­­­­ Московский авиационный институт

(национальный исследовательский университет)

Институт № 8 «Информационные технологии и прикладная математика»

**Лабораторная работа №4**

**по курсу «Теоретическая механика и компьютерное моделирование»**

**Малые колебания**

Выполнил студент группы М8О-207Б-20

Куценко Борис Дмитриевич

Преподаватель: Чекина Евгения Алексеевна

Оценка:

Дата: 27.12.2021

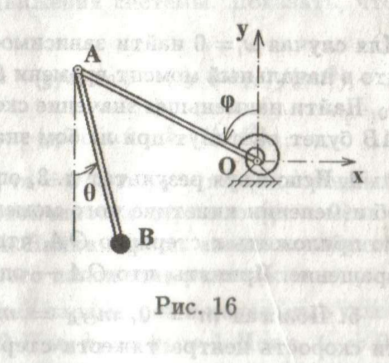
Москва, 2021

**Вариант №«16»**

**Задание:**

Реализовать анимацию движения механической системы в среде Python на основе уравнений Лагранжа 2-го рода для колебаний.

**Механическая система:**

****

**Текст программы:**

**import numpy as np**

**import matplotlib.pyplot as plt**

**from matplotlib.animation import FuncAnimation**

**import matplotlib.patches as pat**

**import sympy as sp**

**from scipy.integrate import odeint**

**import math**

**def form2(y, t, fomega\_phi):**

**y1,y2 = y**

**dydt = [y2, fomega\_phi(y1,y2)]**

**return dydt**

**Steps = 960**

**l = OA = AB = 1**

**R\_Circle = l / 10**

**k1 = 0**

**k2 = 0**

**m = 0**

**m\_oa = 5**

**g = 9.81**

**c = m\_oa \* g \* l \* 2**

**t\_fin = 10**

**phi0 = 0.2**

**thetta0 = -3.1428/3**

**dphi0 = 0**

**dthetta0 = 0**

**t = sp.Symbol('t')**

**phi = sp.Function('phi')(t)**

**thetta = 0;**

**omega\_phi = sp.Function('omega\_phi')(t)**

**omega\_thetta = 0;**

**#1 defining the kinetic energy**

**K = m\_oa \* (omega\_phi \* l/2) \* (omega\_phi \* l/2) / 2 + m\_oa \* 1/3 \* l/2 \* l/2 \* omega\_phi \* omega\_phi / 2**

**#2 defining the potential energy**

**Pmg = m\_oa \* g \* l \* (sp.cos(phi)) / 2**

**P2 = c \* phi \* phi / 2**

**P = Pmg + P2**

**Q1 = - k1 \* omega\_phi**

**#Lagrange function**

**L = K - P**

**#equations**

**ur1 = sp.diff(sp.diff(L,omega\_phi),t)-sp.diff(L,phi) - Q1**

**a11 = ur1.coeff(sp.diff(omega\_phi, t), 1)**

**b1 = -ur1.coeff(sp.diff(omega\_phi, t), 0).subs(sp.diff(phi,t), omega\_phi)**

**print(b1)**

**domega\_phidt = b1/a11**

**Time = np.linspace(0, t\_fin, Steps)**

**fomega\_phi = sp.lambdify([phi, omega\_phi], domega\_phidt, "numpy")**

**y0 = [phi0, dphi0]**

**sol = odeint(form2, y0, Time, args =(fomega\_phi,))**

**Phi = sol[:, 0]**

**Omega\_phi = sol[:, 1]**

**T = 2 \* 3.14 \* sp.sqrt(m\*l\*l/(3\*(c-m\*g\*l)))**

**X\_A = sp.lambdify(phi, l \* sp.cos(phi + 3.14/2))**

**Y\_A = sp.lambdify(phi, l \* sp.sin(phi + 3.14/2))**

**X\_B = sp.lambdify(phi, l \* sp.cos(phi + 3.14/2))**

**Y\_B = sp.lambdify(phi, l \* sp.sin(phi + 3.14/2) - l )**

**XA = X\_A(sol[:, 0])**

**YA = Y\_A(sol[:, 0])**

**XB = X\_B(sol[:, 0])**

**YB = Y\_B(sol[:, 0])**

**Nv= 3**

**R1 = 0.001**

**R2 = l / 12**

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

**ax = fig.add\_subplot(1,1,1)**

**ax.axis('equal')**

**ax.set(xlim=[-l \* 2 - R\_Circle - l/10, l \*2 + R\_Circle + l/10], ylim=[-l\*2 - R\_Circle - l/10,l\*2 + R\_Circle + l/10])**

**alpha = np.linspace(0, Nv\*6.283+Phi[0]+ 3.14/2, 100)**

**X\_SpiralSpr = -(R1 + alpha \* (R2 - R1) / alpha[-1]) \* np.sin(alpha)**

**Y\_SpiralSpr = (R1 + alpha \* (R2 - R1) / alpha[-1]) \* np.cos(alpha)**

**beta = np.linspace(0, 2\*math.pi, 100)**

**X\_Circle = R\_Circle \* np.cos(beta)**

**Y\_Circle = R\_Circle \* np.sin(beta)**

**fig\_for\_graphs = plt.figure(figsize=[13, 7])**

**ax\_for\_graphs = fig\_for\_graphs.add\_subplot(1, 2, 1)**

**ax\_for\_graphs.plot(Time, Phi, color='blue')**

**ax\_for\_graphs.set\_title("Phi(t)")**

**ax\_for\_graphs.set(xlim=[0, t\_fin])**

**ax\_for\_graphs.grid(True)**

**ax\_for\_graphs = fig\_for\_graphs.add\_subplot(1, 2, 2)**

**ax\_for\_graphs.plot(Time, Omega\_phi, color='red')**

**ax\_for\_graphs.set\_title('Phi\'(t)')**

**ax\_for\_graphs.set(xlim=[0, t\_fin])**

**ax\_for\_graphs.grid(True)**

**ax\_for\_graphs.set(xlim=[0, 10])**

**ax\_for\_graphs.grid(True)**

**OX = ax.plot([-l \* 2, l\*2], [-l/10,-l/10], 'black', linestyle = '--')**

**Draw\_Spring = ax.plot(X\_SpiralSpr, Y\_SpiralSpr, color='#666666')[0]**

**Draw\_OA=ax.plot([0, XA[0]], [0, YA[0]], color='#808080')[0]**

**Draw\_AB=ax.plot([XA[0], XB[0]], [YA[0], YB[0]], color='#808080' )[0]**

**PointB = ax.plot(XB[0], YB[0])[0]**

**PointA = ax.plot(XA[0], YA[0], color='#a0a0a0', marker='o')[0]**

**Draw\_Circle = ax.plot(X\_Circle + XB[0], Y\_Circle + YB[0], color='black', linewidth=1)[0]**

**triangle = pat.Polygon([(0,0), (-l/10, -l/10), (l/10, -l/10)], color='#d3d3d3')**

**ax.add\_patch(triangle)**

**def update(i):**

**PointB.set\_data(XB[i],YB[i])**

**Draw\_OA.set\_data([0, XA[i]], [0, YA[i]])**

**PointA.set\_data(XA[i], YA[i])**

**Draw\_AB.set\_data([XA[i], XB[i]], [YA[i], YB[i]])**

**Draw\_Circle.set\_data(X\_Circle + XB[i], Y\_Circle + YB[i])**

**alpha = np.linspace(0, Nv\*6.28+Phi[i] + 3.14/2, 100)**

**X\_SpiralSpr = -(R1 + alpha \* (R2 - R1) / alpha[-1]) \* np.sin(alpha - 1.57)**

**Y\_SpiralSpr = (R1 + alpha \* (R2 - R1) / alpha[-1]) \* np.cos(alpha - 1.57)**

**Draw\_Spring.set\_data(X\_SpiralSpr, Y\_SpiralSpr)**

**return [PointB, Draw\_OA, Draw\_Spring, Draw\_AB, PointA, Draw\_Circle]**

**anima = FuncAnimation(fig, update, frames=Steps, interval=1)**

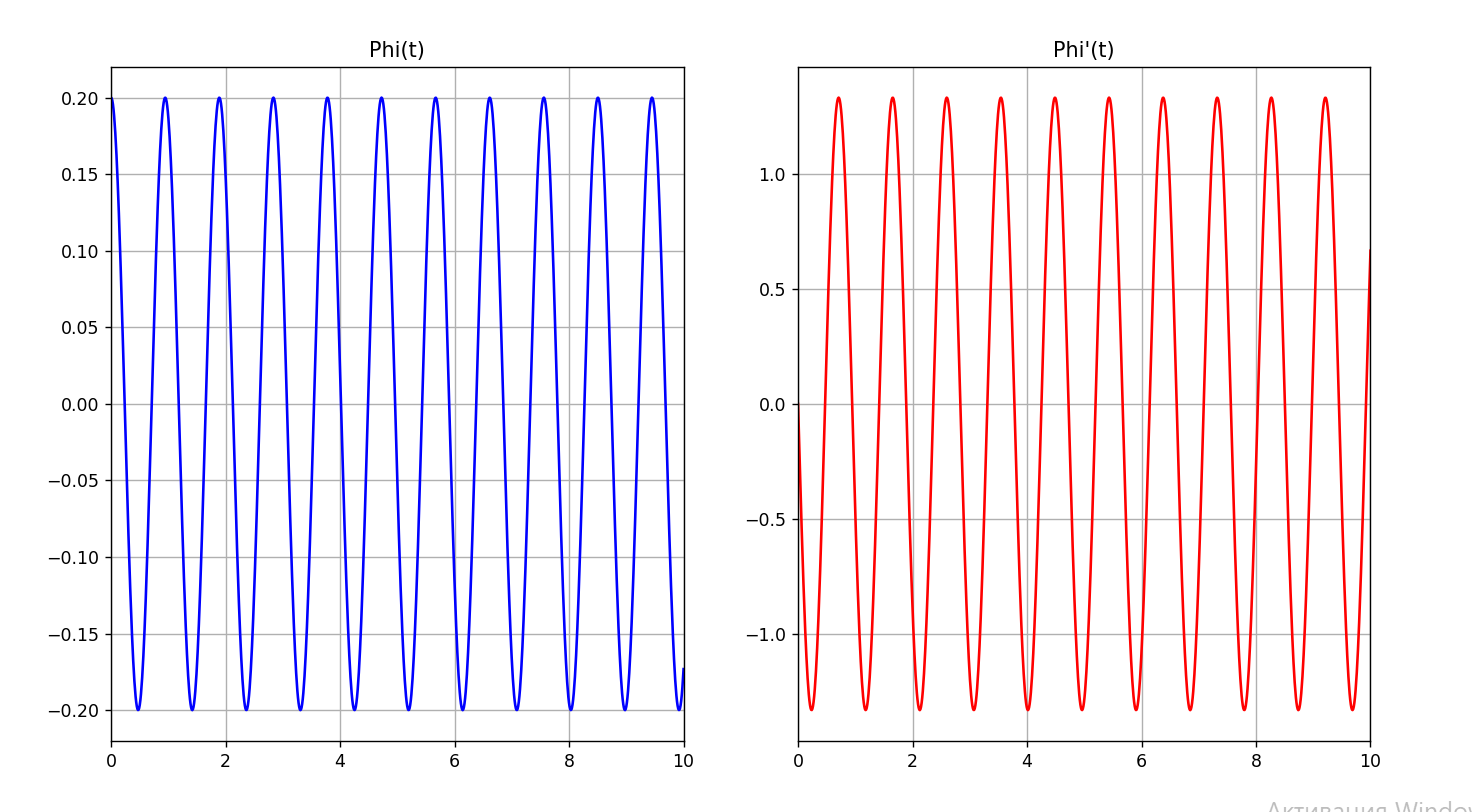
**PERIOD = 2 \* 3.14 \* sp.sqrt(m\_oa\*l\*l/(3\*(c-(m\_oa\*g\*l/2))))**

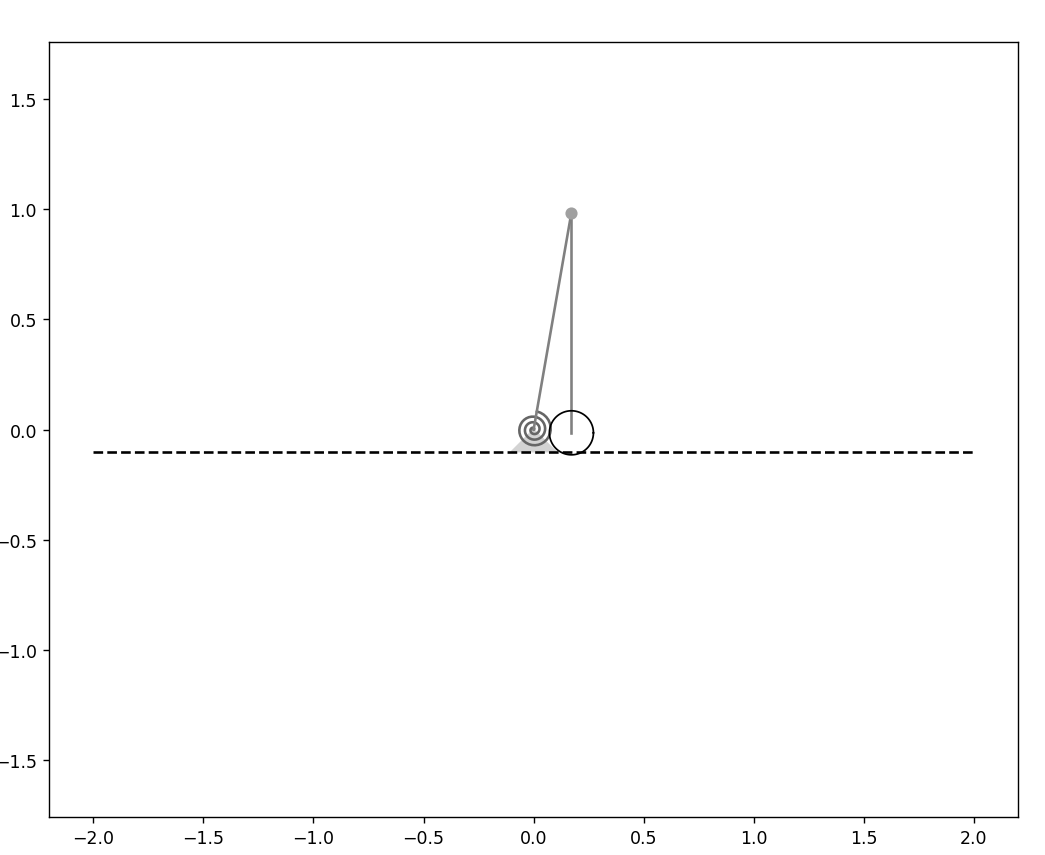
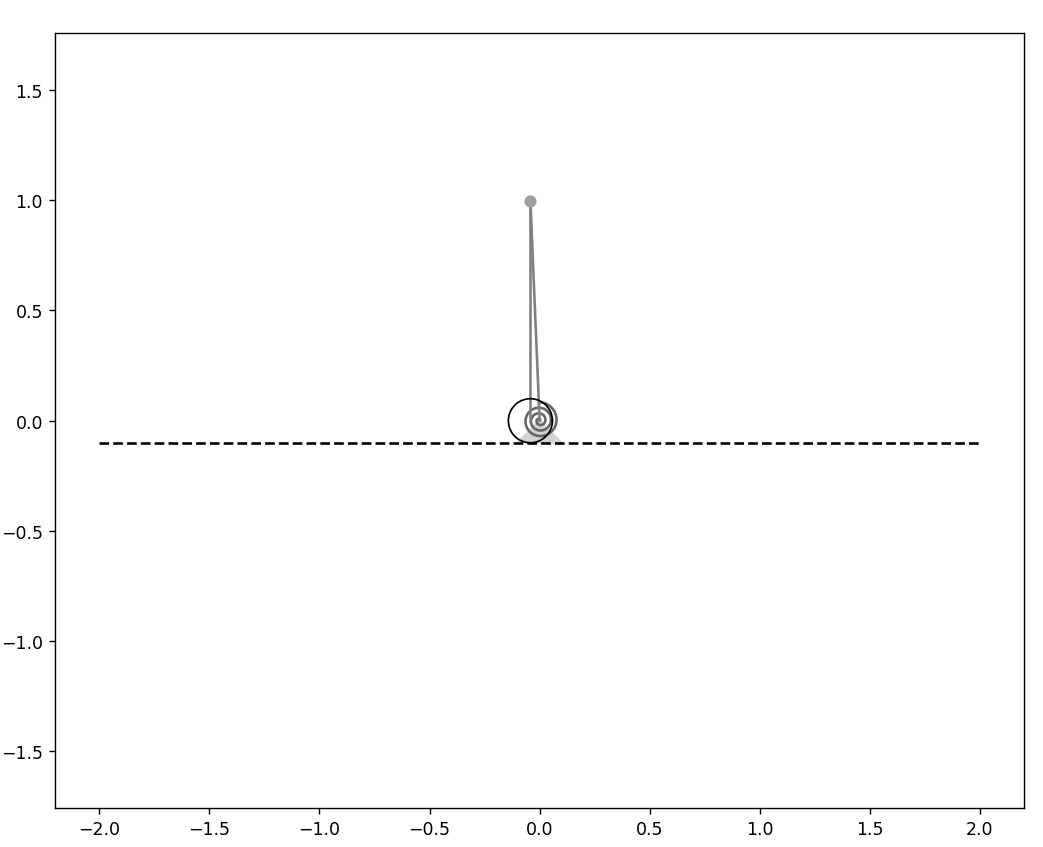
**print("PERIOD = ", PERIOD)**

**plt.show()**

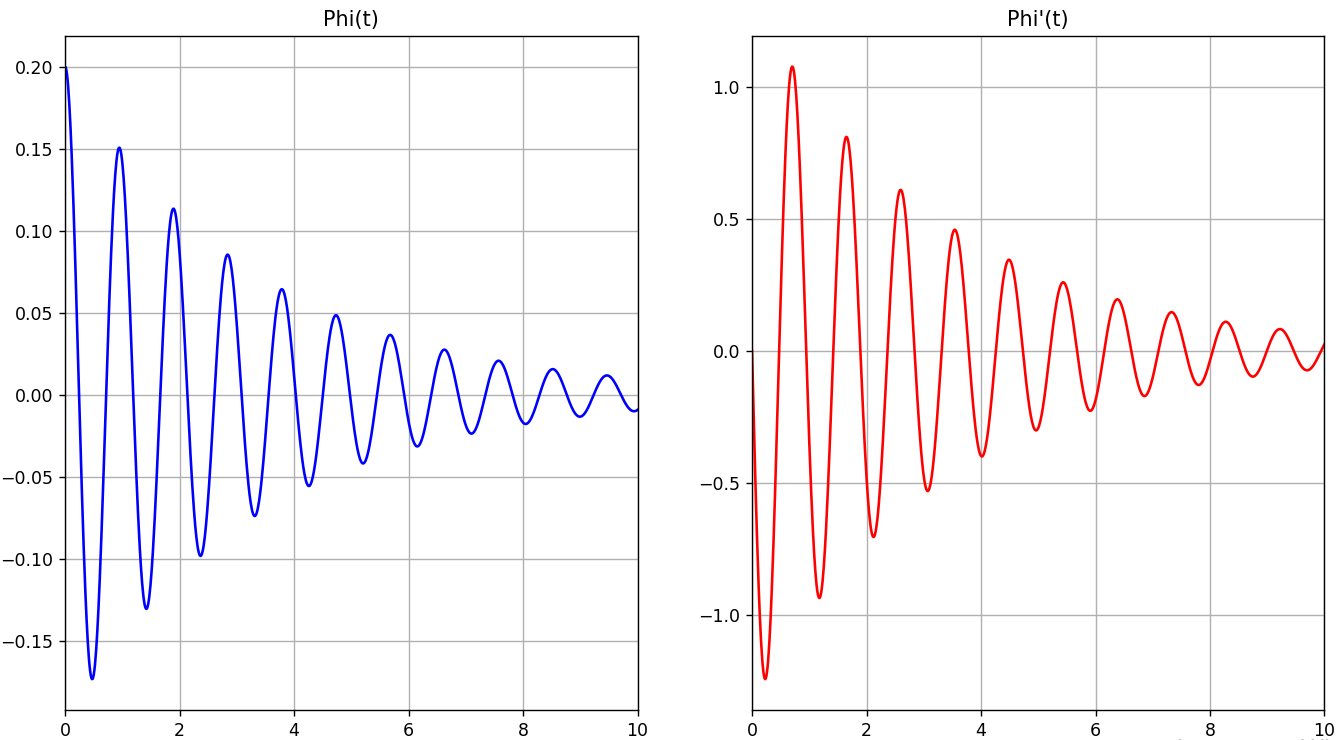
**Результат работы:**

**PERIOD = 0.945189488463463**

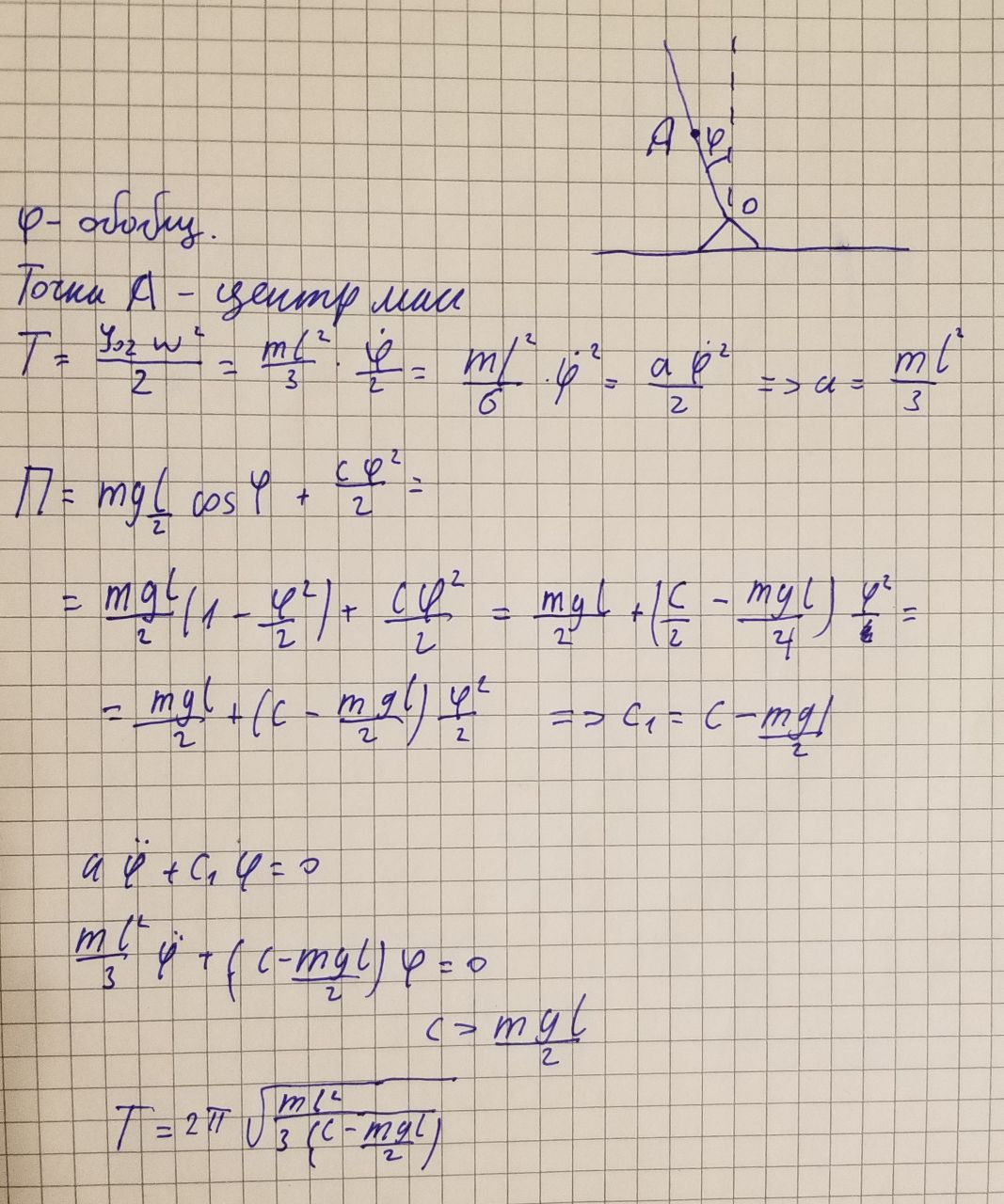
****

****

**Если добавить коеэффициент при непотенциальной силе:**

****

**Вывод периода и уравнений малых колебаний:**



**Выводы:**

В конце программы мы подставляем и вычисляем период по этой формуле. Как мы видим из гафика, период колебанийдействительно чуть меньше единицы, то есть найденное мной значение равно периоду графика. Следовательно, система смоделирована верно.

