```
#practica 5. sigleton
#ejemplo de patron de diseño singleton - sistema de registro de logs,


class logger:
    #Atributo de la clase para guardar la unica
    _instancia = None
#metodo de _new_ controla la creacion del objeto antes del init se
asegura que solo exista una instancia de logger
    def __new__(cls):
     if cls._instancia is None:
       cls._instancia = super().__new__(cls)
       #abrimos un archivo de logs en modo "append"
       cls._instancia.archivo = open("app.log","a")
       #devuelve siempre a la misma instancia
     return cls._instancia


    def registro(self,mensaje):
       self.archivo.write(mensaje)
       self.archivo.flush() #forza al archivo para guardarse en el disco

registro1 = logger() # creamos la unica instancia singleton
registro2 = logger() # devuelve la misma instancia, sin crear una nueva

registro1.registro("Inicio de sesion en la aplicacion")
registro2.registro("El regisro se autentico")

print(registro1 is registro2)#true o false
#si me regresa true: es el mismo objeto en el memoria



# 1. Qué pasaría si eliminamos la verificación if cls.instancia is None
en el método __new_?
que el codigo marcaria no match daria error

# 2.   En la línea print(registro1 is registro2).
¿Qué significa que devuelva True en el contexto del Singleton? que si
creo correctamente el texto que se guarda en la memoria

# 3.   ¿Crees que siempre es buena idea usar Singleton para todo lo que
es global?
Da un ejemplo donde no sería recomendable. pues yo dijo que para varias
cosas si es necesario,
pero por ejemplo en cosass donde las instancias sean multiples o varios
identificadores no es recomendable
```