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 № 17**

Выполнил(а) студент группы М8О-209Б-23

Стрепетов И.О.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

Проверил и принял

Ст. преп. каф. 802, Волков Е.В.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

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

Москва, 2024

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

Вариант:



Текст программы:

import numpy as np

import sympy as sp

import math

import matplotlib.pyplot as plt

from matplotlib.animation import FuncAnimation

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

T = np.linspace(0, 10, 700)

t = sp.Symbol('t')

r = 2 + 0.5 \* sp.sin(12 \* t)

phi = 1.2 \* t + 0.2 \* sp.cos(12 \* t)

x = r \* sp.cos(phi)

y = r \* sp.sin(phi)

X = np.zeros\_like(T)

Y = np.zeros\_like(T)

VX = np.zeros\_like(T)

VY = np.zeros\_like(T)

AX = np.zeros\_like(T)

AY = np.zeros\_like(T)

A = np.zeros\_like(T)

AT = np.zeros\_like(T)

#Curve = np.zeros\_like(T)

Vx = sp.diff(x, t)

Vy = sp.diff(x, t)

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

VxN = Vx / V

VyN = Vy / V

Ax = sp.diff(Vx, t)

Ay = sp.diff(Vy, t)

A\_full = sp.sqrt(Ax \*\* 2 + Ay \*\* 2)

AxN = Ax / A\_full

AyN = Ay / A\_full

A\_tan = sp.diff(V, t)

#R\_cur = (V \*\* 2) / sp.sqrt(A\_full \*\* 2 - A\_tan \*\* 2)

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(VxN, t, T[i])

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

AX[i] = sp.Subs(AxN, t, T[i])

AY[i] = sp.Subs(AyN, t, T[i])

#Curve[i] = sp.Subs(R\_cur, t, T[i])

fig = plt.figure()

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

ax1.axis('equal')

ax1.set(xlim = [-10, 10], ylim = [-10, 10])

ax1.plot(X, Y)

# Точка

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

# Линии скорости, ускорения, радиус-вектора, радиуса кривизны

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

ALine, = ax1.plot([X[0], X[0] + AX[0]], [Y[0], Y[0] + AY[0]], 'b')

RLine, = ax1.plot([0, X[0]], [0, Y[0]], 'pink')

#CurveVec, = ax1.plot([X[0], X[0] + (Y[0] + VY[0]) \* Curve[0] / sp.sqrt((Y[0] + VY[0]) \*\* 2 + (X[0] + VX[0])\*\* 2)],

#[Y[0], Y[0] - (X[0] + VX[0]) \* Curve[0] / sp.sqrt((Y[0] + VY[0]) \*\* 2 + (X[0] + VX[0]) \*\* 2)], 'orange')

# Шаблон стрелки

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

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

# Стрелка скорости

RVArrowX, RVArrowY = Rot2D(ArrowX, ArrowY, math.atan2(VY[0], VX[0]))

VArrow, = ax1.plot(RVArrowX + X[0] + VX[0], RVArrowY + Y[0] + VY[0], 'r')

# Стрелка ускорения

RAArrowX, RAArrowY = Rot2D(ArrowX, ArrowY, math.atan2(AY[0], AX[0]))

AArrow, = ax1.plot(RAArrowX + X[0], RAArrowY + Y[0], 'b') #+AX[0]

# Стрелка радиус-вектора

RRArrowX, RRArrowY = Rot2D(ArrowX, ArrowY, math.atan2(Y[0], X[0]))

RArrow, = ax1.plot(RRArrowX + X[0], RRArrowY + Y[0], 'pink')

def anima(i):

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

# Скорость

VLine.set\_data([X[i], X[i] + VX[i]], [Y[i], Y[i] + VY[i]])

RVArrowX, RVArrowY = Rot2D(ArrowX, ArrowY, math.atan2(VY[i], VX[i]))

VArrow.set\_data(RVArrowX + X[i] + VX[i], RVArrowY + Y[i] + VY[i])

# Ускорение

ALine.set\_data([X[i], X[i] + AX[i]], [Y[i], Y[i] + AY[i]])

RAArrowX, RAArrowY = Rot2D(ArrowX, ArrowY, math.atan2(AY[i], AX[i]))

AArrow.set\_data(RAArrowX + X[i] + AX[i], RAArrowY + Y[i] + AY[i])

# Радиус-вектор

RLine.set\_data([0, X[i]], [0, Y[i]])

RRArrowX, RRArrowY = Rot2D(ArrowX, ArrowY, math.atan2(Y[i], X[i]))

RArrow.set\_data(RRArrowX + X[i], RRArrowY + Y[i])

#Радиус кривизны

#CurveVec.set\_data([X[i], X[i] + (Y[i] + VY[i]) \* Curve[i] / sp.sqrt((Y[i] + VY[i]) \*\* 2 + (X[i] + VX[i])\*\* 2)],

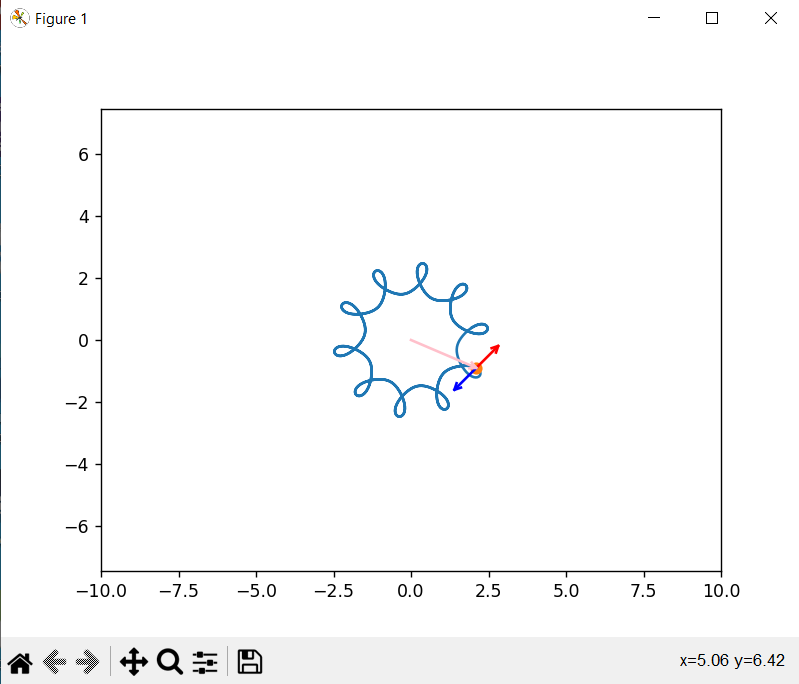
#[Y[i], Y[i] - (X[i] + VX[i]) \* Curve[i] / sp.sqrt((Y[i] + VY[i]) \*\* 2 + (X[i] + VX[i]) \*\* 2)], 'orange')

return P, VLine, VArrow, ALine, AArrow, RLine, RArrow #, CurveVec

anim = FuncAnimation(fig, anima, frames = 1000, interval = 2, repeat = True)

plt.show()

Результат работы программы:



Вывод: Выполнив лабораторную работу, я приобрел практические навыки в области анимации точки.