CFGM: Desenvolupament d'aplicacions multimèdia MP09 Programació de serveis i processos PR2.2 Threads 1 (Coordinació)

ACTIVITAT

Objectius:

- Aprendre a programar threads amb JAVA

Instruccions:

- Es tracta d'un treball en grups de dos, no s'admet cap tipus de còpia.
- Responeu a l'espai de cada pregunta, si ho feu amb diapositives enganxeu la diapositiva en aquest mateix espai.
- Es valorarà la cura en la presentació del document i que segueixi l'estructura indicada.

Criteris d'avaluació:

- Cada pregunta té el mateix pes sobre 90%
- Les metodologies de treball, organització personal i participació conten un 10%

Entrega:

- Aquest document amb les explicacions i captures necessàries i els arxius adjunts necessaris del codi que es demana
- El nom dels arxius adjunts a entregar seràn: nomicognom-nomicognom.zip

Noms i Cognoms: Iván Figueredo

Materials:

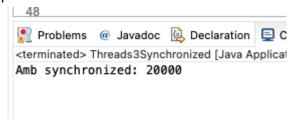
Necessiteu un entorn de desenvolupament en JAVA (per exemple Eclipse) Feu servir Google per buscar els tutorials que us serveixin millor Tens més informació sobre el mètodes d'exclusió mútua en aquest post



Tasques:

Tots els exercicis es basen en fer un programa que:

- 4 fils d'execució
- Cada fil incrementa un contador enter compartit per tots els fils 5.000 vegades
- El programa principal només inicia la variable principal i la mostra al final
- Explica perquè el resultat cada vegada és diferent i menor que 20.000
- Exercici 0 Implementa un programa com el descrit anteriorment, fent ús de exclusió mútua amb 'ThreadsSynchronized' per tal que el resultat final sigui 20.000:
 - Arxiu ThreadsSynchronized.java
 - Sortida esperada:



- Exercici 1 Implementa un programa com el descrit anteriorment, fent ús de exclusió mútua amb 'ReentrantLock' per tal que el resultat final sigui 20.000:
 - Arxiu ThreadsReentrant.java
 - Sortida esperada:

Sortida esperada:





- Exercici 2 - Crea un projecte amb els següents arxius i respon les preguntes:

CampDeTir.java

```
public class CampDeTir {
   public static void main(String[] args) {
      Pistola arma = new Pistola();
      Carregar c = new Carregar(arma, 1);
      Descarregar d = new Descarregar(arma, 1);
      c.start();
      d.start();
   }
}
```

Carregar.java

```
public class Carregar extends Thread {
   private Pistola arma;
   private int cartucho;
   public Carregar(Pistola arma, int cartucho) {
        this.arma = arma;
        this.cartucho = cartucho;
   }
   public void run() {
        for (int i = 0; i < 10; i++) {
            arma.apuntar();
            System.out.println("Apuntar #" + this.cartucho + " bala: " + i);
        }
   }
}</pre>
```

Descarregar.java

```
public class Descarregar extends Thread {
   private Pistola arma;
   private int cartucho;

public Descarregar(Pistola arma, int cartucho) {
     this.arma = arma;
     this.cartucho = cartucho;
}

public void run() {
   for (int i = 0; i < 10; i++) {
      arma.disparar(i);
      System.out.println("Descarregar #" + this.cartucho + " bala: " + i);
   }
}</pre>
```

Pistola.java

```
public class Pistola {
   private boolean enposicio = true;
   public synchronized void disparar(int cartucho) {
      while (enposicio == false) {
         try { wait(); } catch (InterruptedException e) { }
      }
      enposicio = false;
      notifyAll();
   }
   public synchronized void apuntar() {
      while (enposicio == true) {
         try { wait(); } catch (InterruptedException e) { }
      }
      enposicio = true;
      notifyAll();
   }
}
```

Quants fils hi ha en el programa?

En la clase CampDeTir hay dos Threads instanciados y arrancados pero hay un tercero instanciado pero sin arrancar

• Com es diuen?

En la clase CampDeTir hay dos Threads Carregar y Descarregar, Y sin arrancar Pistola.

Per a què serveix la variable enposicio?

Para poder controlar si el programa puede disparar ya que a menos que enposicio este seteado en true el programa no podrá ejecutar el código de la función syncronized "disparar".

Per a què s'invoca al mètode wait()?

Para, en caso de que enposicio sea falso el método pueda esperarse hasta que se vuelva a llamar para poder volver a comprobarlo, ya que se volverá a comprobar ya que está dentro de un while, es por eso que el método se esperará hasta que sea true enposicio.

Per a què s'invoca el mètode notifyAll()?

Para avisar a todos los Threads que estén inactivos (wait) que han de activarse.

El resultat sempre serà el mateix en diferents execucions?



CFGM: Desenvolupament d'aplicacions multimèdia MP09 Programació de serveis i processos PR2.2 Threads 1 (Coordinació)

No, cambiará ligeramente cada vez que lo ejecutemos. Ya que no sabemos en que orden se van a iniciar los Threads.

• Per a què es fa servir el "synchronized"?

No tiene un uso ya que no hemos lockeado el Thread usando syncronized (lock), entonces está mencionado que será synchronized pero no se usará.

Què passaria si no hi hagués en "synchronized"?

Lanza la siguiente excepción:

Lanzamos esta excepción debido a que a la hora de arrancar el Thread no se detecta quien es el propietario del Thread y lanza una excepción de tipo: "IllegalMonitorStateException".