**Jennifer Castro – Arquitectura SO – TUP 2024**

**TP3**

**PARTE 1**

**- ¿Qué se puede notar con respecto al tiempo de ejecución? ¿Es predecible?**

El tiempo de ejecución es variable ya que depende de la velocidad real de la computadora que utilicemos.

No es predecible, porque depende de la velocidad del procesador de cada computadora.

**- Nombrar un proceso o función de la vida real que pueden ser considerados procesos de “máxima velocidad posible” que dependen casi exclusivamente de la velocidad de la máquina que los ejecuta (ej. Ordenar una lista)**

Un ejemplo seria convertir un archivo a otro formato (Por ejemplo pasar un documento de Word a Pdf ) ya que la velocidad de procesamiento depende de la pc que lo ejecute.

**- Nombrar un proceso o función de la vida real que pueden ser considerados procesos de “velocidad de respuesta no dependiente de la velocidad de procesamiento” o que sea de naturaleza impredecible o externa (ej. Leer un archivo externo)**

Un ejemplo seria la descarga de un archivo desde internet, no depende de la velocidad de procesamiento sino de la velocidad de conexión que tenga nuestro servicio de internet.

**PARTE 2**

**¿Qué se puede notar con respecto al tiempo de ejecución? ¿Se mejoró el tiempo de respuesta con respecto al mismo programa sin hilos?**

Se notan variaciones en el tiempo que tarda en responder.

Mejorar el tiempo de demora depe de varias cuestiones, como la cantidad de núcleos de CPU disponibles, la carga del sistema y que tipo de funciones se ejecuten.

El añadir los hilos de ejecución implica una mejora en tiempo de respuesta en situaciones donde las funciones son independientes y pueden ejecutarse simultáneamente.

**- ¿Completan las funciones su ejecución en el orden establecido?**

No siempre se ejecutan en el mismo orden. Eso depende de la programación del sistema operativo y a la asignación de recursos de CPU, por lo que el orden en que se completan las funciones puede variar entre ejecuciones del programa.

**- Nombrar un escenario real donde el multi-hilado puede mejorar considerablemente el tiempo de respuesta de un sistema (ej. Carga de una página WEB en un navegador)**

Un escenario real donde el multi-hilado puede mejorar considerablemente el tiempo de respuesta puede ser en un servidor web que maneja múltiples solicitudes de clientes simultáneamente (por ejemplo un banco). Cuando un servidor web utiliza hilos para manejar cada solicitud de manera independiente, puede atender a varios clientes al mismo tiempo, en lugar de procesar las solicitudes una por una.

**Parte 3**

**- ¿Qué se puede notar con respecto al tiempo de ejecución?**

Con respecto al tiempo de ejecución, se puede notar que el programa tarda más tiempo en completarse. Esto se debe a que los subprocesos están utilizando la función “**time.sleep(0)**”. Aunque esta función pausa el subproceso por un breve tiempo, eso hace que el tiempo total de espera se acumule con cada subproceso utilizado.

**- ¿Qué sucede con el valor final del acumulador?**

En cuanto al valor final del acumulador, podría no ser el valor esperado. Esto se debe a la condición de carrera que existe en el código original. La condición de carrera ocurre cuando múltiples subprocesos intentan acceder y modificar una variable compartida al mismo tiempo, lo que puede llevar a resultados impredecibles.

**- ¿Por qué sucede esto?**

En este caso, dado que tanto el sumador como el restador acceden y modifican el acumulador simultáneamente, podrían ocurrir cambios en medio de la actualización del valor de acumulador. Esto significa que un subproceso podría leer el valor de acumulador, luego ser interrumpido, y otro subproceso podría actualizar acumulador antes de que el primer subproceso complete su operación, lo que resulta en una pérdida de actualización.

**- ¿Cómo se puede corregir esta condición de carrera sin dejar de utilizar hilos?**

Para corregir esta condición de carrera, se puede utilizar un mecanismo de sincronización, como un bloqueo, para garantizar que solo un subproceso pueda acceder y modificar acumulador a la vez. Al utilizarlo, nos aseguramos de que no ocurran cambios de contexto en medio de la actualización de acumulador, lo que garantiza la integridad de los datos y produce el resultado esperado.

**Parte 4:**

**- Detección y corrección del problema.**

**¿Qué sucede con el valor final del acumulador?**

Con el uso del bloqueo, se asegura que solo un hilo pueda acceder y modificar la variable acumulador a la vez, evitando que varios hilos intenten cambiar el valor de acumulador simultáneamente, lo que podría causar resultados inesperados o incorrectos.

Por lo tanto, el valor final de acumulador debería ser consistente y predecible. Dado que ambos hilos operan alternadamente, sumando y restando 5 de acumulador en cada iteración, el resultado final debería ser 0, ya que el número total de suma y resta es el mismo.

**- ¿Qué se puede notar con respecto al tiempo de ejecución?**

En cuanto al tiempo de ejecución, aunque el uso de time.sleep(0) en cada iteración aumene el tiempo total, el impacto debería ser menor en comparación con el código original. Esto es porque el bloqueo nos da una ejecución más ordenada y predecible de los hilos, reduciendo la posibilidad de retrasos debido a cambios de contexto inesperados.