Explicación Detallada del Código de Logging en Python

1. Importaciones

```
import logging
import time
from logging.handlers import RotatingFileHandler
```

- `logging`: Módulo principal para manejar logs en Python.
- `time`: Usado aquí para obtener el tiempo actual en cada iteración del bucle.
- `RotatingFileHandler`: Una clase de `logging` que permite rotar archivos de log cuando alcanzan cierto ta

2. Crear el RotatingFileHandler

```
handler = RotatingFileHandler(
    filename="logs/log.out",
    maxBytes=1024,
    backupCount=3
)
```

- `filename="logs/log.out"`: Establece el nombre y ubicación del archivo de log. En este caso, el archivo de
- `maxBytes=1024`: Define el tamaño máximo en bytes del archivo de log antes de que ocurra la rotación.
- `backupCount=3`: Indica cuántas copias de respaldo mantener. Al alcanzar el tamaño máximo, el archivo

3. Crear el Logger

```
logger = logging.getLogger(__name__)
logger.addHandler(handler)
logger.setLevel("DEBUG")
```

- `logging.getLogger(__name__)`: Crea o recupera un logger asociado al módulo actual, utilizando `__nam
- `logger.addHandler(handler)`: Añade el `RotatingFileHandler` al logger, lo que significa que los mensajes
- `logger.setLevel("DEBUG")`: Establece el nivel de logging. Aquí está en `DEBUG`, por lo que registrará n

4. Generar Mensajes de Log

```
for _ in range(20):
    st = "The time is now {}".format(time.time())
    logger.debug(st)
```

- Este bucle genera 20 mensajes de log. Cada mensaje contiene el tiempo actual, obtenido con `time.time(
- Al ejecutar el código, cada mensaje se guarda en el archivo de log especificado. Si el archivo supera los