MANUAL TECNICO

Este código es una interfaz gráfica simple creada con Tkinter y Matplotlib en Python que nos sirve para visualizar y realizar grafos sin importar la cantidad de aristas o de vértices, con este código también podemos realizar búsquedas en grafos utilizando los algoritmos de búsqueda en anchura y profundidad dependiendo de cual queramos realizar. A continuación, se realizará una explicación más detallada de cada parte del código:

```
import tkinter as tk
import networkx as nx
import matplotlib
```

- **Importación de Bibliotecas:** realizamos la importación adecuada de las bibliotecas necesarias para que todo corra con normalidad.

```
matplotlib.use("TkAgg")
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
from matplotlib.figure import Figure
```

- **Configuración de Matplotlib para Tkinter**

```
G = nx.Graph()
```

- **Creación del grafo:** con este código daremos inicialización a la creación de los grafos.

```
root = tk.Tk()
root.title("Algoritmos de Busqueda en Anchura y Profundidad")
```

- **Inicialización de la ventana Principal:** Con la utilización de Tkinter inicialara la venta general del interfaz donde podr4emos ver el grafo y la creación de este, al igual que todos los botones necesarios para que funcione correctamente.

- **Creación de WIDGETS:** Este el código necesario para mostrar partes de la importast3es de la interfaz

Entradas de texto (Entry) – Nos será de utilidad para agregar los vértices y aristas necesarias para nuestro grafo, donde podremos ingresar cualquiera que nosotros queramos.

Etiquetas (Label) – Nos servirá para mostrar la información sobre los vértices que ingresamos junto a sus aristas.

Botones (Button) – Sera el código necesario para la creación de los botones que agregaran los vértices y aristas, dibujar el grafo y ejecutar los algoritmos de búsqueda que decidimos.

```
def informacion(infoV):
    infoV = añadir_vertice.get()
    labelV["text"] = "Vertice Inicial: " + infoV
```

- **Función para mostrar información sobre el vértice inicial:** Esta parte nos mostrara los daros más importantes y necesarios del primer vértice que es el que más importancia tendrá dependiendo del algoritmo a usar.

```
grafica = Figure(figsize=(4, 4))
axes = grafica.add_subplot(111)
```

- **Configuración de la figura de Matplotlib para dibujar el grafo:** Aquí podremos modificar lo necesario para la ventana de la interfaz que nos mostrara el grafo conforme lo vamos creando.

```
def dibujar_grafo(algoritmo=None):
    axes.clear()
```

- **Función para dibujar el grafo:** Dibuja el grafo con las aristas resaltadas por si se especifica un algoritmo de búsqueda sea mas fácil identificar cada una de ellas.

```
def busqueda_ancho():
    busqueda_ancho_recorrido=list(nx.bfs_edges(G,source=añadir_vertice.get()))
    dibujar_grafo(busqueda_ancho_recorrido)
    canvas.draw()

1 usage
def busqueda_largo():
    busqueda_largo_recorrido = list(nx.dfs_edges(G,source=añadir_vertice.get()))
    dibujar_grafo(busqueda_largo_recorrido)
    canvas.draw()
```

- **Funciones para realizar la búsqueda en anchura y profundidad:** Cada código cumple con su funcionalidad de realizar la búsqueda como sea solicitada.

```
bfs_button = tk.Button(root, text="Busqueda en Anchura", cursor="hand2", command=busqueda_ancho)
bfs_button.pack()
dfs_button = tk.Button(root, text="Busqueda en Profundidad", cursor="hand2", command=busqueda_largo)
dfs_button.pack()
```

- **Configuración de los botones para ejecutar las búsquedas:** Son los botones que nos permitirá realizar la búsqueda de las 2 formas existentes.

```
root.mainloop()
```

- **Bucle principal de la interfaz gráfica**