ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

**ОТЧЕТ**

**О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

**«АНИМАЦИЯ СИСТЕМЫ»**

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

**ВАРИАНТ ЗАДАНИЯ №15**

Выполнил(а) студент группы М8О-207Б-22

КРУТЫХ РОМАН ВЯЧЕСЛАВОВИЧ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

Проверил и принял

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

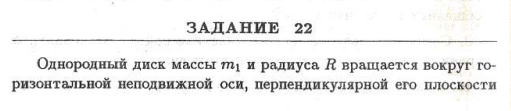
подпись, дата

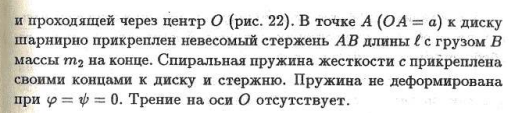
с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

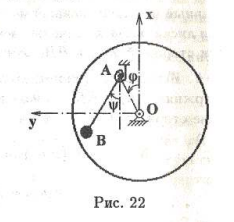
Москва, 2023

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

*Условия задачи для 22 (15 + 7) варианта:*







*Код программы:*

import math

import numpy as np

from matplotlib.animation import FuncAnimation

import matplotlib.pyplot as plt

def Circle1(X, Y, radius):

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

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

return CX, CY

# data of task

t = np.linspace(0, 10, 1001)

phi = 3.5 \* np.sin(math.pi/ 6 + 3\*t)

psi = -9 \* np.sin(3\*t + math.pi / 6)

l = 5

alpha\_angle = 30

alpha = 5

r = 8

c = 10

g = 9.81

#задаем точки системы и строим линии

X\_start = 0

Y\_start = 0

X\_O = X\_start

Y\_O = Y\_start

X\_A = X\_O - alpha \* np.sin(phi)

Y\_A = Y\_O + alpha \* np.cos(phi)

X\_B = X\_A - l \* np.sin(psi)

Y\_B = Y\_A - l \* np.cos(psi)

fig = plt.figure(figsize=[10, 7])

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

ax.axis('equal')

ax.set(xlim=[-25, 25], ylim=[-25, 25])

Point\_O = ax.plot(X\_O, Y\_O, marker='o', color='black')[0]

Point\_A = ax.plot(X\_A[0], Y\_A[0], marker='o', color='black')[0]

Point\_B = ax.plot(X\_B[0], Y\_B[0], marker='o', markersize=20, color='black')[0]

circle1 = ax.plot(\*Circle1(X\_O, Y\_O, r), 'red') # main circle

triangle, = ax.plot([-0.5, 0, 0.5],

[-1, 0, -1], color='black')

line\_tr = ax.plot([- 0.5, 0.5], [-1, -1],

color='black')[0]

Line\_AB = ax.plot([X\_A[0], X\_B[0]], [Y\_A[0], Y\_B[0]], color='black', linewidth=3)[0]

# spiral spring

Nv = 2.6

R1 = 0.2

R2 = 5

thetta = np.linspace(0, Nv \* 6.28 - psi[0], 1001)

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

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

Drawed\_Spiral\_Spring = ax.plot(X\_SpiralSpr + X\_A[0], Y\_SpiralSpr + Y\_A[0], color='black')[0]

def anima(i): #функция анимации

Point\_A.set\_data(X\_A[i], Y\_A[i])

Point\_B.set\_data(X\_B[i], Y\_B[i])

Line\_AB.set\_data([X\_A[i], X\_B[i]], [Y\_A[i], Y\_B[i]])

thetta = np.linspace(0, Nv \* 6 - psi[i], 100)

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

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

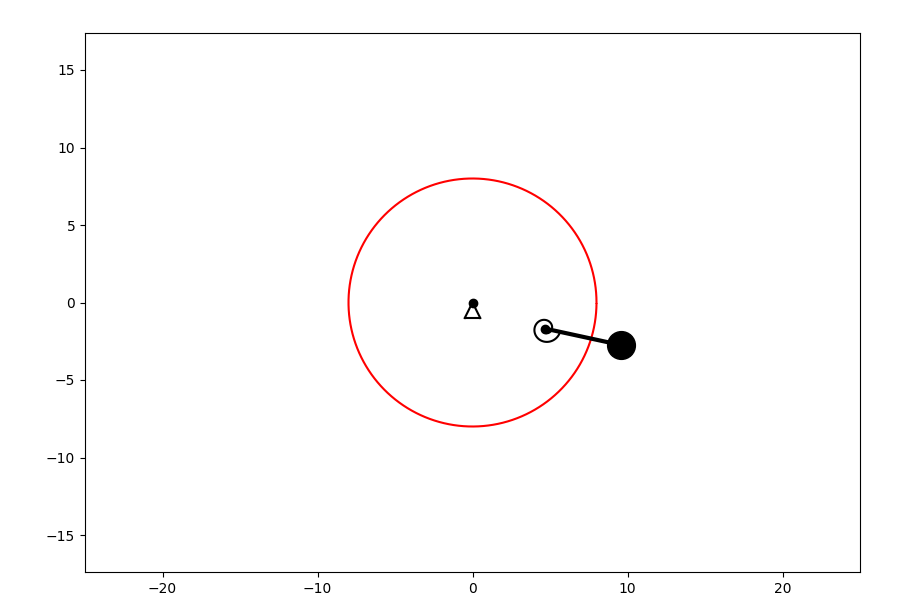
Drawed\_Spiral\_Spring.set\_data(X\_SpiralSpr + X\_A[i], Y\_SpiralSpr +

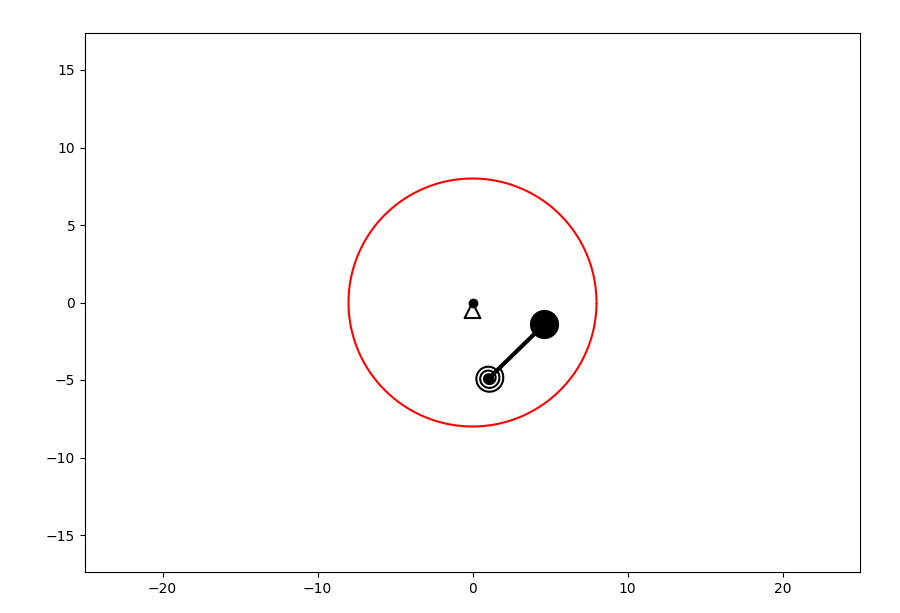
Y\_A[i])

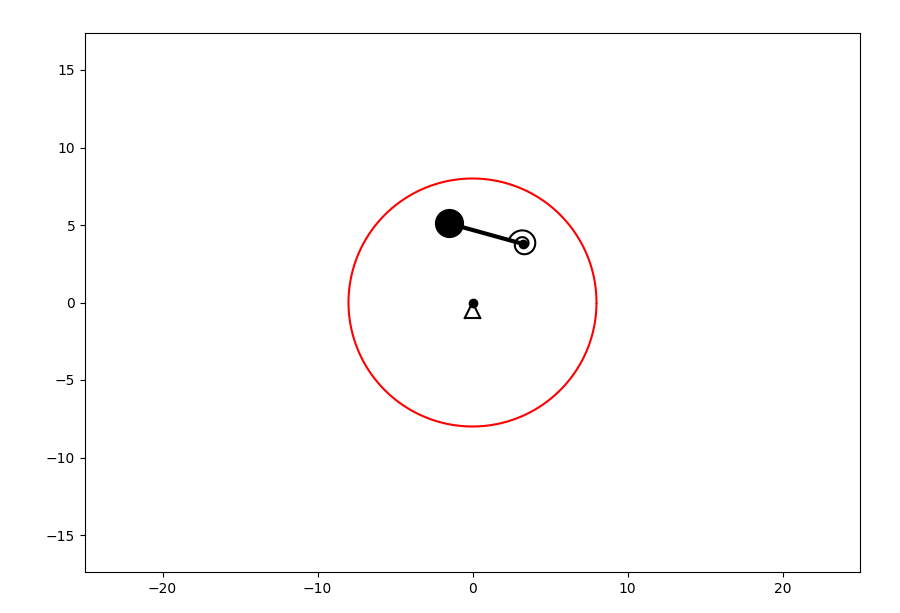
return [Point\_A, Point\_B, Line\_AB, Drawed\_Spiral\_Spring]

anim = FuncAnimation(fig, anima, frames=1000, interval=10)

plt.show()

*Рисунки получившейся физической модели:*





*Вывод:*

*В ходе выполнения этой лабораторной работы я написал Python код, который строит анимацию движения системы. Я реализовал Диск, вращающийся вокруг неподвижной оси, стержень с грузом, и спиральную пружины, прикрепленную к стержню и грузу, а также, она сжимается и разжимается в соответствии с данным законом.  
  
Я научился, как с помощью Matplotlib и NumPy задавать и анимировать движение системы, используя данные законы и средства языка программирования Python.*