Ejercicio1: creación de un proceso.

La creación de un proceso necesita de la “ejecución” de un programa. Cuando el proceso creador termina (el programa java), el proceso sigue estando. El scheduler le reasigna otro proceso “padre”.

```
// crear proceso
public class CrearProcesos {
    public void ejecutar(String ruta){
        ProcessBuilder pb;
        try {
                pb = new ProcessBuilder(ruta);
                pb.start();
        } catch (Exception e) {
                e.printStackTrace();
        }
    }
    /**
     * @param args
     */
    public static void main(String[] args) {
        String ruta= "notepad.exe";
        CrearProcesos proc=new CrearProcesos();
        proc.ejecutar(ruta);
        System.out.println("Ha terminado el programa java");
    }
}               // fin clase
```

La creación de un hilo se realiza desde el programa java (proceso java). Es el propio proceso el encargado de gestionar el inicio i fin del hilo. El scheduler no lo hace. Por ello si el proceso creador termina, el proceso hilo continua indefinidamente.

```
public class Hilo extends Thread {

	public void run() {
		while (!interrupted()) {
			System.out.println("Mensaje");
		}
		System.out.println("Fin Hilo");

	}

	public void interrumpir() {
		interrupt();
	}

	public static void main(String[] args) {

		Hilo h=new Hilo();
		h.start();

		for (int i=0; i< 100000000; i++) {
			// Do nothing
		}
		h.interrumpir();
	}
}
```

Ejercicio3: creación de dos hilos

```
public class sub_hilos extends Thread {

	public void inicio(String text) {
		System.out.println("Inicio "+text);
	}

	public void crear(String text) throws InterruptedException {
		System.out.println("crear "+text);

		sub_hilos h3=new sub_hilos();
		h3.inicio(text);
		h3.start();

		sleep(1000);

		h3.interrumpir(text);


	}

	public void run() {
		while (!interrupted()) {
			//System.out.println("Mensaje");
		}

		System.out.println("Fin Hilo");

	}

	public void interrumpir(String text) {
		System.out.println("cerrar "+text);
		interrupt();
	}

	public static void main(String[] args) throws InterruptedException {

		sub_hilos h=new sub_hilos();
		h.inicio("Hilo 1");
		h.crear("sub Hilo 1");
		h.start();

		Thread.sleep(2000);

		sub_hilos h2=new sub_hilos();
		h2.inicio("Hilo 2");
		h2.start();

		Thread.sleep(2000);

		h.interrumpir("Hilo 1");
		h2.interrumpir("Hilo 2");

	}

}
```

```
public class sub_hilos extends Thread {

	public void inicio(String text) {
		System.out.println("Inicio "+text);
	}

	public void crear(String text) throws InterruptedException {
		System.out.println("crear "+text);

		sub_hilos h3=new sub_hilos();
		h3.inicio(text);
		h3.start();

		sleep(1000);

		h3.interrumpir(text);


	}

	public void run() {
		while (!interrupted()) {
			//System.out.println("Mensaje");
		}

		System.out.println("Fin Hilo");

	}

	public void interrumpir(String text) {
		System.out.println("cerrar "+text);
		interrupt();
	}

	public static void main(String[] args) throws InterruptedException {

		sub_hilos h=new sub_hilos();
		h.inicio("Hilo 1");
		h.crear("sub Hilo 1");
		h.start();

		Thread.sleep(2000);

		sub_hilos h2=new sub_hilos();
		h2.inicio("Hilo 2");
		h2.start();

		Thread.sleep(2000);

		h.interrumpir("Hilo 1");
		h2.interrumpir("Hilo 2");

	}

}
```

Ejercicio5: Comunicación entre hilos

Este es el ejercicio comúnmente llamado productor/consumidor. Un hilo escribe en la zona de memoria y el otro lee en la zona de memoria.

```
class Buffer{
	int a;
	boolean produced = false;

	public synchronized void produce(int x){
		if(produced){
			System.out.println("Crea texto espera...");
			try{
				wait();
			}catch(Exception e){
				System.out.println(e);
			}
		}
		a=x;
		System.out.println("Creado :" + a);
		produced = true;
		notify();
	}

	public synchronized void consume(){
		if(!produced){
			System.out.println("Lector esperando...");
			try{
				wait();
			}catch(Exception e){
				System.out.println(e);
			}
		}
		System.out.println(a + " leido.");
		produced = false;
		notify();
	}
}

class Producer extends Thread{
	Buffer b;
	public Producer(Buffer b){
		this.b = b;
	}

	public void run(){
		System.out.println("creador empieza...");
		for(int i = 1; i <= 10; i++){
			b.produce(i);
		}
	}
}

class Consumer extends Thread{
	Buffer b;
	public Consumer(Buffer b){
		this.b = b;
	}

	public void run(){
		System.out.println("Lector empieza...");
		for(int i = 1; i <= 10; i++){
			b.consume();
		}
	}
}

public class comunicacions_hilos {
	public static void main(String args[]){
		//Crear objeto Buffer
		Buffer b = new Buffer();

		//crear thread crea texto
		Producer p = new Producer(b);

		//crear thread lector texto
		Consumer c = new Consumer(b);

		//Iniciar threads.
		p.start();
		c.start();
	}
}
```