Nombre: Eduardo Quetzal Delgado Pimentel

Fecha: 16/02/2024

Código: 217239716

Materia: Computación Tolerante a fallas

An Introduction to Scaling Distributed Python Applications

Objetivo:

Generar un programa utilizando hilos, procesos, demonios y concurrencia.

introducción:

Hilos para tareas concurrentes: Se utilizan hilos para ejecutar tareas en segundo plano, como el demonio que monitorea el estado del juego cada 30 segundos y la ejecución de operaciones costosas, como la generación de números aleatorios, sin bloquear la interfaz de usuario.

Procesos para operaciones pesadas: Se emplean procesos para realizar operaciones costosas que requieren una cantidad significativa de recursos, como la simulación de operaciones matemáticas complejas. Esto asegura que la carga de trabajo se distribuya eficientemente en múltiples núcleos de CPU.

Desarrollo:

Hilos:

En el programa, se utilizan hilos para realizar tareas concurrentes sin bloquear la interfaz de usuario.

Por ejemplo, se utiliza un hilo para ejecutar el demonio que monitorea el estado del juego cada 30 segundos. Este demonio se implementa en el método demonio_estado_juego.

También se utiliza un hilo para ejecutar operaciones costosas, como la generación de números aleatorios, dentro del método ejecutar_operacion_costosa.

Procesos:

Se emplean procesos para realizar operaciones pesadas que requieren una cantidad significativa de recursos de la CPU.

En el programa, se utiliza un proceso para ejecutar la operación costosa de generar una gran cantidad de números aleatorios en el método ejecutar_operacion_costosa_proceso.

Esta operación se realiza en un proceso separado utilizando la clase multiprocessing.Process.

Demonios:

En Python, los demonios son hilos o procesos que se ejecutan en segundo plano, sin bloquear la finalización del programa principal.

En el programa, se implementa un demonio utilizando un hilo para monitorear el estado del juego cada 30 segundos en el método demonio_estado_juego.

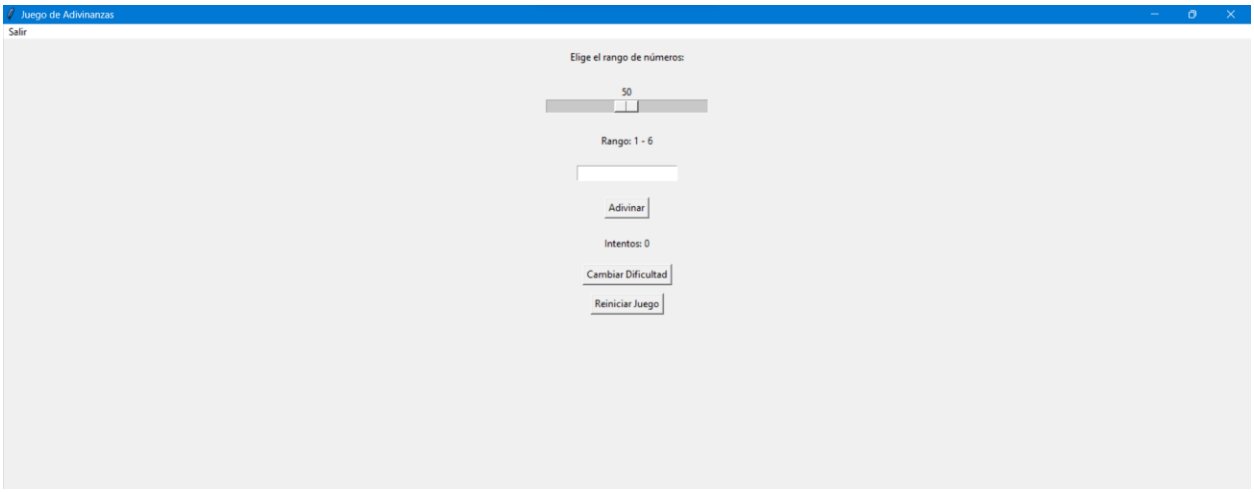
Este demonio imprime un mensaje indicando que está revisando el estado del juego cada vez que se despierta después de un intervalo de tiempo.

Concurrencia:

La concurrencia se refiere a la capacidad de un programa para realizar múltiples tareas simultáneamente.

En el programa, se logra la concurrencia mediante el uso de hilos y procesos para ejecutar tareas en segundo plano mientras la interfaz de usuario sigue siendo receptiva y funcional.

Por ejemplo, mientras el demonio monitorea el estado del juego, el jugador puede seguir interactuando con la interfaz gráfica y realizar acciones como adivinar un número o cambiar la dificultad.



Conclusión:

La implementación de hilos, procesos, demonios y concurrencia en el programa proporciona una experiencia de usuario más fluida y eficiente. Al utilizar hilos, podemos ejecutar tareas en segundo plano sin bloquear la interfaz de usuario, lo que permite una interacción continua y receptiva con la aplicación.

Los procesos son útiles para realizar operaciones pesadas que requieren una cantidad significativa de recursos de la CPU, distribuyendo la carga de trabajo en múltiples núcleos y mejorando así la eficiencia del programa.

Los demonios, tanto en forma de hilos como de procesos, permiten ejecutar tareas en segundo plano de forma periódica sin interferir con la ejecución del programa principal, como la monitorización del estado del juego cada cierto intervalo de tiempo.

En resumen, la combinación de hilos, procesos, demonios y concurrencia mejora la capacidad del programa para manejar múltiples tareas simultáneamente,

proporcionando una experiencia de usuario más receptiva y mejorando la eficiencia general del programa.