**Instrucciones**1.Implementa el algoritmo de similitud mediante diagramas de transición.

2.- Utiliza el ejemplo visto en clase para verificar su funcionamiento.

3.- Inventa otros dos textos y verifica su similitud.

Entregables: código fuente y PDF con los programas funcionando.

import numpy as np

# Función para calcular la matriz de transición de un texto

def calcular\_matriz\_transicion(texto):

  # Obtener el conjunto de caracteres únicos en el texto

  caracteres\_unicos = list(set(texto))

  # Crear una matriz de transición de tamaño NxN, donde N es el número de caracteres únicos

  matriz\_transicion = np.zeros((len(caracteres\_unicos), len(caracteres\_unicos)))

  # Calcular las frecuencias de transición de caracteres

  for i in range(len(texto) - 1):

    caracter\_actual = texto[i]

    siguiente\_caracter = texto[i + 1]

    indice\_caracter\_actual = caracteres\_unicos.index(caracter\_actual)

    indice\_siguiente\_caracter = caracteres\_unicos.index(siguiente\_caracter)

    matriz\_transicion[indice\_caracter\_actual][indice\_siguiente\_caracter] += 1

  # Comprobar si hay filas con suma cero en la matriz de transición

  filas\_con\_suma\_cero = np.where(matriz\_transicion.sum(axis=1) == 0)[0]

  if len(filas\_con\_suma\_cero) > 0:

    # Establecer las filas con suma cero en la matriz de transición a una probabilidad uniforme

    matriz\_transicion[filas\_con\_suma\_cero, :] = 1 / matriz\_transicion.shape[1]

  # Normalizar las frecuencias para obtener probabilidades de transición

  matriz\_transicion = matriz\_transicion / matriz\_transicion.sum(axis=1, keepdims=True)

  return matriz\_transicion

# Función para calcular la similitud entre dos textos basada en matrices de transición

def calcular\_similitud(texto1, texto2):

  # Calcular las matrices de transición para ambos textos

  matriz\_transicion1 = calcular\_matriz\_transicion(texto1)

  matriz\_transicion2 = calcular\_matriz\_transicion(texto2)

  # Alinear las matrices de transición al tamaño máximo

  max\_size = max(matriz\_transicion1.shape[0], matriz\_transicion2.shape[0])

  matriz\_transicion1 = ajustar\_tamaño\_matriz(matriz\_transicion1, max\_size)

  matriz\_transicion2 = ajustar\_tamaño\_matriz(matriz\_transicion2, max\_size)

  # Calcular la diferencia absoluta entre las matrices de transición

  diferencia\_absoluta = np.abs(matriz\_transicion1 - matriz\_transicion2)

  # Calcular la similitud como la inversa de la suma de las diferencias absolutas

  similitud = 1 / (1 + np.sum(diferencia\_absoluta))

  return similitud

# Función para ajustar el tamaño de una matriz de transición

def ajustar\_tamaño\_matriz(matriz, nuevo\_tamaño):

  filas, columnas = matriz.shape

  matriz\_ajustada = np.zeros((nuevo\_tamaño, nuevo\_tamaño))

  matriz\_ajustada[:filas, :columnas] = matriz

  return matriz\_ajustada

# Función para ingresar la cantidad de textos a comparar

def cantidad\_textos(cantidad):

  lista = []

  for x in range(cantidad):

    x = input(f"Ingresa texto [{x + 1}]: ")

    lista.append(x)

  return lista

# Función para calular las similitudes entre los textos ingresados

def calcular\_similitudes(lista):

  for x in range(len(lista)):

    similitud = calcular\_similitud(lista[0], lista[x])

    print(f"Similitud entre texto[1] y texto[{x + 1}]: {similitud}")

# Función main que da flujo al programa

def main():

  cantidad = int(input("Ingresa cuanto textos quieres comparar: "))

  lista = cantidad\_textos(cantidad)

  calcular\_similitudes(lista)

# Arranque de programa

if \_\_name\_\_ == "\_\_main\_\_":

  main()

Este programa está hecho de forma modularizada.

Utilice la librería numpy como apoyo para el manejo de las matrices.

Lo dividí en 5 funciones de algoritmo y 1 función de arranque.

calcular\_matriz\_transicion: Función para calcular la matriz de transición de un texto

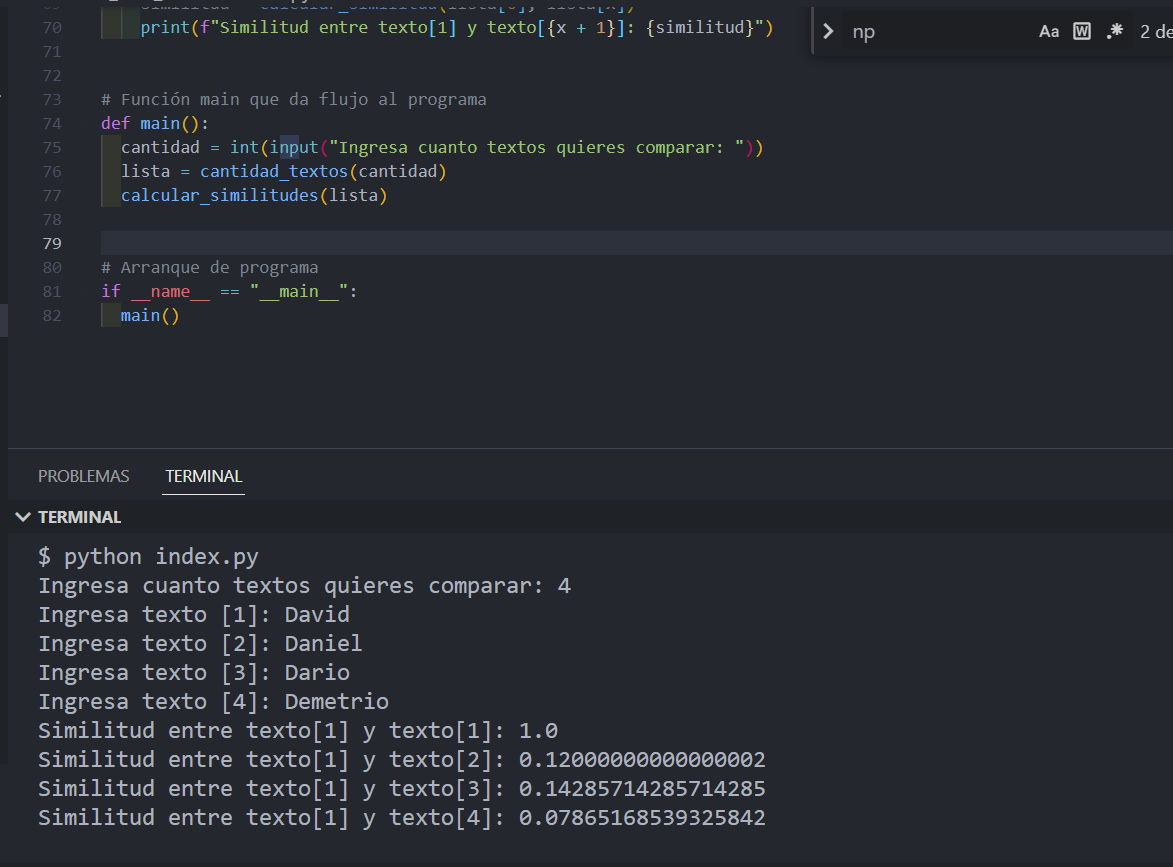
calcular\_similitud: Función para calcular la similitud entre dos textos basada en matrices de transición

ajustar\_tamaño\_matriz: Función para ajustar el tamaño de una matriz de transición

cantidad\_textos: Función para ingresar la cantidad de textos a comparar

calcular\_similitudes: Función para calular las similitudes entre los textos ingresados

Resultados:



En conclusión, utilizar similitud mediante diagramas de transición ofrece una forma visualmente clara y efectiva de analizar datos, identificar patrones, realizar comparaciones y comunicar información de manera comprensible. Esto puede conducir a una mejor toma de decisiones y una comprensión más profunda de los conceptos o fenómenos estudiados.

