

**Universidad de Panamá.**

**Centro Regional Universitario de Coclé.**

**Facultad de Informática, Electrónica y Comunicación.**

**Lic. En Ingeniería en Informática.**

**Grupo: G02-2.**

**Asignatura: Informática Teórica #1.**

**Proyecto Semestral.**

**Profesor: Luis Domínguez.**

**Estudiantes: Rubén Henríquez 8-984-1414**

**Marco Joseph 8-968-1362**

**Año**

**2023**

**Proyecto semestral**

**Librerías que se utilizaron en este proyecto:**

**1. networkx:Esta librería se utiliza para la creación, manipulación y estudio de estructuras, dinámicas y funciones de redes complejas. Es especialmente útil para trabajar con grafos y realizar análisis de redes.**

**2. matplotlib.pyplot: Esta es una librería de visualización en 2D en Python que se utiliza para crear gráficos y visualizaciones, como diagramas, gráficos de dispersión, histogramas, etc. En este caso, se importa para visualizar grafos creados con NetworkX.**

**3. tkinter:Es una librería gráfica de interfaz de usuario (GUI) estándar de Python. Proporciona herramientas para crear interfaces gráficas y ventanas en las aplicaciones. Aquí, se importa como "tk" para trabajar con widgets y elementos de interfaz gráfica.**

**4. ttk:Es un módulo adicional de tkinter que proporciona widgets temáticos (themed widgets). Los widgets temáticos son una versión mejorada de los widgets estándar de tkinter y proporcionan una apariencia más moderna.**

**Puedo decir que, estas librerías se utilizan para crear y visualizar redes complejas (networkx y matplotlib.pyplot) y para construir interfaces gráficas de usuario (tkinter y ttk).**

**Código del proyecto**

import networkx as nx

import matplotlib.pyplot as plt#Importamos las librerias

import tkinter as tk

from tkinter import ttk

def calcular\_ruta():

# Obtenemos los nodos de inicio y destino desde las opciones creadas en la interfaz gráfica

inicio = combo\_inicio.get()

destino = combo\_destino.get()

# Calcular la ruta más corta y su longitud

camino\_corto = nx.shortest\_path(Grafo, inicio, destino, weight='weight')

tiempo = nx.shortest\_path\_length(Grafo, inicio, destino, weight='weight')

# leyenda con la ruta más corta y su longitud

leyenda = f"El Camino Corto es: {camino\_corto} y toma {tiempo} Kilómetros"

label\_resultado.config(text=leyenda)

# Creamos y mostramos el gráfico con la ruta más corta

pos = nx.spring\_layout(Grafo)

# Resaltamos la ruta más corta

edge\_list = [(camino\_corto[i], camino\_corto[i + 1]) for i in range(len(camino\_corto) - 1)]

# Con esto cambiamos el color de todos los nodos a lightblue

node\_colors = ['lightblue' for \_ in Grafo.nodes]

# Crear un mapeo de nodos a índices numéricos

node\_to\_index = {node: index for index, node in enumerate(Grafo.nodes)}

# Cambiar el color del nodo de inicio a rojo

node\_colors[node\_to\_index[inicio]] = 'yellow'

# Cambiar el color del nodo de destino a rosa

node\_colors[node\_to\_index[destino]] = 'pink'

# Dibujar nodos con etiquetas de distancia

nx.draw(Grafo, pos, node\_size=1300, node\_color=node\_colors, font\_size=8, font\_weight='bold', with\_labels=True)

nx.draw\_networkx\_edge\_labels(Grafo, pos, edge\_labels=nx.get\_edge\_attributes(Grafo, "weight"))

# Resaltar la ruta más corta con color rojo

nx.draw\_networkx\_edges(Grafo, pos, edgelist=edge\_list, edge\_color='red', width=2)

# Mostrar el título del gráfico

plt.title("Grafo con la Ruta Más Corta Entre Los Distritos de la Provincia de Coclé")

# Guardar el gráfico como una imagen PNG

plt.savefig("Graph.png", format="PNG")

# Mostrar el gráfico

plt.show()

# Creación el grafo

Grafo = nx.Graph()

Grafo.add\_node("Aguadulce")

Grafo.add\_node("Penonomé")

Grafo.add\_node("Antón")

Grafo.add\_node("Natá")

Grafo.add\_node("La Pintada")

Grafo.add\_node("Olá")

Grafo.add\_node("Cienega Vieja")

Grafo.add\_node("Río Grande")

Grafo.add\_node("Mata Palo")

Grafo.add\_node("Llano Marin")

Grafo.add\_node("Llano Grande")

Grafo.add\_edge("Antón", "Penonomé", weight=34.9)

Grafo.add\_edge("Penonomé", "La Pintada", weight=18.3)

Grafo.add\_edge("La Pintada", "Olá", weight=44.5)

Grafo.add\_edge("Natá", "Olá", weight=37.6)

Grafo.add\_edge("Natá", "Aguadulce", weight=19.3)

Grafo.add\_edge("Antón", "Natá", weight=78.2)

Grafo.add\_edge("Cienega Vieja","Penonomé", weight=20.5)

Grafo.add\_edge("Río Grande","Aguadulce", weight=33.2)

Grafo.add\_edge("Mata Palo","La Pintada", weight=39.1)

Grafo.add\_edge("Llano Marin","Natá", weight=52.4)

Grafo.add\_edge("Antón","Llano Grande", weight=35.1)

# interfaz gráfica

ventana = tk.Tk()

ventana.title("Calculadora de Ruta Más Corta En la provincia de Coclé")

# etiquetas y widgets para nodos de inicio y destino

etiqueta\_inicio = ttk.Label(ventana, text="Lugar de Inicio:")

etiqueta\_inicio.grid(column=0, row=0, padx=10, pady=10)

combo\_inicio = ttk.Combobox(ventana, values=list(Grafo.nodes), width=15)

combo\_inicio.grid(column=1, row=0, padx=10, pady=10)

etiqueta\_destino = ttk.Label(ventana, text="Destino:")

etiqueta\_destino.grid(column=0, row=1, padx=10, pady=10)

combo\_destino = ttk.Combobox(ventana, values=list(Grafo.nodes), width=15)

combo\_destino.grid(column=1, row=1, padx=10, pady=10)

# botón para calcular la ruta

boton\_calcular = ttk.Button(ventana, text="Mostrar Ruta", command=calcular\_ruta)

boton\_calcular.grid(column=0, row=2, columnspan=2, pady=10)

# Etiquetas para mostrar el resultado

label\_resultado = ttk.Label(ventana, text="")

label\_resultado.grid(column=0, row=3, columnspan=2, pady=10)

# Iniciamos el bucle de la interfaz gráfica

ventana.mainloop()

**Imagen del proyecto**

Diagrama

Descripción generada automáticamente