ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

**ОТЧЕТ**

**О ВЫПЛОНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

**«АНИМАЦИЯ СИСТЕМЫ»**

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

**ВАРИАНТ ЗАДАНИЯ № 16**

Выполнил(а) студент группы М8О-203Б-22

Вертоградский Д.А.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

Проверил и принял

Авдюшкин А.Н.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

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

Москва, 2023

*Задание:* построить анимацию движения системы с помощью Python.

Вариант 16

Невесомый стержень OA длины l закреплен шарнирно в точке O. Спиральная пружина жесткости c соединаяет стержень с неподвижной осью O. При вертикальном положениии стержня пружина деформирована. К концу A стержня шарнирно прикреплен невесомый стержень AB длины l с точечным грузом массы m на конце B.

*Реализация:*

import numpy as np

import matplotlib.pyplot as plt

from matplotlib.animation import FuncAnimation

steps = 1000

t = np.linspace(0, 10, steps)

x = np.sin(t)

phi = 5 + np.sin(2\*t)

theta = np.sin(4\*t)

figr = plt.figure()

gr = figr.add\_subplot(1,1,1)

gr.set\_xlim([-3.5, 2.5])

gr.set\_ylim([-2.5, 3.5])

gr.plot([-0.5, 0.5, 0, -0.5], [0.5, 0.5, 1, 0.5], linewidth=1)

lenOA = 2.25

lenAB = 2

xO = 0

yO = 1

xA = xO + lenOA \* np.sin(phi)

yA = yO + lenOA \* np.cos(phi)

xB = xA - lenAB \* np.sin(theta)

yB = yA - lenAB \* np.cos(theta)

pO = gr.plot(xO, yO)[0]

pA = gr.plot(xA[0], yA[0], marker='.')[0]

pB = gr.plot(xB[0], yB[0], marker='.')[0]

lineAB = gr.plot([xA[0], xB[0]], [yA[0], yB[0]], color='black')[0]

lineOA = gr.plot([xO, xA[0]], [yO, yA[0]], color='black')[0]

Ns = 3

r1 = 0.1

r2 = 0.4

numpnts = np.linspace(0, 1, 50\*Ns+1)

Betas = numpnts \* (Ns \* 2 \* np.pi - phi[0]+1.5)

Xs = (r1 + (r2 - r1) \* numpnts \* np.cos(Betas)) - 0.1

Ys = (r1 + (r2 - r1) \* numpnts \* np.sin(Betas)) - 0.1

SpPruzh = gr.plot(Xs + xO, Ys + yO)[0]

def run(i):

pA.set\_data(xA[i], yA[i])

pB.set\_data(xB[i], yB[i])

lineAB.set\_data([xA[i], xB[i]], [yA[i], yB[i]])

lineOA.set\_data([xO, xA[i]], [yO, yA[i]])

Betas = numpnts \* (Ns \* 2 \* np.pi - phi[i]+1.5)

Xs = (r1 + (r2 - r1) \* numpnts \* np.cos(Betas)) - 0.1

Ys = (r1 + (r2 - r1) \* numpnts \* np.sin(Betas)) - 0.1

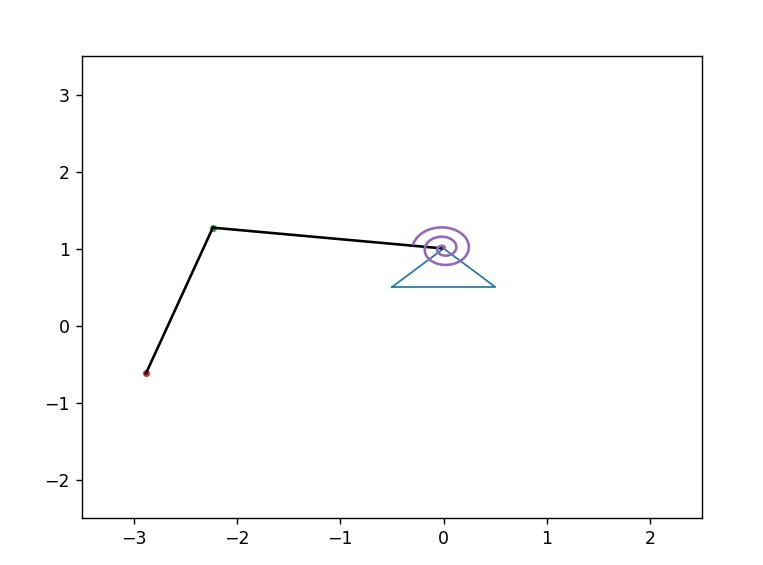
SpPruzh.set\_data(Xs + xO, Ys + yO)

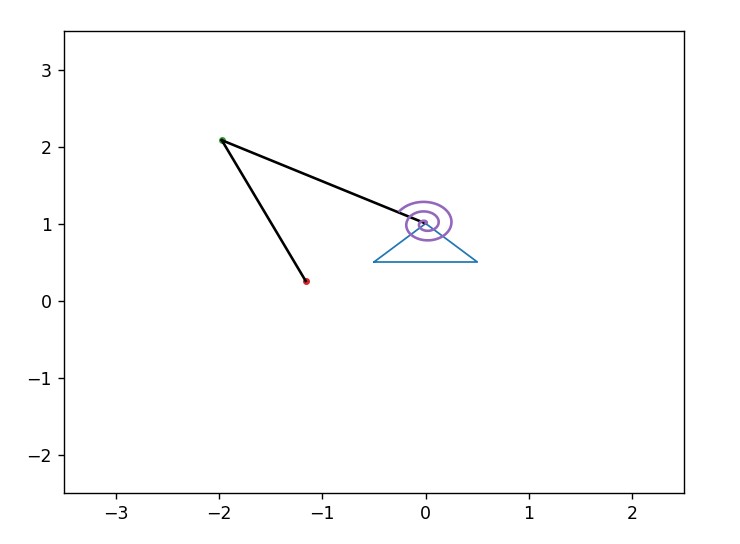
return

anim = FuncAnimation(figr, run, frames=steps, interval=0.5)

plt.show()

*Результаты:*





*Выводы:* Построение анимации движения системы с помощью Python позволяет наглядно представить и изучить динамику системы.

Использование численных методов позволяет точно вычислить новые значения координат и скоростей для каждого шага времени.

Визуализация результатов в виде анимации и графиков позволяет более детально изучить изменение координат и скоростей элементов системы во времени.

Анимация движения системы с помощью Python может быть полезна для анализа и предсказания движения объектов в реальных ситуациях, таких как движение тел в физических системах или траектории движения транспортных средств.

Таким образом, выполнение лабораторной работы позволяет более глубоко изучить и понять движение системы, а также оценить его характеристики и динамику.