Práctica 4

Objetivos:

Seguir trabajando con clases y además con matrices utilizando la librería *numpy* para realizar operaciones básicas como creación, modificación, visualización, suma y resta de matrices en 1D y 2D.

Implemente la siguiente clase:

```
import numpy as np

class CMatFloat:
    """
    Clase que representa una matriz dinámica 1D/2D.

    Atributos:
        _Matriz       # Almacena la matriz (utilice numpy)
        _m_nFilas     # Almacena el número de filas de la matriz
        _m_nColumnas  # Almacena el número de columnas de la matriz
    """

    def __init__(self):
        """
        Método para inicializar el atributo matriz con None
        Y los atributos filas y columnas a 0.
        """
        pass

    def CrearMatriz2D(self, nFilas, nColumnas):
        """
        Método para crear una matriz bidimensional de ceros.
        Asigna valores de filas y columnas según parámetros.
        """
        pass

    def CrearMatriz1D(self, nElementos):
        """
        Método para crear una matriz unidimensional de ceros.
        Usa CrearMatriz2D para asignar 1 fila y n columnas.
        """
        pass

    def Introducir(self):
        """
        Método para introducir los elementos de la matriz.
```

```
        Los elementos de la matriz son de tipo decimal.
        """
        pass

    def Mostrar(self):
        """
        Método para mostrar los elementos de la matriz.
        """
        pass

    def Existe(self):
        """
        Método que verifica si matriz está creada y no está vacía.
        Retorna True si existe, de lo contrario retorna False.
        """
        pass
```

Se implementarán las siguientes funciones que se reutilizarán en prácticas futuras:

```
def leer_int(mensaje="Introduce un número entero: "):
    """
    Función auxiliar que lee un número entero del teclado.
    Si se introduce un valor no válido, se solicita de nuevo.
    """
    Parámetros:
        mensaje (str): El mensaje que se muestra al usuario
                       solicitando la entrada.

    Retorna:
        int: El valor entero introducido por el usuario.

    pass

def leer_float(mensaje="Introduce un número decimal: "):
    """
    Función auxiliar que solicita al usuario un número decimal.
    Si se introduce un valor no válido, se solicitar de nuevo.

    Parámetros:
        mensaje (str): El mensaje que se muestra al usuario
                       solicitando la entrada.

    Retorna:
        float: El valor decimal introducido por el usuario.
    """
```

```
    pass

def crear_menu(opciones_menu):
    """
    Función que muestra un menú de opciones y solicita al usuario
    que seleccione una opción válida.

    Parámetros:
        opciones_menu (list): Lista de opciones a mostrar en el
                              menú.

    Retorna:
        int: El número de opción seleccionado por el usuario.
    """
    pass
```

El programa incluirá un menú principal con las siguientes opciones:

1. Construir matriz 1D: permite crear un vector unidimensional con un número definido de elementos.
2. Construir matriz 2D: permite crear una matriz bidimensional definiendo el número de filas y columnas.
3. Introducir matriz: permite al usuario ingresar los valores para la matriz creada.
4. Mostrar matriz: muestra los valores actuales de la matriz.
5. Operaciones con matrices: Lleva al usuario a un submenú donde se pueden realizar las operaciones de suma y resta de matrices.
6. Terminar: finaliza el programa.

Para la opción 5 del menú principal, habrá un submenú de Operaciones con matrices con las siguientes opciones:

1. Sumar matrices: solicita al usuario una segunda matriz y suma ambas matrices, mostrando el resultado.
2. Restar matrices: solicita al usuario una segunda matriz y resta ambas matrices, mostrando el resultado.
3. Volver al menú principal: regresa al menú principal.

Estructura del programa

- Cree un archivo llamado *matriz.py* para la clase CMatFloat y las funciones relacionadas.
- Implemente una función main() que ejecute el menú.