ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

**ОТЧЕТ**

**О ВЫПЛОНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

**«ДИНАМИКА СИСТЕМЫ»**

**ПО ДИСЦИПЛИНЕ «ТЕОРЕТИЧЕСКАЯ МЕХАНИКА И ОСНОВЫ КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ»**

**ВАРИАНТ ЗАДАНИЯ № 22**

Выполнил(а) студент группы М8О-212Б-22

Куценко М.Д.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

Проверил и принял

Зав. каф. 802, Бардин Б.С.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

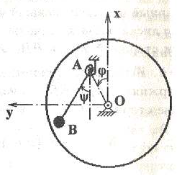
подпись, дата

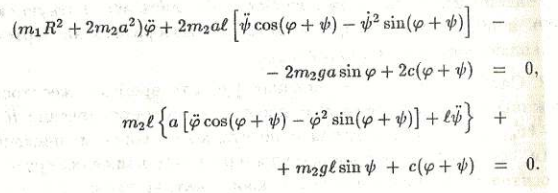
с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва, 2023

*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.

*Вариант 22:*

Однородный диск массы **m1** и радиуса **R** вращается вокруг горизонтальной неподвижной оси, перпендикулярной его плоскости и проходящей через центр O. В точке A (OA=**a**) к диску шарнирно прикреплён невесомый стержень AB длины **l** с грузом B массы **m2** на конце. Спиральная пружина жесткости с прикреплена своими концами к диску и стрежню. Пружина не деформирована при **φ**=**ψ**=0. Трение на оси O отсутствует.

 Дифференциальные уравнения движения системы имеют вид:

*Код:*

import numpy as np

import math

import matplotlib.pyplot as plt

from matplotlib.animation import FuncAnimation

import sympy as sp

from scipy.integrate import odeint

def odesys(y, t, r, a, l, m1, m2, c, g): # Функция создания системы диффуров

dy = np.zeros(4)

dy[0] = y[2]

dy[1] = y[3]

a11 = m1\*r\*\*2+2\*m2\*a\*\*2

a12 = 2\*m2\*a\*l\*np.cos(y[0]+y[1])

a21 = m2\*l\*a\*np.cos(y[0]+y[1])

a22 = m2\*l\*\*2

b1 = 2\*m2\*a\*l\*np.sin(y[0]+y[1])\*y[3]\*\*2+2\*m2\*g\*a\*np.sin(y[0])-2\*c\*(y[0]+y[1])

b2 = m2\*l\*a\*np.sin(y[0]+y[1])\*y[2]\*\*2-m2\*g\*l\*np.sin(y[1])-c\*(y[0]+y[1])

dy[2] = (b1\*a22-b2\*a12)/(a11\*a22-a12\*a21)

dy[3] = (b2\*a11-b1\*a21)/(a11\*a22-a12\*a21)

return dy

### ИЗМЕНЯЕМЫЕ ПАРАМЕТРЫ СИСТЕМЫ

R = 1 # радиус диска

A = 0.5 # расстояние между шарниром и центром диска

L = 1 # длина стержня, на котором шарнирно прикреплён груз

M1 = 1 # масса диска

M2 = 1 # масса груза

C = 1 # жёсткость спиральной пружины

G = 9.81 # ускорение свободного падения

### НАЧАЛЬНЫЕ ЗНАЧЕНИЯ

PHI0 = math.pi/6

PSI0 = 0

DPHI0 = 0

DPSI0 = 0

Y0 = [PHI0, PSI0, DPHI0, DPSI0]

### СТАТИЧЕСКАЯ ЧАСТЬ

X\_C = R+A+L # координаты центра диска

Y\_C = R+A+L

RM = R/20 # радиус маленького круга в центре диска

ang = np.linspace(0, 2\*math.pi, 80) # углы для отрисовки кругов

X\_Disk = X\_C + R\*np.cos(ang) # координаты диска

Y\_Disk = Y\_C + R\*np.sin(ang)

X\_Sm = X\_C + RM\*np.cos(ang) # координаты маленького круга в центре диска

Y\_Sm = Y\_C + RM\*np.sin(ang)

X\_Side\_1 = [X\_C+RM\*np.cos(math.pi\*5/4), X\_C+RM\*3\*np.cos(math.pi\*5/4)] # боковые линии (центр)

Y\_Side\_1 = [X\_C+RM\*np.sin(math.pi\*5/4), Y\_C+RM\*3\*np.sin(math.pi\*5/4)]

X\_Side\_2 = [X\_C+RM\*np.cos(math.pi/-4), X\_C+RM\*3\*np.cos(math.pi/-4)]

Y\_Side\_2 = [X\_C+RM\*np.sin(math.pi/-4), Y\_C+RM\*3\*np.sin(math.pi/-4)]

X\_Bottom = [X\_Side\_1[1]-R/40, X\_Side\_2[1]+R/40] # линия-закреп центра

Y\_Bottom = [Y\_Side\_1[1], Y\_Side\_2[1]]

X\_Lines\_1 = np.linspace(float(X\_Bottom[0])+R/50, float(X\_Bottom[1])-R/50, 5) # полоски на линии-закрепа центра диска

X\_Lines\_2 = X\_Lines\_1 + R/20\*np.cos(math.pi\*9/8)

Y\_Lines\_1 = np.full(5, Y\_Bottom[0])

Y\_Lines\_2 = Y\_Lines\_1 + R/20\*np.sin(math.pi\*9/8)

### ДИНАМИЧЕСКАЯ ЧАСТЬ

Steps = 1000

t\_fin = 20

t = np.linspace(0, t\_fin, Steps) # время

X\_Sh = np.zeros\_like(t) # координаты шарнира

Y\_Sh = np.zeros\_like(t)

X\_Gr = np.zeros\_like(t) # координаты груза

Y\_Gr = np.zeros\_like(t)

Sol = odeint(odesys, Y0, t, (R, A, L, M1, M2, C, G)) # решение диффура

phi = Sol[:,0] # угол между вертикальной осью и радиус-вектором к шарниру

psi = Sol[:,1] # угол между вертикальной осью и стержнем

dphi = Sol[:,2] # угловые скорости

dpsi = Sol[:,3]

ddphi = [odesys(y, t, R, A, L, M1, M2, C, G)[2] for y,t in zip(Sol,t)] # угловые ускорения

ddpsi = [odesys(y, t, R, A, L, M1, M2, C, G)[3] for y,t in zip(Sol,t)]

for i in np.arange(len(t)): # просчёт основных величин

X\_Sh[i] = X\_C + A\*np.cos(phi[i]+math.pi/2)

Y\_Sh[i] = Y\_C + A\*np.sin(phi[i]+math.pi/2)

X\_Gr[i] = X\_Sh[i] + L\*np.cos(-psi[i]-math.pi/2)

Y\_Gr[i] = Y\_Sh[i] + L\*np.sin(-psi[i]-math.pi/2)

### ПЕРЕХОД К ОТРИСОВКЕ

fig = plt.figure() # задаём пространство для отрисовки

ax = fig.add\_subplot(1, 1, 1)

ax.axis('equal')

ax.set(xlim = [0, X\_C\*2], ylim = [0, Y\_C\*2])

ax.set(xlabel="x", ylabel="y")

### СТАТИЧЕСКАЯ ОТРИСОВКА

ax.plot(X\_C, Y\_C, marker = 'o', markersize=2, color = 'blue') # отрисовка центра диска

ax.plot(X\_Disk, Y\_Disk, color = 'blue') # отрисовка диска

ax.plot(X\_Sm, Y\_Sm, color = 'blue') # отрисовка кружка вокруг центра диска

ax.plot(X\_Side\_1, Y\_Side\_1, color = 'blue') # отрисовка боковых линий от центра диска

ax.plot(X\_Side\_2, Y\_Side\_2, color = 'blue')

ax.plot(X\_Bottom, Y\_Bottom, color = 'blue') # отрисовка линии-закрепа центра диска

for i in np.arange(len(X\_Lines\_1)): # отрисовка штрихов на линии-закрепе центра диска

ax.plot([X\_Lines\_1[i], X\_Lines\_2[i]], [Y\_Lines\_1[i], Y\_Lines\_2[i]], color = 'darkblue')

### ДИНАМИЧЕСКАЯ ОТРИСОВКА

LEN = R/6 # длина линии-закрепа пружинки

WIDE = R/8 # ширина линии-закрепа пружинки

X\_DSHT = R/20\*np.cos(math.pi/4) # сдвиги штрихов по координатам

Y\_DSHT = R/20\*np.cos(math.pi/4)

R1 = R/8 # радиусы спиральной пружины

R2 = R/64

thetta = np.linspace(0, 3/2\*math.pi+psi[0], 100) # угол проворота спиральной пружины

X\_SpiralSpr = (R1 + thetta\*(R2-R1)/thetta[-1])\*np.cos(thetta) # координаты точек спиральной пружины

Y\_SpiralSpr = -(R1 + thetta\*(R2-R1)/thetta[-1])\*np.sin(thetta)

spr, = ax.plot(X\_SpiralSpr+X\_Sh[0], Y\_SpiralSpr+Y\_Sh[0], color = 'green') # отрисовка спиральной пружины

pl1, = ax.plot([X\_Sh[0]+R1-WIDE/2, X\_Sh[0]+R1-WIDE/2+X\_DSHT], [Y\_Sh[0]+LEN, Y\_Sh[0]+LEN+Y\_DSHT], color = 'darkgreen') # штрихи на линии-закрепе спиральки

pl2, = ax.plot([X\_Sh[0]+R1, X\_Sh[0]+R1+X\_DSHT], [Y\_Sh[0]+LEN, Y\_Sh[0]+LEN+Y\_DSHT], color = 'darkgreen')

pl3, = ax.plot([X\_Sh[0]+R1+WIDE/2, X\_Sh[0]+R1+WIDE/2+X\_DSHT], [Y\_Sh[0]+LEN, Y\_Sh[0]+LEN+Y\_DSHT], color = 'darkgreen')

hl, = ax.plot([X\_Sh[0]+R1-WIDE/2-R/10, X\_Sh[0]+R1+WIDE/2+R/10], [Y\_Sh[0]+LEN, Y\_Sh[0]+LEN], color = 'green') # отрисовка вертикальной линии от спиральки

upl, = ax.plot([X\_Sh[0]+R1, X\_Sh[0]+R1], [Y\_Sh[0], Y\_Sh[0]+LEN], color = 'green') # отрисовка линии-закрепа спирали

sh, = ax.plot(X\_Sh[0], Y\_Sh[0], marker='o', markersize = 5, color = 'orange') # отрисовка шарнира

st, = ax.plot([X\_Sh[0], X\_Gr[0]], [Y\_Sh[0], Y\_Gr[0]], color = 'orange') # отрисовка стержня

gr, = ax.plot(X\_Gr[0], Y\_Gr[0], marker = 'o', markersize = 20, color = 'orange') # отрисовка грузика

def anima(i): # функция анимации

thetta = np.linspace(0, 3/2\*math.pi+psi[i], 100)

X\_SpiralSpr = (R1 + thetta\*(R2-R1)/thetta[-1])\*np.cos(thetta)

Y\_SpiralSpr = -(R1 + thetta\*(R2-R1)/thetta[-1])\*np.sin(thetta)

spr.set\_data(X\_SpiralSpr+X\_Sh[i], Y\_SpiralSpr+Y\_Sh[i])

pl1.set\_data([X\_Sh[i]+R1-WIDE/2, X\_Sh[i]+R1-WIDE/2+X\_DSHT], [Y\_Sh[i]+LEN, Y\_Sh[i]+LEN+Y\_DSHT])

pl2.set\_data([X\_Sh[i]+R1, X\_Sh[i]+R1+X\_DSHT], [Y\_Sh[i]+LEN, Y\_Sh[i]+LEN+Y\_DSHT])

pl3.set\_data([X\_Sh[i]+R1+WIDE/2, X\_Sh[i]+R1+WIDE/2+X\_DSHT], [Y\_Sh[i]+LEN, Y\_Sh[i]+LEN+Y\_DSHT])

hl.set\_data([X\_Sh[i]+R1-WIDE/2, X\_Sh[i]+R1+WIDE/2], [Y\_Sh[i]+LEN, Y\_Sh[i]+LEN])

upl.set\_data([X\_Sh[i]+R1, X\_Sh[i]+R1], [Y\_Sh[i], Y\_Sh[i]+LEN])

sh.set\_data(X\_Sh[i], Y\_Sh[i])

st.set\_data([X\_Sh[i], X\_Gr[i]], [Y\_Sh[i], Y\_Gr[i]])

gr.set\_data(X\_Gr[i], Y\_Gr[i])

return spr, pl1, pl2, pl3, hl, upl, sh, st, gr

anim = FuncAnimation(fig, anima, frames=Steps, interval=40, repeat=False) # создаём разовую анимацию

fig.suptitle('Kutsenko LW3', fontsize=14) # добавляем название

anim.save("./Animation.mp4", writer="ffmpeg") # сохраняем анимацию

### ГРАФИКИ ЗАВИСИМОСТЕЙ ВЕЛИЧИН ОТ ВРЕМЕНИ

Nox = (M1+M2)\*G-M2\*(A\*(ddphi\*np.sin(phi)+dphi\*\*2\*np.cos(phi))-L\*(ddpsi\*np.sin(psi)+dpsi\*\*2\*np.cos(psi))) # проекция реакции оси диска

k1 = 1 # коэффициенты для нахождения силы сопротивления R

k2 = 10

pls = plt.figure()

p1 = pls.add\_subplot(3, 2, 1) # строим графики величин

p1.set(xlim = [0,t\_fin])

p1.plot(t, phi)

p1.grid()

plt.title('Phi(t)')

p2 = pls.add\_subplot(3, 2, 3)

p2.plot(t, psi)

p2.grid()

p2.set(xlim = [0,t\_fin])

plt.title('Psi(t)')

p3 = pls.add\_subplot(3, 2, 2)

p3.plot(t, Nox)

p3.grid()

p3.set(xlim = [0,t\_fin])

plt.title('Nox(t)')

p4 = pls.add\_subplot(3, 2, 4)

p4.plot(t, -k1\*dpsi)

p4.grid()

p4.set(xlim = [0,t\_fin])

plt.title('R1(t)')

p5 = pls.add\_subplot(3, 2, 6)

p5.plot(t, -k2\*dpsi)

p5.grid()

p5.set(xlim = [0,t\_fin])

plt.title('R2(t)')

plt.tight\_layout() # чтобы не накладывались названия

plt.savefig('Plots.png') # сохраняем графики

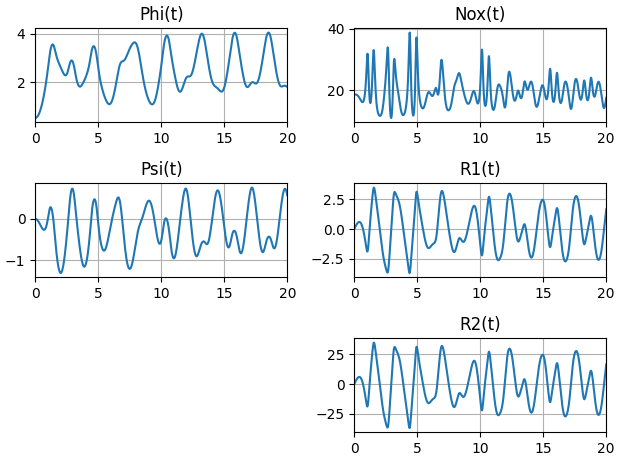
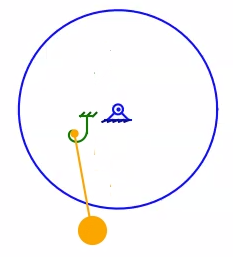
*Процесс выполнения работы:*

Добавил функцию образования системы дифференциальных уравнений odesys, которая потом решается через функцию odeint. Получившиеся решения используются для построения анимации с использованием кода из предыдущей лабораторной работы.

Также с учётом изменения величин для отрисовки пришлось поменять код программы, чтобы система масштабировалась в зависимости от R, a, l.

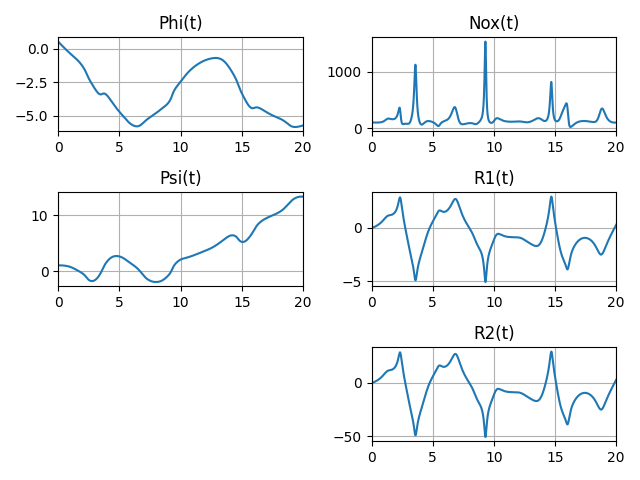
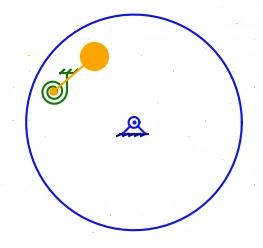
Добавил графики зависимостей различных величин системы в зависимости от времени.

*Результаты работы программы:*

**1.** Отрисовка по данным из учебника (R = 1, a = 0.5, l = 1, m1 = 1, m2 = 2, c = 1, g = 9.81, φ0 = π/6, ψ0 = 0, dφ0 = 0, dψ0 = 0):

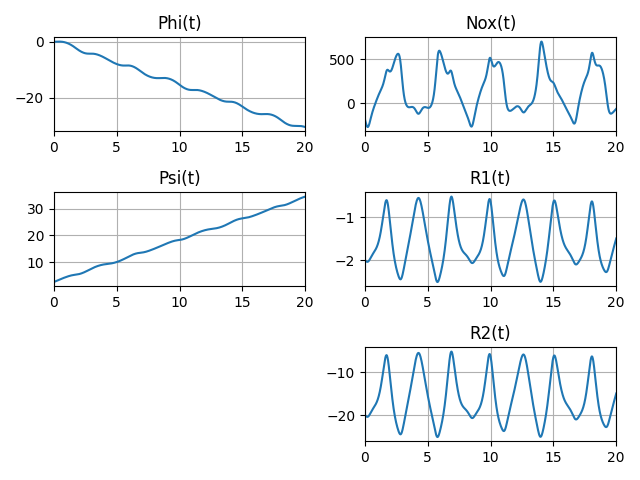
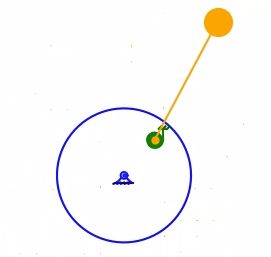
Диск закрутится влево, и груз начнёт двигаться относительно положения равновесия.

**2.**  Отрисовка по данным (R = 10, a = 8, l = 5, m1 = 10, m2 = 8, c = 20, g = 9.81, φ0 = π/6, ψ0 = π/3, dφ0 = -1, dψ0 = 0):



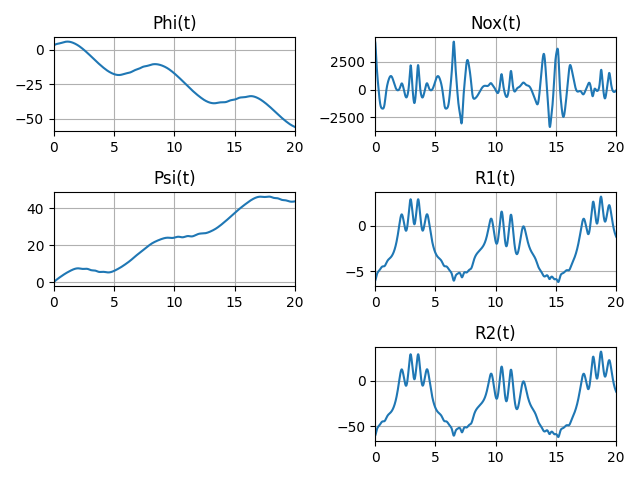
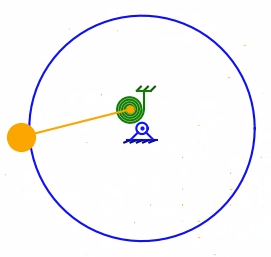
Диск под действием начальных сил закрутится вправо и начнётся движение вокруг положения равновесия с закручиванием пружины по часовой стрелке.

**3.** Отрисовка по данным (R = 10, a = 7, l = 20, m1 = 10, m2 = 5, c = 30, g = 9.81, φ0 = 0, ψ0 = 5/6\*π, dφ0 = 0, dψ0 = 2):



Под действием начальной силы груз закрутится вправо и диск начнёт крутиться вправо, а спираль — закручиваться по часовой стрелке.

**4.** Отрисовка по данным (R = 10, a = 2, l = 10, m1 = 10, m2 = 10, c = 200, g = 9.81, φ0 = π, ψ0 = 0, dφ0 = 5, dψ0 = 6):

****

Из-за того, что приложенные силы в начальный момент времени достаточно значительны, диск начнёт крутиться с большой скоростью, в основном по часовой стрелке. Спираль также будет закручиваться.

*Вывод:* успешно выполнил лабораторную работу по теоретической механике. С помощью языка программирования Python и библиотек matplotlib и numpy я схематично проанимировал движение стержня с грузом, шарнирно закреплённого на диске и решил систему дифференциальных уравнений.

В моей программе используются реальные законы движения, благодаря чему можно посмотреть, как эта система будет вести себя в реальной жизни (без учёта трения).