```python
# Practica 5. Patrones de diseño

class Logger:
    #Creamos un atributo de clase donde se guarda la unica instancia
    _instancia = None

    # __new__ es el metodo que controla la creación del objeto antes de init. Sirve para asegurarnos de que solo exista una unica
    # instancia de la clase Logger
    def __new__(cls, *args, **kwargs):
        # *args es un argumento posicional que permite reciber multiples parametros.
        # **kwargs permite cualquier cantidad de parametros con nombre
        #Validar si existe o no la instancia aun:
        if cls._instancia is None:
            cls._instancia = super().__new__(cls) #Creamos instancia de logger
            # Agregando un atributo "archivo" que apunta a un archivo físico
            # "a" significa appened = Todo lo que se escriba se agrega al final del archivo.
            cls._instancia.archivo = open("app.log", "a")
        return cls._instancia #Devolvermos siempre la misma instancia

    def log(self, mensaje):
        #Simulando un registro de logs
        self.archivo.write(mensaje + "/n")
        self.archivo.flush() #Método para guardar en el disco

logger1 = Logger() #Creamos la primera y unica instancia
logger2 = Logger() #Devolver la misma instancia, sin crear una nueva

logger1.log("Inicio de sesión en la aplicación")
logger2.log("El usuario se utenticó")

# Comprobar que son el mismo objeto de memoria
print(logger1 is logger2) #Devuelve true o false



#Actividad de la practica


class Presidente:
     _instancia = None

     def __new__(cls, nombre):
        if cls._instancia is None:
            cls._instancia = super().__new__(cls)
            cls._instancia.nombre = nombre
            cls._instancia.historial = []
        return cls._instancia

     def accion (self, accion):
        evento = f"{self.nombre} {accion}"
        self.historial.append(evento)
        print(evento)

#Varios presidentes inytentar tomar el poder
p1 = Presidente ("AMLO")
p2 = Presidente ("Penanieto")
p3 = Presidente ("Fox")

#Todos apuntan al mismo presidente
p1.accion("firmo decreto")
p2.accion("visito USA")
p3.accion("Aprobo presupuesto")


print("\n Historial del presidente:")
print(p1.historial)


#Validacion de singleton
print(p1 is p2 is p3)#True o False



#1. ¿Que pasaria si eliminamos la verificacion id cls._instancia is None en el metodo new?
#Se crearian nuevas instancias cada vez y ya no sería un Singleton. p1, p2 y p3 serían distintos.

#2. ¿Que significa el "True" en p1 is p2 is p3 en el contexto del metod singleston?
#Que son la misma instancia.

#3. ¿Es buena idea usar Singleston para tod lo que sea global? Menciona un ejemplo donde no seria recomendable.
#No. No siempre conviene porque puede dar problemas.
#Ejemplo: una conexión a base de datos, ahí es mejor tener varias conexiones y no una sola.
```