EXAMEN SERVICIOS Y PROCESOS

## Teoria:

* ¿Qué método hay que llamar para ejecutar un hilo?
  + Start
* ¿Se puede hacer en una línea la creación de un hilo y ejecutarlo?
* ¿Qué es un socket? (tema 2)
  + Un socket es un túnel de comunicación que ayuda a que aplicaciones se comuniquen.
* ¿Qué se necesita para crear un socket?
  + Url y puerto
* ¿Cuántos tipos de socket existen?
* ¿Para que sirve las funciones wait y notify all?
  + Permiten enviar las señales de sincronización del monitor.
* ¿Se le pueden asignar prioridades a un hilo?
  + Si, pero no se ejecutan como queremos
* ¿Para que sirve ReentrantLock?
  + Sirve para hacer condiciones, nosotros hemos simples y se pueden hacer multiples.
* ¿Cuáles son los estados de proceso?
  + Ejecución: proceso está actualmente en ejecución.
  + Listo: el proceso está listo para ser ejecutado, solo está esperando que el planificador así lo disponga.
  + Bloqueado: el proceso no puede se puede ejecutar hasta que no se produzca cierto suceso, como operación de entrada/salida
  + Nuevo: el proceso se acaba de crear y aun no ha sido admitido por el sistema operativo.
  + Terminado: proceso fue expulsado del grupo de procesos ejecutables, por que termino o por que fallo.
* ¿Cómo crear procesos en java?
  + Con la clase ProcessBuilder
  + Los métodos ProcessBuilder.start () y Runtime.exec crean un proceso nativo y devuelven una instancia de una subclase de *Process* que se puede usar para controlar el proceso y obtener información sobre éste.
  + La clase *Process* proporciona métodos para realizar la entrada del proceso, realizar la salida al proceso, esperar a que el proceso se complete, verificar el estado de salida del proceso y destruir (kill) el proceso.
* ¿Se puede esperar a la finalización de un proceso? Como.
  + Si se puede esperar a la finalización de un proceso, llamando al método

waitfor.

* ¿Qué pasa cuando tienes un proceso dormido y lo interrumpes?
  + Cuando se detiene la ejecución del hilo pueden ejecutarse otros hilos.
* ¿Puedo interrumpir un proceso y lo puedo reinterrumpir dentro de una excepción?
* ¿Qué es un demonio?
  + Es un hilo que antes de crearlo hemos indicado que será ejecutado como demonio. Esto hace que el hilo se ejecute en un segundo plano
  + Tienen un prioridad mas baja
  + Es una cualidad que se hereda es decir que si un hilo demonio crea otro hilo el otro también será un demonio.
  + Una vez el hilo se ha inicializado será siempre demonio.
* ¿Cuáles son las formas de crear un hilo?
  + Extendiendo la clase thread.
    - Crear una nueva clase que herede de la clase thread.
    - Redefinir en la nueva clase el método run con el código asociado al hilo.
    - Crear un objeto de la nueva clase thread. Esta será realmente el hilo.
    - Iniciar el hilo con el método start.
  + Mediante la interfaz runnable.
    - Declara una nueva clase que implemente Runnable
    - Redefinir en la nueva clase el método run con el código asociado al hilo.
    - Crear un objeto de la nueva clase.
    - Crear un objeto de la clase thread pasando como argumento al constructor el objeto cuya clase tiene el método run.
    - Ponerlo en marcha llamando a start.
* ¿Qué es una sección critica?
  + Trozo de código que se accede a un recurso compartido que no debe ser accedido por mas de un proceso o hilo en ejecución.
* ¿Cómo se puede hacer que los trozos de código se ejecuten solo por un proceso (individualmente)?
* ¿Cuál es la función para soltar un semáforo?
  + Release
* Lock y reetrant

## Practica:

* Un hilo y preguntar que falta para que funcione
  + Ejecucion de hilo mediante clase Thread.

public class HelloThread extends Thread {

public void run() {

System.out.println("Hello from a thread!");

}

public static void main(String args[]) {

(new HelloThread()).start();

}

}

* + Mediante la interfaz runable

public class HelloRunnable implements Runnable {

public void run() {

System.out.println("Hello from a thread!");

}

public static void main(String args[]) {

Thread t = new Thread(new HelloRunnable());

t.start();

}

}

* Accept

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

ServerSocket servidor = new ServerSocket(9999);

while (true) {

Socket miSocket = servidor.accept(); PUEDE QUITAR LA LINEA ESTA

DataInputStream flujoEntrada = new DataInputStream(miSocket.getInputStream());

String texto = flujoEntrada.readUTF();

System.out.println(texto);

miSocket.close();

}

}

* Entender cada línea de código que hemos ido haciendo.
* Como se crear procesos en java
* Señalar una sección critica
* Codigo de un semáforo (para que funcione el semáforo hay que pasar al constructor de el numero de hilos).