




| | | |
|--|-------------------------------|-------------------------------|
|  | VICERRECTORADO DOCENTE | Código: GUIA-PRL-001 |
| | CONSEJO ACADÉMICO | Aprobación: 2016/04/06 |
| Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación | | |

| | | | |
|--|--|--|--|
|  | | FORMATO DE INFORME DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA ESTUDIANTES | |
| CARRERA: Computación | | ASIGNATURA: Programación Aplicada | |
| NRO. PRÁCTICA: | | TÍTULO PRÁCTICA: Hilos en java | |
| OBJETIVO ALCANZADO: Identificar los cambios importantes de Java Diseñar e Implementar las nuevas técnicas de programación concurrente Entender cada una de las características de Thread en Java. | | | |
| ACTIVIDADES DESARROLLADAS | | | |
| 1. Para la practica de hilos se pidió crear una aplicación que simule gráficamente el problema de los filósofos comensales. Para este programa se creo tres paquetes <ul style="list-style-type: none"> • Ec.edu.ups.modelo • Ec.edu.ups.vista • Ec.edu.ups.imagenes | | | |
| 2. Ec.edu.ups.modelo En este paquete se crearon todos los objetos que se necesitaran para simular el problema de los filósofos comensales <ul style="list-style-type: none"> • Filosofo • Mesa • Palillo | | | |
| 3. Palillo.java Esta clase simula un palillo de los filósofos, ya que el problema nos dice que cada filosofo tiene un palillo. Atributos: private ReentrantLock llave; para esta clase se creo un constructor en el cual solamente inicializa el atributo “llave” metodos: <ul style="list-style-type: none"> • quitarPalillo: Este método no retorna ningún objeto. Lo único que hace este método es obtener la llave del palillo y lo bloquea para los demás Thread. • devolverPalillo: Este método no retorna ningún objeto. Este método pregunta si la llave esta ocupada, en caso de que si este ocupado desbloquea el palillo y lo retorna para que pueda ser accedido por otros Thread • palilloOcupado: este metodo retorna un boolean. Pregunta si el palillo esta ocupado o no y retorna un boolean. | | | |
| 4. Filosofo.java | | | |

| | | |
|---|------------------------|------------------------|
|  | VICERRECTORADO DOCENTE | Código: GUIA-PRL-001 |
| | CONSEJO ACADÉMICO | Aprobación: 2016/04/06 |
| Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación | | |

Esta clase simula un filosofo en el cual va a tener sus propios atributos y además un palillo. Esta clase implementa la interfaz Runnable

Atributos:

- private int id;
- private Mesa mesa;
- private Palillo izq, der;
- private boolean esZurdo;

para esta clase se creo un constructor en el cual se ha incluido todos los atributos en su parámetro.

Metodos:

- pensar: Este método no retorna ningún objeto. Este método simula cuando un filosofo esta pensando. Obtiene el tiempo que paso pensando y los pasa al método pasarTiempo().
- comer: Este método no retorna ningún objeto. Este método simula cuando un filosofo esta comiendo y lo que hace es llamar al método quitarPalillo() obtiene el tiempo que paso comiendo y lo pasa al método pasarTiempo() por ultimo devuelve los palillos a la mesa.
- run: Este método sobre escribe el método run de la interfaz Runnable. Este método corre en un while y corre siempre y cuando la condición sea True. Por ultimo llama a los métodos pensar() y comer().
- quitarPalillo: Este método tiene una condición en la cual pregunta si el filosofo es zurdo, si es que el filosofo es zurdo recoge el palillo desde la izquierda y luego recoge el de la derecha. En caso de que no sea zurdo recoge los palillos desde la derecha a la izquierda.
- devolverPalillo: Este método tiene una condición en la cual pregunta si el filosofo es zurdo, si es que el filosofo es zurdo libera el palillo desde la izquierda y luego libera el de la derecha. En caso de que no sea zurdo libera los palillos desde la derecha a la izquierda.
- pasarTiempo: Este método recibe un objeto de tipo Long en el cual detiene el Thread por el lapso recibido en su parámetro.

5.

6. Mesa.java

Esta clase simula una mesa en la cual los filósofos estarán sentados comiendo o pensando. Esta clase implementa la interfaz Runnable.

Atributos:


- private List<Palillo> palillos;
- private List<Filosofo> filosofos;
- private Iterator<Long> times;

Esta clase tiene un constructor en el cual recibe el numero de filósofos a crear que el usuario haya ingresado. Una vez que recibe el numero de filósofos primero comprueba que el numero de filósofos sea mayor que dos. En caso de que no sea mayor que dos un error aparecerá diciendo que no pueden existir pocos filósofos. En caso de que el numero de filósofos sea mayor a dos se inicializa las listas que se tienen como atributos. una vez inicializado las listas se recorre con un for() y se crea un palillo por filosofo y se crean de una vez el filosofo y se los agrega a su lista respectivamente.

Metodos:

- Tiempo: Este método es un método sincronizado en el cual retorna el siguiente elemento en el listado de tiempo.
- Run: Este método crea un ExecutorService() en el cual corre cada filosofo y for each los mandamos a correr.

7. Ec.edu.ups.imagenes

| | | |
|--|-------------------------------|-------------------------------|
|  | VICERRECTORADO DOCENTE | Código: GUIA-PRL-001 |
| | CONSEJO ACADÉMICO | Aprobación: 2016/04/06 |
| Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación | | |

Este paquete solamente contiene las imágenes para poder simular el problema de los filósofos comensales.

8. Ec.edu.ups.vista

Este paquete muestra el procedimiento para los filósofos comensales

- Test

RESULTADO(S) OBTENIDO(S):

Ingrese el numero de filosofos que desea

5

```

filosofo 1 esta pensando por un tiempo de: 4059
filosofo 5 esta pensando por un tiempo de: 4925
filosofo 2 esta pensando por un tiempo de: 4428
filosofo 3 esta pensando por un tiempo de: 3328
filosofo 4 esta pensando por un tiempo de: 4808
filosofo 3 esta comiendo por un tiempo de: 4046
filosofo 1 esta comiendo por un tiempo de: 1939
filosofo 1 esta pensando por un tiempo de: 1792
filosofo 3 esta pensando por un tiempo de: 4419
filosofo 4 esta comiendo por un tiempo de: 3722
filosofo 2 esta comiendo por un tiempo de: 2816
filosofo 1 esta comiendo por un tiempo de: 1258
filosofo 2 esta pensando por un tiempo de: 4545
filosofo 4 esta pensando por un tiempo de: 3903
filosofo 5 esta comiendo por un tiempo de: 4371
filosofo 1 esta pensando por un tiempo de: 3713
filosofo 3 esta comiendo por un tiempo de: 4500
filosofo 1 esta comiendo por un tiempo de: 2917
filosofo 5 esta pensando por un tiempo de: 3749
filosofo 3 esta pensando por un tiempo de: 2867
filosofo 4 esta comiendo por un tiempo de: 1133
filosofo 4 esta pensando por un tiempo de: 4001
filosofo 1 esta pensando por un tiempo de: 2578
filosofo 2 esta comiendo por un tiempo de: 2216

```

Como se puede observar se muestra que en la consola los filósofos comen y piensan mientras el programa corre. De la misma manera también se muestra al final un lapso que el filósofo esta pensando o comiendo.

CONCLUSIONES: En conclusión, se debe haber comprendido como se utilizan hilos para poder correr múltiples procesos a la vez sin caer en alguno error, también se debe haber memorizado los diferentes tipos de Hilos que existen. Se debe haber comprendido que la clase Thread es una clase que se comporta de manera abstracta mientras que la clase Runnable se comporta como una interfaz donde debemos importar todos sus métodos. Por último, también se debió haber repasado como utilizar el método synchronized.

RECOMENDACIONES:

Nombre de estudiante: _____ Denys Dutan

Firma de estudiante: _____