ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

**ОТЧЕТ**

**О ВЫПОЛОНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

**«АНИМАЦИЯ СИСТЕМЫ»**

**ПО ДИСЦИПЛИНЕ «Основы Компьютерного Моделирования Математических Систем»**

**ВАРИАНТ ЗАДАНИЯ №29**

Выполнил(а) студент группы М8О-215Б-23

Самарский Я.В.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

Проверил и принял

Ст. преп. каф. 802 Волков Е.В.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

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

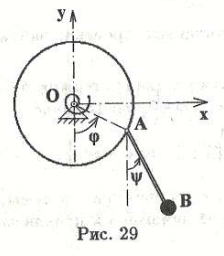
Москва, 2024

**Вариант №29**

**Задание:**

Реализовать анимацию движения механической системы в среде Octave (или Matlab)

**Механическая система:**

****

**Текст программы**

from matplotlib.patches import Circle

import matplotlib.pyplot as plt

import numpy as np

import sympy

from matplotlib.animation import FuncAnimation

from sympy.utilities.lambdify import implemented\_function

steps = 200

t = np.linspace(0, 1, steps)

# Произвольные законы движения

phi = t \* 2 \* np.pi

psi = np.cos(t) \* 3.3 \* np.pi

figure = plt.figure(figsize=[9, 9])

subplot = figure.add\_subplot(1, 1, 1)

subplot.axis('equal')

subplot.set\_xlim((-3, 3))

subplot.set\_ylim((-3, 3))

R\_MAIN = 1

R\_B = 0.1

L = 1.5

x\_a = np.cos(phi)

y\_a = np.sin(phi)

x\_b = x\_a + np.cos(psi) \* L

y\_b = y\_a + np.sin(psi) \* L

# Центральная окружность

main\_circle = Circle((0, 0), R\_MAIN, fill=False)

subplot.add\_patch(main\_circle)

# Прямые AB и AO

line\_ab = subplot.plot([x\_a[0], x\_b[0]], [y\_a[0], y\_b[0]], marker='o')[0]

line\_oa = subplot.plot([0, x\_a[0]], [0, y\_a[0]], 'k--')[0]

# Точка B

b\_circle = Circle((x\_b[0], y\_b[0]), R\_B, fill=True, color='b', zorder=2)

b\_circle\_patch = subplot.add\_patch(b\_circle)

# Спираль

R\_SPIRAL\_1 = 0.05

R\_SPIRAL\_2 = 0.2

radians = np.linspace(0, 2 \* np.pi, 50)

radiuses = np.linspace(R\_SPIRAL\_1, R\_SPIRAL\_2, 50)

X\_n = np.cos(radians) \* radiuses

Y\_n = np.sin(radians) \* radiuses

spiral = subplot.plot(X\_n, Y\_n)[0]

def Rot2D(X, Y, phi):

    # Поворот двумерной ДСК с помощью матрицы поворота

    X\_r = X \* np.cos(phi) - Y \* np.sin(phi)

    Y\_r = X \* np.sin(phi) + Y \* np.cos(phi)

    return X\_r, Y\_r

def update(i):

    print(i)

    line\_ab.set\_data([x\_a[i], x\_b[i]], [y\_a[i], y\_b[i]])

    line\_oa.set\_data([0, x\_a[i]], [0, y\_a[i]])

    b\_circle.set\_center((x\_b[i], y\_b[i]))

    new\_X\_n, new\_Y\_n = Rot2D(X\_n, Y\_n, phi[i])

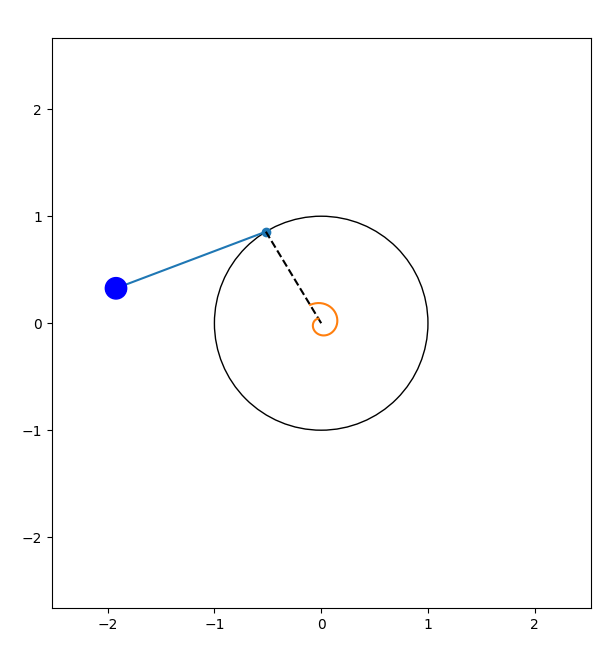
    spiral.set\_data(new\_X\_n, new\_Y\_n)

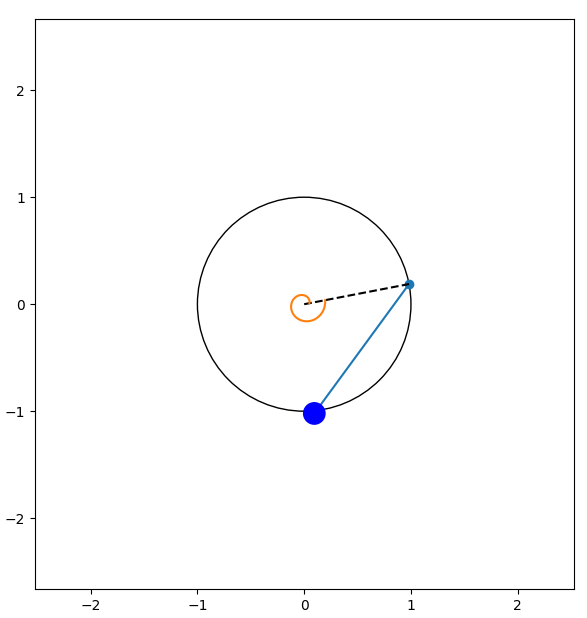
    return line\_ab, line\_oa, b\_circle\_patch, spiral

animation = FuncAnimation(figure, update, frames=steps, interval=10, blit=True)

plt.show()

**Результат работы:**





**Вывод:**

Во второй лабораторной работе я изучил способ анимации механической системы для произвольных законов движения элементов. С помощью этих законов движения я рассчитал координаты и отрисовал по ним механическую систему.