1 Matrix Operations with Streamlit

Welcome to the Matrix Operations app powered by Streamlit! This interactive tool allows you to perform various operations on matrices, including displaying the original matrix, its transpose, and its triangular forms.

Simply upload a text file containing the matrix data, and the app will automatically process it. You can visualize the original matrix, its transpose, as well as its upper and lower triangular forms.

Explore different matrix operations and visualize their results easily with this user-friendly interface. Whether you're a student studying linear algebra or a professional working with matrices, this app provides a convenient way to perform matrix operations on the go.

```python
import streamlit as st

class Matriz:
    def __init__(self, data):
        self.__fred = data

    def show_matriz(self):
        return "\n".join([" ".join(map(str, fila)) for fila in self.__fred])

    def transpuesta(self):
        max_longitud = max(len(fila) for fila in self.__fred)
        matriz_cuadrada = [fila + [0] * (max_longitud - len(fila)) for fila in self.__fred]
        transpuesta = []
        for i in range(max_longitud):
            fila_transpuesta = []
            for fila in matriz_cuadrada:
                fila_transpuesta.append(fila[i])
            transpuesta.append(fila_transpuesta)
        return transpuesta

    def suma(self):
        total = 0
        for fila in self.__fred:
            total += sum(fila)
        return total

    def promedio(self):
        total_elementos = sum(len(fila) for fila in self.__fred)
        return self.suma() / total_elementos

    def matriz_triangular_inferior(self):
        dim = len(self.__fred)
        matriz_triangular = []

        st.write("LECTURA DE ASIGNACION DE MATRIZ")
        for i in range(dim):
            fila_triangular = []
            for j in range(dim):
                if j >= i:  # visualiza solo matriz triangular inferior
                    fila_triangular.append(self.__fred[i][j])
                else:
                    fila_triangular.append(0)
            matriz_triangular.append(fila_triangular)

        return matriz_triangular

    def matriz_triangular_superior(self):
        dim = len(self.__fred)
        matriz_triangular = []

        st.write("LECTURA DE ASIGNACION DE MATRIZ")
        for i in range(dim):
            fila_triangular = []
            for j in range(dim):
                if j < dim - i:  # visualiza solo matriz triangular superior izquierda
```

```
                    fila_triangular.append(self.__fred[i][j])
                else:
                    fila_triangular.append(0)
            matriz_triangular.append(fila_triangular)

        return matriz_triangular

def main():
    st.title("Matriz Triangular App")

    uploaded_file = st.file_uploader("Elija un archivo de texto con la matriz", type="txt")

    if uploaded_file is not None:
        datos = []
        for linea in uploaded_file:
            fila = list(map(int, linea.decode("utf-8").strip().split()))
            datos.append(fila)

        m = Matriz(datos)

        st.header("Matriz Original")
        st.text(m.show_matriz())

        st.header("Matriz Transpuesta")
        transpuesta = m.transpuesta()
        for fila in transpuesta:
            st.text(" ".join(map(str, fila)))

        st.header("Matriz Triangular Superior")
        matriz_triangular_superior = m.matriz_triangular_superior()
        for fila in matriz_triangular_superior:
            st.text(" ".join(map(str, fila)))

        st.header("Matriz Triangular Inferior")
        matriz_triangular_inferior = m.matriz_triangular_inferior()
        for fila in matriz_triangular_inferior:
            st.text(" ".join(map(str, fila)))

        st.header("C lculos")
        st.write(f"Suma de todos los elementos: {m.suma()}")
        st.write(f"Promedio de los elementos: {m.promedio()}")

if __name__ == "__main__":
    main()
```

Listing 1: wrk1.py

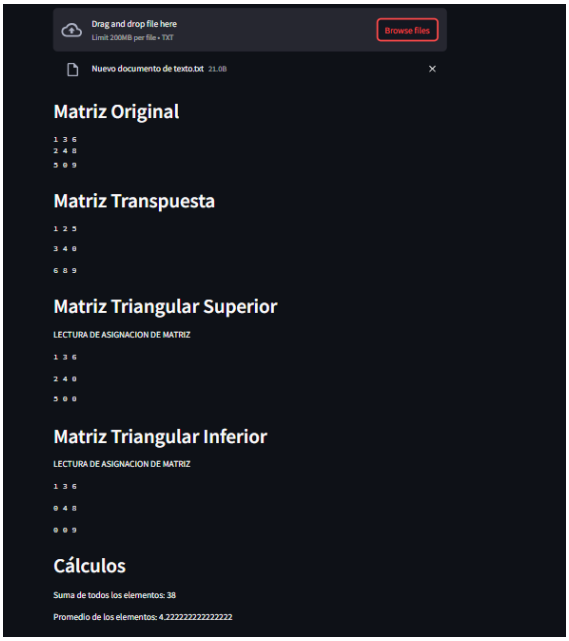

Figure 1: Resultados del código.

Figure 2: Link QR.