ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

**ОТЧЕТ**

**О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

**«АНИМАЦИЯ СИСТЕМЫ»**

**ПО ДИСЦИПЛИНЕ «ОСНОВЫ КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ ДИНАМИЧЕСКИХ СИСТЕМ»**

**ВАРИАНТ ЗАДАНИЯ №23**

Выполнил(а) студент группы М8О-201Б-23

Терентьев М.А.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

Проверил и принял

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

подпись, дата

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

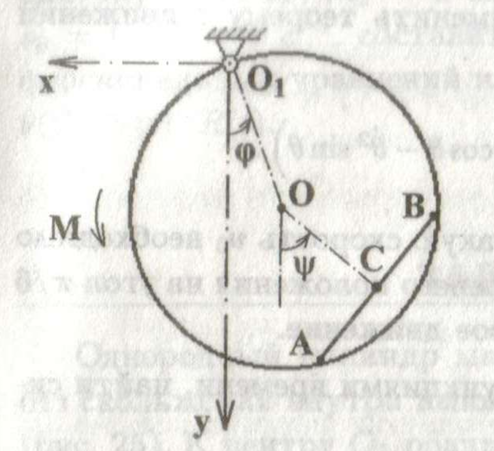
Москва, 2024

**Вариант №20**

**Задание:**

Реализовать анимацию движения механической системы используя язык программирования Python.

**Механическая система:**

****

**Код программы:**

import numpy as np

import matplotlib.pyplot as plt

from matplotlib.animation import FuncAnimation

import sympy as sp

import math

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

R = 5

l = 3

r = math.sqrt(R \* R - l \* l)

t = sp.Symbol('t')

phi = 0.3 \* t

ksi = 0.5 \* t

x\_O = -R \* sp.sin(phi)

y\_O = R \* sp.cos(phi)

x\_C = x\_O - r \* sp.sin(ksi)

y\_C = y\_O + r \* sp.cos(ksi)

x\_rel = -r \* sp.sin(ksi)

y\_rel = r \* sp.cos(ksi)

Vx\_C = sp.diff(x\_C, t)

Vy\_C = sp.diff(y\_C, t)

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

Ax\_C = sp.diff(x\_C, t, 2)

Ay\_C = sp.diff(y\_C, t, 2)

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

T = np.linspace(0, 45, 1000)

X\_O\_def = sp.lambdify(t, x\_O, modules='numpy')

Y\_O\_def = sp.lambdify(t, y\_O, modules='numpy')

X\_C\_def = sp.lambdify(t, x\_C, modules='numpy')

Y\_C\_def = sp.lambdify(t, y\_C, modules='numpy')

X\_REL\_def = sp.lambdify(t, x\_rel, modules='numpy')

Y\_REL\_def = sp.lambdify(t, y\_rel, modules='numpy')

X\_O = X\_O\_def(T)

Y\_O = Y\_O\_def(T)

X\_C = X\_C\_def(T)

Y\_C = Y\_C\_def(T)

X\_REL = X\_REL\_def(T)

Y\_REL = Y\_REL\_def(T)

fig = plt.figure(figsize=(10, 8))

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

ax1.set\_aspect('equal')

ax1.set\_xlim([-10, 10])

ax1.set\_ylim([-20, 20])

ax1.set\_xlabel('Ось x')

ax1.set\_ylabel('Ось y')

ax1.invert\_xaxis()

ax1.invert\_yaxis()

ax1.grid(True)

PointO1, = ax1.plot([0], [0], 'bo')

Circ\_Angle = np.linspace(0, 2 \* np.pi, 100)

Circ, = ax1.plot(X\_O[0] + R \* np.cos(Circ\_Angle), Y\_O[0] +

R \* np.sin(Circ\_Angle), 'g')

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

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

R\_Stick\_ArrowX, R\_Stick\_ArrowY = Rot2D(

ArrowX, ArrowY, math.atan2(Y\_REL[0], X\_REL[0]))

Stick\_Arrow, = ax1.plot(

R\_Stick\_ArrowX + X\_C[0], R\_Stick\_ArrowY + Y\_C[0], 'r-')

O1O, = ax1.plot([0, X\_O[0]], [0, Y\_O[0]], 'b:')

OC, = ax1.plot([X\_O[0], X\_C[0]], [Y\_O[0], Y\_C[0]], 'b:')

def anima(i):

O1O.set\_data([0, X\_O[i]], [0, Y\_O[i]])

OC.set\_data([X\_O[i], X\_C[i]], [Y\_O[i], Y\_C[i]])

Circ.set\_data(X\_O[i] + R \* np.cos(Circ\_Angle),

Y\_O[i] + R \* np.sin(Circ\_Angle))

angle = math.atan2(Y\_REL[i], X\_REL[i])

R\_Stick\_ArrowX, R\_Stick\_ArrowY = Rot2D(ArrowX, ArrowY, angle)

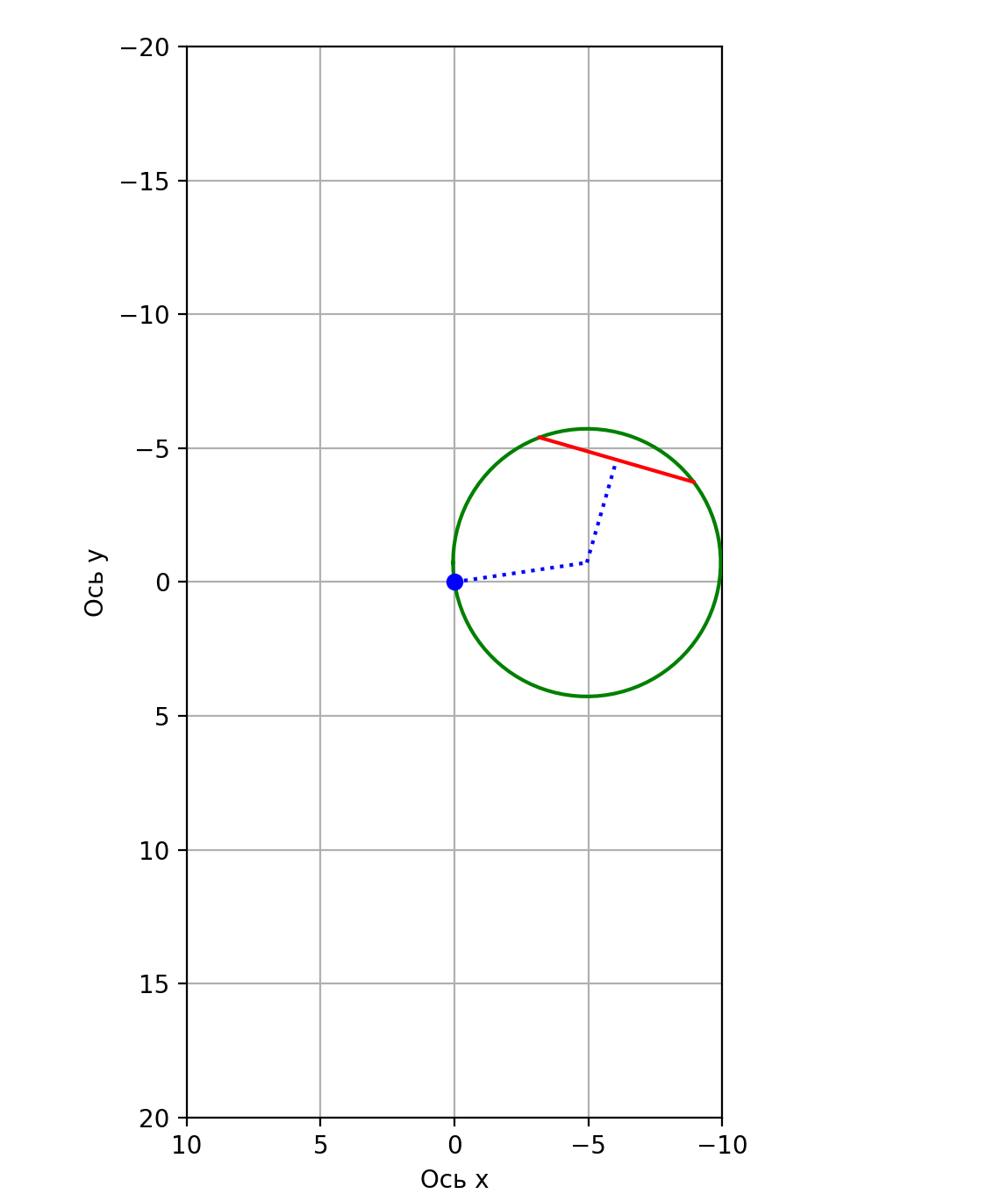
Stick\_Arrow.set\_data(R\_Stick\_ArrowX + X\_C[i], R\_Stick\_ArrowY + Y\_C[i])

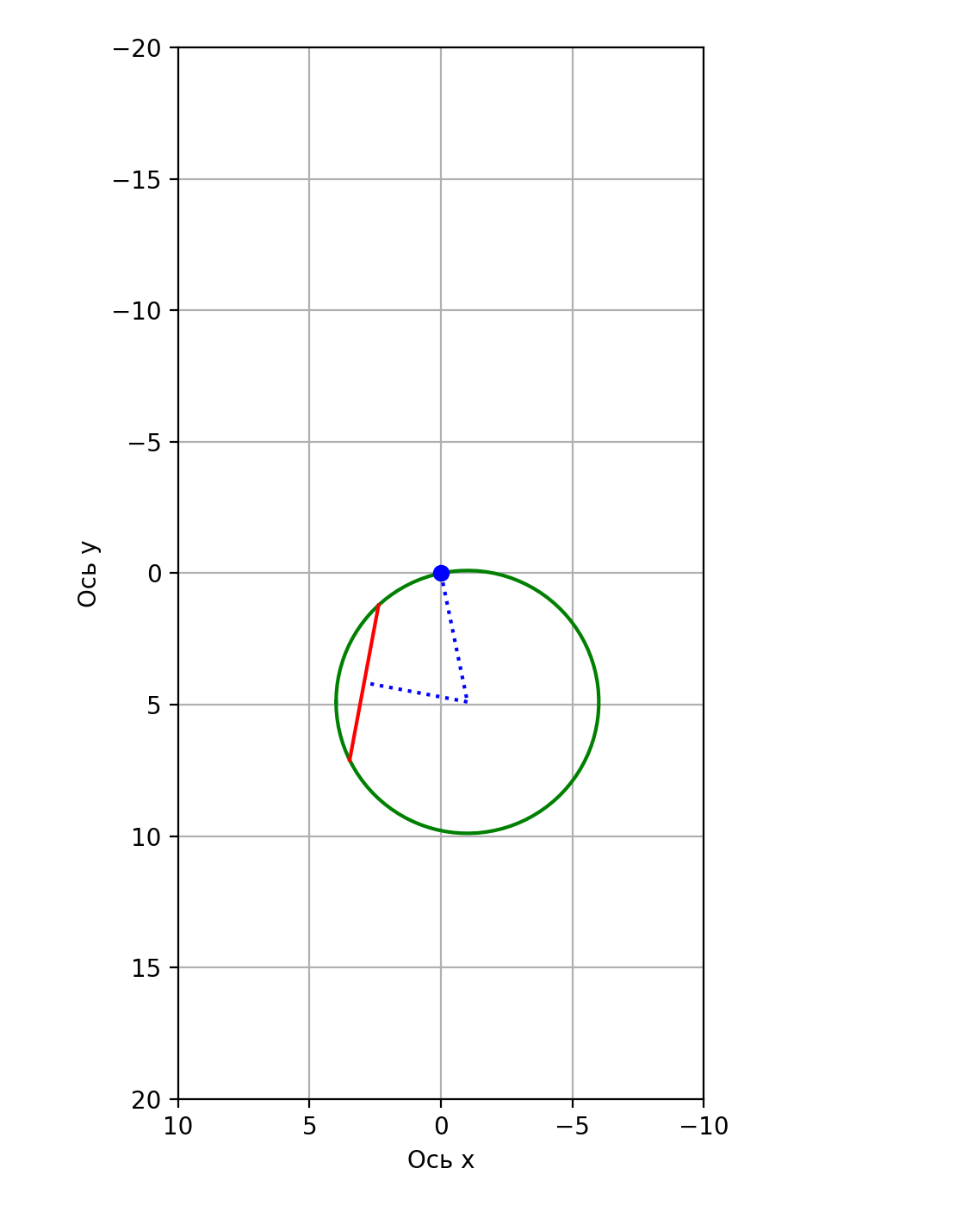
return O1O, OC, Circ, Stick\_Arrow, PointO1

anim = FuncAnimation(fig, anima, frames=len(T), interval=20, blit=True)

plt.show()

**Работа программы:**





**Вывод:**

В ходе лабораторной работы была создана анимация механической системы с использованием библиотек numpy и matplotlib. Шаги включали в себя определение параметров системы, расчет координат точек, и создание анимации. Комментарии были добавлены для понимания кода, и использовались различные параметры для улучшения визуального восприятия. Также рассматривались вопросы управления скоростью анимации и решения проблем с её остановкой.

Начало формы