ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

**ОТЧЕТ**

**О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

**«ДИНАМИКА СИСТЕМЫ»**

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

**ВАРИАНТ ЗАДАНИЯ №15**

Выполнил(а) студент группы М8О-212Б-22

Жаворонков Михаил Николаевич\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

Проверил и принял

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

подпись, дата

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

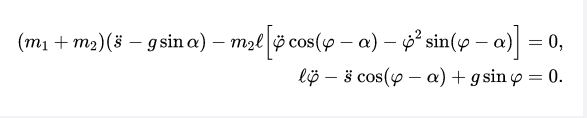
Москва, 2023

**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.

**Задание системы 15 варианта формулируется следующим образом:**

Тело А массой m1, скользит без трения по неподвижной плоскости, наклоненной под углом α к горизонту. К телу с помощью шарнира прикреплен невесомый стержень AB длины l с точечным грузом массы m2 на конце.

Дифференциальные уравнения движения системы имеют вид:



**Код программы**

**import math**

**import numpy as np**

**import matplotlib.pyplot as plt**

**from matplotlib.animation import FuncAnimation**

**from scipy.integrate import odeint**

**def SystemOfEquations(y, t, m1, m2, l, g, alpha):**

**# y: массив, содержащий значения координат и их производных**

**# t: текущее время**

**# m1: масса тела A**

**# m2: масса точечного груза на конце стержня**

**# l: длина стержня AB**

**# g: ускорение свободного падения (гравитационная постоянная)**

**# alpha: угол наклона плоскости к горизонту**

**yt = np.zeros\_like(y)**

**# Распаковка значений из массива y**

**yt[0] = y[2] # s'**

**yt[1] = y[3] # phi'**

**# Коэффициенты матрицы для решения системы уравнений**

**a11 = m1 + m2**

**a12 = -m2 \* l \* np.cos(y[1] - alpha)**

**a21 = -np.cos(y[1] - alpha)**

**a22 = l**

**b1 = g \* np.sin(alpha) \* (m1 + m2) - m2 \* l \* ((y[3]) \*\* 2) \* np.sin(y[1] - alpha)**

**b2 = -g \* np.sin(y[1])**

**# Решение системы уравнений методом Крамера**

**yt[2] = (b1 \* a22 - a12 \* b2) / (a11 \* a22 - a12 \* a21)**

**yt[3] = (b2 \* a11 - a21 \* b1) / (a11 \* a22 - a12 \* a21)**

**return yt**

**def rotate(x, y, theta):**

**xr = math.cos(theta) \* x - math.sin(theta) \* y**

**yr = (math.sin(theta)) \* x + np.cos(theta) \* y**

**return xr, yr**

**def Film(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]])**

**Box.set\_data(X\_A[i] + X\_Box, Y\_A[i] + Y\_Box)**

**return [Point\_A, Point\_B, Box, Line\_AB]**

**m1 = 50 # масса тела**

**m2 = 50 # масса точечного груза**

**L = 2 # длина стержня**

**Phi0 = 0**

**alpha = np.pi/8**

**g = 9.81**

**S0 = 0 #начальное значение координаты тела A по оси, параллельной наклонной плоскости.**

**DS0 = 0 #представляет начальную скорость тела A, то есть производную координаты ss по времени в начальный момент времени.**

**DPhi0 = 12 \* 3.33564095e-09 #Начальная угловая скорость стержня. Это значение влияет на начальное вращение стержня.**

**y0 = [S0, Phi0, DS0, DPhi0] #ектор начальных условий**

**Tfin = 10**

**NT = 1001 #Количество шагов для численного интегрирования.**

**t = np.linspace(Tfin, 0, NT) #Создание массива временных точек для численного интегрирования.**

**Y = odeint(SystemOfEquations, y0, t, (m1, m2, L, g, alpha))#Решение системы дифференциальных уравнений с использованием численного интегрирования odeint.**

**x = Y[:, 0]**

**phi = Y[:, 1]**

**Dx = Y[:, 2]**

**Dphi = Y[:, 3]**

**DDx = [SystemOfEquations(y, t, m1, m2, L, g, alpha)[2] for y, t in zip(Y, t)]**

**DDphi = [SystemOfEquations(y, t, m1, m2, L, g, alpha)[3] for y, t in zip(Y, t)]**

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

**ax\_for\_graphs = fig\_for\_graphs.add\_subplot(2, 3, 1)**

**ax\_for\_graphs.plot(t, x, color='blue')**

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

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

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

**ax\_for\_graphs = fig\_for\_graphs.add\_subplot(2, 3, 4)**

**ax\_for\_graphs.plot(t, phi, color='red')**

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

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

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

**ax\_for\_graphs = fig\_for\_graphs.add\_subplot(2, 3, 2)**

**ax\_for\_graphs.plot(t, Dx, color='green')**

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

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

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

**ax\_for\_graphs = fig\_for\_graphs.add\_subplot(2, 3, 5)**

**ax\_for\_graphs.plot(t, Dphi, color='black')**

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

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

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

**ax\_for\_graphs = fig\_for\_graphs.add\_subplot(2, 3, 3)**

**ax\_for\_graphs.plot(t, DDx, color='blue')**

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

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

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

**ax\_for\_graphs = fig\_for\_graphs.add\_subplot(2, 3, 6)**

**ax\_for\_graphs.plot(t, DDphi, color='red')**

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

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

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

**fig = plt.figure(figsize=[20, 8])**

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

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

**ax.set(xlim=[-3, 10], ylim=[-3, 10])**

**BoxX = 3**

**BoxY = 2**

**bx = BoxX / 2**

**by = BoxY / 2**

**l = L**

**Pi = np.pi**

**tg = np.tan(alpha)**

**# Отрисовка земли**

**xMax = x[0] + (math.fabs(x[-1] - x[0]))/10**

**xMin = x[-1]\*(-1)**

**coef = by\*1.1**

**X\_Ground = [xMax - BoxX\*2\*tg, xMax, xMin, xMax]**

**Y\_Ground = [xMax\*tg - coef + BoxX\*2, xMax \***

**tg - coef, xMin\*tg - coef, xMin\*tg - coef]**

**ax.plot(X\_Ground, Y\_Ground, color='black', linewidth=3)**

**#**

**# Коробки**

**x1, y1 = rotate(-bx, by, alpha)**

**x2, y2 = rotate(bx, by, alpha)**

**x3, y3 = rotate(bx, -by, alpha)**

**x4, y4 = rotate(-bx, -by, alpha)**

**X\_Box = np.array([x1, x2, x3, x4, x1])**

**Y\_Box = np.array([y1, y2, y3, y4, y1])**

**#**

**# Точки А и В**

**X\_A = BoxX / 2 - x + 10**

**Y\_A = X\_A \* tg**

**X\_B = X\_A + l \* np.sin(phi)**

**Y\_B = Y\_A - l \* np.cos(phi)**

**#**

**# Плоты**

**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, color='black')[0]**

**Box = ax.plot(X\_A[0] + X\_Box, Y\_A[0] + Y\_Box, color='black', linewidth='2')[0]**

**Line\_AB = ax.plot([X\_A[0], X\_B[0]], [Y\_A[0], Y\_B[0]],**

**color='black', linewidth='2')[0]**

**#**

**anima = FuncAnimation(fig, Film, frames=NT, interval=10,repeat=True)**

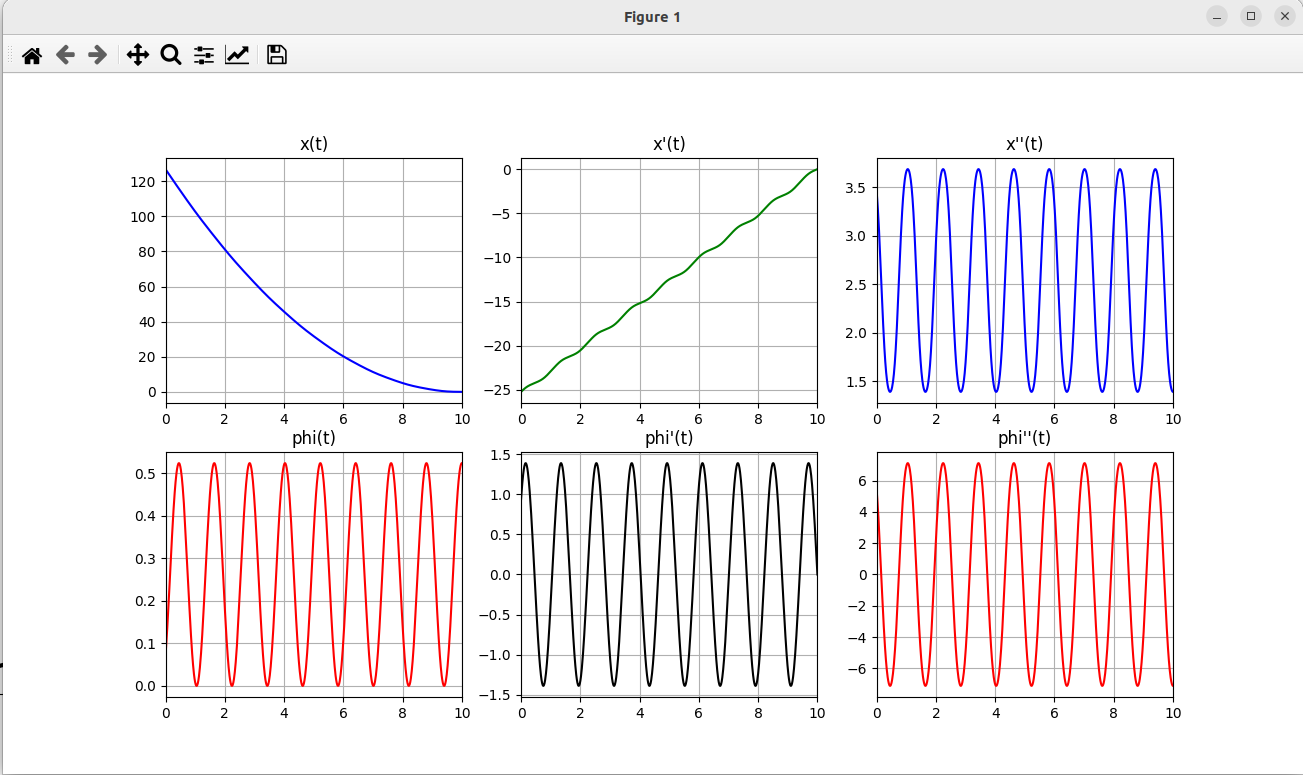
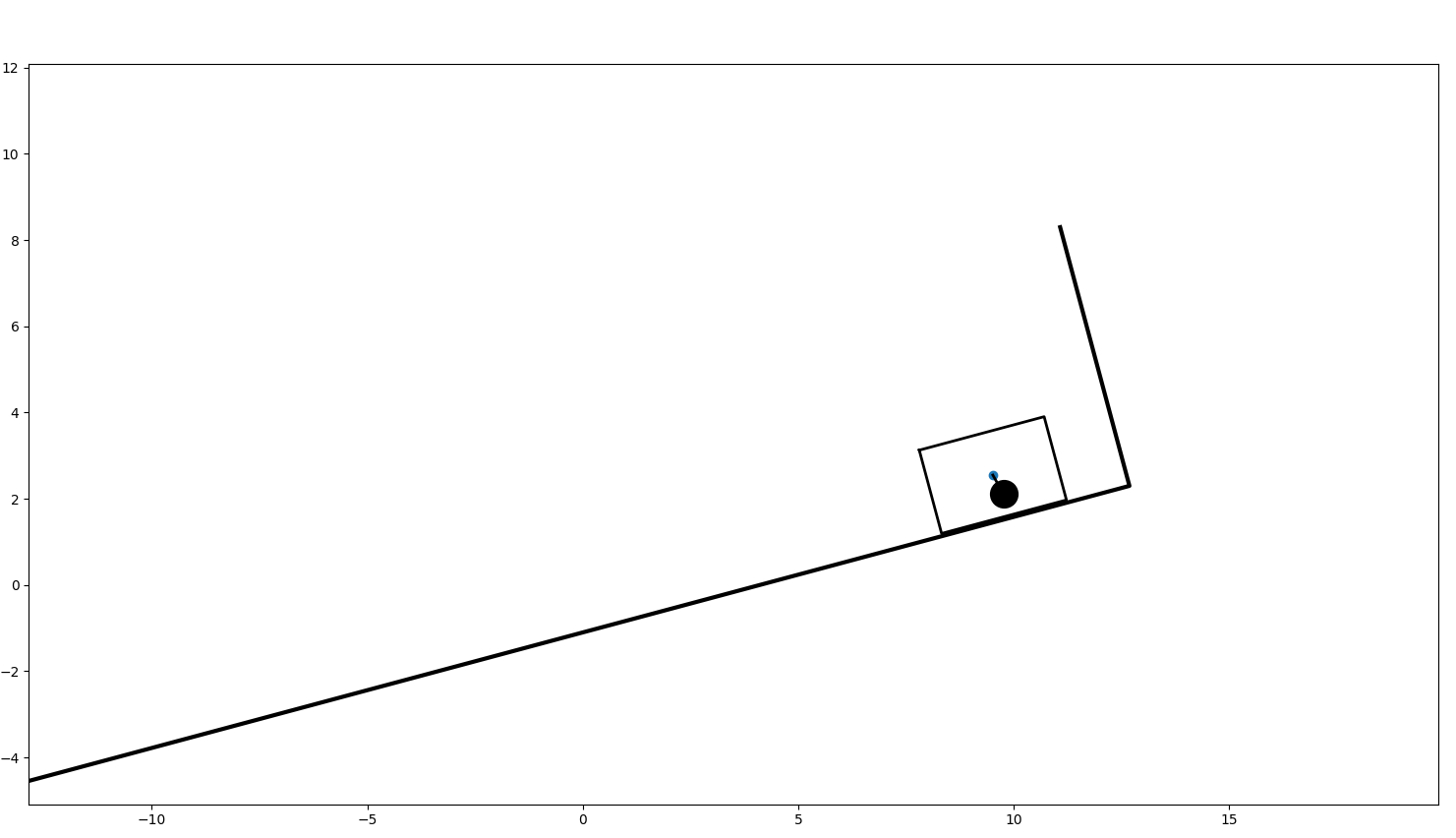
**plt.show()**

**Примеры работы:**

Приведу несколько примеров работы моей программы.

1. Начальные условия из учебника

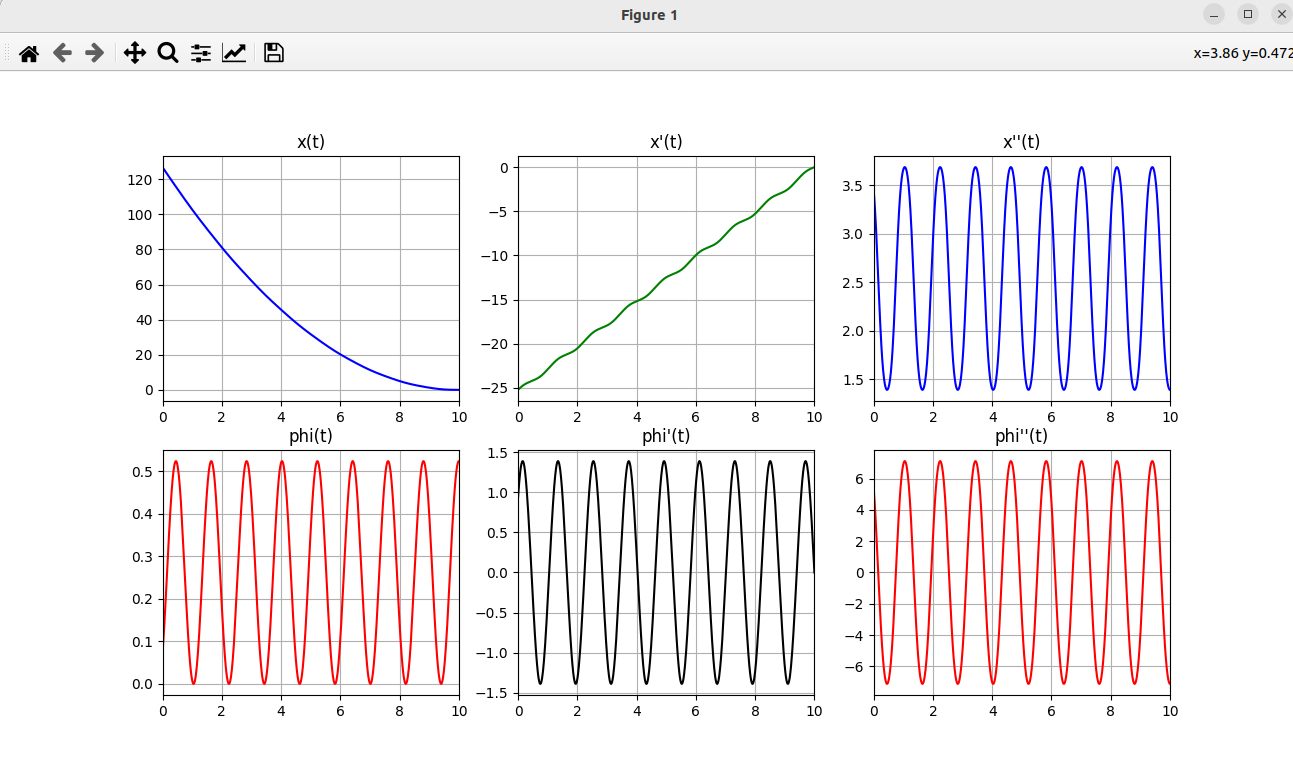
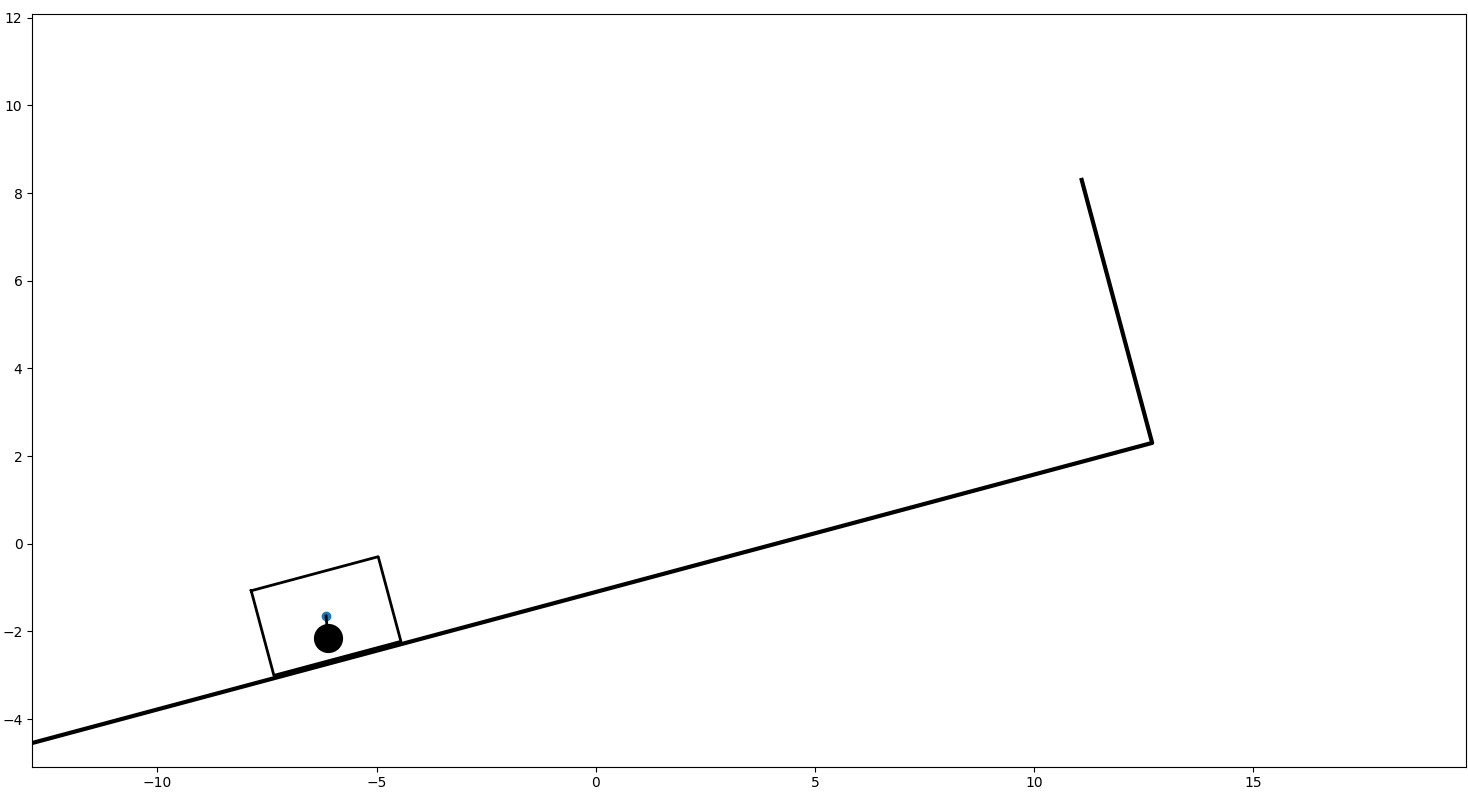
m1=1 кг, m2=0,5 кг, l=0,5 м, t0=0,s0=0, φ0 =π/6 ,ds0=0,dφ0 =12с^-1



Движение системы можно охарактеризовать как медленное, но поддерживаемое значительным влиянием начальной угловой скорости. Тело A и стержень AB совершают колебательное движение с относительно небольшой амплитудой и высокой частотой колебаний. Точечный груз на конце стержня начинает двигаться с некоторым ускорением, но из-за относительно небольшой массы его влияние на общее движение системы ограничено.

1. Увеличим массу тела А в 10 раз относительно начальных значений

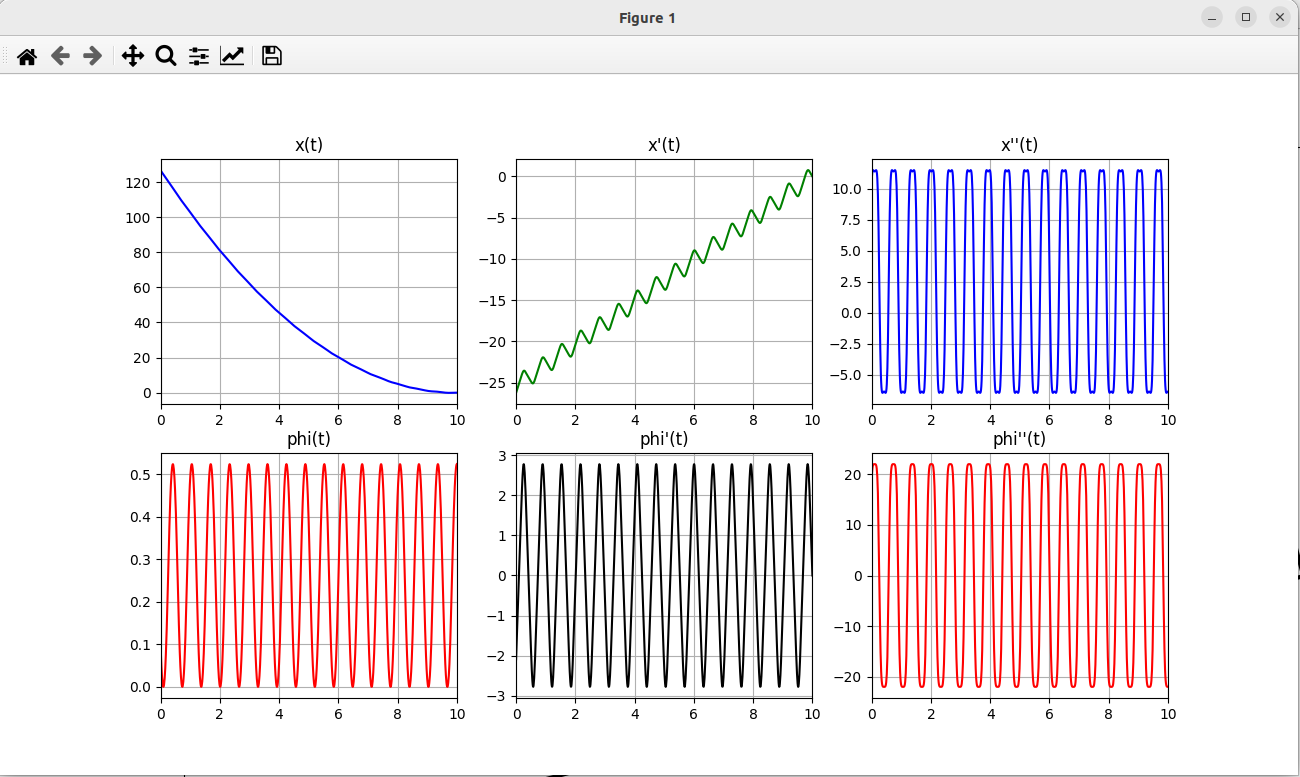
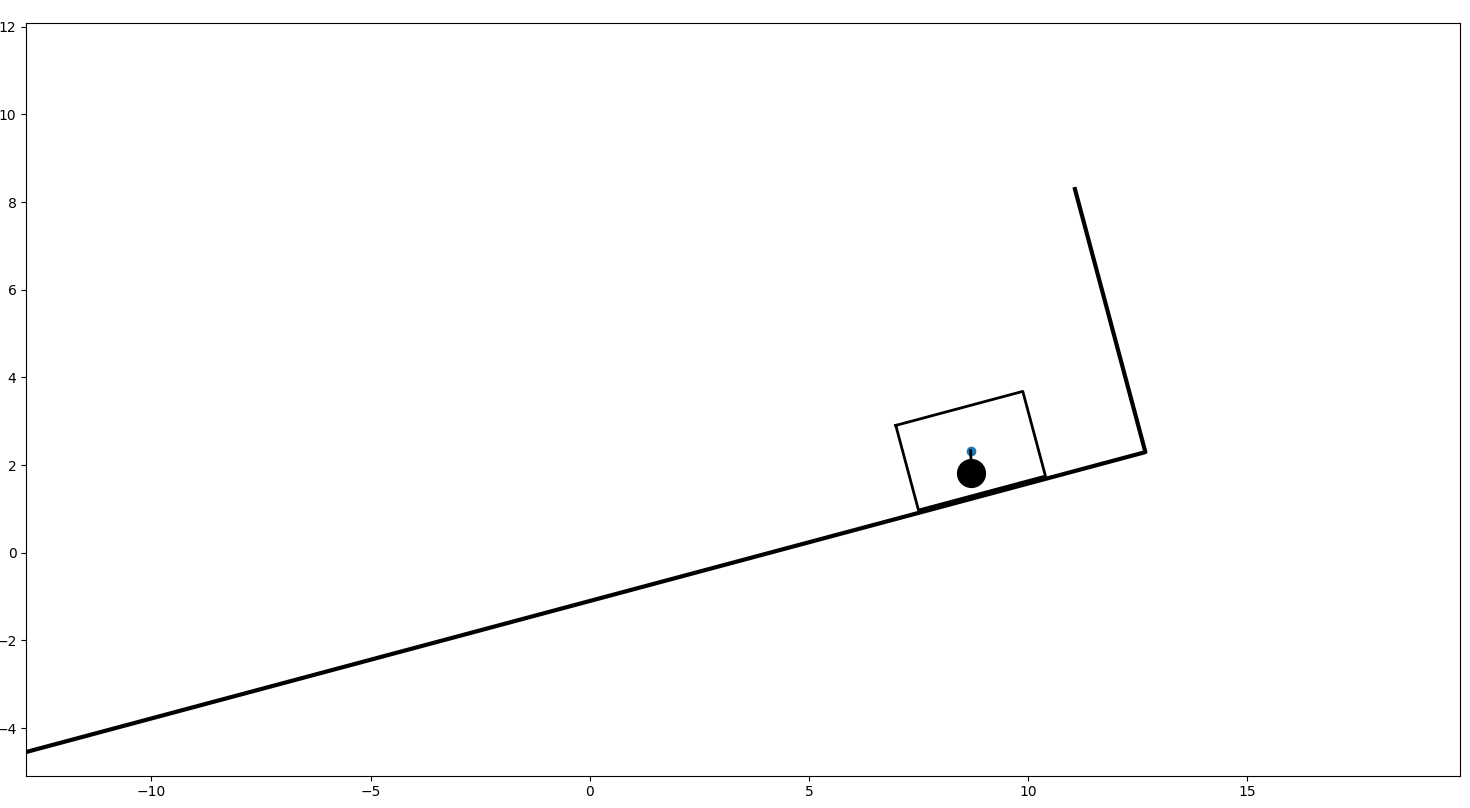
m1=10 кг, m2=0,5 кг, l=0,5 м, t0=0,s0=0, φ0 =π/6 ,ds0=0,dφ0 =12с^-1



Движение системы становится более замедленным из-за увеличения инерции. Амплитуда колебаний может уменьшиться, но система всё ещё совершает колебательное движение с высокой частотой. Влияние точечного груза остается ограниченным, и движение становится менее чувствительным к его воздействию из-за увеличения массы тела A.

1. Уменьшим массу тела А в 10 раз относительно начальных значений

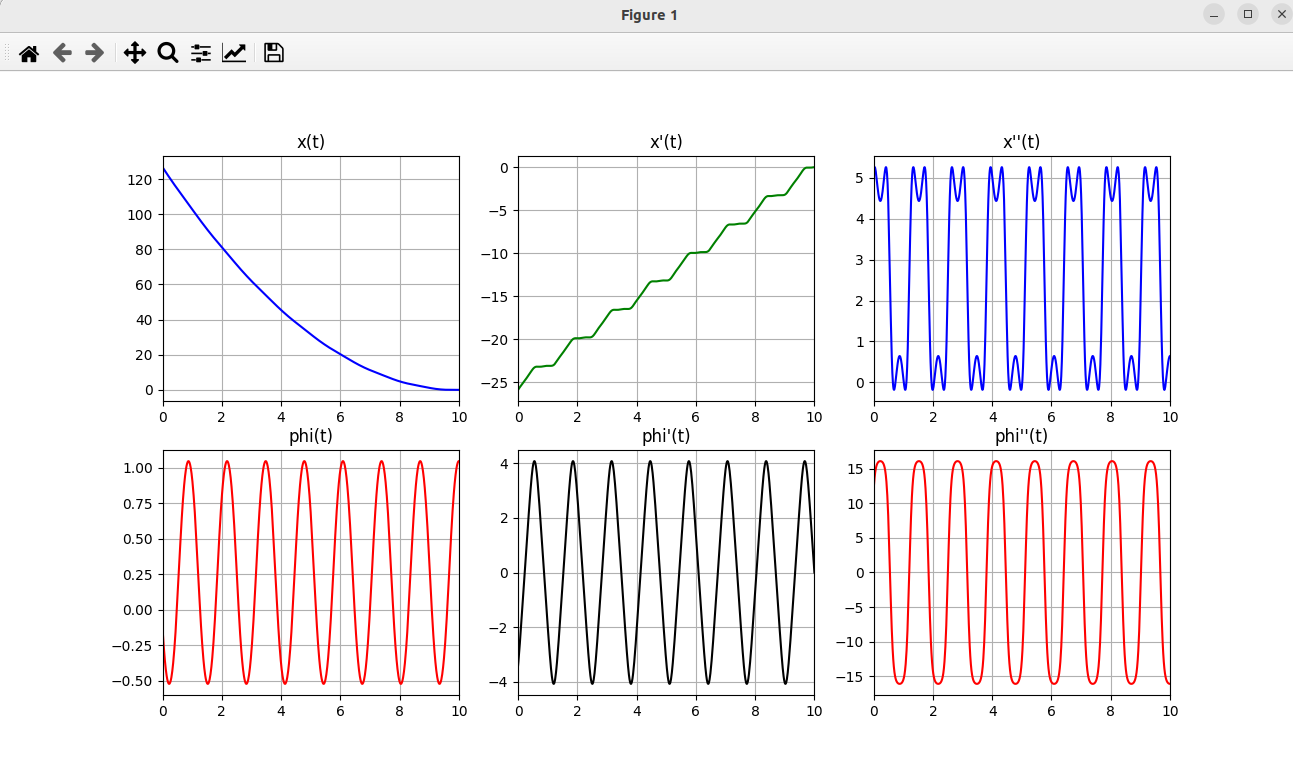
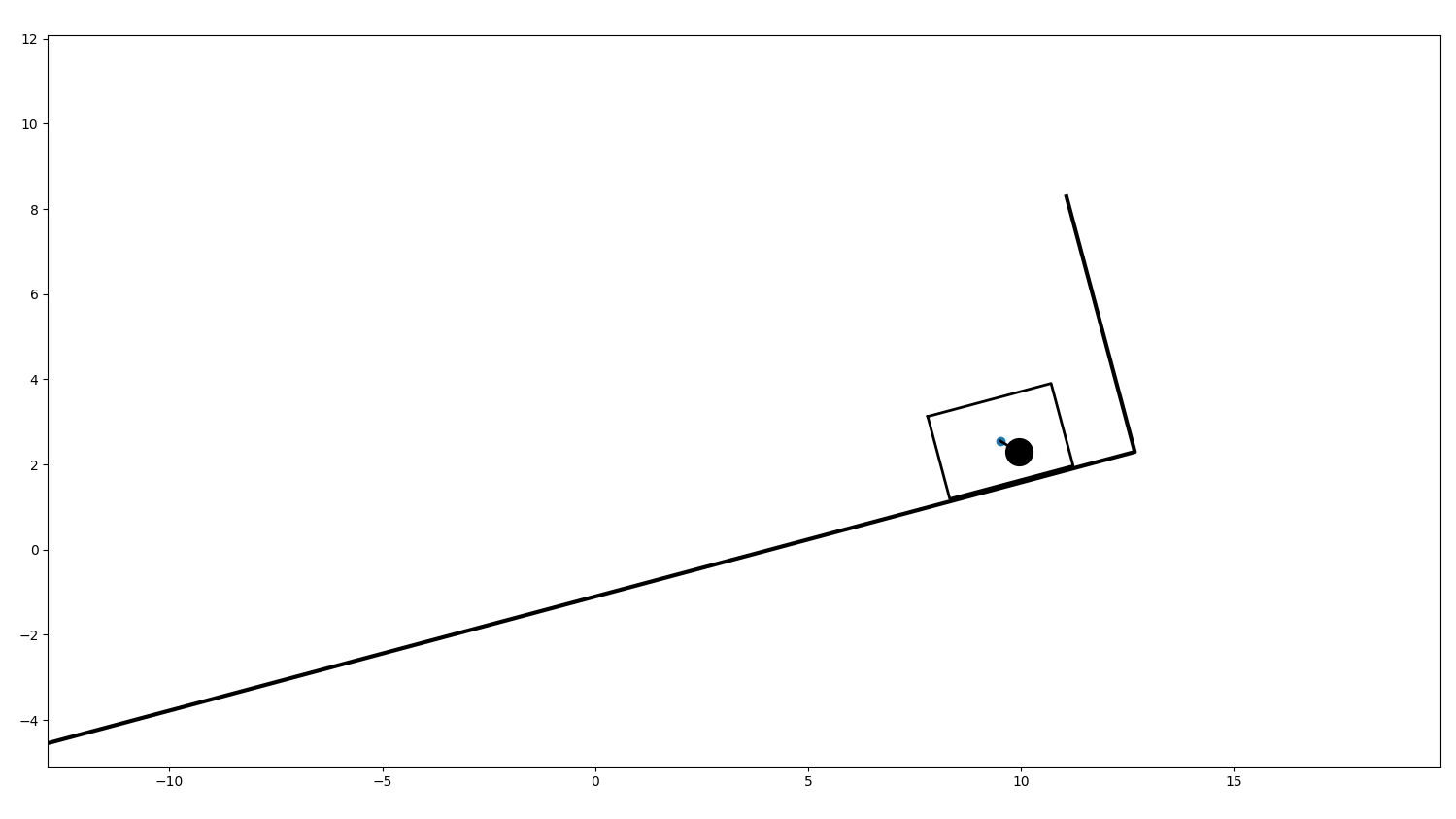
m1=0,1 кг, m2=0,5 кг, l=0,5 м, t0=0,s0=0, φ0 =π/6 ,ds0=0,dφ0 =12с^-1



Движение системы становится более подвижным и быстрым из-за уменьшения инерции. Амплитуда колебаний может быть более высокой, и система сохраняет высокую частоту колебаний. Влияние точечного груза остается ограниченным, и движение становится более чувствительным к его воздействию из-за уменьшения массы тела A.

4.Когда α=π/4, сила тяжести будет действовать по диагонали, равномерно распределенной между направлением стержня и направлением вдоль плоскости. Это приведет к изменению уравнений движения и, следовательно, изменению динамики системы.

5. Увеличение угла Фи на π/3



Увеличение угла Φ0​ приводит к более значительному начальному отклонению стержня от вертикальной оси. Это изменение приводит к более сложным колебательным движениям системы, где стержень может совершать большие амплитудные колебания. Также обратите внимание, что изменение угла начального отклонения может существенно влиять на форму траектории и период колебаний.

**Вывод**

Через использование Python, библиотек matplotlib и numpy, я разработал программу для анимации движения системы, состоящей из двух стержней и пружины. Была создана 2D-анимация, визуализирующая движение системы. Это позволило наглядно представить, как изменяются положение стержней и пружины во времени. Система дифференциальных уравнений, описывающая движение стержней и пружины, была успешно решена с использованием функции odeint. Программа учитывает физические законы движения, что позволяет получить более реалистичное представление о том, как данная система вела бы себя в реальной жизни.