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 №21

Выполнил(а) студент группы М80-201Б-22

Парфенов Михаил Максимович_____
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 системы, а также графики законов движения системы и указанных в задании реакций для разных случаев системы.

Механическая система:

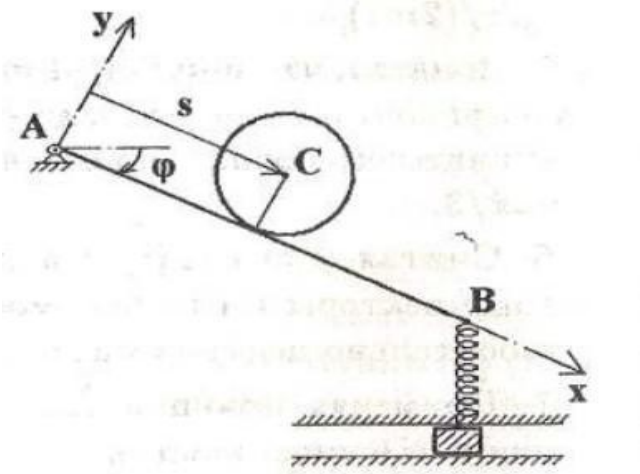


Рис. 32

Текст программы:

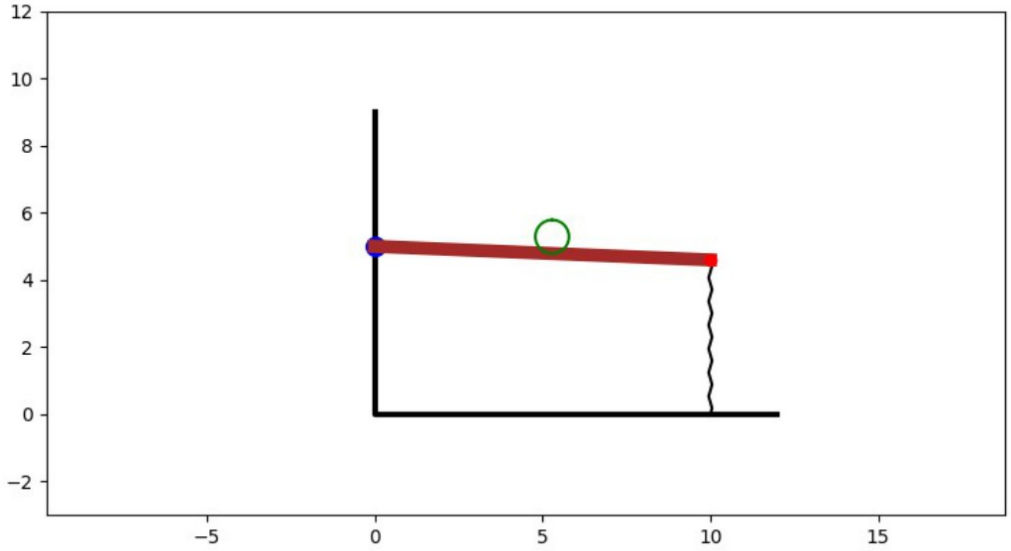
```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation
from scipy.integrate import odeint


def odesys(y, t, P1, P2, l, r, phi_, g): # функция системы диффуров
    dy = np.zeros(4)

    dy[0] = y[2]
    dy[1] = y[3]

#       Правило Краммера
#       a11 * s'' + a12 * phi'' = b1
#       a21 * s'' + a22 * phi'' = b2

#       detA = a11 * a22 - a12 * a21
#       detA1 = b1 * a22 - a12 * b2
#       detA2 = a11 * b2 - b2 * a21

    a11 = 3 * P2 * r / 2
    a12 = ((P1 * l**2) / 3) + P2*y[0]**2 + ((3 * P2 * r**2) / 2)
    a21 = 3 / 2
    a22 = (3 * r) / 2

    b1 = (P1 / 2) * g * l * (1 - (np.sin(y[1]) / np.sin(phi_))) * np.cos(y[1]) + P2 * g
* (y[0] * np.cos(y[1]) + r * np.sin(y[1])) - 2 * P2 * y[0] * y[2] * y[3]
    b2 = g * np.sin(y[1]) + y[0] * y[3]**2

#       dy[2] = detA1 / detA
#       dy[3] = detA2 / detA
    dy[2] = (b1*a22 - b2*a12)/(a11*a22 - a12*a21)
```

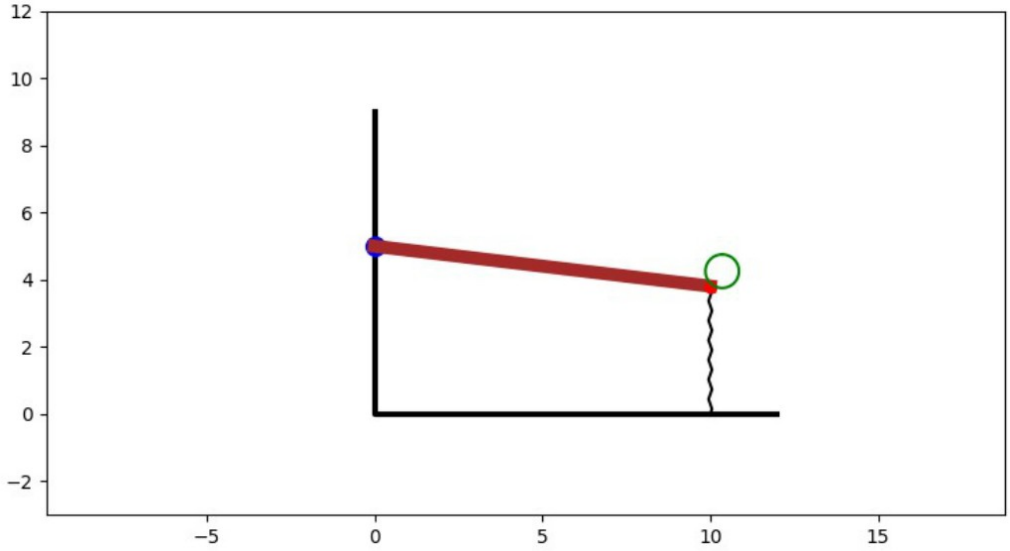
```
    dy[3] = (b2*a11 - b1*a21)/(a11*a22 - a12*a21)

    return dy


#ЗАДАЕМ НАЧАЛЬНЫЕ ЗНАЧЕНИЯ!!!
P1 = 150 # вес балки
P2 = 50 # вес диска
l = 10 # длинная балки
r = 0.5 # радиус диска
g = 9.8
phi_ = np.pi/36

s0 = 0
phi0 = np.pi/18
ds = 0
dphi0 = 0

t_fin = 15 #сетка времени
t=np.linspace(0,t_fin,501)

y0 = [s0, phi0, ds, dphi0]
# интегрируем систему диффуров
Y = odeint(odesys, y0, t, (P1, P2, l, r, phi_, g))

s = Y[:, 0]  # получили решения
phi = Y[:, 1]
ds = Y[:, 2]
dphi = Y[:, 3]

# 2 производная по phi
ddphi = np.array([odesys(yi, ti, P1, P2, l, r, phi_, g)[3] for yi,ti in zip(Y,t)])

# давелние N диска на балку
N = (P2 / g) * (g * np.cos(phi) - (2 * ds + r * dphi) * dphi - s * ddphi)

fig_for_graphs = plt.figure(figsize=[13,7])  # построим их графики
ax_for_graphs = fig_for_graphs.add_subplot(2,3,1)
ax_for_graphs.plot(t,s,color='blue')
ax_for_graphs.set_title("s(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,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 = fig_for_graphs.add_subplot(2,3,3)
ax_for_graphs.plot(t,N,color='blue')
ax_for_graphs.set_title('N(t)')
ax_for_graphs.set(xlim=[0,t_fin])
ax_for_graphs.grid(True)


xA = 0 #точки крепления шарнира и балки
yA = 5
xB = l #точки крепления пружины и балки
```

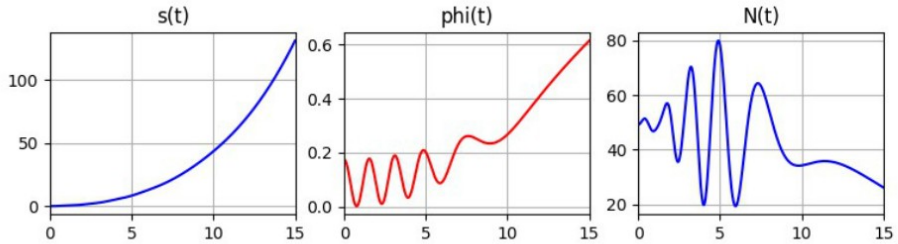
```
yB = 0


l0 = 5 #длина недеформированной пружины в начальном состоянии с учетом phi_
n = 13
h = 0.05
xP = np.zeros(2*n + 1)
yP = np.linspace(0, 1, 2*n + 1)
ss = 0
#   делаем пружину пружиной for i in range(2*n+1):
      xP[i] = h*np.sin(ss)
      ss += np.pi/2

yB = l0 - l * np.sin(phi)

#prizuna
fig = plt.figure(figsize=[15,7])
ax = fig.add_subplot(1,1,1)
ax.axis('equal')
ax.set(xlim=[-3,12], ylim=[-3,12])
Pruzzhina = ax.plot(xP + xB, yP*(yB[0] + l0), color='black')[0]

#enviroment
X_Ground = [0, 0, 12]
Y_Ground = [9, 0, 0]
ax.plot(X_Ground,Y_Ground,color='black',linewidth=3)
ax.plot([0], [5], '-ro', markersize=10, color='blue')

#балка
Stick = ax.plot([xA, xB], [yA, yB[0]], linewidth=7, color='brown')[0]

#точка крепления пружины и балки
B = ax.plot(xB, yB[0], 'o',color=[1,0,0])[0]

#   диск по точкам, используя уравнение окружности def disk(x, y, r):
      cx = [x + r * np.sin(i / 100) for i in range(0,
      628)] cy = [y + r * np.cos(i / 100) for i in
      range(0, 628)] return (cx, cy)

x_disk = s * np.cos(phi)
#   l - s - r * np.tan(phi) - вертикальное расстояние от точки B до центра диска
#   sin(phi) - учитывает горизонтальное перемещение диска в результате поворота балки.
#   r/cos(phi) - компенсация вертикального перемещения диска вдоль наклоненной балки, что
учитывает наклон балки.
y_disk = (l - s - r * np.tan(phi)) * np.sin(phi) + r / np.cos(phi)

disk_ = ax.plot(disk(x_disk[0], y_disk[0] + yB[0], r)[0], disk(x_disk[0], y_disk[0] +
yB[0], r)[1], 'green')[0]

def kadr(i):
    B.set_data(xB, np.abs(yB[i]))
    Pruzzhina.set_data(xP + xB, np.abs(yP*(yB[i])))
    Stick.set_data([xA, xB], [yA, np.abs(yB[i])])
    disk_.set_data(disk(x_disk[i], y_disk[i] + yB[i], r)[0], disk(x_disk[i], y_disk[i]
+ yB[i], r)[1])
    return [B, Pruzzhina, Stick, disk]
```

```
kino = FuncAnimation(fig,kadr,interval = 10,frames=len(t))

plt.show()
```

Результат работы:







Вывод: В ходе выполнения лабораторной работы, проинтегрировав систему дифференциальных уравнений движения системы, создал анимацию движения системы. В этом мне помог Python и библиотеки matplotlib, numpy и scipy. Также я построил графики законов движения.