¿Qué es un monitor?

Un monitor en programación concurrente es como un guardia de seguridad que cuida un recurso compartido para que solo una persona a la vez pueda usarlo. Técnicamente, es una estructura que combina datos y métodos sincronizados, garantizando que solo un hilo pueda acceder a sus métodos críticos al mismo tiempo. Así se evitan errores cuando varios hilos intentan modificar algo al mismo tiempo.

¿Qué problemática se resolvió al utilizar los monitores?

El uso de monitores resuelve la problemática de la concurrencia descontrolada. Es decir, sin un monitor, varios hilos podrían ejecutar código crítico al mismo tiempo, provocando errores, resultados inesperados o pérdida de datos. Con monitores, logramos que el acceso se dé por turnos y de forma segura.

¿Cuántos procedimientos se ocuparon en el monitor?

En el monitor del problema de los hermanos sembrando árboles, se implementaron tres procedimientos principales, uno para cada hermano:

1. SiembraHermano_1()

2. SiembraHermano_2()

3. SiembraHermano_3()

También se usaron métodos auxiliares como 'debeContinuar()' y 'dormir()'.

Explica con tus propias palabras la estructura y componentes del monitor

La estructura del monitor es como una pequeña clase organizada y protegida que maneja el acceso a un recurso compartido. Está compuesta por:

- Variables compartidas: como el contador de árboles o el turno.

- Métodos sincronizados: que controlan el acceso seguro a esas variables.

- Condiciones de espera (wait y notifyAll): permiten que los hilos esperen su turno y se notifiquenentre ellos.
- Lógica de turnos: que decide qué hilo puede actuar en cada momento.

En conclusión, un monitor es como un coordinador justo que da el turno a cada quien y asegura el orden.