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 № 2

Выполнил(а) студент группы М8О-208Б-21

Белоносов К.А.______________________________

подпись, дата

Проверил и принял

Зав. каф. 802, Бардин Б.С.______________________________

подпись, дата

с оценкой ______________________________

Москва, 2022

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

Схема программы:

Для решения поставленных задач требуется сделать следующие шаги.

Отдельно от основной программы с помощью уравнений движения системы требуется сформировать функцию, которая будет принимать в себя значения $(q_1, q_2, \dot{q}_1, \dot{q}_2)$, а на выход вернёт значения $(\dot{q}_1, \dot{q}_2, \ddot{q}_1, \ddot{q}_2)$.

В основной программе требуется задать значения всех параметров, начальное положение системы, и запустить процедуру численного интегрирования системы.

Результаты численного интегрирования системы далее следует использовать при построении анимации движения системы.

Составление функции уравнений движения:

Запишем функцию f, которая будет принимать в качестве аргументов время t (даже в случае автономных уравнений этот аргумент нельзя опускать) и вектор состояния системы $y = (y_1, y_2, y_3, y_4) = (q_1, q_2, \dot{q}_1, \dot{q}_2)$, а на выход вернёт $\dot{y} = (\dot{y}_1, \dot{y}_2, \dot{y}_3, \dot{y}_4) = (\dot{q}_1, \dot{q}_2, \ddot{q}_1, \ddot{q}_2)$.

Рассмотрим механическую систему, изображённую на рисунке.
Уравнения её движения имеют вид:

$$[(5/6)m_1 + (4/3)m_2]r^2\ddot{\varphi} + 2m_2le[\ddot{\theta}sin\alpha - \dot{\theta}cos\alpha] = [m_1sin\varphi + 2m_2cos(\varphi - \frac{\pi}{6})]eg - c\varphi,$$

$$2e[\ddot{\varphi}\, sin\alpha + \dot{\varphi}\, cos\alpha] + l\ddot{\theta} = -gsin\theta.$$

Вектор y имеет вид $y = (y_1, y_2, y_3, y_4) = (\theta, \varphi, \dot{\theta}, \dot{\varphi})$. Система уравнений движения является линейной относительно величин $\ddot{\theta}, \ddot{\varphi}$, и её можно решить, правилом Крамера.

Функция имеет вид:

```
def odesys(y, t, m1, m2, c, R, l, g):
    dy = np.zeros(4)
    dy[0] = y[2]
    dy[1] = y[3]
    e = R / (3 ** 0.5)
    a = y[0] - y[1] - (math.pi / 6)
    a11 = (5/6 * m1 + 4/3 * m2) * (R ** 2)
    a12 = 2 * m2 * l * e * np.sin(a)
    a21 = 2 * e * np.sin(a)
    a22 = l

    b1 = (y[3] ** 2) * np.cos(a) * 2 * m2 * l * e + \
         (m1 * np.sin(y[0]) + 2 * m2 * np.cos(y[0] - math.pi / 6)) * e * g - c * y[0]
    b2 = -2 * e * (y[2] ** 2) * np.cos(a) - g * np.sin(y[1])

    dy[2] = (b1*a22 - b2*a12)/(a11*a22 - a12*a21)
    dy[3] = (b2*a11 - b1*a21)/(a11*a22 - a12*a21)
    return dy
```

Численное интегрирование системы уравнений:

Численное интегрирование системы дифференциальных уравнений в указанной выше форме производится с помощью функции odeint.

В начале основной программы запишем:

```
Y = odeint(odesys, y0, t, (m1, m2, c, R, l, g))


phi = Y[:, 0]
ksi = Y[:, 1]
dphi = Y[:, 2]
dksi = Y[:, 3]
```

```
ddphi = [odesys(y, t, m1, m2, c, R, l, g)[2] for y, t in zip(Y, t)]

ddksi = [odesys(y, t, m1, m2, c, R, l, g)[3] for y, t in zip(Y, t)]
```

<u>Построение графиков:</u>

Построим графики решения $\theta(t), \varphi(t)$ и реакции. Для построения графика реакции потребуется дополнительно вычислить значения $\ddot{\theta}, \ddot{\varphi}$. Это можно сделать, вызвав функцию с уравнениями движения, отправив её в качестве аргументов полученное решение.

```
fig_for_graphs = plt.figure(figsize=[13, 7])
ax_for_graphs = fig_for_graphs.add_subplot(2, 3, 1)
ax_for_graphs.plot(t, 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(2, 3, 2)
ax_for_graphs.plot(t, ksi, color='red')
ax_for_graphs.set_title('ksi(t)')
ax_for_graphs.set(xlim=[0, t_fin])
ax_for_graphs.grid(True)

ax_for_graphs = fig_for_graphs.add_subplot(2, 3, 3)
ax_for_graphs.plot(t, dphi, color='green')
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(2, 3, 4)
ax_for_graphs.plot(t, dksi, color='black')
ax_for_graphs.set_title('ksi\'(t)')
ax_for_graphs.set(xlim=[0, t_fin])
ax_for_graphs.grid(True)

ax_for_graphs = fig_for_graphs.add_subplot(2, 3, 5)
ax_for_graphs.plot(t, ddphi, color='green')
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(2, 3, 6)
ax_for_graphs.plot(t, ddksi, color='black')
ax_for_graphs.set_title('ksi\'\'(t)')
ax_for_graphs.set(xlim=[0, t_fin])
ax_for_graphs.grid(True)
```

Построение анимации:

Теперь построим анимацию движения системы, используя полученные результаты. Оформим графическое окно, создадим нарисованные объекты в начальном положении и запустим цикл, переставляющий объекты в положения, отвечающие новым моментам времени.

```
psi = np.linspace(0, 2 * math.pi, 100)

C = 1

R1 = 0.2

R2 = R/(3**0.5)/2


X_Wheel = R*np.sin(psi)

Y_Wheel = R*np.cos(psi)

X_C = (R/(3**0.5))*np.sin(phi)

Y_C = (R/(3**0.5))*np.cos(phi)

X_A = X_C + R*np.sin(phi - math.pi / 2)

Y_A = Y_C + R*np.cos(phi - math.pi / 2)

X_B = X_A + l*np.cos(ksi)

Y_B = Y_A + l*np.sin(ksi)

thetta = np.linspace(0, C*math.pi*2-phi[0], 100)

X_SpiralSpr = -(R1 + thetta*(R2-R1)/thetta[-1])*np.sin(thetta)

Y_SpiralSpr = (R1 + thetta*(R2-R1)/thetta[-1])*np.cos(thetta)


fig = plt.figure(figsize=[15, 7])


ax = fig.add_subplot(1, 1, 1)


ax.axis('equal')

ax.set(xlim=[-6, 6], ylim=[-6, 6])
```

```
Circ = ax.plot(X_C[0]+R * np.cos(psi), Y_C[0]+R * np.sin(psi), 'yellow')[0]

Point_O = ax.plot(0, 0, marker='o')[0]

Point_C = ax.plot(X_C[0], Y_C[0], marker='o')[0]

Point_A = ax.plot(X_A[0], Y_A[0], marker='o')[0]

Point_B = ax.plot(X_B[0], Y_B[0], marker='o', markersize=20)[0]

Line_CO = ax.plot([0, X_C[0]], [0, Y_C[0]], 'blue')[0]

Line_AB = ax.plot([X_A[0], X_B[0]], [Y_A[0], Y_B[0]])[0]

Drawed_SpiralSpring = ax.plot(X_SpiralSpr, Y_SpiralSpr)[0]


def anima(i):

    Point_C.set_data(X_C[i], Y_C[i])

    Point_A.set_data(X_A[i], Y_A[i])

    Point_B.set_data(X_B[i], Y_B[i])

    Line_AB.set_data([X_A[i], X_B[i]], [Y_A[i], Y_B[i]])

    Circ.set_data(X_C[i] + R * np.cos(psi), Y_C[i] + R * np.sin(psi))

    Line_CO.set_data([0, X_C[i]], [0, Y_C[i]])

    thetta = np.linspace(0, C * math.pi * 2 - phi[i], 100)

    X_SpiralSpr = -(R1 + thetta * (R2 - R1) / thetta[-1]) * np.sin(thetta)

    Y_SpiralSpr = (R1 + thetta * (R2 - R1) / thetta[-1]) * np.cos(thetta)

    Drawed_SpiralSpring.set_data(X_SpiralSpr, Y_SpiralSpr)

    return [Point_C, Point_A, Point_B, Circ, Line_CO, Line_AB, Drawed_SpiralSpring]


anim = FuncAnimation(fig, anima, frames=len(t), interval=3)


plt.show()
```

Примеры работы программы:

1. Начальные условия:

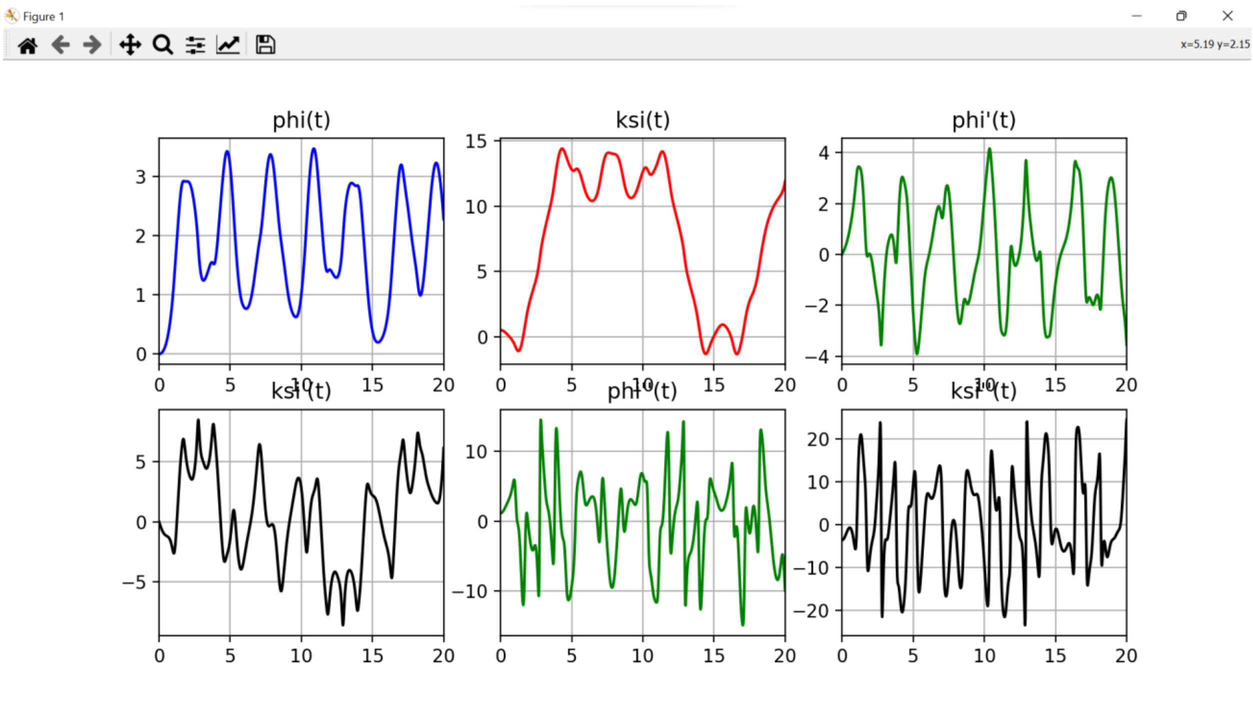
M1 = 5кг

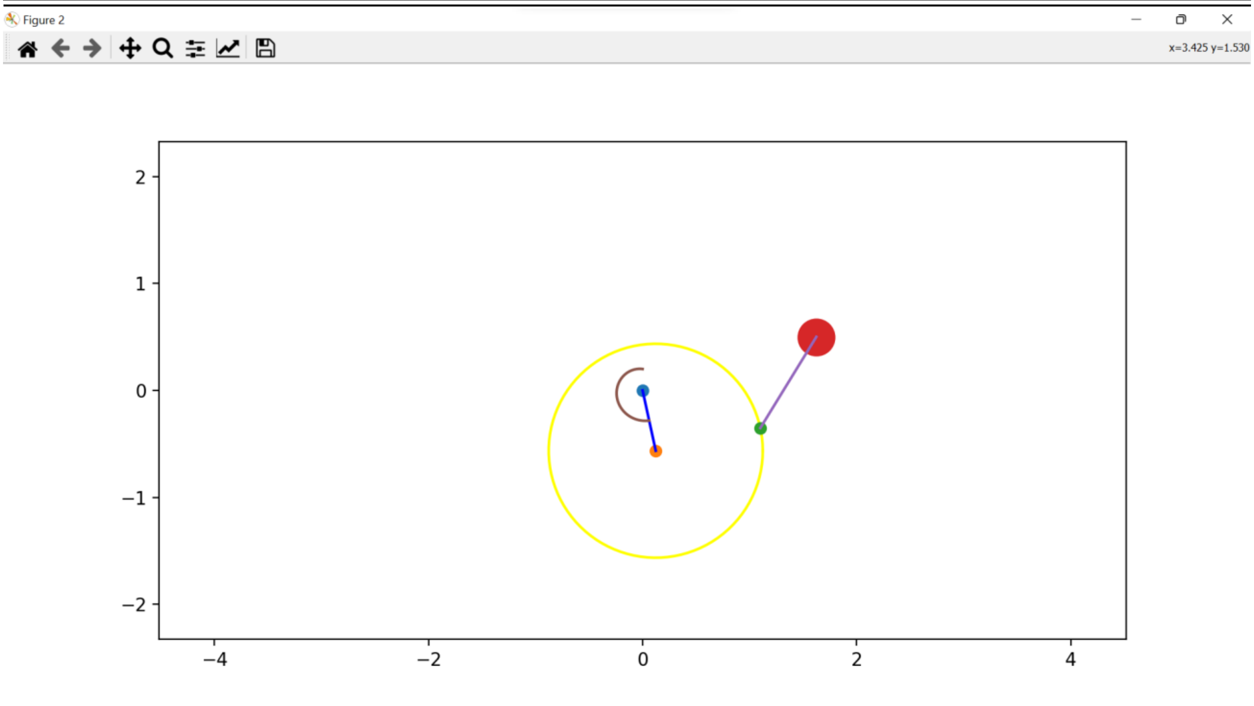
M2 = 1кг

C = 10 Н*м

r = 1 м

l = 1 м





При данных начальных условиях, система совершает затухающие колебания. Причем колебания с большим периодом.

2. Начальные условия

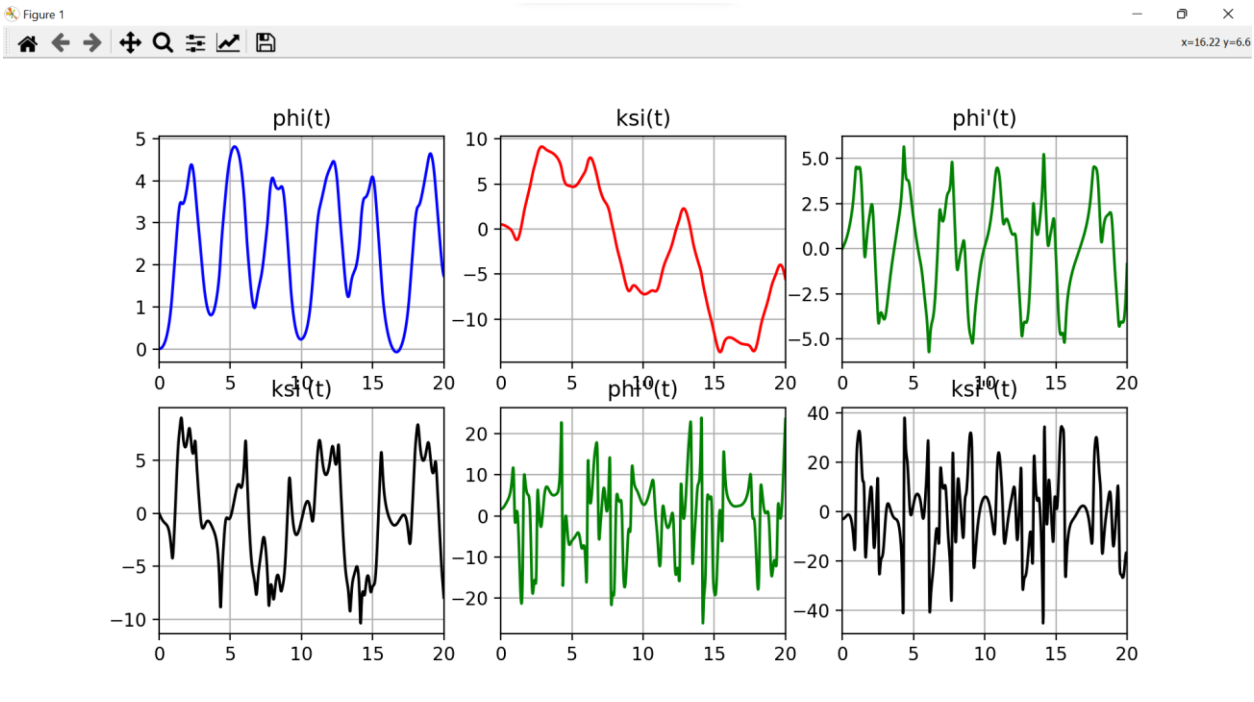
M1 = 10кг

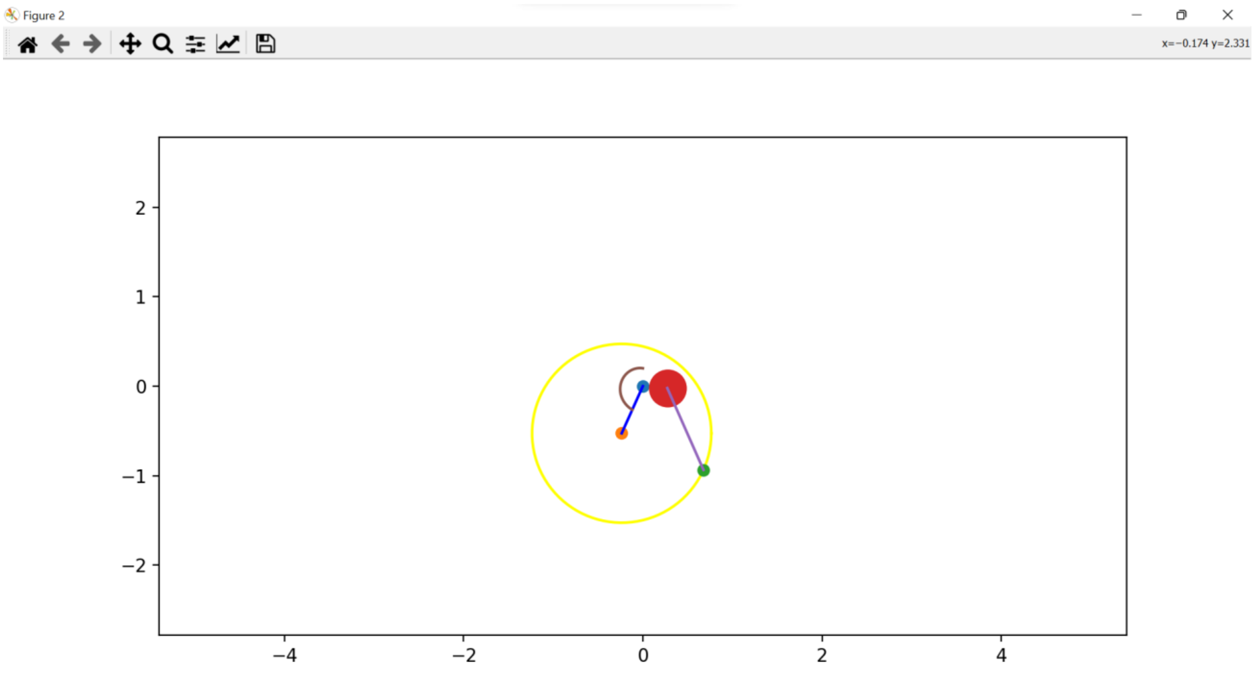
M2 = 3кг

C = 1 Н*м

r = 1 м

l = 1 м





Можно заметить, что с увеличением массы и понижением жесткости пружины уменьшилась амплитуда колебаний грузика, при этом колебания амплитуда колебаний диска увеличилась

3. Начальные условия

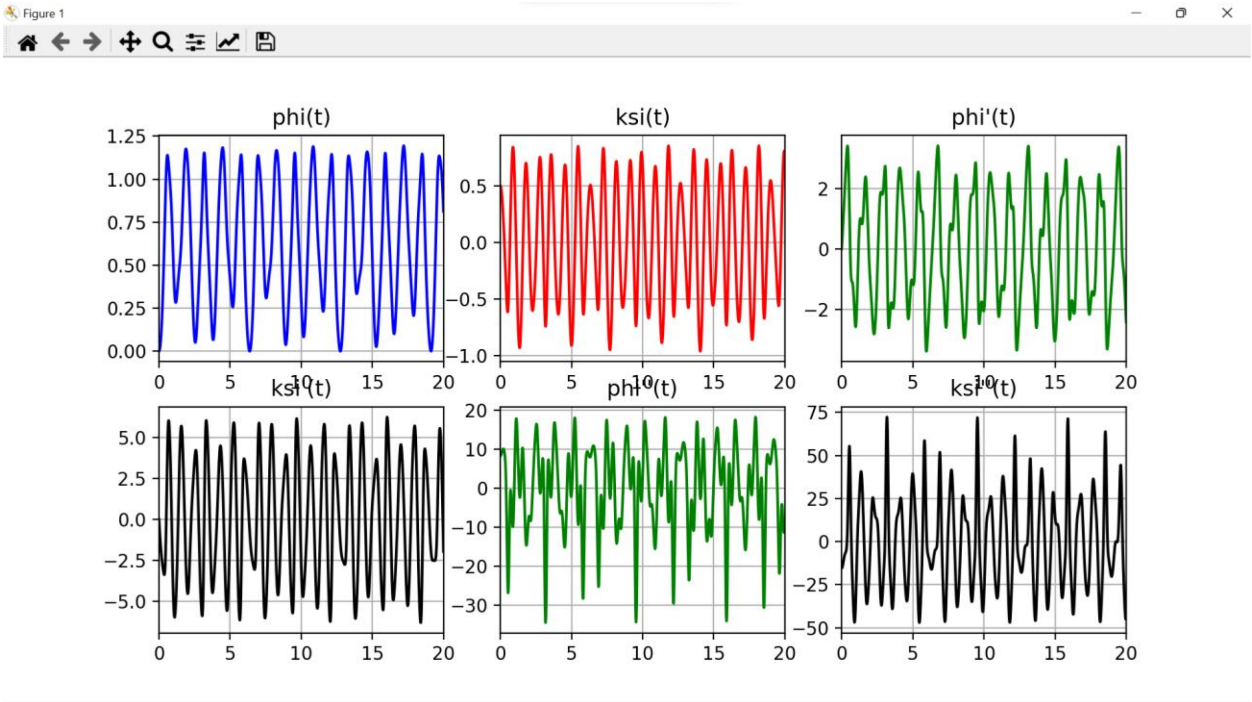
M1 = 10кг

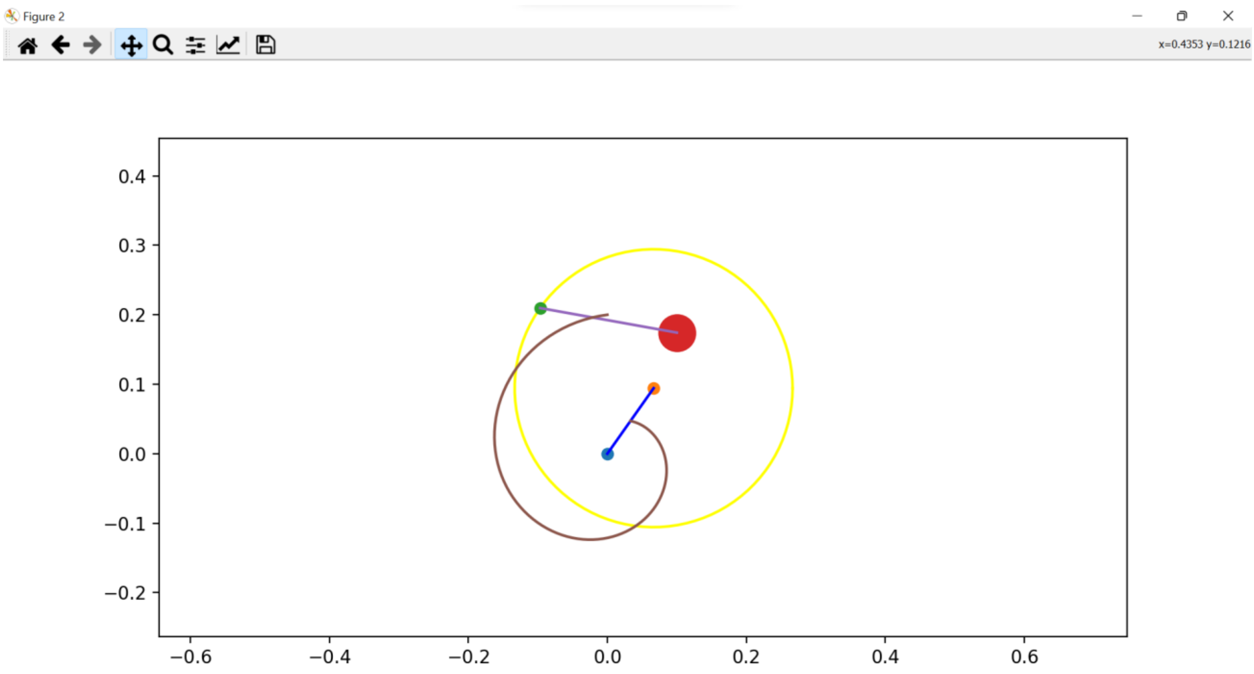
M2 = 3кг

C = 20 Н*м

r = 0.2 м

l = 0.2 м





Изменение длины стержня и оси сильно сказалось на характере колебаний. Наша система стала совершать малые колебания.

4. Начальные условия

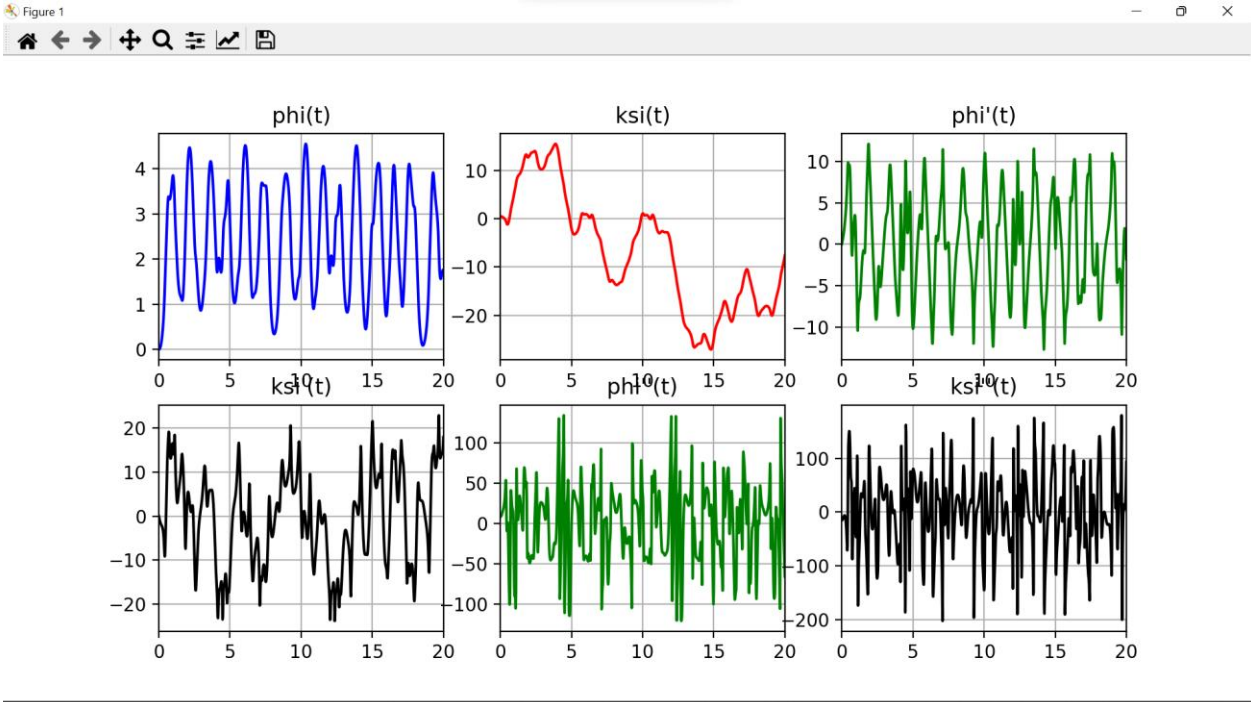
M1 = 10кг

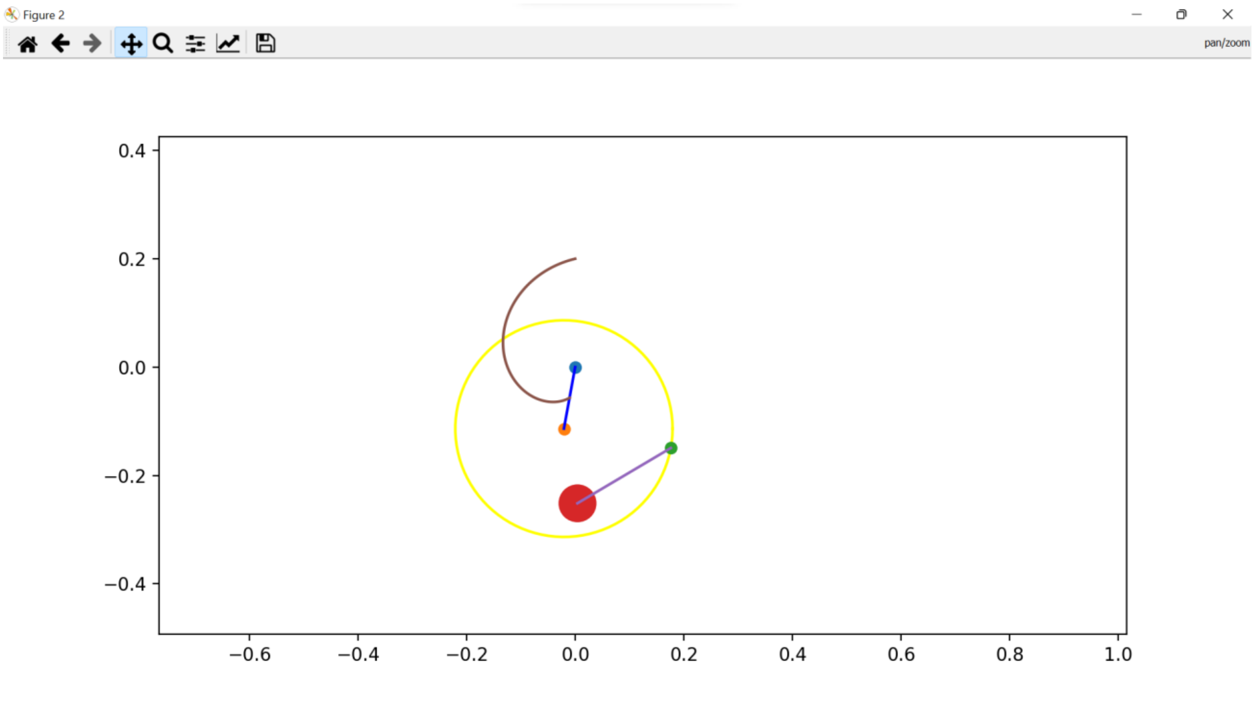
M2 = 3кг

C = 1 Н*м

r = 0.2 м

l = 0.2 м





При уменьшении жесткости пружины, амплитуда системы резка возросла и она перестала совершать малые колебания.

Вывод: В результате данной лабораторной работы мы изучили данную нами систему, составили уравнения Лагранжа 2 рода, посмотрели на поведение системы при различных начальных условиях и изучили её устойчивость.

__Вычисления:__

$$U = -m_1 \cos\varphi\, e g + 2 e g\, m_2 \sin\left(\varphi - \frac{\pi}{6}\right) + \frac{c\varphi^2}{2} +$$
$$+ m_2 g l \cos\theta$$

$$T = \frac{5 m_1 r^2 \dot{\varphi}^2}{2} + \frac{4 m_2 r^2 \dot{\varphi}^2}{3} + m_2 l e \sin\alpha\, \dot{\theta}^2 +$$
$$+ m_1 l e \sin\alpha\, \dot{\varphi}^2$$

$T_0 = 0 \Rightarrow$

$U^* = U \qquad m_2 = 0 \quad \varphi = 0$

$$\frac{\partial U^*}{\partial \varphi} = m_1 e g \sin\varphi - c\varphi = 0$$

$$m_1 e g \sin\varphi - c\varphi = 0$$

$$m_1 e g \sin\varphi = c\varphi \Big|_{\varphi = 0}$$

$$0 = 0$$

$$\frac{\partial^2 U^*}{\partial \varphi^2} = e g m_1 \cos\varphi - c > 0 \Big|_{\varphi = 0}$$

$$e g m_1 > c$$

$$m_1 > \frac{c}{e g}$$