

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 _____

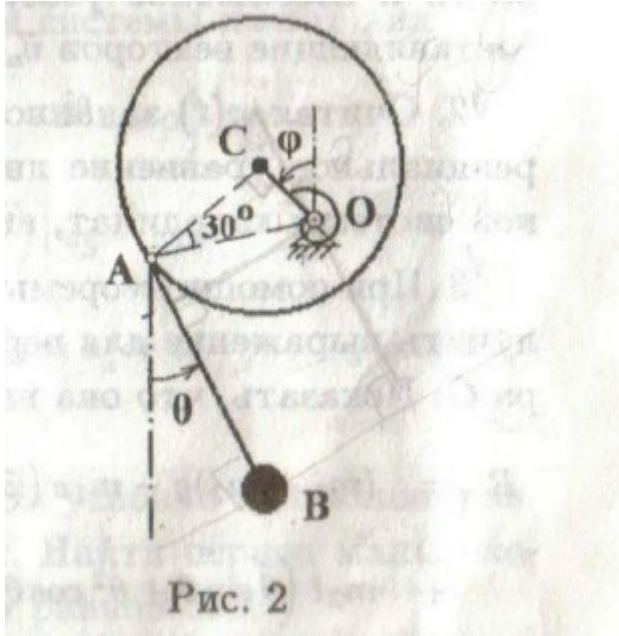
Москва, 2021

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

theta = np.linspace(0, Nv * 6.28 - phi[0], 100)
X_SpiralSpr = -(R1 * theta * (R2 - R1) / theta[-1]) * np.sin(theta)
Y_SpiralSpr = (R1 * theta * (R2 - R1) / theta[-1]) * np.cos(theta)
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)

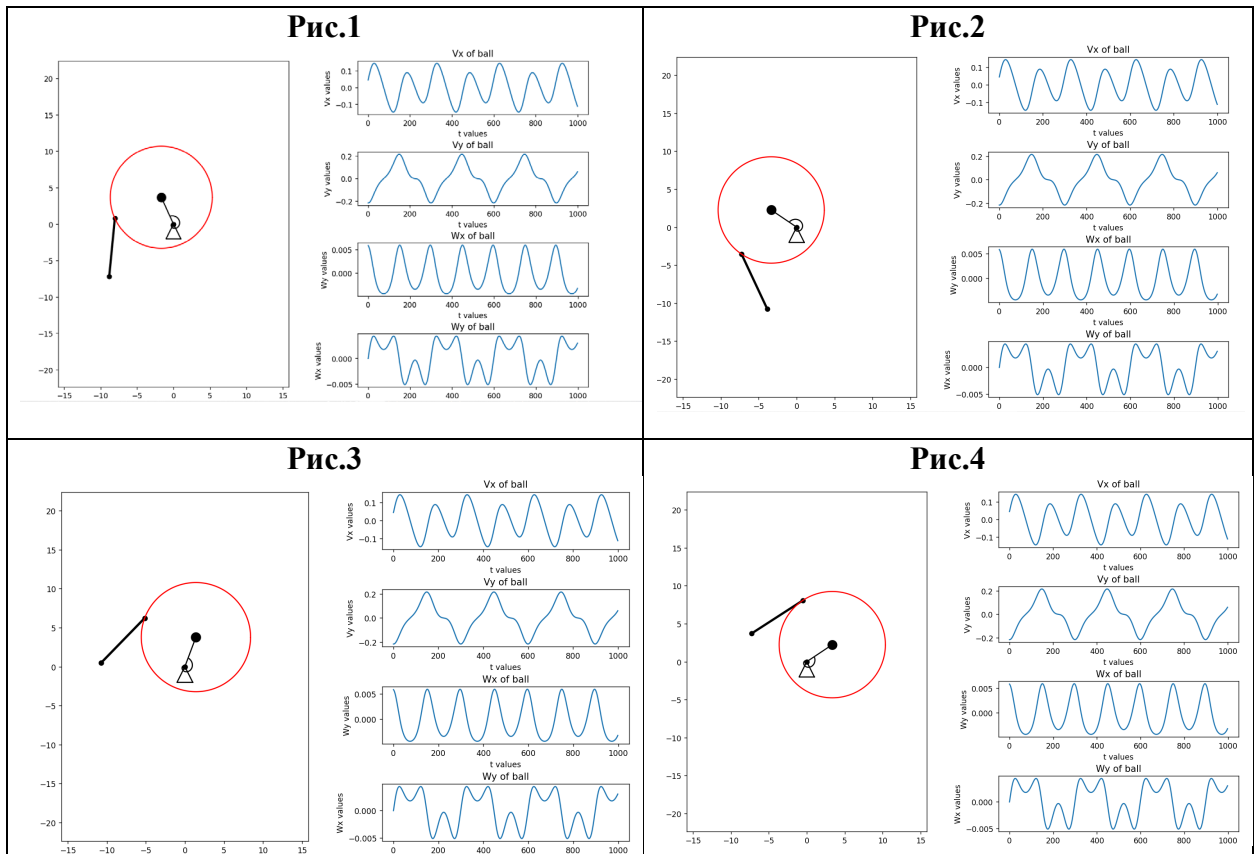
def Dordge(i): #функция анимации
    circle1.set_data(*Circle1(X_C[i], Y_C[i], r))
    Point_O.set_data(X_O, Y_O)
    Point_C.set_data(X_C[i], Y_C[i])
    Point_A.set_data(X_A[i], Y_A[i])
    Line_OC.set_data([X_O, X_C[i]], [Y_O, Y_C[i]])
    Point_B.set_data(X_B[i], Y_B[i])
    Line_AB.set_data([X_A[i], X_B[i]], [Y_A[i], Y_B[i]])
    theta = np.linspace(0, Nv * 5.6 + phi[i], 100)
    X_SpiralSpr = -(R1 * theta * (R2 - R1) / theta[-1]) *
np.sin(theta)
    Y_SpiralSpr = (R1 * theta * (R2 - R1) / theta[-1]) *
np.cos(theta)
    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:



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