

# Ayuda para Práctica 2

## Espera activa y señales con sincronización

Descarga este documento en:

Enlace corto (puede no funcionar desde la facultad): <http://goo.gl/Eo2yM>

Enlace largo: <https://www.dropbox.com/s/fg0pk0i8lythoip/Ayuda%20Practica%202.pptx>

Más ayuda: en tutorías o  
Pablo.Bermejo@uclm.es

- Recordad poner cada ejercicio en un paquete distinto.
- Cuidado con la ruta de las clases que se llaman igual y están en paquetes distintos!

## Ejercicio 1.A.

- Cread las clases Cola, Productor y Consumidor copiando el código que aparece en el guion de prácticas, cread una clase principal que lance un productor y un consumidor con **la misma cola**. Fijaos que la salida es incorrecta.
- Para obtener una salida correcta, hay que:
  - Sincronizar a Productor y Consumidor
- Para ello, hay que trabajar sobre el recurso compartido: Cola

# Ejercicio 1.A

## Salida que hay que conseguir

Coloco: 0  
Obtengo: 0  
Coloco: 1  
Obtengo: 1  
Coloco: 2  
Obtengo: 2  
Coloco: 3  
Obtengo: 3  
Coloco: 4  
Obtengo: 4  
Coloco: 5  
Obtengo: 5  
Coloco: 6  
Obtengo: 6  
Coloco: 7  
Obtengo: 7  
Coloco: 8  
Obtengo: 8  
Coloco: 9  
Obtengo: 9

# Ejercicio 1.A.

## Sincronizar con Espera Activa (*busy wait*)

```
class ColaBusyWait{
    int n;

    //piensa cuál debe ser el primer valor de esta variable
    boolean turnoConsumidor= ...;

    public int get() {
        //espera activa: no hacer nada mientras que turnoConsumidor tenga el valor...
        while (...) { }
        System.out.println("Obtengo: " + n);
        //cambio el valor de tocaConsumir
        tocaConsumir = ...;
        return n;
    }

    public void put(int n) {

        //espera activa: no hacer nada mientras que turnoConsumidor tenga el valor...
        while (...) { }
        this.n = n;
        System.out.println("Coloco: " + n);
        //cambio el valor de tocaConsumir
        tocaConsumir = ...;
    }
}
```

## Ejercicio 1.B.

### Sincronizar con Señales

- Hay que añadir el modificador *synchronized* a los métodos, lo cual asegura exclusión mutua.
- Para convertir la espera activa en sincronización con señales:
  - Cambiar el *while infinito* por un *if*
  - Poner a la espera al hilo con *wait()* dentro del *if*
  - Tras cambiar el valor de la variable lógica, usar *notify()* para mandar una señal al hilo que está esperando.

## Ejercicio 2

### Sincronizar con Señales más de 2 hilos

- Tenemos 2 consumidores y 1 productor.
- Gracias a *synchronized* tenemos exclusión mutua pero:
  - notifyAll() despierta a todos los hilos durmiendo
  - por lo que **no se puede asegurar *fairness* usando solo señales.**
- La salida debe leer siempre el último valor de la cola, pero no podremos controlar quién lee ni cuándo
- Hay que mezclar *synchronized*+señales+espera activa

# Ejercicio 2

## Sincronizar con Señales más de 2 hilos

```
public class ColaConSeñales2Consumidores{  
    int n;  
    int turno=0;
```

```
    public synchronized int get(int id) {  
        //espera activa que hace wait() mientras no sea el turno indicado por id  
        ...  
        System.out.println("[ "+id+" ]Obtengo: " + n + " para Consumidor "+Thread.currentThread().getName());  
        turno--;  
        notifyAll();  
        return n;  
    }  
}
```

```
    public synchronized void put() {  
        //espera activa que hace wait() mientras el turno no sea 0  
        ...  
        System.out.println("Coloco: " + ++n);  
        turno+=2;  
        notifyAll();  
    }  
}
```

## Ejercicio 2 - Salida deseada

Coloco: 1  
[2]Obtengo: 1 para Consumidor 2  
[1]Obtengo: 1 para Consumidor 1  
Coloco: 2  
[2]Obtengo: 2 para Consumidor 2  
[1]Obtengo: 2 para Consumidor 1  
Coloco: 3  
[2]Obtengo: 3 para Consumidor 2  
[1]Obtengo: 3 para Consumidor 1  
Coloco: 4  
[2]Obtengo: 4 para Consumidor 2  
[1]Obtengo: 4 para Consumidor 1  
Coloco: 5  
[2]Obtengo: 5 para Consumidor 2  
[1]Obtengo: 5 para Consumidor 1  
Coloco: 6  
[2]Obtengo: 6 para Consumidor 2  
[1]Obtengo: 6 para Consumidor 1  
Coloco: 7  
[2]Obtengo: 7 para Consumidor 2  
[1]Obtengo: 7 para Consumidor 1  
Coloco: 8  
[2]Obtengo: 8 para Consumidor 2  
[1]Obtengo: 8 para Consumidor 1  
Coloco: 9  
[2]Obtengo: 9 para Consumidor 2  
[1]Obtengo: 9 para Consumidor 1  
Coloco: 10  
[2]Obtengo: 10 para Consumidor 2  
[1]Obtengo: 10 para Consumidor 1

## Ejercicio 3

<u>Torno 1</u>	<u>Contador</u>	<u>Torno 2</u>
3	5	3

De nuevo, la sincronización ha de implementarse sobre el recurso compartido; en este caso: **clase Contador**, para conseguir que la suma sea correcta:

<u>Torno 1</u>	<u>Contador</u>	<u>Torno 2</u>
4	8	4

# Ejercicio 3.A

## Exclusión mutua con algoritmo de Dekker

```
class ContadorDekker {
    int value=0;
    NumberCanvas display;

    // Variables para Dekker
    volatile boolean[] flags={false,false};
    volatile int turno=0;

    ContadorDekker(NumberCanvas n) {
        display=n;
        display.setvalue(value);
    }

    void incrementa(int id) { //ojo!, tenéis que modificar la clase Torno para crearla con un id y lo use al llamar a people.incrementa
        //Dekker (los ids de los tornos deben ser 0 y 1)
        ... //levanto la bandera para pedir entrar
        while(flags[1-id]){ //mientras el otro tenga la bandera levantada
            if(turno==1-id){ //si es el turno del otro
                ... //yo no puedo aún y tengo que bajar la bandera
                ... // hago espera activa mientras el turno sea el del otro
                ... // el otro ya ha acabado así que puedo pedir entrar volviendo a subir la bandera
            }
        }
        //entro porque ya tengo la bandera a true y el otro la ha bajado y ha cambiado el turno
        //sección crítica
        int temp = value; //read[v]
        CC.FuerzaCC();
        value=temp+1;    //write[v+1]
        display.setvalue(value);

        //Dekker
        //cambio el turno y bajo la bandera
        ...
    }
}
```

## Ejercicio 3.B

### Exclusión mutua con *synchronized*

- Añade a incrementa este modificador y observa qué ocurre.
- ¿Cómo es más sencillo obtener la exclusión mutua, con el alg. de Dekker o con el modificador *synchronized*?

- Me interesa conocer mejor al grupo.
- **Por favor, si tenéis tiempo me gustaría que contestarais de forma anónima este formulario:**

<https://docs.google.com/forms/d/1bROjBwoYFSCvbbYfXL0ccCnr-6gXz0XBQClr6F2nLSw/viewform>

Gracias!