Programación de Servicios y Procesos. 2018. Primera Evaluación. Recupera Marzo

La NSA ha desarrollado un sistema para decodificar palabras cifradas mediante el uso paralelo de los procesadores de la tarjeta gráfica (Cudas). Para ello generaremos tantos hilos de descifrado como la longitud de la palabra a descifrar-1:

```
//array compartido para guardar los resultados
public static String palabras[] = null;
//Comando contiene la palabra cifrada
String comando = new String("ISACAM");
int maximo_cudas = comando.length()-1;
//Hilos de decodificación
Decode [] cuda = new Decode[maximo_cudas];
```

El hilo de ejecución principal **NSA, debe esperar a que todos los Cudas estén preparados**, imprimiendo "Inicializando los cudas..." y cuando todos los hilos sean creados (sincronizar, p.e. mediante CountDownLatch), entonces imprimirá "Todos los cudas operativos ..." y **les enviará** a todos los cudas **mediante un pipe** la palabra a descifrar y **activará una variable compartida controlada por semáforo (lanzar)** para que los cudas se pongan a descifrar la variable.

```
//variable compartida controlada por semáforo
public static boolean lanzar = false;
```

Tras esto, esperará a que todos los cudas hayan descifrado su palabra (sincronizar) e **imprimirá los valores del array palabras[]** que es dónde los cudas guadarán el resultado de su ejecución.

Una vez inicializados los cudas (son hilos de ejecución, instancias de la clase Decode), informarán a la clase principal NSA que están operativos mediante algún mecanismo de sincronización e imprimirán **"Cuda <nºcuda> listo para decodificar....", leerán la palabra a decodificar por el pipe de entrada y quedarán funcionando a la espera de detectar la orden de iniciar.** Para ello cada cuda leerá la **variable compartida controlada por semáforo (lanzar)** a la espera de que obtenga el valor true.

La clase NSA dispondrá de un array de pipes para comunicar con los cudas. Cada cuda tiene su propio canal de comunicaciones.

```
//Array de comunicaciones
PipedWriter emisor[] = new PipedWriter[maximo_cudas];
PrintWriter flujoS[] = new PrintWriter[maximo_cudas];
```

Una vez que el cuda haya **decodificado la palabra** (ver el código de la **función de descifrar**) **guardará el valor en el array *palabras***, en la posición correspondiente con su número de hilo. **Para acceder al array *palabras* también se utilizará un semáforo**, evitando que 2 o más hilos accedan concurrentemente a esta variable. Tras esto, informará de su finalización mediante algún mecanismo de sincronización y terminará.

```
private String descifrar(int posi, String comando){
        String c1 = comando.substring(posi+1,comando.length());
        String c2 = comando.substring(0,posi+1);
        return (c1+c2);
}
```

Nuestra aplicación constará de la siguientes clases (no son necesarias más clases):

1. **NSA**, que es la aplicación principal que inicia todos los hilos.
2. **Decode**, que es la clase que decodifica las palabra y almacena su resultado.

Criterios de evaluación:

1. Las clases se ha implementado adecuadamente, realizan la función sugerida y los parámetros de las clases son coherentes con su funcionalidad. (2 puntos)

2. La sincronización de espera, tanto para que los cudas se creen, como la espera a que terminen se ha realizado adecuadamente, utilizando las clases de concurrencia que java proporciona. (2 puntos)

3. La sincronización de la generación de los resultados, y la puesta en marcha de los cudas se ha realizado adecuadamente, considerando los problemas de la sección crítica. (2 puntos)

4. La aplicación presenta una lógica coherente y funciona correctamente. (2 puntos)

Entrega del ejercicio

El ejercicio deberá ser entregado en un archivo comprimido con el nombre y apellidos y todos los archivos fuente utilizados.

Para una mejor compresión, se indica detalladamente el **funcionamiento del NSA**

1. Crear hilos, pipes y demás objetos y variables necesarias
2. Inicializar todos los hilos
3. Esperar que todos los hilos se creen y estén listos antes de continuar (SYNC)
4. Crear todos los flujos de comunicación con los hilos (pipes)
5. Enviar por los pipes la palabra a descifrar
6. Activar la variable ***lanzar*** mediante un mecanismo de concurrencia (SYNC)
7. Esperar a que todos los cudas terminen su labor (SYNC)
8. Imprimir el array con todas las palabras descifradas

El funcionamiento del **cuda (instancia de la clase Decode**) es el siguiente:

1. Inicializar variables, semáforos u lo que sea necesario
2. Comunicar que estamos operativos a NSA (SYNC)
3. Leer palabra a decodificar por el pipe de entrada
4. Esperar a que la variable ***lanzar*** tenga el valor true (SYNC)
5. Decodificar la palabra
6. Almacenar el valor decodificado en el array ***palabras*** (SYNC)
7. Esperar un valor aleatorio de hasta 1 sg
8. Notificar a NSA que hemos terminado (SYNC)
9. Finalizar el hilo

Seguidamente y a modo de ejemplo se muestran los posibles **parámetros de la clase Decode**

```
public Decode(int posi,CountDownLatch preparaCudas,CountDownLatch
              finCudas,PipedWriter emisor, Semaphore semaforo)
```