

Bloque 4. Paradigmas de la Programación

Tarea 1: Ejercicio camareros y cocineros utilizando Monitores.

A tener en cuenta:

- Usar únicamente locks implícitos y monitores
- Es un productor-consumidor.
- Tendremos 4 clases: Main, 2 clases para el productor (camarero) y el consumidor (cocinero) y por último la clase compartida Estantería.
- Main: crea el objeto compartido, 2 productores y 2 consumidores, y los lanza .
- Estantería: objeto compartido con operaciones de escribirComanda y leerComanda.
- Productor: genera una comanda y la inserta en el ArrayList.
- Consumidor: consume una comanda y la elimina del ArrayList.

Código Clase Main:

```
public class Main {

    public static void main(String[] args) {
        Estanteria est = new Estanteria(20); //Clase común al productor y consumidor
        //2 Hilos para la clase Productora
        Camarero ca1 = new Camarero("Camarero 1", est);
        Camarero ca2 = new Camarero("Camarero 2", est);
        //2 Hilos para la clase Consumidora
        Cocinero co1 = new Cocinero("Cocinero 1", est);
        Cocinero co2 = new Cocinero("Cocinero 2", est);

        ca1.start();
        ca2.start();
        co1.start();
        co2.start();
    }
}
```

Código Clase Productor - Camarero:

```
import static java.lang.Thread.sleep;

public class Camarero extends Thread {

    private String id;
    private Estanteria est;

    public Camarero(String id, Estanteria est) {
        this.id = id;
        this.est = est;
    }

    public void run() {
        String comanda;
        int i = 0;
```

```

while (true) { // El camarero no se sabe cuántas comandas llevará

    try {
        comanda = id + " ---->" + i;
        i++;
        est.escribirComanda(comanda);
        sleep(500);
    } catch (InterruptedException e) {
    }
}
}
}
}

```

Código Clase Consumidor - Cocinero:

```

import static java.lang.Thread.sleep;

public class Cocinero extends Thread {

    private String id;
    private Estanteria est;

    public Cocinero(String id, Estanteria est) {
        this.id = id;
        this.est = est;
    }

    public void run() {
        String comanda;
        while (true) { // El cocinero no se sabe cuántas comandas llevará
            try {
                comanda = est.leerComanda();
                System.out.println("El " + id + " prepara la comanda para el " + comanda); //Imprimiremos 20
                lecturas de temperatura
                sleep(400);
            } catch (InterruptedException e) {
            }
        }
    }
}
}

```

Código Clase Estantería:

```

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

public class Estanteria {

    private ArrayList<String> arl; //Array de comandas
    private int maximo = 0, in = 0, out = 0, numElem = 0;

    public Estanteria(int max) {
        this.maximo = max;
        this.arl = new ArrayList<>();
    }

    public synchronized void escribirComanda(String comanda) throws InterruptedException {
        while (numElem == maximo) { //Si el ArrayList está lleno... Espera
            wait();
        }

        //Vamos añadiendo las comandas de la clase productor-cocinero
        arl.add(comanda);
        numElem++;
        in = (in + 1) % maximo;
        notifyAll(); //El ArrayList ya no está vacío
    }

    public synchronized String leerComanda() throws InterruptedException {
        while (numElem == 0) { //Si el ArrayList está vacío... Espera
            wait();
        }
        String comanda = arl.get(0);
        arl.remove(0);
        numElem = numElem - 1;
        out = (out + 1) % maximo;
        notifyAll(); //El ArrayList ya no está lleno
        return (comanda);
    }
}

```

Resultado:

```

El Cocinero 1 prepara la comanda para el Camarero 2 ---->7
El Cocinero 2 prepara la comanda para el Camarero 1 ---->7
El Cocinero 1 prepara la comanda para el Camarero 1 ---->8
El Cocinero 2 prepara la comanda para el Camarero 2 ---->8

```

El Cocinero 2 prepara la comanda para el Camarero 1 ---->9
El Cocinero 1 prepara la comanda para el Camarero 2 ---->9
El Cocinero 2 prepara la comanda para el Camarero 2 ---->10
El Cocinero 1 prepara la comanda para el Camarero 1 ---->10
El Cocinero 2 prepara la comanda para el Camarero 2 ---->11
El Cocinero 1 prepara la comanda para el Camarero 1 ---->11
El Cocinero 2 prepara la comanda para el Camarero 1 ---->12
El Cocinero 1 prepara la comanda para el Camarero 2 ---->12
El Cocinero 1 prepara la comanda para el Camarero 1 ---->13
El Cocinero 2 prepara la comanda para el Camarero 2 ---->13
BUILD STOPPED (total time: 7 seconds)