ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

**ОТЧЕТ**

**О ВЫПЛОНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

**«АНИМАЦИЯ ТОЧКИ»**

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

**ВАРИАНТ ЗАДАНИЯ № 14**

Выполнил(а) студент группы М8О-208Б-21

Марков И. И. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

Проверил и принял

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

подпись, дата

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

Москва, 2022

**Лабораторная работа 3.**

*Задание:* Построить заданную траекторию и анимацию движения точки, а также отобразить стрелки скорости и ускорения.

**1. Закон движения точки:**

r = 2 + cos(t\*6)

phi = 7 \* t + 1.2 \* cos(t\*6)

**2. Текст программы:**

import sympy as sp

import numpy as np

import math

from matplotlib.animation import FuncAnimation

import matplotlib.pyplot as plt

T = np.linspace(1, 15, 1000)

t = sp.Symbol('t')

R\_ = 4

r = 2 + sp.cos(t\*6)

phi = 7 \* t + 1.2 \* sp.cos(t\*6)

x = r \* sp.cos(phi)

y = r \* sp.sin(phi)

Vx = sp.diff(x, t)

Vy = sp.diff(y, t)

Wx = sp.diff(Vx, t)

Wy = sp.diff(Vy, t)

V = sp.sqrt(Vx\*\*2 + Vy\*\*2)

Wfull = sp.sqrt(Wx\*\*2 + Wy\*\*2)

Wtan = sp.diff(V)

СurveVector = V\*\*2 / sp.sqrt(Wfull\*\*2 - Wtan\*\*2) # радиус кривизны

X = np.zeros\_like(T)

Y = np.zeros\_like(T)

R = np.zeros\_like(T)

VX = np.zeros\_like(T)

VY = np.zeros\_like(T)

WX = np.zeros\_like(T)

WY = np.zeros\_like(T)

CVector = np.zeros\_like(T)

for i in np.arange(len(T)):

X[i] = sp.Subs(x, t, T[i])

Y[i] = sp.Subs(y, t, T[i])

VX[i] = sp.Subs(Vx, t, T[i])

VY[i] = sp.Subs(Vy, t, T[i])

WX[i] = sp.Subs(Wx, t, T[i])

WY[i] = sp.Subs(Wy, t, T[i])

CVector[i] = sp.Subs(СurveVector, t, T[i])

def Rot2D(X, Y, Alpha):

RX = X \* np.cos(Alpha) - Y \* np.sin(Alpha)

RY = X \* np.sin(Alpha) + Y \* np.cos(Alpha)

return RX, RY

def anima(j):

P.set\_data(X[j], Y[j])

Vvector.set\_data([X[j], X[j] + VX[j]], [Y[j], Y[j] + VY[j]]) # vector of speed

Rvector.set\_data([0, X[j]], [0, Y[j]]) # radius-vector

Wvector.set\_data([X[j], X[j] + WX[j]], [Y[j], Y[j] + WY[j]]) # vector of speed-up

Cvector.set\_data([X[j], X[j] + (Y[j] + VY[j]) \* CVector[j] / sp.sqrt((Y[j] + VY[j])\*\*2 +

(X[j] + VX[j])\*\*2)], [Y[j], Y[j] - (X[j] + VX[j]) \* CVector[j] /

sp.sqrt((Y[j] + VY[j])\*\*2 + (X[j] + VX[j])\*\*2)])

RArrowX, RArrowY = Rot2D(ArrowX, ArrowY, math.atan2(VY[j], VX[j]))

RArrowWx, RArrowWy = Rot2D(ArrowWx, ArrowWy, math.atan2(WY[j], WX[j]))

RArrowRx, RArrowRy = Rot2D(ArrowRx, ArrowRy, math.atan2(Y[j], X[j]))

VArrow.set\_data(RArrowX + X[j] + VX[j], RArrowY + Y[j] + VY[j])

WArrow.set\_data(RArrowWx + X[j] + WX[j], RArrowWy + Y[j] + WY[j])

RArrow.set\_data(RArrowRx + X[j], RArrowRy + Y[j])

return P, Vvector, Rvector, Wvector, VArrow, WArrow, RArrow, Cvector,

fig = plt.figure()

ax1 = fig.add\_subplot(1, 1, 1)

ax1.axis('equal')

ax1.set(xlim=[-R\_, R\_], ylim=[-R\_, R\_])

ax1.plot(X, Y)

P, = ax1.plot(X[0], Y[0], 'r', marker='o')

Vvector, = ax1.plot([X[0], X[0] + VX[0]], [Y[0], Y[0] + VY[0]], 'red')

Wvector, = ax1.plot([X[0], X[0] + WX[0]], [Y[0], Y[0] + WY[0]], 'green')

Rvector, = ax1.plot([0, X[0]], [0, Y[0]], 'black')

Cvector, = ax1.plot([X[0], X[0] + (Y[0] + VY[0]) \* CVector[0] / sp.sqrt((Y[0] + VY[0])\*\*2 +

(X[0] + VX[0])\*\*2)], [Y[0], Y[0] - (X[0] + VX[0]) \* CVector[0] /

sp.sqrt((Y[0] + VY[0])\*\*2 + (X[0] + VX[0])\*\*2)], 'purple')

ArrowX = np.array([-0.1 \* R\_, 0, -0.1 \* R\_])

ArrowY = np.array([0.05 \* R\_, 0, -0.05 \* R\_])

ArrowRx = np.array([-0.1 \* R\_, 0, -0.1 \* R\_])

ArrowRy = np.array([0.05 \* R\_, 0, -0.05 \* R\_])

ArrowWx = np.array([-0.1 \* R\_, 0, -0.1 \* R\_])

ArrowWy = np.array([0.05 \* R\_, 0, -0.05 \* R\_])

RArrowX, RArrowY = Rot2D(ArrowX, ArrowY, math.atan2(VY[0], VX[0]))

RArrowWx, RArrowWy = Rot2D(ArrowWx, ArrowWy, math.atan2(WY[0], WX[0]))

RArrowRx, RArrowRy = Rot2D(ArrowRx, ArrowRy, math.atan2(Y[0], X[0]))

VArrow, = ax1.plot(RArrowX + X[0] + VX[0], RArrowY + Y[0] + VY[0], 'red')

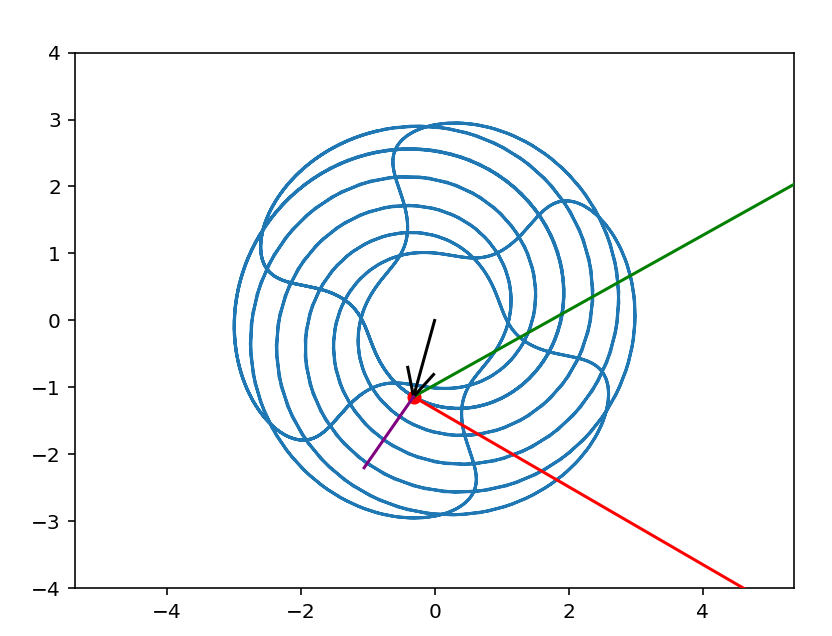
WArrow, = ax1.plot(RArrowWx + X[0] + WX[0], RArrowWy + Y[0] + WY[0], 'green')

RArrow, = ax1.plot(RArrowRx + X[0], RArrowRy + Y[0], 'black')

anim = FuncAnimation(fig, anima, frames=1500, interval=60, blit=True)

plt.show()

**Результат работы программы:**

****