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 № 14

Выполнил(а) студент группы М8О-208Б-20

Марков И. И. _____
подпись, дата

Проверил и принял

Зав. каф. 802, Бардин Б.С._____
подпись, дата

с оценкой _____

Москва, 2022

Лабораторная работа 2.

Задание: реализовать анимацию движения механической системы.

1. Механическая система

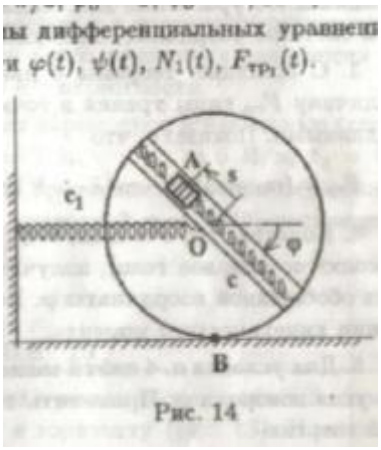

Рис. 14

2. Код программы

```
import numpy as np

from math import pi

from matplotlib.animation import FuncAnimation

from matplotlib import pyplot as plt


# константы

t_fin = 20

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


x0 = 5 # положение кольца в начальный момент времени

R = 3 # радиус кольца


s = [0.9 * R * np.cos(2 * x) for x in t] # отклонение груза

phi = [1 + 0.5*np.cos(4 * x + pi) for x in t] # угол поворота кольцы


angles = np.linspace(0, 2 * pi , 360)


box_w = 0.4 # ширина груза

box_h = 0.2  # высота груза


# генерирует пружинку высотой h с количеством витков k и шириной w

def spring(k, h, w):

    x = np.linspace(0, h, 100)

    return np.array([

        x,

        np.sin(2 * pi / (h / k) * x) * w

    ])
```

```
# массивы, где будем хранить просчитанные точки
ring_dots_x_tmp = R * np.cos(angles)
ring_dots_y_tmp = R * np.sin(angles)

box_x_tmp = np.array([-box_h / 2, -box_h / 2, box_h / 2, box_h / 2, -box_h / 2])
box_y_tmp = np.array([-box_w / 2, box_w / 2, box_w / 2, -box_w / 2, -box_w / 2])

ring_dots_x = np.zeros([len(t), len(angles)])
ring_dots_y = np.zeros([len(t), len(angles)])

box_dots_x = np.zeros([len(t), 5])
box_dots_y = np.zeros([len(t), 5])
spring_a_x = np.zeros([len(t), 100])
spring_a_y = np.zeros([len(t), 100])

spring_b_x = np.zeros([len(t), 100])
spring_b_y = np.zeros([len(t), 100])

spring_c_x = np.zeros([len(t), 100])
spring_c_y = np.zeros([len(t), 100])

# считаем точки
for i in range(len(t)):
    # центр кольца
    ring_x = x0 + phi[i] * R
    ring_y = R

    # точки окружности кольца
    ring_dots_x[i] = np.cos(phi[i]) * ring_dots_x_tmp + np.sin(phi[i]) * ring_dots_y_tmp + ring_x
    ring_dots_y[i] = - np.sin(phi[i]) * ring_dots_x_tmp + np.cos(phi[i]) * ring_dots_y_tmp + ring_y

    # точки груза
    bx = box_x_tmp - s[i]
    by = box_y_tmp
    box_dots_x[i] = np.cos(phi[i]) * bx + np.sin(phi[i]) * by + ring_x
    box_dots_y[i] = - np.sin(phi[i]) * bx + np.cos(phi[i]) * by + ring_y

    # горизонтальная пружина
    spring_a_x[i] = spring(5, ring_x, 0.2)[0]
    spring_a_y[i] = spring(5, ring_x, 0.2)[1] + ring_y

    # верхняя пружина в кольце
```

```
    b_x = R - spring(10, R + s[i] - box_h / 2, 0.2)[0]
    b_y = spring(10, R - s[i], 0.2)[1]
    spring_b_x[i] = np.cos(phi[i]) * b_x + np.sin(phi[i]) * b_y + ring_x
    spring_b_y[i] = -np.sin(phi[i]) * b_x + np.cos(phi[i]) * b_y + ring_y

    # нижняя пружина в кольце
    c_x = spring(10, R - s[i] - box_h / 2, 0.2)[0] - R
    c_y = spring(10, R - s[i], 0.2)[1]
    spring_c_x[i] = np.cos(phi[i]) * c_x + np.sin(phi[i]) * c_y + ring_x
    spring_c_y[i] = -np.sin(phi[i]) * c_x + np.cos(phi[i]) * c_y + ring_y

# рисуем график
fig = plt.figure() # создаем холст, на котором будем рисовать фигуры
ax = fig.add_subplot(1, 1, 1)
ax.axis("equal")

surface = ax.plot([0, 0, 10], [10, 0, 0]) # пол и стена
ring, = ax.plot(ring_dots_x[0], ring_dots_y[0]) # кольцо
box, = ax.plot(box_dots_x[0], box_dots_y[0]) # груз
spring_a, = ax.plot(spring_a_x[0], spring_a_y[0]) # горизонтальная пружина
spring_b, = ax.plot(spring_b_x[0], spring_b_y[0]) # нижняя пружина
spring_c, = ax.plot(spring_c_x[0], spring_c_y[0]) # верхняя пружина

# функция анимации
def animate(i):
    ring.set_data(ring_dots_x[i], ring_dots_y[i])
    box.set_data(box_dots_x[i], box_dots_y[i])
    spring_a.set_data(spring_a_x[i], spring_a_y[i])
    spring_b.set_data(spring_b_x[i], spring_b_y[i])
    spring_c.set_data(spring_c_x[i], spring_c_y[i])

    return ring, box, spring_a, spring_b, spring_c

animation = FuncAnimation(fig, animate, frames=len(t), interval=60)
plt.show()
```

3. Результат работы программы