Guarderia-Junio2017Monitores.pdf

TTronc0

Programación de Sistemas y Concurrencia

2º Grado en Ingeniería Informática

Escuela Técnica Superior de Ingeniería Informática
Universidad de Málaga

```
package Junio2017Mon;

import java.util.concurrent.locks.Condition;
import java.util.concurrent.locks.Lock;
import java.util.concurrent.locks.ReentrantLock;

public class Guarderia {

	private int numBebes = 0;
	private int numAdultos = 0;

	private int quiereSalirBebe = 0;
	private int quiereSalirAdulto = 0;

	private Lock l = new ReentrantLock();
	private Condition bebe = l.newCondition();
	private Condition adulto = l.newCondition();

	/**
	 * Un bebe que quiere entrar en la guarderia llama a este
metodo. Debe esperar
	 * hasta que sea seguro entrar, es decir, hasta que cuado
entre haya, al menos,
	 * 1 adulto por cada 3 bebes
	 *
	 */
	public void entraBebe(int id) throws InterruptedException {

		l.lock();

		try {

			System.out.println("El bebe " + id + " quiere
entrar en la guarderia");

			while ((numBebes + 1) > (3 * numAdultos)) {
				System.out.println("El bebe " + id + " debe
esperar para entrar");
				bebe.await();
			}

			numBebes++;
			System.out.println(
					"Entra el bebe " + id + " a la
guarderia. Hay " + numBebes + " bebes y " + numAdultos + "
adultos");

		} finally {
			// TODO: handle finally clause
			l.unlock();
		}
	}
```

```
	/**
	 * Un bebe que quiere irse de la guarderia llama a este
metodo *
	 */
	public void saleBebe(int id) throws InterruptedException {

		l.lock();

		try {

			System.out.println("El bebe " + id + " quiere
salir de la guarderia");

			while ((numBebes - 1) > (3 * numAdultos)) {
				System.out.println("El bebe " + id + " debe
esperar para salir");
				quiereSalirBebe = 1;
				bebe.await();
			}

			numBebes--;
			System.out.println(
					"Sale el bebe " + id + " de la
guarderia. Hay " + numBebes + " bebes y " + numAdultos + "
adultos");
		} finally {
			// TODO: handle finally clause
			l.unlock();
		}
	}

	/**
	 * Un adulto que quiere entrar en la guarderia llama a este
metodo *
	 */
	public void entraAdulto(int id) throws InterruptedException {

		l.lock();

		try {

			numAdultos++;
			System.out.println("Entra el adulto " + id + " a
la guarderia. Hay " + numBebes + " bebes y " + numAdultos + "
adultos");

			if (numBebes <= (3 * numAdultos)) {

				if (quiereSalirBebe == 1) {

					bebe.signal();
```

```
                    }

                    if (quiereSalirAdulto == 1) {

                        adulto.signal();
                    }
                }


            } finally {
                // TODO: handle finally clause
                l.unlock();
            }

        }

        /**
         * Un adulto que quiere irse de la guarderia llama a este
metodo. Debe esperar
         * hasta que sea seguro salir, es decir, hasta que cuando se
vaya haya, al
         * menos, 1 adulto por cada 3 bebes
         *
         */
        public void saleAdulto(int id) throws InterruptedException {

            l.lock();

            try {

                System.out.println("El adulto " + id + " quiere
salir de la guarderia");

                while (numBebes > (3 * (numAdultos -1 ))) {
                    System.out.println("El adulto " + id + " debe
esperar para salir");
                    quiereSalirAdulto = 1;
                    adulto.await();
                }

                numAdultos--;
                System.out.println("Sale el adulto " + id + " de
la guarderia. Hay " + numBebes + " bebes y " + numAdultos + "
adultos");
            } finally {
                // TODO: handle finally clause
                l.unlock();
            }

        }

}
```