Додаток А

Код програми

```
import tkinter as tk
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

# Параметри маніпулятора
l1 = 30
l2 = 25
l3 = 25

# Маси ланок (в кг)
m1 = 1.0
m2 = 0.8
m3 = 0.6

# Початкові кути повороту (в градусах)
theta1 = 0
theta2 = 0
theta3 = 0
phi = 0  # Кут для вертикального підйому

# Початкові координати
x = l1 + l2 + l3
y = 0
z = 0
```

```
    # Функція обчислення координат за новими кутами
    def calculate_coordinates():
        global x, y, z
        x = (l1 * np.cos(np.radians(theta1)) +
             l2 * np.cos(np.radians(theta1 + theta2)) +
             l3 * np.cos(np.radians(theta1 + theta2 + theta3))) *
np.cos(np.radians(phi))
        y = (l1 * np.sin(np.radians(theta1)) +
             l2 * np.sin(np.radians(theta1 + theta2)) +
             l3 * np.sin(np.radians(theta1 + theta2 + theta3))) *
np.cos(np.radians(phi))
        z = (l1 * np.cos(np.radians(theta1)) +
             l2 * np.cos(np.radians(theta1 + theta2)) +
             l3 * np.cos(np.radians(theta1 + theta2 + theta3))) *
np.sin(np.radians(phi))

    # Функція розрахунку кінетичної енергії
    def calculate_kinetic_energy():
        omega1 = np.radians(theta1)
        omega2 = np.radians(theta2)
        omega3 = np.radians(theta3)

        ke1 = 0.5 * m1 * (l1 * omega1)**2
        ke2 = 0.5 * m2 * (l2 * omega2)**2
        ke3 = 0.5 * m3 * (l3 * omega3)**2

        return ke1, ke2, ke3

    # Функція оновлення графіку
```

```
    def update_plot():
        fig = plt.figure()
        ax = fig.add_subplot(111, projection='3d')
        ax.set_xlim(-80,  80)
        ax.set_ylim(-80,  80)
        ax.set_zlim(-80,  80)
        ax.plot([0, l1 * np.cos(np.radians(theta1)) * np.cos(np.radians(phi))],
                [0, l1 * np.sin(np.radians(theta1)) * np.cos(np.radians(phi))],
                [0, l1 * np.sin(np.radians(phi))], color='r', marker='o')
        ax.plot([l1 * np.cos(np.radians(theta1)) * np.cos(np.radians(phi)),
                 (l1 * np.cos(np.radians(theta1)) + l2 * np.cos(np.radians(theta1 + theta2))) * np.cos(np.radians(phi))],
                [l1 * np.sin(np.radians(theta1)) * np.cos(np.radians(phi)),
                 (l1 * np.sin(np.radians(theta1)) + l2 * np.sin(np.radians(theta1 + theta2))) * np.cos(np.radians(phi))],
                [l1 * np.sin(np.radians(phi)),
                 (l1 * np.sin(np.radians(phi)) + l2 * np.sin(np.radians(phi)))], color='g', marker='o')
        ax.plot([(l1 * np.cos(np.radians(theta1)) + l2 * np.cos(np.radians(theta1 + theta2))) * np.cos(np.radians(phi)),
                 (l1 * np.cos(np.radians(theta1)) + l2 * np.cos(np.radians(theta1 + theta2)) + l3 * np.cos(np.radians(theta1 + theta2 + theta3))) * np.cos(np.radians(phi))],
                [(l1 * np.sin(np.radians(theta1)) + l2 * np.sin(np.radians(theta1 + theta2))) * np.cos(np.radians(phi)),
                 (l1 * np.sin(np.radians(theta1)) + l2 * np.sin(np.radians(theta1 + theta2)) + l3 * np.sin(np.radians(theta1 + theta2 + theta3))) * np.cos(np.radians(phi))],
                [(l1 * np.sin(np.radians(phi)) + l2 * np.sin(np.radians(phi))),
```

```
                (l1 * np.sin(np.radians(phi)) + l2 * np.sin(np.radians(phi)) + l3 *
np.sin(np.radians(phi)))], color='b', marker='o')
        ax.scatter([x], [y], [z], color='black', marker='o', label='End effector')
        ax.legend()
        plt.show()

    # Функція для зміни кутів повороту
    def change_angles(theta_num, angle_change):
        global theta1, theta2, theta3, phi
        if theta_num == 1:
            theta1 += angle_change
        elif theta_num == 2:
            theta2 += angle_change
        elif theta_num == 3:
            theta3 += angle_change
        elif theta_num == 4:
            phi += angle_change
        calculate_coordinates()
        update_coordinates()
        update_kinetic_energy()
        update_plot()

    # Функція для оновлення координат в інтерфейсі
    def update_coordinates():
        coordinates_label.config(text=f"Поточні координати захватного
пристрою: x = {x:.2f}, y = {y:.2f}, z = {z:.2f}")

    # Функція для оновлення кінетичної енергії в інтерфейсі
    def update_kinetic_energy():
```

```
    ke1, ke2, ke3 = calculate_kinetic_energy()
    kinetic_energy_label.config(text=f"Кінетична енергія ланок: KE1 = {ke1:.2f} Дж, KE2 = {ke2:.2f} Дж, KE3 = {ke3:.2f} Дж")

# Створення GUI
root = tk.Tk()
root.title("Управління триланковим маніпулятором")

# Функції для зміни кутів за допомогою кнопок
def button1_pressed():
    change_angles(1, 10)

def button2_pressed():
    change_angles(1, -10)

def button3_pressed():
    change_angles(2, 10)

def button4_pressed():
    change_angles(2, -10)

def button5_pressed():
    change_angles(3, 10)

def button6_pressed():
    change_angles(3, -10)

def button7_pressed():
    change_angles(4, 10)
```

```
    def button8_pressed():
        change_angles(4, -10)

    # Кнопки управління
    button1 = tk.Button(root, text="Збільшити кут 1", command=button1_pressed)
    button1.pack()

    button2 = tk.Button(root, text="Зменшити кут 1", command=button2_pressed)
    button2.pack()

    button3 = tk.Button(root, text="Збільшити кут 2", command=button3_pressed)
    button3.pack()

    button4 = tk.Button(root, text="Зменшити кут 2", command=button4_pressed)
    button4.pack()

    button5 = tk.Button(root, text="Збільшити кут 3", command=button5_pressed)
    button5.pack()

    button6 = tk.Button(root, text="Зменшити кут 3", command=button6_pressed)
    button6.pack()

    button7 = tk.Button(root, text="Підняти (Z)", command=button7_pressed)
```

```
button7.pack()

button8 = tk.Button(root, text="Опустити (Z)", command=button8_pressed)
button8.pack()

# Мітка для виведення координат
coordinates_label = tk.Label(root, text=f"Поточні координати захватного пристрою: x = {x:.2f}, y = {y:.2f}, z = {z:.2f}")
coordinates_label.pack()

# Мітка для виведення кінетичної енергії
kinetic_energy_label = tk.Label(root, text="Кінетична енергія ланок: KE1 = 0.00 Дж, KE2 = 0.00 Дж, KE3 = 0.00 Дж")
kinetic_energy_label.pack()

# Початкові координати маніпулятора
calculate_coordinates()
update_coordinates()
update_kinetic_energy()
update_plot()

root.mainloop()
```

ДОДАТОК Б

Демонстраційний матеріал

Формат	Позиція	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість	Примітки
			Текстові документи		
А4	1	ГЮІК 501310.043 ПЗ	Пояснювальна	63	с.
			записка		
			Інші документи		
	2		Додаток А	7	с.
			Код програми		
	3		Додаток Б	1	с.
			Демонстраційний		
			матеріал		
			ГЮІК 501310.043 ПЗ		

Змін	Арк.	№ докум.	Підпис	Дата	Моделювання триланкового маніпулятора типової модульної конструкції з механічними шарнірами	Літера	Аркуш	Аркуш
Розробив		Шкітін Д.О.				Н	1	1
Перевірив		Демська Н.П.				Кафедра КІТАР ХНУРЕ		
Норм. контроль		Бронніков А.І.						
Затв.		Невлюдов І.Ш.						