Laboratorio de Infraestructura Computacional

Taller 1b

Nombre: Geisson Said Ponce Solano

Código: 201913908

Preguntas Parte 1 + Solución:

1. *¿Al ejecutar el programa, el resultado corresponde al valor esperado?*

- *Sí,* con el uso del *Thread* principal de aplicación se logra el resultado 10000000 que es correspondiente a sumar 1000 veces 10000 (1000*10000 = 10000000).

2. *¿Al ejecutar el programa, el resultado corresponde al valor esperado? Explique.*

- No, esto debido a que cada *Thread* se ejecuta en paralelo con los que también están accediendo a la variable "contador"

3. *Ejecute cinco veces el programa y escriba el resultado obtenido en cada ejecución.*

Ejecución	Valor Obtenido
1	6140000
2	5140013
3	5610000
4	3860947
5	5260000

4. ¿Hay acceso concurrente a alguna variable compartida? Si es así, diga en dónde.

- *Sí,* la variable "**contador"** está siendo usada por todos lo *threads* generando una concurrencia de esa variable. Esto sucede cuando cada *thread* ejecuta el método run() el cual usa la variable **contador.**

Preguntas Parte 2 + Solución:

1. *Ejecute cinco veces el programa y escriba el resultado obtenido en cada ejecución.*

Ejecución	Valor Obtenido	Valor Esperado
1	100363	100363
2	93215	93215
3	94586	94586
4	49749	84987
5	81038	92616

2. ¿Hay acceso concurrente a alguna variable compartida? Si es así, diga en dónde

- *Sí,* a la variable "**mayor**", la cual es exactamente la variable que presenta problemas debido a que depende del orden que se ejecuten los Threads, por lo tanto, crea que el resultado muchas veces no sea el esperado sino un máximo local. Es concurrente justo cuando se llama para comparar el "mayorFila"

y esta última varía en función de la fila que se está analizando que en este caso corresponde a la de cada Thread.

3. *¿Puede obtener alguna conclusión?*

- El uso de variables concurrentes entre threads suele ser peligroso en torno a la aleatoriedad de lo que se está analizando, como en el ejemplo anterior, el algoritmo multithread solo funciona cuando el mayor corresponde al mayor del último thread (que corresponde a una fila) comparado, lo cual implica aleatoriedad en la respuesta. Por otro lado, las variables no concurrentes, es decir, las no usadas por todos los threads, como la de “mayorFila” si funcionan correctamente en todos los casos debido a que cada thread funciona independientemente de forma correcta.