# PROGRAMACION CONCURRENTE (Curso 2023/2024)

## PRÁCTICA 3

**Objetivos de la práctica:**

1. Uso de semáforos
2. Clase *Semaphore* en Java

**Evitar condición de carrera práctica 2 utilizando semáforos.** Este ejercicio consiste en evitar una condición de carrera usando semáforos. Tenemos el mismo número de procesos incrementadores que decrementadores que, respectivamente, incrementan y decrementan, en un mismo número de pasos una variable compartida. El objetivo es asegurar la exclusión mutua en la ejecución de los incrementos y decrementos de la variable y el objetivo es hacerlo utilizando exclusivamente un semáforo de la clase Semaphore.

**Productor-consumidor con semáforos.** Existen procesos de dos tipos diferentes: *Productores*, su hilo de ejecución consiste, repetidamente (puedes poner un número máximo de iteraciones P), en crear un producto y hacerlo llegar a uno de los consumidores. *Consumidores*: su hilo de ejecución consiste en, repetidamente (puedes poner un número máximo de iteraciones C), recoger productos producidos por los productores y consumirlos. La comunicación entre productores y consumidores se realizará través de un "almacén" compartido por todos los procesos. La interfaz de Almacen es:

```
public interface Almacen {
      /**
        * Almacena (como ultimo) un producto en el almacén. Si no hay
        * hueco el proceso que ejecute el método bloqueará hasta que lo
        *  haya.
        */
      public void almacenar(Producto producto);
      /**
        * Extrae el primer producto disponible. Si no hay productos el
        * proceso que ejecute el método bloqueará hasta que se almacene un
        *  dato.
        */
      public Producto extraer();
}
```

En este apartado el tamaño del almacén es uno, es decir puede haber almacenado como máximo un producto. Si un proceso quiere almacenar debe esperar hasta que el almacén esté libre y si un proceso quiere extraer espera hasta que haya un producto.

Para valorar si el problema está bien resuelto, el objetivo es asegurar

- que todos los productos producidos acaban por ser consumidos,
- que no se consume un producto dos veces y
- que no se consume ningún producto no válido

**Recomendación**: probar con diferentes números de productores y consumidores y observar con atención las trazas del programa.

**Extensión a almacén con N productos.** En esta apartado, el almacén a implementar tiene una capacidad de hasta N productos, lo que permite a los productores seguir trabajando aunque los consumidores se vuelvan, momentáneamente, lentos.

**Lectores y escritores con paso de testigo.** Re-implementa el apartado anterior para que los procesos productores se conviertan en procesos **escritores** y los procesos consumidores en **lectores**. Es decir, la operación almacenar de los productores se reemplaza por escribir, que lleva un parámetro adicional indicando la posición del almacén en la que se quiere almacenar el producto: `escribir(Producto producto, int pos)`. La operación extraer de los consumidores se reemplaza por leer, que lleva como parámetro la posición del producto a leer: `leer(int pos)`. Por tanto, los productos no se extraen del almacen, solamente se consultan (el mismo producto puede ser leído por múltiples lectores). También un producto puede ser reescrito por un escritor antes de que éste haya sido consultado.