

## PRÁCTICA N7 - THREADING VS MULTIPROCESSING EN PYTHON

### 1. ¿Qué es el Global Interpreter Lock (GIL) en Python?

- a. Un mecanismo que permite la ejecución paralela de hilos.
- b. Un bloqueo que asegura que solo un hilo se ejecute a la vez en un proceso de Python.
- c. Una herramienta para mejorar el rendimiento de los hilos en entornos multi- núcleo.
- d. Un sistema para sincronizar la ejecución de varios procesos.

### 2. ¿Cómo afecta el GIL al rendimiento del threading en Python?

- a. Permite la ejecución simultánea de múltiples hilos, aumentando el rendimiento.
- b. No tiene ningún efecto en el rendimiento de los hilos.
- c. Limita la ejecución de los hilos a secuencias concurrentes, no paralelas.
- d. Aumenta el rendimiento de las tareas limitadas por I/O.

### 3. ¿Cuál es la principal ventaja del multiprocessing sobre el threading en Python para tareas intensivas en CPU?

- a. Menor uso de memoria.
- b. Mejor uso de los núcleos de CPU para ejecución paralela.
- c. No requiere sincronización entre procesos.
- d. Reduce la complejidad del código.

### 4. ¿Qué mecanismos utiliza Python para la comunicación segura entre hilos?

- a. Global Interpreter Lock (GIL) y semáforos.
- b. Colas y tuberías (pipes).
- c. Colas y bloqueos (locks).
- d. Multiprocessing y semáforos.

**5. Al elegir entre threading y multiprocessing en Python, ¿qué aspecto NO es correcto considerar?**

- a. El número de núcleos de CPU disponibles.
- b. La cantidad de memoria RAM en el sistema.
- c. La necesidad de ejecución simultánea de código Python.
- d. La compatibilidad con versiones anteriores de Python 2.

**6. ¿Qué afirmación describe mejor los procesos en Python?**

- e. Los procesos comparten el mismo espacio de memoria, facilitando la comunicación entre ellos.
- f. Los procesos en Python no pueden ejecutarse en paralelo debido al GIL.
- g. Cada proceso tiene su propio espacio de memoria y puede ejecutarse en paralelo en diferentes núcleos de CPU.
- h. La creación de procesos es menos costosa en recursos que la creación de hilos.

**7. En el contexto de Python, ¿qué tarea se beneficiaría más del uso de threading en lugar de multiprocessing?**

- a. Un programa que realiza cálculos matemáticos intensivos.
- b. Una aplicación que necesita realizar múltiples operaciones de entrada/salida (I/O) de manera concurrente.
- c. Un proceso que requiere un alto grado de ejecución paralela para aumentar la velocidad.
- d. Una tarea que necesita un gran volumen de intercambio de datos entre procesos.

**8. ¿Cuál es el propósito principal de utilizar colas en la programación multi-hilo en Python?**

- a. Asegurar que solo un hilo se ejecute a la vez para evitar el GIL.
- b. Permitir la ejecución paralela de hilos sin restricciones.
- c. Facilitar la comunicación segura y el intercambio de datos entre hilos.
- d. Aumentar la velocidad de ejecución de las operaciones de entrada/salida (I/O).

**9. ¿Qué mecanismo de sincronización se recomienda para proteger los datos compartidos en un entorno multi-hilo en Python?**

- a. Utilizar el Global Interpreter Lock (GIL) exclusivamente.

- b. Implementar tuberías (pipes) entre hilos.
- c. Usar bloqueos (locks) y semáforos para controlar el acceso a los datos.
- d. Dividir el código en procesos en lugar de hilos para evitar la necesidad de sincronización.

**10. ¿Cuál de las siguientes afirmaciones es verdadera sobre el impacto del GIL en las aplicaciones Python?**

- a. El GIL mejora el rendimiento de las aplicaciones multi-hilo en sistemas multi- núcleo.
- b. El GIL no tiene ningún impacto en las aplicaciones que utilizan multiprocessing.
- c. El GIL asegura que múltiples procesos puedan acceder simultáneamente a objetos de Python sin conflictos.
- d. El GIL incrementa la eficiencia de las operaciones de entrada/salida (I/O) en programas multi-hilo.