ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

**ОТЧЕТ**

**О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

**«АНИМАЦИЯ ТОЧКИ»**

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

**ВАРИАНТ ЗАДАНИЯ №10**

Выполнил(а) студент группы М8О-203Б-22

Клименко В.М.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

Проверил и принял

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

подпись, дата

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

Москва, 2023

*Задание:* построить заданную траекторию, запустить анимацию движения точки, построить стрелки радиус-вектора, вектора скорости, вектора ускорения и радиуса кривизны.

Код:

import math

import numpy

import sympy

import matplotlib.pyplot as plt

from matplotlib.animation import FuncAnimation

def animate(i):

P.set\_data(Xs[i], Ys[i])

VVec.set\_data([Xs[i], Xs[i] + Xs\_velocity[i]], [Ys[i], Ys[i] + Ys\_velocity[i]])

RVecArrowX, RVecArrowY = rotation2D(ArrowX, ArrowY, math.atan2(Ys\_velocity[i], Xs\_velocity[i]))

VVecArrow.set\_data(RVecArrowX + Xs[i] + Xs\_velocity[i], RVecArrowY + Ys[i] + Ys\_velocity[i])

AVec.set\_data([Xs[i], Xs[i] + Xs\_acceleration[i]], [Ys[i], Ys[i] + Ys\_acceleration[i]])

RVecArrowX\_A, RVecArrowY\_A = rotation2D(ArrowX, ArrowY, math.atan2(Ys\_acceleration[i], Xs\_acceleration[i]))

AVecArrow.set\_data(RVecArrowX\_A + Xs[i] + Xs\_acceleration[i], RVecArrowY\_A + Ys[i] + Ys\_acceleration[i])

RVec.set\_data([0, Xs[i]], [0, Ys[i]])

RVecArrowX\_R, RVecArrowY\_R = rotation2D(ArrowX, ArrowY, math.atan2(Ys[i], Xs[i]))

RVecArrow.set\_data(RVecArrowX\_R + Xs[i], RVecArrowY\_R + Ys[i])

return P, VVec, VVecArrow, AVec, AVecArrow, RVec, RVecArrow

def rotation2D(x, y, angle):

Rx = x \* numpy.cos(angle) - y \* numpy.sin(angle)

Ry = x \* numpy.sin(angle) + y \* numpy.cos(angle)

return Rx, Ry

t = sympy.Symbol('t')

r = 2 + sympy.sin(6 \* t)

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

x = r \* sympy.cos(phi)

y = r \* sympy.sin(phi)

x\_velocity = sympy.diff(x, t)

y\_velocity = sympy.diff(y, t)

x\_acceleration = sympy.diff(x\_velocity, t)

y\_acceleration = sympy.diff(y\_velocity, t)

T = numpy.linspace(1, 10, 2000)

Xs = numpy.zeros\_like(T)

Ys = numpy.zeros\_like(T)

Xs\_velocity = numpy.zeros\_like(T)

Ys\_velocity = numpy.zeros\_like(T)

Xs\_acceleration = numpy.zeros\_like(T)

Ys\_acceleration = numpy.zeros\_like(T)

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

Xs[i] = sympy.Subs(x, t, T[i])

Ys[i] = sympy.Subs(y, t, T[i])

Xs\_velocity[i] = sympy.Subs(x\_velocity, t, T[i])

Ys\_velocity[i] = sympy.Subs(y\_velocity, t, T[i])

Xs\_acceleration[i] = sympy.Subs(x\_acceleration, t, T[i])

Ys\_acceleration[i] = sympy.Subs(y\_acceleration, t, T[i])

fig = plt.figure()

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

ax1.axis('equal')

ax1.set(xlim = [-3.5, 3.5], ylim = [-3.5, 3.5])

ax1.plot(Xs, Ys)

P, = ax1.plot(Xs[0], Ys[0], marker = 'o')

ArrowX = numpy.array([-0.2, 0, -0.2])

ArrowY = numpy.array([0.1, 0, -0.1])

VVec, = ax1.plot([Xs[0], Xs[0] + Xs\_velocity[0]], [Ys[0], Ys[0] + Ys\_velocity[0]], 'r')

RVecArrowX, RVecArrowY = rotation2D(ArrowX, ArrowY, math.atan2(Ys\_velocity[0], Xs\_velocity[0]))

VVecArrow, = ax1.plot(RVecArrowX + Xs\_velocity[0] + Xs[0], RVecArrowY + Ys\_velocity[0] + Ys[0], 'r')

RVec, = ax1.plot([0, Xs[0]], [0, Ys[0]], 'b')

RVecArrowX\_R, RVecArrowY\_R = rotation2D(ArrowX, ArrowY, math.atan2(Ys[0], Xs[0]))

RVecArrow, = ax1.plot(RVecArrowX\_R + Xs\_velocity[0] + Xs[0], RVecArrowY\_R + Ys\_velocity[0] + Ys[0], 'b')

AVec, = ax1.plot([Xs[0], Xs[0] + Xs\_acceleration[0]], [Ys[0], Ys[0] + Ys\_acceleration[0]], 'g')

RVecArrowX\_A, RVecArrowY\_A = rotation2D(ArrowX, ArrowY, math.atan2(Ys\_acceleration[0], Xs\_acceleration[0]))

AVecArrow, = ax1.plot(RVecArrowX\_A + Xs[0], RVecArrowY\_A + Ys[0], 'g')

animation = FuncAnimation(fig, animate, frames = 1000, interval = 10, blit = True)

plt.show()

Скриншот выполняющейся программы:

