ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

**ОТЧЕТ**

**О ВЫПЛОНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

**«ДИНАМИКА СИСТЕМЫ»**

**ПО ДИСЦИПЛИНЕ «ТЕОРЕТИЧЕСКАЯ МЕХАНИКА И ОСНОВЫ КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ»**

**ВАРИАНТ ЗАДАНИЯ №13**

Выполнил студент группы М8О-208Б-22

Григорьев Т.А.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

Проверил и принял

Авдюшкин А.Н.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

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

Москва, 2023

*Задание:* проинтегрировать систему дифференциальных уравнений движения системы с двумя степенями свободы с помощью средств Python. Построить анимацию движения система, а также графики законов движения системы и указанных в задании реакций для разных случаев системы.

*Код:*

import math

import numpy as np

import sympy

import matplotlib.pyplot as plt

from scipy.integrate import odeint

from matplotlib.animation import FuncAnimation

def SystDiffEq(y, t, m1, m2, a, b, g):

# y = [phi, psi, phi', psi'] -> dy = [phi', psi', phi'', psi'']

dy = np.zeros\_like(y)

dy[0] = y[2]

dy[1] = y[3]

phi = y[0]

psi = y[1]

dphi = y[2]

dpsi = y[3]

# a11 \* phi'' + a12 \* psi'' = b1

# a21 \* phi'' + a22 \* psi'' = b2

a11 = (m1 + m2)\*2\*a

a12 = -m2\*b\*(1 - np.cos(psi - phi))

b1 = -(m1 + m2)\*g\*np.sin(phi) + m2\*b\*np.sin(psi - phi)\*dpsi\*\*2

a21 = a\*(1 - np.cos(psi - phi))

a22 = -2\*b

b2 = g\*np.sin(psi) + a\*np.sin(psi - phi)\*dphi\*\*2

detA = a11 \* a22 - a12 \* a21

detA1 = b1 \* a22 - a12 \* b2

detA2 = a11 \* b2 - b1 \* a21

dy[2] = detA1 / detA

dy[3] = detA2 / detA

return dy

# Дано:

g = 9.8

m1 = 4

m2 = 2

r1 = 1

r2 = 0.125

r3 = 0.05

a = r1 - r3

b = r1 - r2

t0 = 0

phi0 = np.pi / 6

psi0 = np.pi / 3

dphi0 = 0

dpsi0 = np.pi / 3

# Задаю функции phi(t) и psi(t)

step = 1000

t = np.linspace(0, 10, step)

y0 = np.array([phi0, psi0, dphi0, dpsi0])

Y = odeint(SystDiffEq, y0, t, (m1, m2, a, b, g))

phi = Y[:,0]

psi = Y[:,1]

dphi = Y[:,2]

dpsi = Y[:,3]

ddphi = np.zeros\_like(t)

ddpsi = np.zeros\_like(t)

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

ddphi[i], ddpsi[i] = SystDiffEq(Y[i], t[i], m1, m2, a, b, g)[2:]

# Задаю функции N1 и Fft1 - реакции опоры и трения

N1 = (m1 + m2)\*(g\*np.cos(phi) + a\*dphi\*\*2) +\

m2\*b\*(ddpsi\*np.sin(phi - psi) + dpsi\*\*2\*np.cos(phi - psi))

Ffr1 = (m1 + m2)\*(g\*np.sin(phi) + a\*ddphi) +\

m2\*b\*(ddpsi\*np.cos(psi - phi) - dpsi\*\*2\*np.sin(psi - phi))

fgrt = plt.figure()

phiplt = fgrt.add\_subplot(4, 1, 1)

plt.title("phi(t)")

phiplt.plot(t, phi, color = 'r')

psiplt = fgrt.add\_subplot(4, 1, 2)

plt.title("psi(t)")

psiplt.plot(t, psi)

n1plt = fgrt.add\_subplot(4, 1, 3)

plt.title("N1(t)")

n1plt.plot(t, N1)

ffr1plt = fgrt.add\_subplot(4, 1, 4)

plt.title("Ffr1(t)")

ffr1plt.plot(t, Ffr1)

fgrt.show()

# Анимация

fig = plt.figure()

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

gr.axis('equal')

def rotation2D(x, y, angle):

Rx = x \* np.cos(angle) - y \* np.sin(angle)

Ry = x \* np.sin(angle) + y \* np.cos(angle)

return Rx, Ry

Ox, Oy = 0, 1

pO = plt.Circle((Ox, Oy), r3, color='black')

C1x = np.linspace(0, 10, step)

C1y = np.linspace(0, 10, step)

for i in range(len(phi)):

Rx, Ry = rotation2D(0, -a, phi[i])

C1x[i] = Ox + Rx

C1y[i] = Oy + Ry

C2x = np.linspace(0, 10, step)

C2y = np.linspace(0, 10, step)

for i in range(len(phi)):

Rx, Ry = rotation2D(0, -b, psi[i])

C2x[i] = C1x[i] + Rx

C2y[i] = C1y[i] + Ry

gr.add\_patch(pO)

A1 = plt.Circle((C1x[0], C1y[0]), r1, color='black', fill=False)

A2 = plt.Circle((C2x[0], C2y[0]), r2, color='black', fill=False)

gr.add\_patch(A1)

gr.add\_patch(A2)

def animate(i):

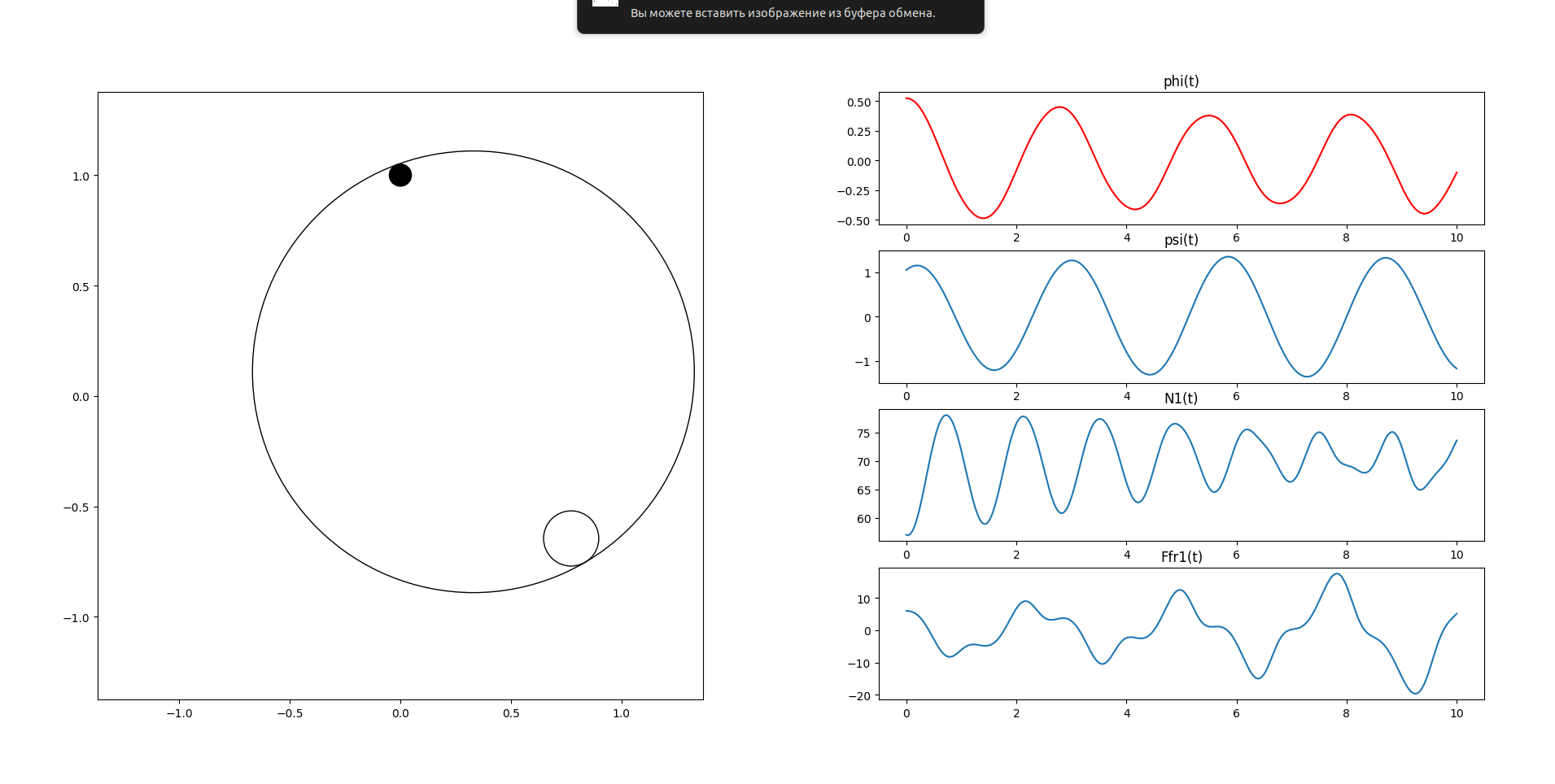
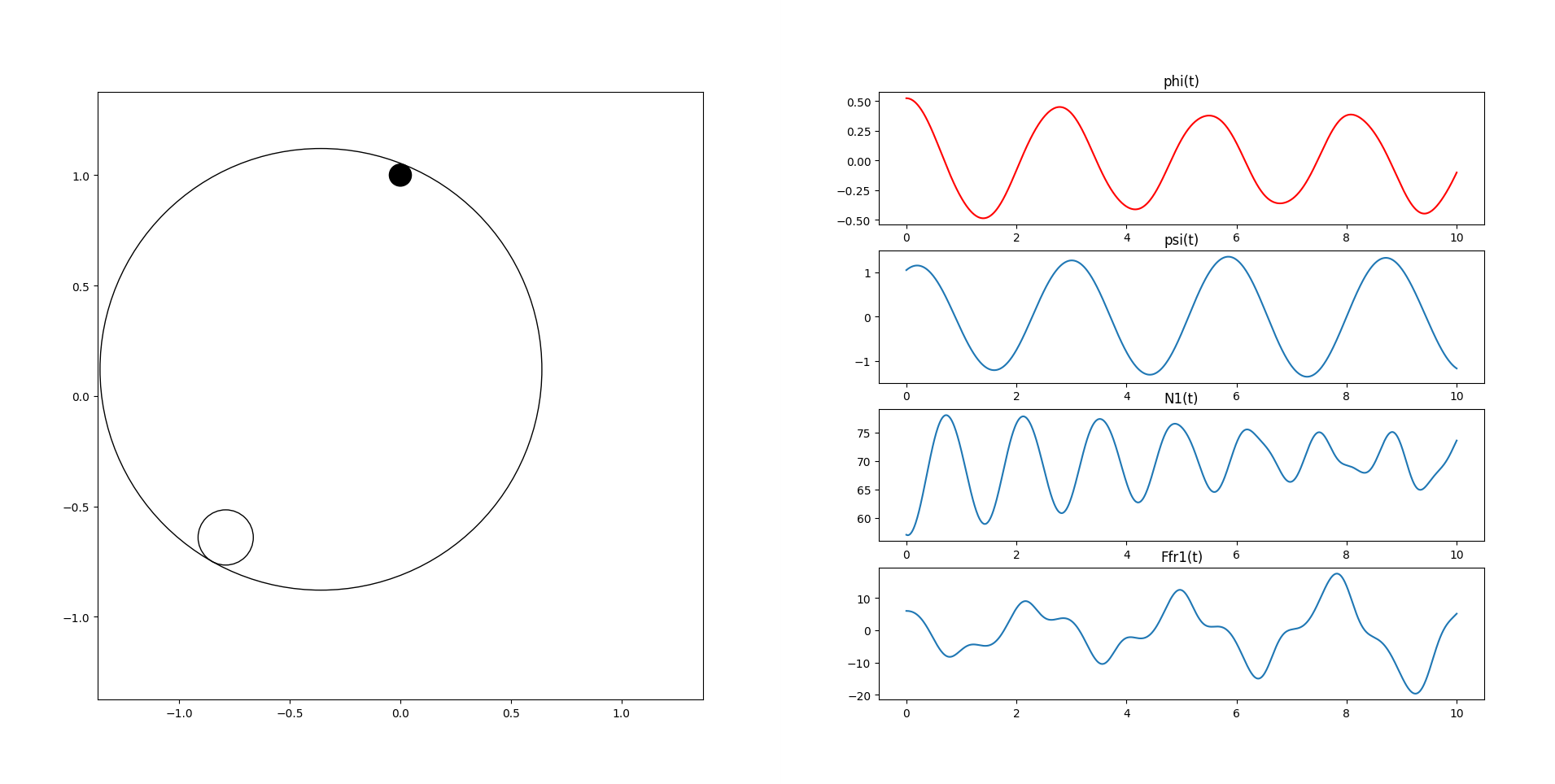
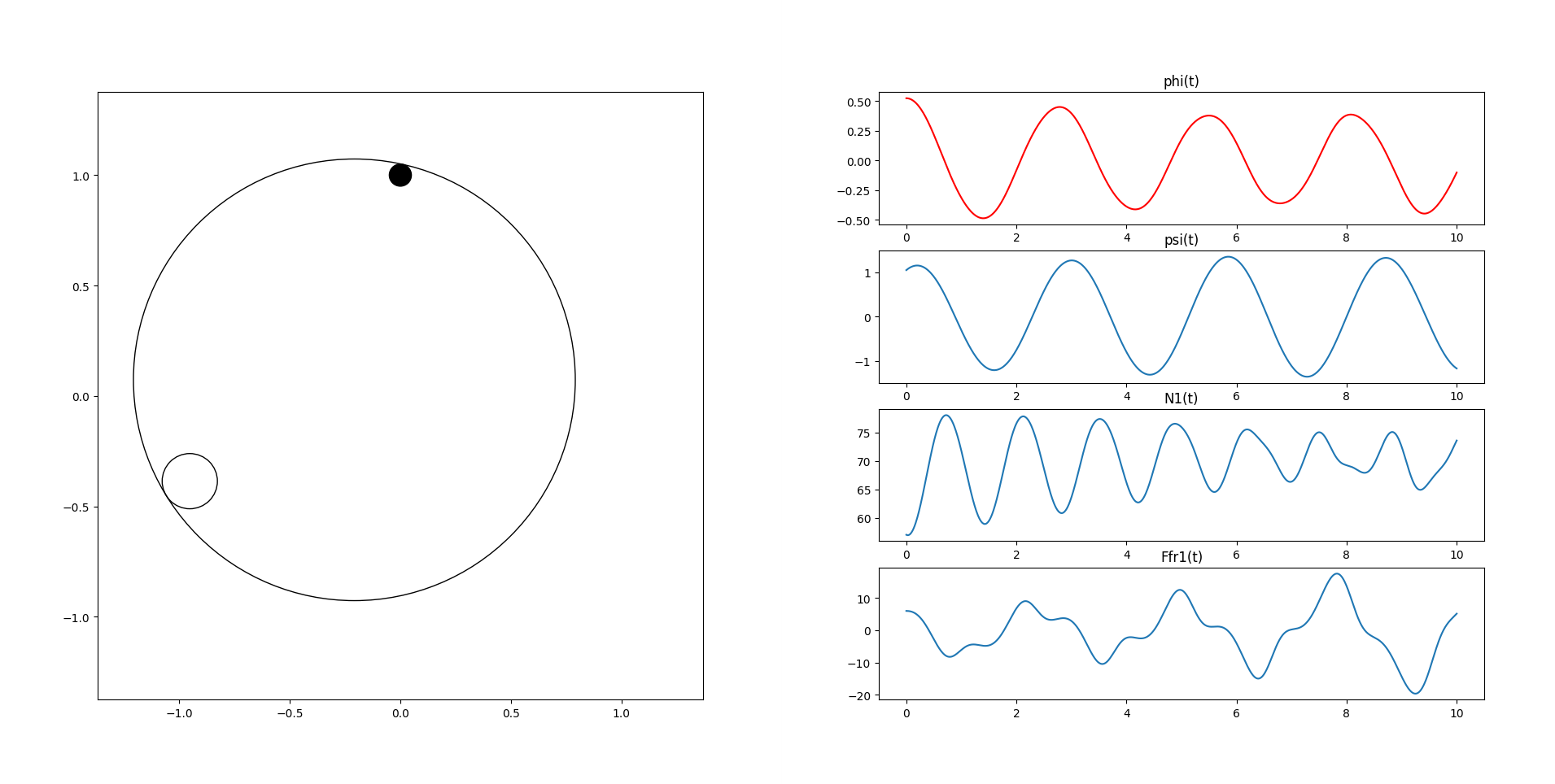
A1.center = (C1x[i], C1y[i])

A2.center = (C2x[i], C2y[i])

gr.set(xlim=[-2, 2], ylim=[-2, 2])

anim = FuncAnimation(fig, animate, frames = step, interval = 1)

plt.show()



*Скриншот анимации:*