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 №18**

Выполнил(а) студент группы М8О-212Б-22

Козырев П.А.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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.

**Задание системы формулируется следующим образом:**

Колечко M веса P прикреплено к концу пружины OM и может скользить вдоль стержня, который качается в вертикальной плоскости вокруг шарнира О. Жесткость пружины с. Длина недеформированной пружины ОО1 = l.

**Рисунок получившейся анимации движения:**

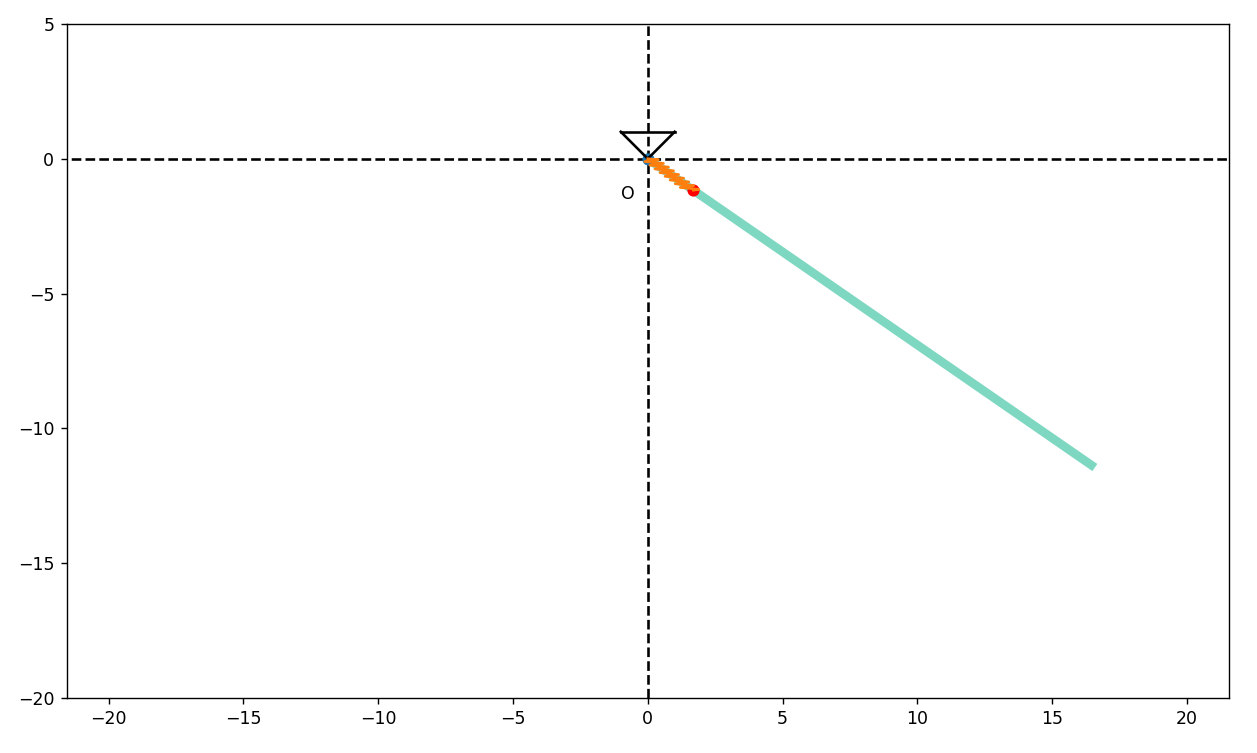


Рис. 1.1

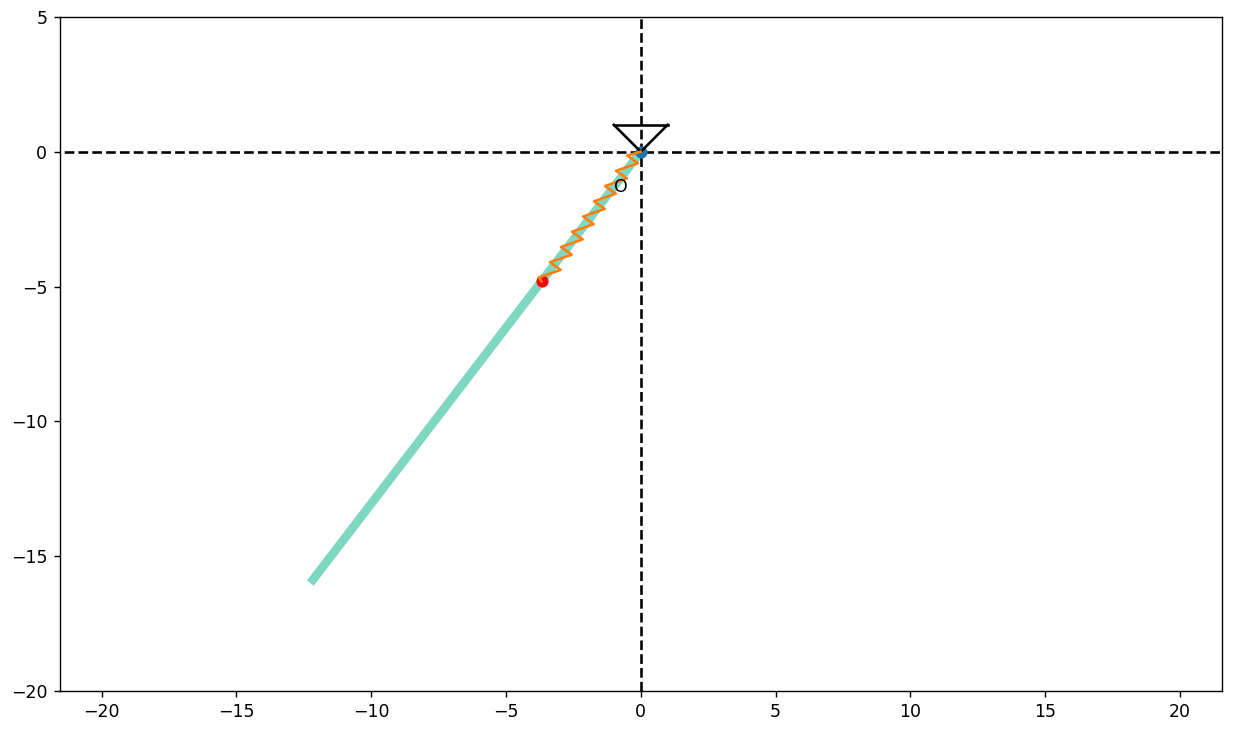


Рис.1.2

**Код программы:**

import math

import numpy as np

import matplotlib.pyplot as plt

from matplotlib.animation import FuncAnimation

from scipy.integrate import odeint

Steps = 1001

t\_fin = 20

t = np.linspace(0, t\_fin, Steps)

l\_kernel = 20 # длина стержня

s\_0 = 8 # длина O-О1

s = np.sin(t)

phi = np.cos(t)

M\_x = s\_0 \* np.sin(phi) \* s

M\_y = (-1) \* s\_0 \* np.cos(phi) \* s

Kernel\_x = l\_kernel \* np.sin(phi)

Kernel\_y = (-1) \* l\_kernel \* np.cos(phi)

SprX\_0 = 4

K = 19

Sh = 0.3

b = 1 / (K - 2)

X\_Spr = np.zeros(K)

Y\_Spr = np.zeros(K)

X\_Spr[0] = 0

Y\_Spr[0] = 0

X\_Spr[K - 1] = 0

Y\_Spr[K - 1] = 1

for i in range(K - 2):

Y\_Spr[i + 1] = b \* ((i + 1) - 1 / 2)

X\_Spr[i + 1] = Sh \* (-1) \*\* i

L\_Spr = (s\_0 \* np.cos(phi) \* s \* (-1))

X\_U = np.linspace(0, 1, K)

fig = plt.figure(figsize=[12, 7])

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

ax.axis('equal')

ax.set(xlim=[-20, 20], ylim=[-20, 5])

Kernel = ax.plot([0, Kernel\_x[0]], [0, Kernel\_y[0]], linewidth=5, color="#7ED7C1")[0]

ax.plot([0, 0], [-20, 20], "k--")

ax.plot([-50, 50], [0, 0], "k--")

ax.plot(0, 0, marker='o')

ax.annotate("O", xy=(0, 0), xytext=(-1, -1.5))

ax.plot([-1, 0, 1], [1, 0, 1], "black")

ax.plot([-1, 1], [1, 1], "black")

Point\_M = ax.plot(M\_x[0], M\_y[0], marker='o', color='r')[0]

Drawed\_Spring = ax.plot(X\_Spr + X\_U \* np.sin(phi)[0] \* (s\_0 \* s[0]), Y\_Spr \* L\_Spr[0])[0]

def anima(i):

if s[i] < 0: tmp = -1

else: tmp = 1

Point\_M.set\_data([M\_x[i] \* tmp], [M\_y[i] \* tmp])

Kernel.set\_data([0, Kernel\_x[i]], [0, Kernel\_y[i]])

Drawed\_Spring.set\_data((X\_Spr + X\_U \* np.sin(phi)[i] \* (s\_0 \* s[i])) \* tmp, Y\_Spr \* L\_Spr[i]\*tmp)

return [Point\_M, Kernel, Drawed\_Spring]

anim = FuncAnimation(fig, anima, frames=len(t), interval=40, repeat=False)

plt.show()

**Пояснения:**

M\_x, M\_y – координаты колечка;

Kernel\_x, Kernel\_y – координаты стержня;

X\_Spr, Y\_Spr – шаблон пружинки единичной длины.

Функция anima(i) – функция смены кадра. На каждой итерации меняет координаты стержня, колечка и пружинки.

**Вывод:**

Я успешно выполнил лабораторную работу. В результате я построил анимацию движения системы с помощью Python.