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 №14**

Выполнил(а) студент группы М8О-207Б-22

Кострюков Евгений Сергеевич\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

Проверил и принял

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

подпись, дата

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

Москва, 2023

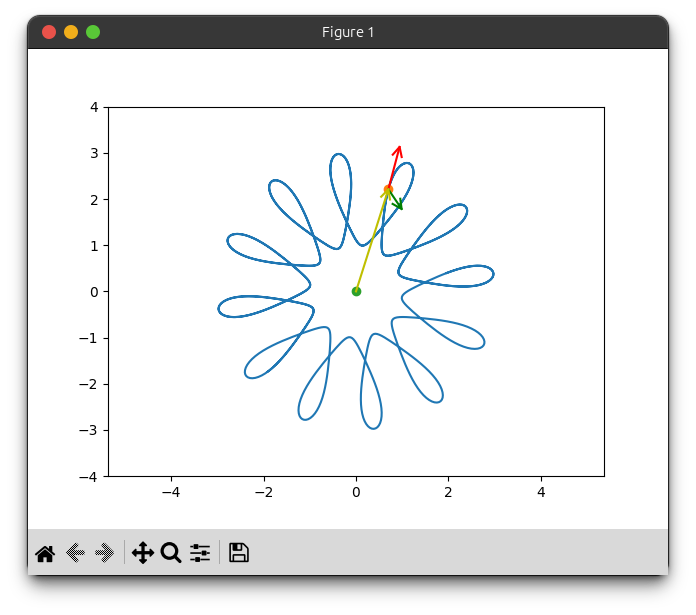
*Задание:* построить заданную траекторию, запустить анимацию движения точки, построить стрелки радиус-вектора, вектора скорости, вектора ускорения и радиуса кривизны.

*1)Условия задачи 6 варианта:*

r(t) = 2+sin(12\*t)

φ(t) = t + 0.2\*cos(12\*t)

*2)Рисунок получившейся физической модели:*



Вектор скорости — красным цветом

Вектор ускорения — зелёным цветом

Радиус-вектор — жёлтым цветом

*3)Код программы:*

import numpy as np

import sympy as sp

import math

import matplotlib.pyplot as plt

from matplotlib.animation import FuncAnimation

# Поворот в двумерной плоскости

def Rot2D(X, Y, Alpha):

# Координаты после поворота

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, 1000) # Равномерно распределяем 1000 элементов на участке (0; 10)

t = sp.Symbol('t') # Cимвольная переменная

scale = 12 # Константа для масштабирования векторов

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

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

# Функции

x = r \* sp.cos(phi)

y = r \* sp.sin(phi)

Vx = sp.diff(x, t)/scale

Vy = sp.diff(y, t)/scale

Ax = sp.diff(Vx, t)/scale

Ay = sp.diff(Vy, t)/scale

# Создаём нулевые массивы типа данных T

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)

# Обновляем нулевые массивы

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

# Функция, заменяемый элемент, на что меняем

X[i] = sp.Subs(x, t, T[i])

Y[i] = sp.Subs(y, t, T[i])

VX[i] = sp.Subs(Vx, t, T[i])

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

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

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

fig = plt.figure() # Создаём окно

ax1 = fig.add\_subplot(1, 1, 1) # Создаём поле для отрисовки. *Подграфик имеет одну строку, один столбец и индекс 1, означающий, что график занимает всю доступную область фигуры*

ax1.axis('equal') # Делаем единичные отрезки осей равными

ax1.set(xlim=[-4, 4], ylim=[-4, 4]) # Пределы по X и Y

ax1.plot(X, Y) # Рисуем траекторию, соединяя точки

# Создаём объекты с их начальным положением

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

Pstart, = ax1.plot(0, 0, marker='o')

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

ALine, = ax1.plot([X[0], X[0]+AX[0]], [Y[0], Y[0]+AY[0]], 'g') # Вектор ускорения

RVLine, = ax1.plot([0, X[0]], [0, Y[0]], 'y') # Радиус-вектор

# Координаты для стрелки в начальный момент времени(массив из 3 элементов)

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

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

# Координаты для стрелки после поворота

RArrowX, RArrowY = Rot2D(ArrowX, ArrowY, math.atan2(VY[0], VX[0]))

RArrowAX, RArrowAY = Rot2D(ArrowX, ArrowY, math.atan2(AY[0], AX[0]))

RArrowRVX, RArrowRVY = Rot2D(ArrowX, ArrowY, math.atan2(Y[0], X[0]))

# Отрисовываем стрелку на конце нужного вектора

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

AArrow, = ax1.plot(RArrowAX+X[0]+AX[0], RArrowAY+Y[0]+AY[0], 'g')

RVArrow, = ax1.plot(RArrowRVX+X[0], RArrowRVY+Y[0], 'y')

# Обновление текущего кадра

def anima(i):

Pstart.set\_data(0,0)

# Изменяем нулевые элементы на i-ые:

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

# Скорость

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

RArrowX, RArrowY = Rot2D(ArrowX, ArrowY,math.atan2(VY[i], VX[i])) # Поворот стрелки

VArrow.set\_data(RArrowX+X[i]+VX[i], RArrowY+Y[i]+VY[i]) # Изменение координат стрелки

# Ускорение

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

RArrowAX, RArrowAY = Rot2D(ArrowX, ArrowY,math.atan2(AY[i], AX[i]))

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

# Радиус-вектор

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

RArrowRVX,RArrowRVY = Rot2D(ArrowX, ArrowY, math.atan2(Y[i], X[i]))

RVArrow.set\_data(RArrowRVX+X[i], RArrowRVY+Y[i])

return P ,VLine ,VArrow ,ALine ,AArrow, RVLine, RVArrow

anim = FuncAnimation(fig, anima, frames=1000, interval=50, repeat=False) # Создаём анимацию

plt.show() # Отрисовываем

*4) Пояснение к коду:*

*1. Подключаем библиотеки для работы с математическими формулами и графическим интерфейсом для отображения физических моделей.*

*2. Записываем условие задачи и переводим его в декартовую систему координат.*

*3. Записываем функции скорости и ускорения, с помощью которых затем заполняем массивы для хранения векторов скорости и ускорения в каждый момент времени.*

*4. Создаем окно для построения графиков с помощью функции plt.figure().*

*5. Добавляем подграфик с помощью метода add\_subplot().*

*6. С помощью метода axis("equal") делаем единичные отрезки осей равными.*

*7. Устанавливаем пределы для осей x и y с помощью метода set(), чтобы они лежали в границах от -4 до 4.*

*8. Создаём объекты (точку P, начальную точку Pstart, линию радиус вектора RVline, линию скорости VLine и линию ускорения точки ALine) на подграфике.*

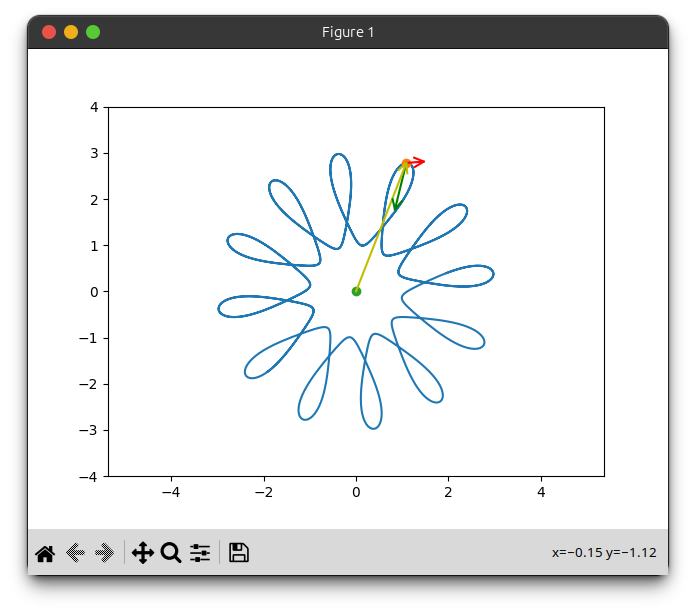
*9. Создаём массивы ArrowX и ArrowY с координатами для стрелок к этим объектам.*

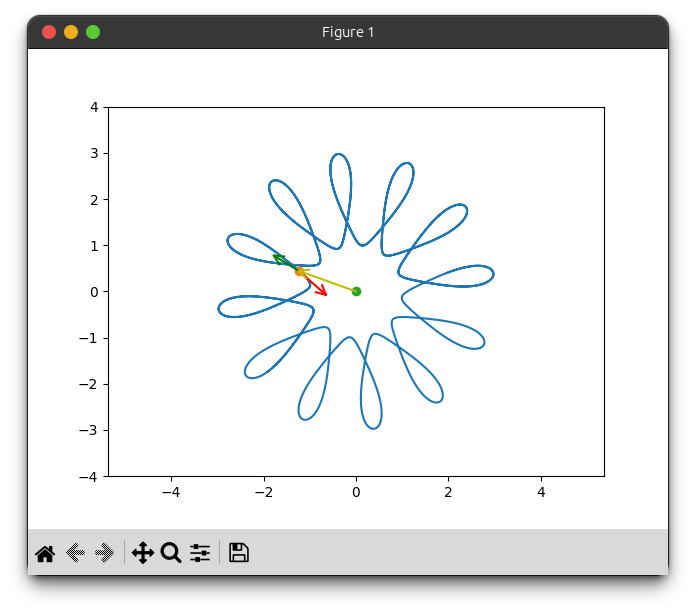
*10. Обновляем текущие кадры в функции anima(i), изменяя нулевые элементы на i-ые.*

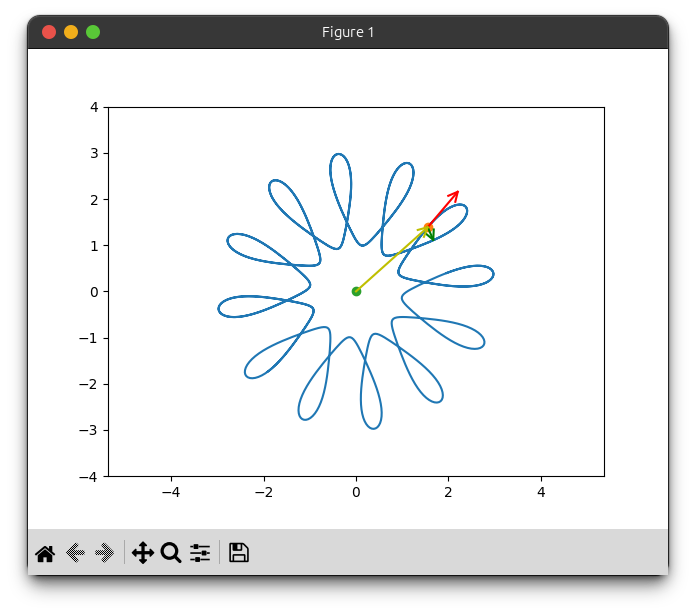
*11. Функция Rot2D(X, Y, Alpha) через матрицу поворота высчитывает координаты стрелок в соответствии с их углом поворота в каждый момент времени.*

*12. Создаём анимацию с помощью функции FuncAnimation().*

*5) Результат работы программы*







*6) Вывод:*

Лабораторная работа успешно сделана. Мы построили траекторию движения точки, после чего запустили анимацию движения точки по заданной траектории на языке программирования Python. Кроме того, у вектора скорости, вектора ускорения и радиус-вектора были реализованы стрелки, указывающие направление, что позволило детально показать характеристики движения точки. Благодаря данной работе я смог лучше разобраться в основных концепциях теоретической физики, которые связаны с движением объектов в пространстве, а также визуализировать их, используя современный язык программирования Python.