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 8

Выполнил(а) студент группы М8О-206Б-22

Жаднов Михаил Денисович_____
подпись, дата

Проверил и принял

Зав. каф. 802, Бардин Б.С._____
подпись, дата

с оценкой _____

Москва, 2023

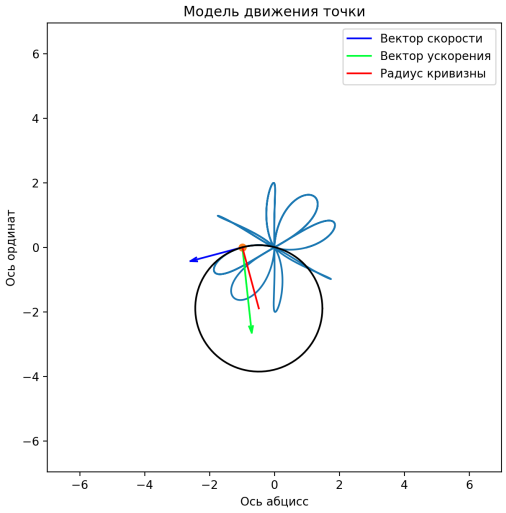
Задание: построить заданную траекторию (r(t) = 2 + sin(8t), φ(t) = t + 0.2*sin(6t)), запустить анимацию движения точки, построить стрелки радиус-вектора, вектора скорости, вектора ускорения и радиуса кривизны.

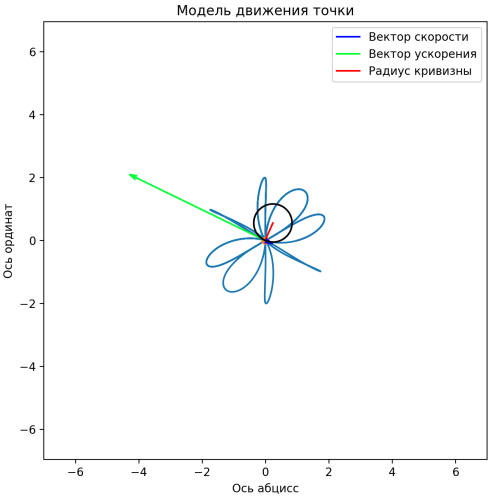
Код:

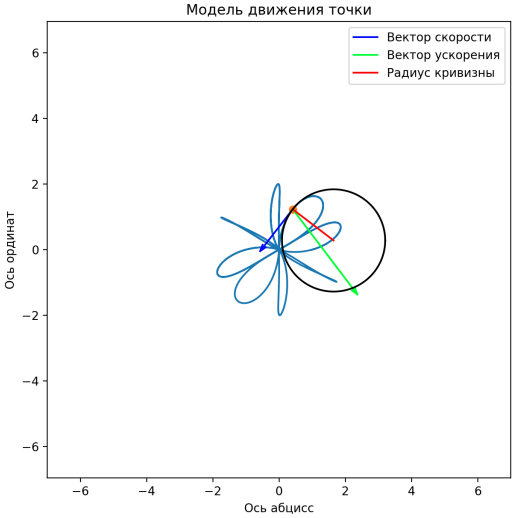
```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation
import sympy as sp


# Var 8: r(t) = 2 + sin(8t), phi(t) = t + 0.2*sin(6t)

# Функция поворота на угол альфа
def Rot2D(X, Y, Phi):
RotX = X * np.cos(Phi) - Y * np.sin(Phi)
RotY = X * np.sin(Phi) + Y * np.cos(Phi)
return RotX, RotY

t = sp.Symbol('t')

r = 1 + sp.sin(8 * t)
phi = t + 0.2 * sp.sin(6 * t)

# Переход из полярных координат в Декартовы координаты
x = r * sp.cos(phi)
y = r * sp.sin(phi)

# Вычисление скорости по x и y
Vx = sp.diff(x, t)
Vy = sp.diff(y, t)

# Вычисление ускорения по x и y
Wx = sp.diff(Vx, t)
Wy = sp.diff(Vy, t)

# Вычисление общей скорости
v = sp.sqrt(Vx ** 2 + Vy ** 2)

# Вычисление тангенциального ускорения
Wtan = sp.diff(v, t)
# Вычисление нормального ускорения
Wnorm = (sp.sqrt(Wx ** 2 + Wy ** 2 - Wtan ** 2))
R = v ** 2 / Wnorm

NORMx = (Wx - ((Vx / v) * Wtan)) / Wnorm
NORMy = (Wy - ((Vy / v) * Wtan)) / Wnorm
# Подстановка в функции числовых значений
F_x = sp.lambdify(t, x)
F_y = sp.lambdify(t, y)
```

```
F_Vx = sp.lambdify(t, Vx)
F_Vy = sp.lambdify(t, Vy)

F_Wx = sp.lambdify(t, Wx)
F_Wy = sp.lambdify(t, Wy)

F_R = sp.lambdify(t, R)

F_NORMx = sp.lambdify(t, NORMx)
F_NORMy = sp.lambdify(t, NORMy)

# Равномерное распределение по массиву 1001 чисел от 0 до 10.
t = np.linspace(0, 10, 1001)

# Присваивание переменным значения фукции
x = F_x(t)
y = F_y(t)

Vx = F_Vx(t)
Vy = F_Vy(t)

Wx = F_Wx(t)
Wy = F_Wy(t)

RO = F_R(t)

NORMx = F_NORMx(t)
NORMy = F_NORMy(t)

# Создать окно
fig = plt.figure(figsize=[7, 7])

# ax - окно с графиком
ax = fig.add_subplot(1, 1, 1)
ax.axis('equal')
ax.set_title("Модель движения точки")
ax.set_xlabel('Ось абцисс')
ax.set_ylabel('Ось ординат')
ax.set(xlim=[-7, 7], ylim=[-7, 7])

k_V = 0.2
k_W = 0.075

# Построение траектории
ax.plot(x, y)
# Построение точки
P = ax.plot(x[0], y[0], marker='o')[0]
# Построение векторов скорости, ускорения, радиуса-кривизны
V_line = ax.plot([x[0], x[0] + k_V * Vx[0]], [y[0], y[0] + k_V * Vy[0]], color=[0, 0, 1], label='Вектор
скорости')[0]
W_line = ax.plot([x[0], x[0] + k_W * Wx[0]], [y[0], y[0] + k_W * Wy[0]], color=[0, 1, 0], label='Вектор
ускорения')[0]
```

```
RO_Line = \
ax.plot([x[0], x[0] + RO[0] * NORMx[0]], [y[0], y[0] + RO[0] * NORMy[0]], color=[1, 0, 0], label='Радиус кривизны')[
0]
ax.legend()

Alpha_V = np.arctan2(Vy, Vx)
Alpha_W = np.arctan2(Wy, Wx)
a = 0.2
b = 0.06
# Построение стрелочек на концах векторах
x_arr = np.array([-a, 0, -a])
y_arr = np.array([b, 0, -b])
V_RotX, V_RotY = Rot2D(x_arr, y_arr, Alpha_V[0])
W_RotX, W_RotY = Rot2D(x_arr, y_arr, Alpha_W[0])
V_Arrow = ax.plot(x[0] + k_V * Vx[0] + V_RotX, y[0] + k_V * Vy[0] + V_RotY, color=[0, 0, 1])[0]
W_Arrow = ax.plot(x[0] + k_W * Wx[0] + W_RotX, y[0] + k_W * Wy[0] + W_RotY, color=[0, 1, 0])[0]

Phi1 = np.linspace(0, 6.28, 100)
Circ = ax.plot(x[0] + RO[0] * NORMx[0] * np.cos(Phi1), y[0] + RO[0] * NORMy[0] * np.sin(Phi1), color=[0, 0, 0])[0]

# Функция для анимации
def TheMagicOfTheMovement(i):
P.set_data(x[i], y[i])
V_line.set_data([x[i], x[i] + k_V * Vx[i]], [y[i], y[i] + k_V * Vy[i]])
W_line.set_data([x[i], x[i] + k_W * Wx[i]], [y[i], y[i] + k_W * Wy[i]])
RO_Line.set_data([x[i], x[i] + RO[i] * NORMx[i]], [y[i], y[i] + RO[i] * NORMy[i]])
Circ.set_data(x[i] + RO[i] * NORMx[i] + RO[i] * np.cos(Phi1), y[i] + RO[i] * NORMy[i] + RO[i] * np.sin(Phi1))
V_RotX, V_RotY = Rot2D(x_arr, y_arr, Alpha_V[i])
W_RotX, W_RotY = Rot2D(x_arr, y_arr, Alpha_W[i])
V_Arrow.set_data(x[i] + k_V * Vx[i] + V_RotX, y[i] + k_V * Vy[i] + V_RotY)
W_Arrow.set_data(x[i] + k_W * Wx[i] + W_RotX, y[i] + k_W * Wy[i] + W_RotY)
return [P, V_line, W_line, RO_Line, V_Arrow, W_Arrow]

kino = FuncAnimation(fig, TheMagicOfTheMovement, frames=len(t), interval=50)

plt.show()
```


Модель движения точки
Вектор скорости
Вектор ускорения
Радиус кривизны
Ось ординат
Ось абцисс


Модель движения точки
Вектор скорости
Вектор ускорения
Радиус кривизны
Ось ординат
Ось абцисс


Модель движения точки
Вектор скорости
Вектор ускорения
Радиус кривизны
Ось ординат
Ось абцисс