## **Escuela Colombiana de Ingeniería**

### **Arquitecturas de Software**

#### **Laboratorio - Programación concurrente, condiciones de carrera, esquemas de sincronización, colecciones sincronizadas y concurrentes.**

**Integrantes:** Stefany Morón Castro - Juan Pablo Arévalo Merchán

**Ejercicio**

1. **Analice el código para entender cómo hace uso de hilos para crear un comportamiento autónomo de las N serpientes.**
2. **De acuerdo con lo anterior, y con la lógica del juego, identifique y escriba claramente (archivo RESPUESTAS.txt):**
   * **Posibles condiciones de carrera.**
     1. La condición de carrera se da en el momento que dos o más serpientes toman algún objeto que esté dentro del tablero, ya sea comida o turbo o alguno de estos objetos, si dos serpientes llegan al tiempo, ambas cogerán el mismo objeto porque no existe un bloqueo.



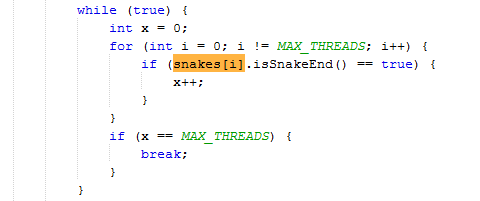
* + 1. Otra posible condición de carrera es cuando dos culebras se estrellan entre sí, actualmente cada una de ellas puede pasar por encima de la otra y no ocurre nada, por lo tanto, las dos están accediendo a la misma celda en el mismo momento.

* + **Uso inadecuado de colecciones, considerando su manejo concurrente (para esto, aumente la velocidad del juego y ejecútelo varias veces hasta que se genere un error).**



Como vemos en el error anterior, existe una falla en la utilización del LinkedList de las serpientes, pues en algún momento de la ejecución se está intentando acceder (cuando se están dibujando las serpientes en el tablero) y al mismo tiempo se está modificando algún dato dentro de ellos.

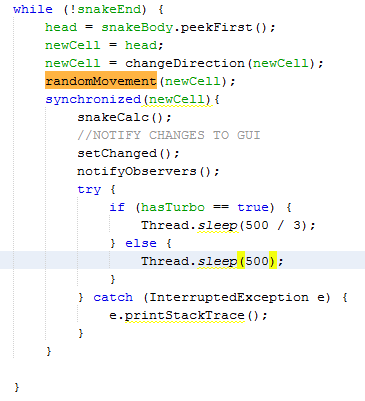
* **Uso innecesario de esperas activas.**



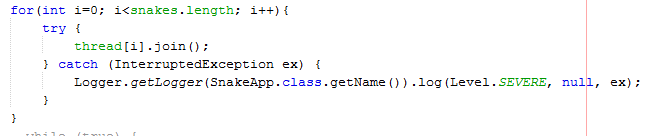
En este método encontramos que se está utilizando una espera activa innecesaria, ya que cada ciclo está preguntando si todas las serpientes están muertas, lo que hace que el consumo de CPU sea constante e innecesario. Ya que lo único que necesitamos es saber el momento en que todas las serpientes mueran (Todos los hilos finalicen).

**3. Identifique las regiones críticas asociadas a las condiciones de carrera, y haga algo para eliminarlas. Tenga en cuenta que se debe sincronizar estrictamente LO NECESARIO. En su documento de respuestas indique, la solución realizada para cada ítem del punto 2. Igualmente tenga en cuenta que en los siguientes puntos NO se deben agregar más posibles condiciones de carrera.**

Nosotros nombramos 2 condiciones de carrera, ambas las hemos solucionado sincronizando los hilos, utilizando como llave la celda por donde la serpiente pasará. Para esto tenemos que sacar del método “snakeCalc()” la función de obtener la siguiente celda, para que nos sirva de llave. Esto nos permite controlar y bloquear los demás intentos de las serpientes de ingresar a esta celda, en el momento en que la serpiente suelte la llave, las otras serpientes podrán ingresar a la celda, pero ya no existirá el elemento que habia ahi, por lo tanto 2 o más serpientes no obtendrán el mismo objeto 2 veces.

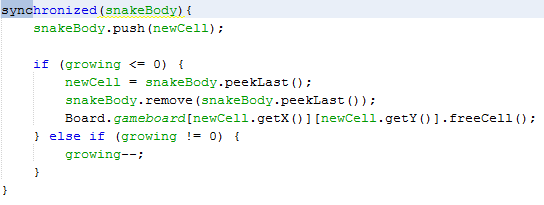


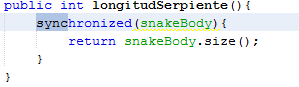
Para las esperas activas, lo que vamos a realizar será un join de todos los hilos, eso nos garantiza, que cuando todas las serpientes se mueran (se acaban los hilos), se terminará el juego, por lo tanto no tenemos que realizar un while(true) ya que esto consume mucho recurso del computador.

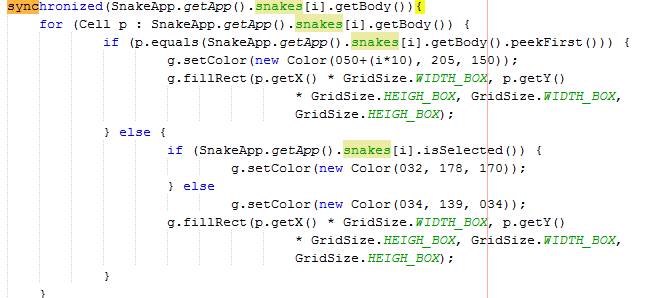


Para el uso inadecuado de colecciones lo que hicimos, fue sincronizar el uso de la colección que maneja el cuerpo de la serpiente, para ello, en las partes del código donde consultaba o modificar este arreglo, se utilizaba un synchronized, con la llave que sería la misma colección (es decir snakeBody)









4. Como se puede observar, el juego está incompleto. Haga los ajustes necesarios para que a través de botones en la interfaz se pueda Iniciar/Pausar/Reanudar el juego: iniciar el juego no se ha iniciado aún, suspender el juego si está en ejecución, reactivar el juego si está suspendido. Para esto tenga en cuenta:

* + Al pausar (suspender) el juego, en alguna parte de la interfaz (agregue los componentes que desee) se debe mostrar:
    - La serpiente viva más larga.
    - La peor serpiente: la que primero murió.
  + Recuerde que la suspensión de las serpientes NO es instantánea, y que se debe garantizar que se muestre información consistente.