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 13**

Выполнил студент группы М8О-212Б-22

Корнев Максим Сергеевич\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

Проверил и принял

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

подпись, дата

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

Москва, 2023

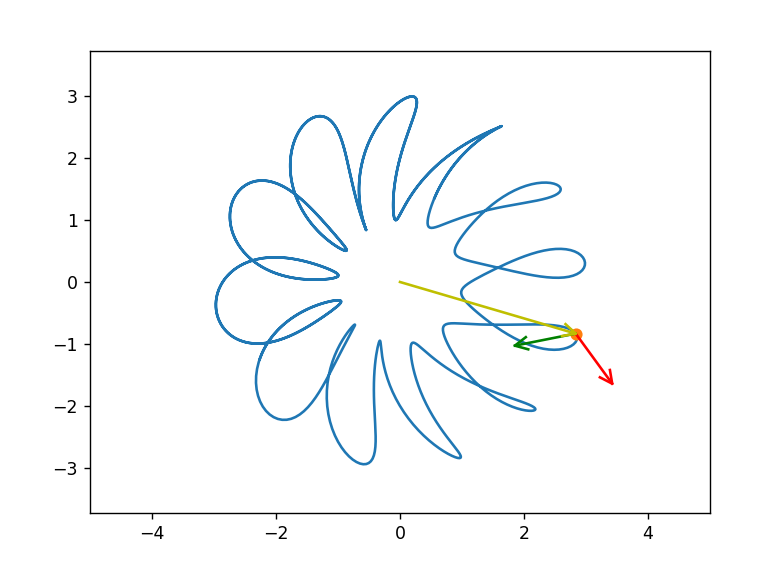
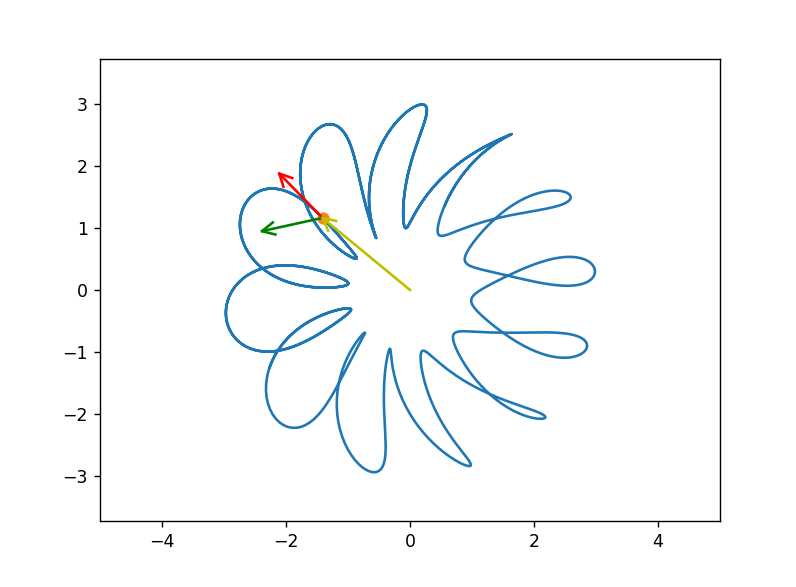
**Задание:**

Построить заданную траекторию и анимацию движения точки, а также отобразить стрелки скорости и ускорения, радиус - вектор.

**Вариант 13: r(t) = 2 + sin(12t), ф(t) = t + 0.2cos(13t)**

import numpy as np  
import matplotlib.pyplot as plt  
from matplotlib.animation import FuncAnimation  
import sympy as sp  
import math  
  
def anime(i):  
 P.set\_data(X[i], Y[i])  
 VVec.set\_data([X[i], X[i] + VX[i]], [Y[i], Y[i] + VY[i]])  
 RVecArrowX, RVecArrowY = rotation2D(ArrowX, ArrowY, math.atan2(VY[i], VX[i]))  
 VVecArrow.set\_data(RVecArrowX + X[i] + VX[i], RVecArrowY + Y[i] + VY[i])  
 AVec.set\_data([X[i], X[i] + AX[i]], [Y[i], Y[i] + AY[i]])  
 RVecArrowX\_A, RVecArrowY\_A = rotation2D(ArrowX, ArrowY, math.atan2(AY[i], AX[i]))  
 AVecArrow.set\_data(RVecArrowX\_A + X[i] + AX[i], RVecArrowY\_A + Y[i] + AY[i])  
 RVec.set\_data([0, X[i]], [0, Y[i]])  
 RVecArrowX\_R, RVecArrowY\_R = rotation2D(ArrowX, ArrowY, math.atan2(Y[i], X[i]))  
 RVecArrow.set\_data(RVecArrowX\_R + X[i], RVecArrowY\_R + Y[i])  
 return P, VVec, VVecArrow, AVec, AVecArrow, RVec, RVecArrow  
  
def rotation2D(x, y, a):  
 Rx = x \* np.cos(a) - y \* np.sin(a)  
 Ry = x \* np.sin(a) + y \* np.cos(a)  
 return Rx, Ry  
  
T = np.linspace(1, 10, 1000)  
t = sp.Symbol('t')  
  
r = 2 + sp.sin(12\*t) #сюда вставлять  
phi = t + 0.2 \* sp.cos(13 \* t) # и сюда  
  
X = np.zeros\_like(T)  
Y = np.zeros\_like(T)  
x = r \* sp.cos(phi)  
y = r \* sp.sin(phi)  
VX = np.zeros\_like(T)  
VY = np.zeros\_like(T)  
Vx = sp.diff(x, t)  
Vy = sp.diff(y, t)  
V = sp.sqrt(Vy\*\*2 + Vx\*\*2)  
VxN = Vx/V  
VyN = Vy/V  
AX = np.zeros\_like(T)  
AY = np.zeros\_like(T)  
Ax = sp.diff(Vx, t)  
Ay = sp.diff(Vy, t)  
A = sp.sqrt(Ax\*\*2 + Ay\*\*2)  
AxN = Ax/A  
AyN = Ay/A  
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])  
fig = plt.figure()  
ax1 = fig.add\_subplot(1, 1, 1)  
ax1.axis('equal')  
ax1.set(xlim = [-5, 5], ylim = [-5, 5])  
ax1.plot(X, Y)  
P, = ax1.plot(X[0], Y[0], marker = 'o')  
ArrowX = np.array([-0.2, 0, -0.2])  
ArrowY = np.array([0.1, 0, -0.1])  
VVec, = ax1.plot([X[0], X[0] + VX[0]], [Y[0], Y[0] + VY[0]], 'r')  
RVecArrowX, RVecArrowY = rotation2D(ArrowX, ArrowY, math.atan2(VY[0], VX[0]))  
VVecArrow, = ax1.plot(RVecArrowX + VX[0] + X[0], RVecArrowY + VY[0] + Y[0], 'r')  
RVec, = ax1.plot([0, X[0]], [0, Y[0]], 'y')  
RVecArrowX\_R, RVecArrowY\_R = rotation2D(ArrowX, ArrowY, math.atan2(Y[0], X[0]))  
RVecArrow, = ax1.plot(RVecArrowX\_R + VX[0] + X[0], RVecArrowY\_R + VY[0] + Y[0], 'y')  
AVec, = ax1.plot([X[0], X[0] + AX[0]], [Y[0], Y[0] + AY[0]], 'g')  
RVecArrowX\_A, RVecArrowY\_A = rotation2D(ArrowX, ArrowY, math.atan2(AY[0], AX[0]))  
AVecArrow, = ax1.plot(RVecArrowX\_A + X[0], RVecArrowY\_A + Y[0], 'g')  
anim = FuncAnimation(fig, anime, frames = 1000, interval = 30, blit = True)  
plt.show()

**Результат работы программы:**



**Вывод:**

Программирование на Python с применением библиотек для научных вычислений обеспечивает удобное моделирование и визуализацию разнообразных физических явлений. Создание анимации движения материальной точки становится полезным инструментом для обучения и постижения физических законов, поскольку она позволяет наглядно демонстрировать их действие. Использование библиотеки numpy для вычислений и matplotlib для визуализации открывает возможность создавать динамические анимации, наглядно отражающие физические процессы. Визуализация движения материальной точки эффективно демонстрирует изменение ее координат, скорости и ускорения в течение времени.