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 № 2

Выполнил(а) студент группы М8О-208Б-22

Былькова Кристина Алексеевна_____
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.

Текст программы:

```
import math
import sympy as s
import matplotlib.pyplot as plt
import numpy as n
from matplotlib.animation import FuncAnimation

# Var №2: r(t) = 1 + sin(5*t), phi(t) = t
# x = r * cos(phi)
# y = r * sin (phi)

t = s.Symbol('t')
# В полярных координатах
r = 1 + s.sin(5 * t)
phi = t
# Переход в декартовы координаты
x = r * s.cos(phi)
y = r * s.sin(phi)

# Скорость
Vx, Vy = s.diff(x), s.diff(y)
# Ускорение
Wx, Wy = s.diff(Vx, t), s.diff(Vy, t)
#Радиус кривизны
v = s.sqrt(Vx**2 + Vy**2)
Wt = s.diff(v) # тангенциальное ускорение
Wn = (s.sqrt(Wx**2 + Wy**2 - Wt**2)) # нормальное ускорение
R = v**2 / Wn # радиус кривизны
# Касательная к траектории
TAUx = Vx / v
TAUy = Vy / v
Wtx = TAUx * Wt
Wty = TAUy * Wt
Wnx = Wx - Wtx
Wny = Wy - Wty
# Нормаль
NORMx = Wnx / Wn
NORMy = Wny / Wn


step = 1800
T = n.linspace(0, 10, step)
X, Y = n.zeros_like(T), n.zeros_like(T)
VX, VY = n.zeros_like(T), n.zeros_like(T)
WX, WY = n.zeros_like(T), n.zeros_like(T)
RO = n.zeros_like(T)
NORMX, NORMY = n.zeros_like(T), n.zeros_like(T)
```

```
for i in n.arange(len(T)):
    X[i], Y[i] = s.Subs(x, t, T[i]), s.Subs(y, t, T[i])
    VX[i], VY[i] = s.Subs(Vx, t, T[i]), s.Subs(Vy, t, T[i])
    WX[i], WY[i]  = s.Subs(Wx, t, T[i]), s.Subs(Wy, t, T[i])
    RO[i] = s.Subs(R, t, T[i])
    NORMX[i], NORMY[i] = s.Subs(NORMx, t, T[i]), s.Subs(NORMy,
t, T[i])

fig = plt.figure()

axis = fig.add_subplot(1, 1, 1)
axis.axis('equal')
axis.set_title('Модель движения точки')
axis.set(xlim = [-3, 3], ylim = [-3, 3])
axis.plot(X,Y)

# Точка
Pnt = axis.plot(X[0], Y[0], 'magenta', marker = 'o')[0]
# Радиус-вектор
RLine = axis.plot([0, X[0]], [0, Y[0]], 'black', label =
'Радиус-вектор')[0]
# Вектор скорости
VLine = axis.plot([X[0], X[0] + VX[0]], [Y[0], Y[0] + VY[0]],
'red', label = 'Вектор скорости')[0]
# Вектор ускорения
WLine = axis.plot([X[0], X[0] + WX[0]], [Y[0], Y[0] + WY[0]],
'green', label = 'Вектор ускорения')[0]
# Вектор радиуса кривизны
ROLine = axis.plot([X[0], X[0] + RO[0] * NORMX[0]], [Y[0], Y[0]
+ RO[0] * NORMY[0]], 'orange', label = 'Радиус кривизны')[0]

axis.legend()

# Функция для отрисовки стрелки
def Vect_arrow(X, Y, ValX, ValY):
    a = 0.05
    b = 0.1
    Arx = n.array([-b, 0, -b])
    Ary = n.array([a, 0, -a])
    alp = math.atan2(ValY, ValX)
    RotArx = Arx * n.cos(alp) - Ary * n.sin(alp)
    RotAry = Arx * n.sin(alp) + Ary * n.cos(alp)

    Arx = X + ValX + RotArx
    Ary = Y + ValY + RotAry
    return Arx, Ary

# Стрелка радиус-вектора
RAx, RAy = Vect_arrow(0, 0, X[0], Y[0])
Varrow = axis.plot(RAx, RAy, 'black')[0]
# Стрелка вектора скорости
RVAx, RVAy = Vect_arrow(X[0], Y[0], VX[0], VY[0])
```

```
VVarrow = axis.plot(RVAx, RVAy, 'red')[0]
# Стрелка вектора ускорения
RWAx, RWAy = Vect_arrow(X[0], Y[0], WX[0], WY[0])
VWarrow = axis.plot(RWAx, RWAy, 'green')[0]
# Стрелка вектора радиуса кривизны
ROAx, ROAy = Vect_arrow(X[0], Y[0], RO[0] * NORMX[0], RO[0] *
NORMY[0])
VROarrow = axis.plot(ROAx, ROAy, 'orange')[0]

def anim(i):
    # Анимация точки
    Pnt.set_data(X[i], Y[i])
    # Анимация Радиус-вектора
    RLine.set_data([0, X[i]], [0, Y[i]])
    RAx, RAy = Vect_arrow(0, 0, X[i], Y[i])
    Varrow.set_data(RAx, RAy)
    # Анимация вектора скорости
    VLine.set_data([X[i], X[i] + VX[i]], [Y[i], Y[i] + VY[i]])
    RVAx, RVAy = Vect_arrow(X[i], Y[i], VX[i], VY[i])
    VVarrow.set_data(RVAx, RVAy)
    # Анимация вектора ускорения
    WLine.set_data([X[i], X[i] + WX[i]], [Y[i], Y[i] + WY[i]])
    RWAx, RWAy = Vect_arrow(X[i], Y[i], WX[i], WY[i])
    VWarrow.set_data(RWAx, RWAy)
    # Анимация вектора радиуса кривизны
    ROLine.set_data([X[i], X[i] + RO[i] * NORMX[i]], [Y[i], Y[i]
+ RO[i] * NORMY[i]])
    ROAx, ROAy = Vect_arrow(X[i], Y[i], RO[i] * NORMX[i], RO[i]
* NORMY[i])
    VROarrow.set_data(ROAx, ROAy)

an = FuncAnimation(fig, anim, frames = step, interval = 30)

fig.show()

plt.show()
```

Результат работы программы: