Московский авиационный институт

(национальный исследовательский университет)

Институт № 8 «Информационные технологии и прикладная математика»

**Лабораторная работа №4**

**по курсу «Теоретическая механика»**

**Малые колебания**

Выполнил студент группы М8О-206Б-20

Семин Александр Витальевич

Преподаватель: Сухов Е. А.

Дата: 23.12.21

Оценка:

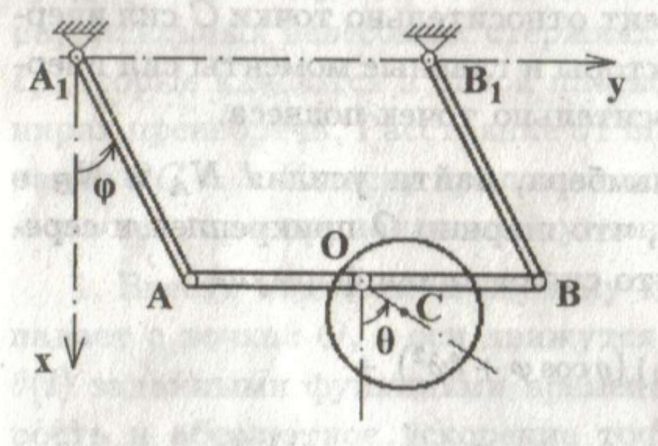
Москва, 2021

**Вариант № «21»**

**Задание:**

Реализовать анимацию движения механической системы в среде Python на основе уравнений Лагранжа 2-го рода для малых колебаний, точка D закреплена в точке B.

**Механическая система:**

****

**Текст программы:**

import numpy as np

import matplotlib.pyplot as plt

from matplotlib.animation import FuncAnimation

from scipy.integrate import odeint

import sympy as sp

import math

def Circle(X, Y):

CX = [X + 0.75 \* math.cos(i/100) \* R for i in range(0, 628)]

CY = [Y + 0.75 \* math.sin(i/100) \* R for i in range(0, 628)]

return CX, CY

def anima(i):

Beam\_1.set\_data([-0.5, -0.5], [-0.5, 0])

Beam\_2.set\_data([0.5, 0.5], [-0.5, 0])

Beam\_3.set\_data([-0.5, 0.5], [-0.5, -0.5])

Beam\_4.set\_data([0, 0 + X\_o[i]], [-0.5, -0.5 + Y\_o[i]])

circle.set\_data(\*Circle(0 + X\_o[i], -0.5 + Y\_o[i]))

return Beam\_1, Beam\_2, Beam\_3, Beam\_4, circle,

def formY(y, t, fV, fOm):

y1,y2,y3,y4 = y

dydt = [y3,y4,fV(y1,y2,y3,y4),fOm(y1,y2,y3,y4)]

return dydt

def formY2(y, t, fOm):

y1,y2 = y

dydt = [y2,fOm(y1,y2)]

return dydt

m\_2 = 5

m\_1 = 5

g = 10

l = 1

b = 0.125

k = 0.5

R = 0.25

t = sp.Symbol('t')

phi = 0

tetta = sp.Function('tetta')(t)

om = 0

om\_2 = sp.Function('om\_2')(t)

E\_ab = (m\_2 \* om\*\*2 \* l\*\*2) / 2

Vx\_c = om \* l \* sp.sin(phi)

Vy\_c = om \* l \* sp.cos(phi)

Vx\_o = om\_2 \* sp.sin(tetta) \* b

Vy\_o = om\_2 \* sp.cos(tetta) \* b

J\_cir = (m\_1 \* R\*\*2) / 2

E\_cir = m\_1 \* ((Vx\_o + Vx\_c)\*\*2 + (Vy\_o + Vy\_c)\*\*2) / 2 + (J\_cir \* om\_2\*\*2)/2

Ekin = E\_ab + E\_cir

Pi1 = m\_2 \* g \* (l \* (1 - sp.cos(phi)) + b)

Pi2 = m\_1 \* g \* (l \* (1 - sp.cos(phi)) + b \* (1 - sp.cos(tetta)))

Epot = Pi1 + Pi2

M = - k \* om\_2

L = Ekin - Epot

ur1 = sp.diff(sp.diff(L, om\_2), t) - sp.diff(L, tetta) - M

print(ur1)

# ur2 = sp.diff(sp.diff(L, om), t) - sp.diff(L, phi)

#вычисление вторых производных (DV/dt и dom/dt) с использованием метода Крамера

a11 = ur1.coeff(sp.diff(om\_2, t), 1)

# a12 = ur1.coeff(sp.diff(om, t), 1)

# a21 = ur2.coeff(sp.diff(om\_2, t), 1)

# a22 = ur2.coeff(sp.diff(om, t), 1)

# b1 = -(ur1.coeff(sp.diff(om\_2, t), 0)).coeff(sp.diff(om, t), 0).subs([(sp.diff(tetta, t), om\_2), (sp.diff(phi, t), om)])

b1 = -ur1.coeff(sp.diff(om\_2, t),0).subs(sp.diff(tetta, t), om\_2);

# b2 = -(ur2.coeff(sp.diff(om\_2, t), 0)).coeff(sp.diff(om, t), 0).subs([(sp.diff(tetta, t), om\_2), (sp.diff(phi, t), om)])

# detA = a11\*a22-a12\*a21

# detA1 = b1\*a22-b2\*a21

# detA2 = a11\*b2-b1\*a21

# dom\_tettadt = detA2/detA

# dom\_phidt = detA1/detA

print(b1, a11)

domdt = b1/a11

print(domdt)

countOfFrames = 100

# Построение системы д/у

T = np.linspace(0, 10, countOfFrames)

# f\_om\_tetta = sp.lambdify([phi, tetta, om\_2, om], dom\_tettadt, "numpy")

f\_om\_phi = sp.lambdify([tetta, om\_2], domdt, "numpy")

y0 = [math.pi/2, 0]

sol = odeint(formY2, y0, T, args = (f\_om\_phi, ))

# X\_ab\_func = sp.lambdify(phi, sp.sin(phi) \* l)

# Y\_ab\_func = sp.lambdify(phi, - sp.cos(phi) \* l)

X\_o\_func = sp.lambdify(tetta, - sp.sin(tetta) \* b)

Y\_o\_func = sp.lambdify(tetta, - sp.cos(tetta) \* b)

X\_o = X\_o\_func(sol[:, 1])

Y\_o = Y\_o\_func(sol[:, 1])

fig = plt.figure(figsize = (17, 8))

ax1 = fig.add\_subplot(1, 2, 1)

ax1.axis('equal')

ax1.set(xlim = [-2, 2], ylim = [-2, 2])

ax2 = fig.add\_subplot(4, 2, 2)

ax2.plot(T, sol[:, 1])

ax2.set\_xlabel('T')

ax2.set\_ylabel('Om\_tetta')

ax3 = fig.add\_subplot(4, 2, 4)

ax3.plot(T, sol[:, 0])

ax3.set\_xlabel('T')

ax3.set\_ylabel('tetta')

# ax3 = fig.add\_subplot(4, 2, 4)

# ax3.plot(T, sol[:, 2])

# ax3.set\_xlabel('T')

# ax3.set\_ylabel('Om\_tetta')

Beam\_1, = ax1.plot([-0.5, -0.5], [-0.5, 0], 'black')

Beam\_2, = ax1.plot([0.5, 0], [0.5, -0.5], 'black')

Beam\_3, = ax1.plot([-0.5, 0.5], [-0.5, -0.5], 'black')

Beam\_4, = ax1.plot([0, 0 + X\_o[0]], [-0.5, -0.5 + Y\_o[0]], 'black')

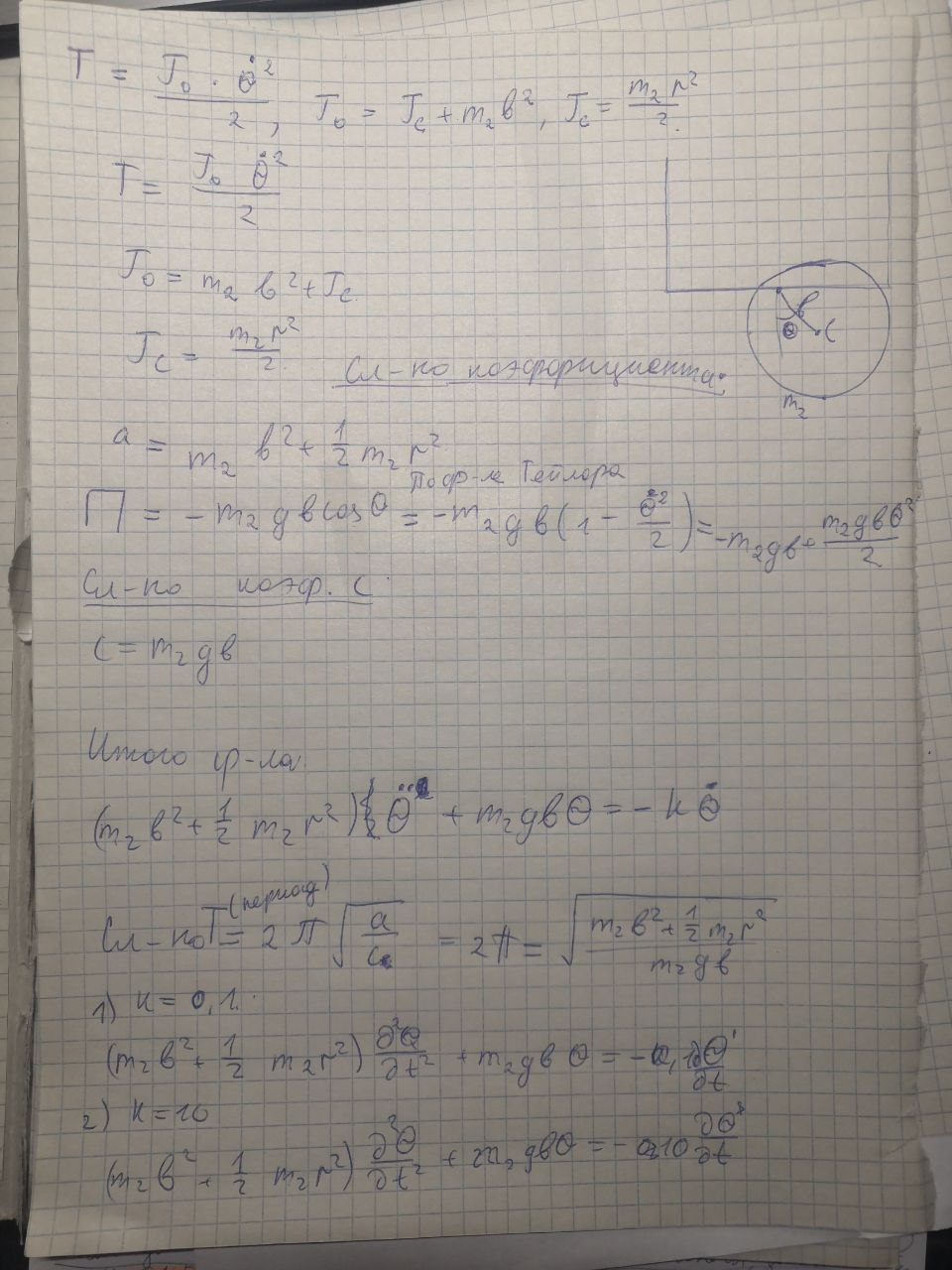
circle, = ax1.plot(\*Circle(0 + X\_o[0], -0.5 + Y\_o[0]), 'black')

anim = FuncAnimation(fig, anima, frames=countOfFrames, interval=100, blit=True)

plt.show()

**Результат работы:**

|  |
| --- |
|  |
|  |



**Вывод:**

В процессе выполнения работы я познакомился с явлением малых колебаний. Они представляют собой распространенный тип движения механических систем, имеющих положение устойчивого равновесия, при малом отклонении от которого возникают силы, стремящиеся вернуть систему в исходное положение. Так что все движение происходит в малой окрестности этого устойчивого положения равновесия.