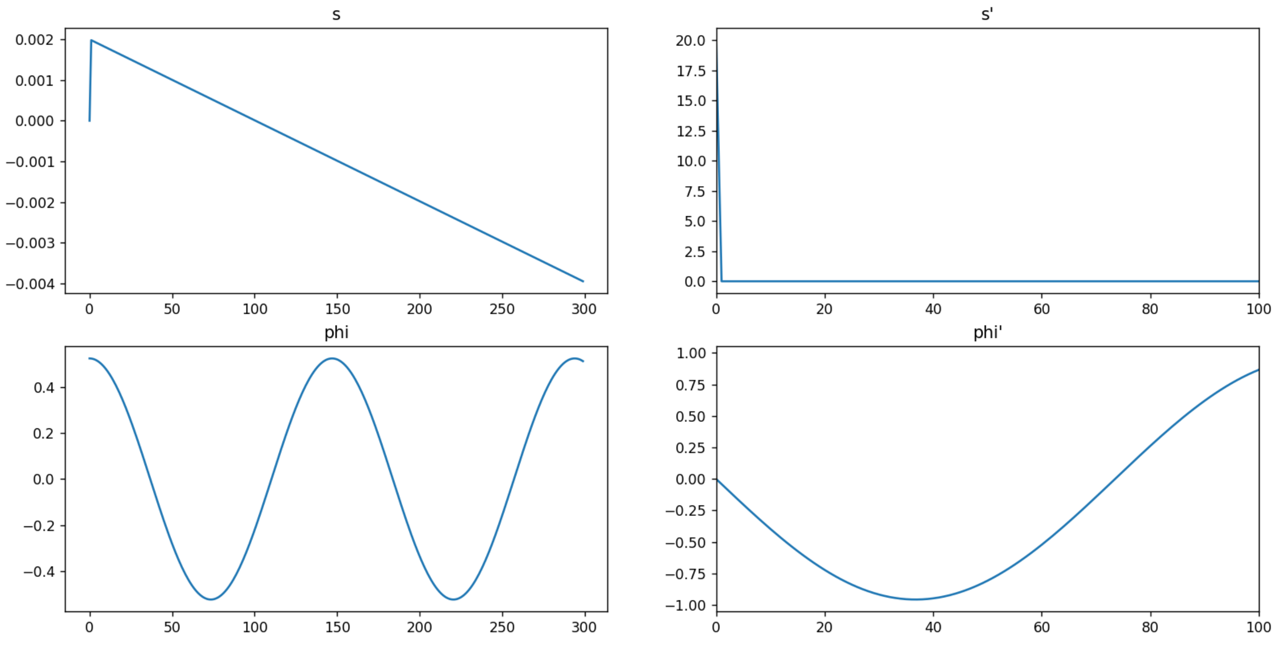


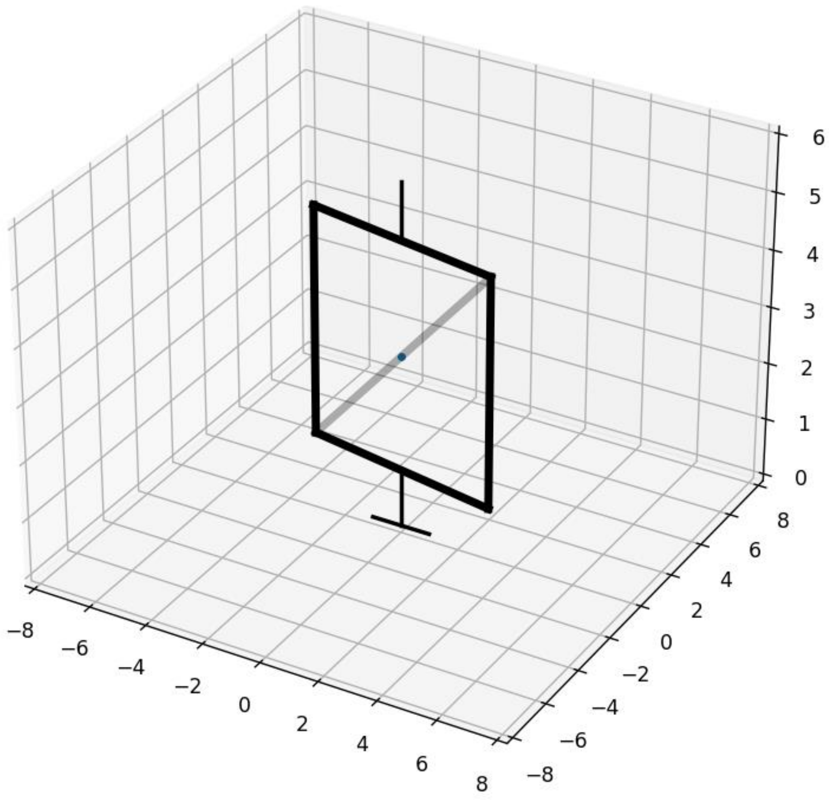


**Результат:** При этих начальных условиях, взятых из условия задачи, шарик медленно сползает вниз под действием силы тяжести, а пластина колеблется из-за приложенного к ней момента силы

**2.** m = 000.1, J = 3, alpha = pi/6, k = 10, c = 10, $phi_0$ = pi/6, $s_0$ = 0, $ds_0$ = 20, $dphi_0$ = 0

Этот эксперимент отличается от первого тем, что теперь шарик имеет очень маленькую массу.

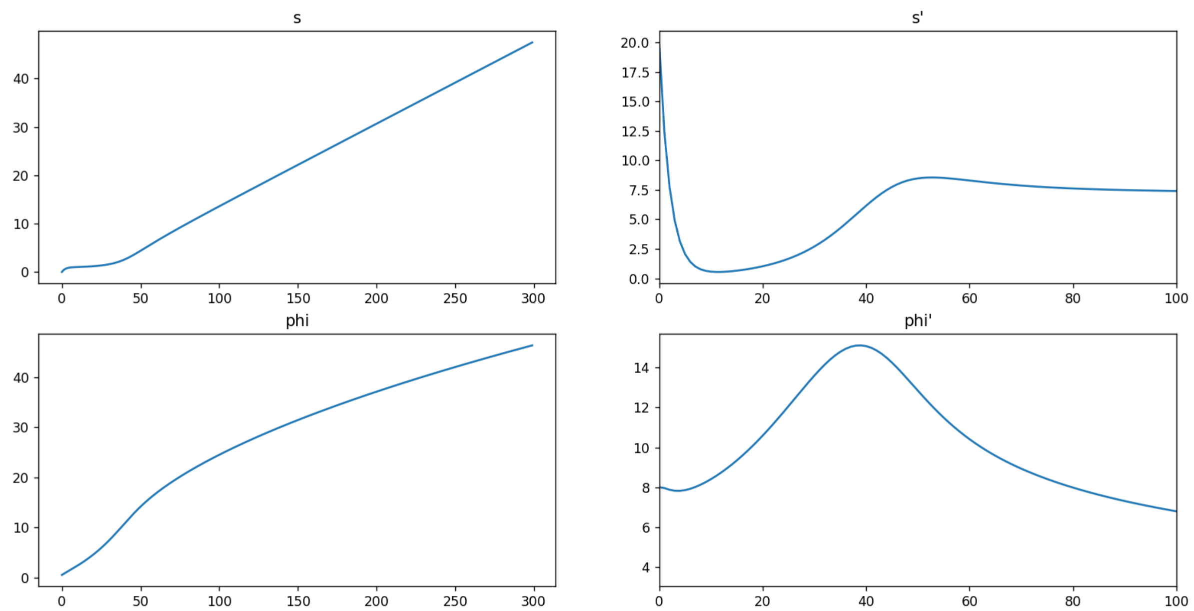


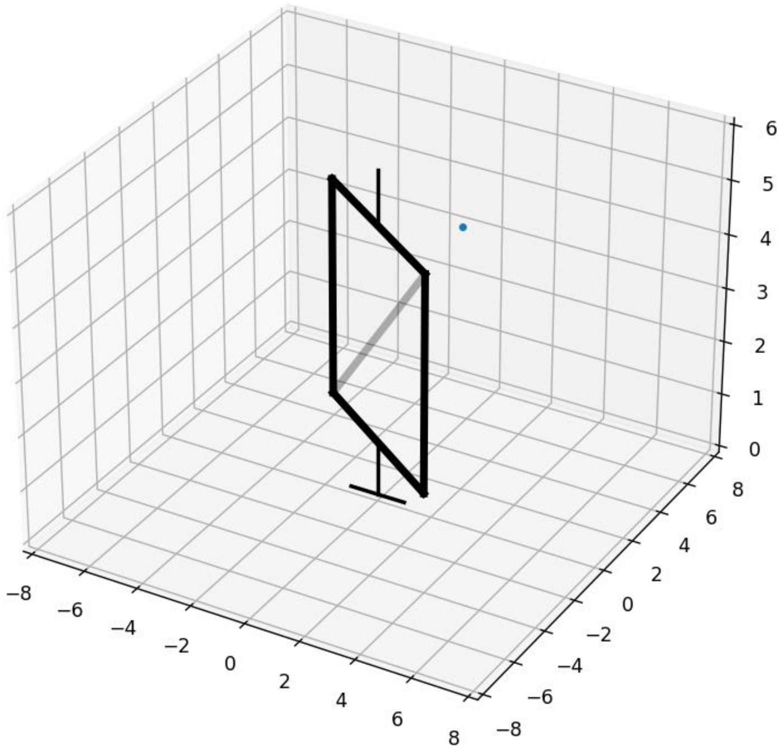



**Результат:** Однородная пластина совершает гармонические колебания, а шарик очень медленно сползает вниз, но из-за небольшой силы тяжести и наличия силы трения, это передвижение почти незаметно.

**3.** m = 1, J = 3, alpha = pi/6, k = 20, c = -10, $phi_0$ = pi/6, $s_0$ = 0, $ds_0$ = 20, $dphi_0$ = 6

В этом случае зададим пластине начальную скорость, и отрицательный коэффициент момента силы.

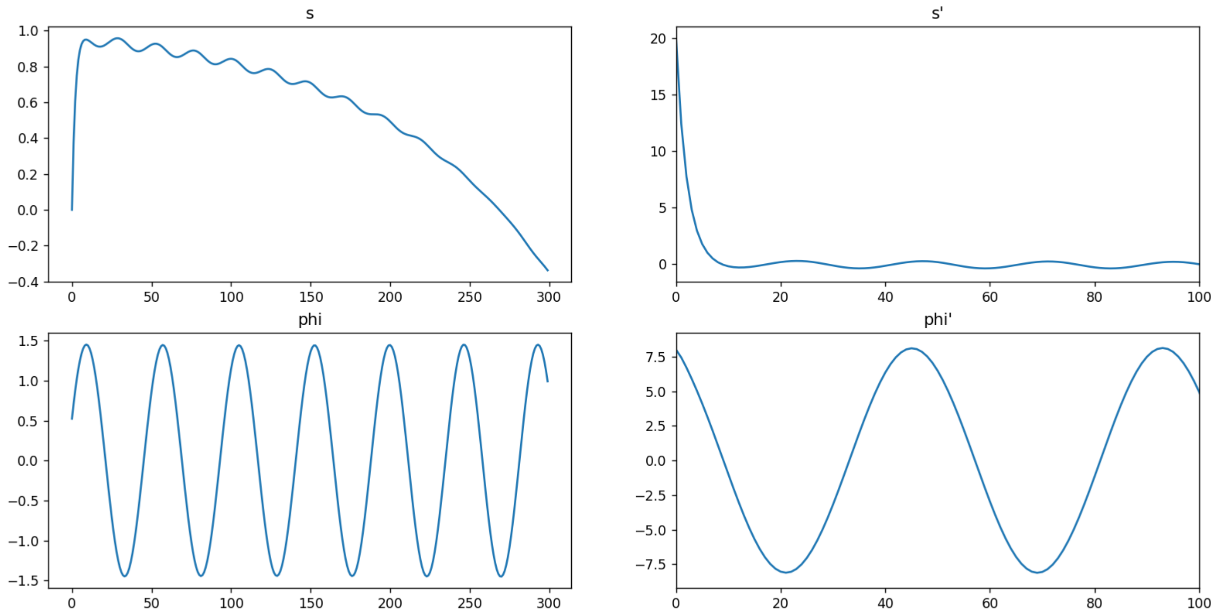


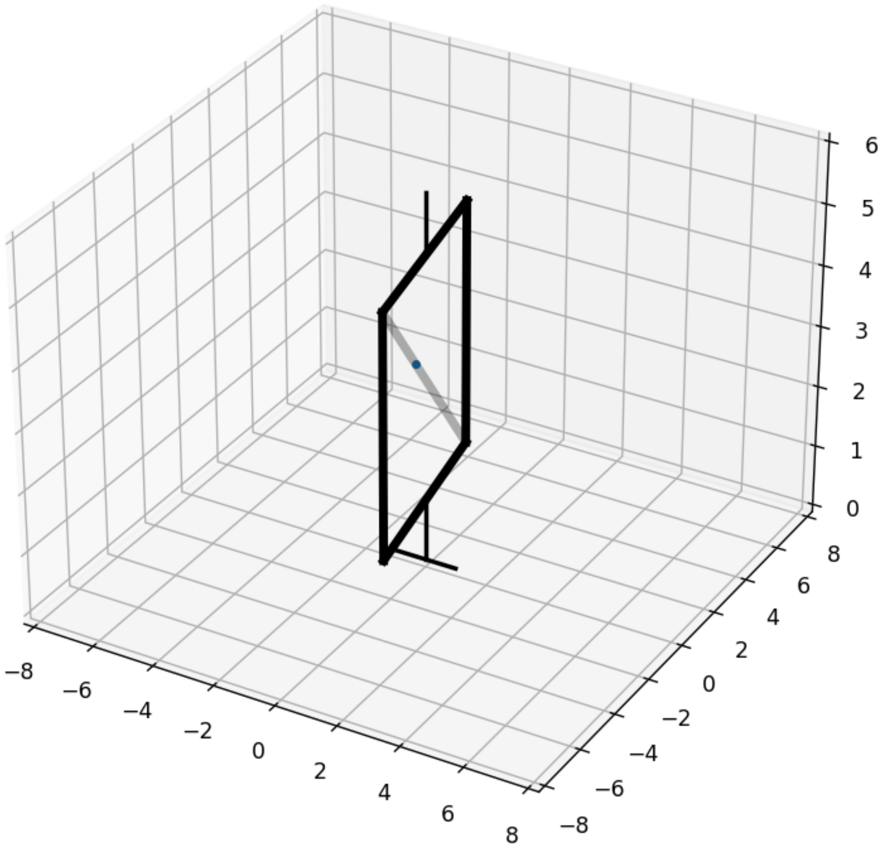



**Результат:** Момент силы, сообщаемой пластине будет положительным, теперь пластина не колеблется, а вращается против часовой стрелки. Шарик вылетает из канала вверх из-за воздействия силы инерции.

**4.** m = 1, J = 3, alpha = pi/6, k = 20, c = 100, $phi_0$ = pi/6, $s_0$ = 0, $ds_0$ = 20, $dphi_0$ = 8

Зададим однородной пластине начальную скорость и большой коэффициент момента силы.




**Результат:** Некоторое время шарик будет колебаться у неустойчивого положения равновесия, но по итогу все равно скатится вниз.