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 №2**

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

Борисов Я.А\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

Проверил и принял

Доцент каф. 802, Чекина Е.А.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

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

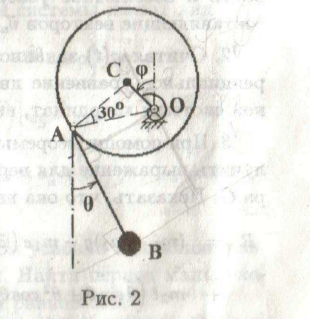
Москва, 2021

**Вариант № 2**

**Задание:**

Реализовать анимацию движения механической системы в среде Python

**Механическая система:**

****

**Текст 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*

l = 8

alpha = 30

r = 7

c = 10

e = r / math.sqrt(3)

g = 9.81

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

phi = np.sin(2.1 \* t)

tau = np.sin(phi - math.pi / 6)

#задаем точки системы и строим линии

X\_O = 0

Y\_O = 0

X\_C = X\_O - e \* np.sin(phi)

Y\_C = Y\_O + e \* np.cos(phi)

X\_A = X\_C - r \* np.sin(math.pi / 2 + phi)

Y\_A = Y\_C + r \* np.cos(math.pi / 2 + phi)

X\_B = X\_A + l \* np.sin(tau)

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

fig = plt.figure(figsize=[13, 9])

ax = fig.add\_subplot(1, 2, 1)

ax.axis('equal')

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

*# spiral spring*

Nv = 1.1

R1 = 0.2

R2 = 6

thetta = np.linspace(0, Nv \* 6.28 - phi[0], 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 = ax.plot(X\_SpiralSpr + X\_O, Y\_SpiralSpr + Y\_O, color='black')[0]

Point\_C = ax.plot(X\_C[0], Y\_C[0], marker='o', markersize=12, color='black')[0]

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

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

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

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

Line\_OC = ax.plot([X\_O, X\_C[0]], [Y\_O, Y\_C[0]], color='black')[0]

circle1, = ax.plot(\*Circle1(X\_C[0], Y\_C[0], r), 'red') *# main circle*

triangle, = ax.plot([-1, 0, 1],

[-2, 0, -2], color='black')

line\_tr = ax.plot([- 1, 1], [-2, -2],

color='black')[0]

*# plots(строим графики)*

VXB = np.diff(X\_B)

VYB = np.diff(Y\_B)

WXB = np.diff(VXB)

WYB = np.diff(VYB)

ax2 = fig.add\_subplot(4, 2, 2)

ax2.plot(VXB)

plt.title('Vx of ball')

plt.xlabel('t values')

plt.ylabel('Vx values')

ax3 = fig.add\_subplot(4, 2, 4)

ax3.plot(VYB)

plt.title('Vy of ball')

plt.xlabel('t values')

plt.ylabel('Vy values')

ax4 = fig.add\_subplot(4, 2, 6)

ax4.plot(WXB)

plt.title('Wx of ball')

plt.xlabel('t values')

plt.ylabel('Wy values')

ax5 = fig.add\_subplot(4, 2, 8)

ax5.plot(WYB)

plt.title('Wy of ball')

plt.xlabel('t values')

plt.ylabel('Wx values')

plt.subplots\_adjust(wspace=0.3, hspace=0.7)

**def** Dordge(i): #функция анимации

circle1.set\_data(\*Circle1(X\_C[i], Y\_C[i], r))

Point\_O.set\_data(X\_O, Y\_O)

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

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

Line\_OC.set\_data([X\_O, X\_C[i]], [Y\_O, Y\_C[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 \* 5.6 + phi[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\_O, Y\_SpiralSpr + Y\_O)

**return** [circle1, Point\_O, Point\_C, Line\_OC, Drawed\_Spiral\_Spring, Point\_A, Point\_B, Line\_AB]

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

plt.show()

**Результат работы:**

|  |  |
| --- | --- |
| **Рис.1** | **Рис.2** |
| **Рис.3** | **Рис.4** |

**Вывод:** реализовали анимацию движения механической системы в среде Python, научились строить графики для характеристик движения механической системы.