CICLE FORMATIU: CFGS Desenvolupament Aplicacions Multiplataforma DAM

MÒDUL PROFESSIONAL: MP09 Programació de serveis i processos

UNITAT FORMATIVA: UF2. Processos i fils

==========

Pràctica 5. Sincronització multifil.

Treballarem a partir del següent exemple de codi en java:

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();
    }
 }
```

Pistola.java

```
public class Pistola {
    private int cartucho;
    private boolean enposicio = true;

    public synchronized void disparar(int cartucho) {
        while (enposicio == false) {
            try {
                // Esperar a apuntar
                wait();
            } catch (InterruptedException e) { }
        }
        enposicio = false;
        notifyAll();
    }

    public synchronized void apuntar() {
        while (enposicio == true) {
            try {
                // Esperar a disparar
                wait();
            } catch (InterruptedException e) { }
        }
        enposicio = true;
        notifyAll();
    }

}
```

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);
        }
    }
}
```

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);
        }
    }
}
```

Lliura el codi font comentat i un exemple de la seva execució. Contesta a les següents qüestions:

1. Explica detalladament el funcionament general del programa (4 punts).

   El programa el que fa es carregar i descarregar una pistola. El programa executa dos threads el c.start y e d.start aquest dos threads executen dues mètodes que contenen un wait() cadascun i un notifyAll(), per els qual se executen depenen d'un boolean.
   Quan executem el programa veiem com ens surts el resultats barrejat i es perquè el dos threads se executen simultàniament.
   Per entendre millor el programa el que fa es executar el dos threads que executen els mètodes que estan dintre dels for dels threads que son apuntar(), disparar() el motiu per el quan no s'executen en ordre es perquè hi ha vegades que un thread s'executa abans que el altre i això fa que imprimeixi el print que conte el for.

   Exemple: El programa executa el dos thread, un thread executa el mètode que conte dins però aquest mètode entra en espera perquè la seva condició es true y el boolean esta true, doncs aquest thread ni printa el que te el for es queda esperant fins que el altre thread no pugui entrar al seva condició perquè es false y el boolean esta true. Així que canvia el boolean a false i llença el notifyAll() perquè s'executen tots el processos que estan aturats. Perquè mentre un thread esta executen el seu for es possible que el l'altre este donen voltes i estigui deixen molt processos en wait() fins que el altre executa el notifyAll() y fa que tots el processos en wait() terminin i facin el print. Per això hi ha vegades que el print d'un mateix for surt varies vegades perquè hi ha processos del thread en wait().

   També el perquè es possible que el print del segon thread surti abans que el del primer es perquè el primer es possible que arribi després al wait() que el segon thread. Així que el segon thread al tenir el boolean a true no entra, executa el notifyAll() (Que no faria res perquè no hi ha cap wait() i el synchronized faria que no peti el programa) però encara no hi ha cap procés del primer thread en espera. Així que fa print i després torna a entrar i ara si que el primer thread a arribat al wait i esta en espera però encara no ha fet print, però ja finalment el mètode executa el notifyAll() i ja fa print del primer thread.

2. Quants fils hi ha en el programa? Com es diuen?

   Son dos fils els que actuan en aquest programa.
   Un es el c.start(); (Carregar c = new Carregar(arma, 1);
   Altre es d.start(); (Descarrgar d = new Descarregar(arma, 1);

3. Per a què serveix la variable enposicio?

   Serveix per posar els thread en wait() depenen del seu valor. Això el que fa es que hi ha processos que es quedant en espera i després de que aquest boolean canvi el valor mitjançant els mètodes executa el notifyAll() per executar tots el processos que es troben en wait() per això hi ha vegades que surten print molt seguit d'un mateix thread.

4. Per a què s'invoca al mètode wait()?

   Per fer que els thread es quedin en espera fins que un notifyAll() s'executa per llençar-los.

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

S’utilitza per poder despertar als mètodes que estén en wait(). Es a dir, el que fa es posar en marxa de nou als altres mètodes que estiguin en modo wait() (en espera).

6. El resultat sempre serà el mateix en diferents execucions?

   Si, sempre serà diferent perquè un thread pot anar mes ràpid que l’altre així que mentre que un thread esta en el trajecte de posar-se en wait() al entrar el mètode l’altre pot anar fer print. Per això depèn molt de la velocitat que s’estigui executant el thread si en aquell moment es mes ràpid que un o l’altre.

7. Per a què es fa servir el “`synchronized`”? Què passaria si no hi fos?

   El syncronized es fa servir perquè quan un mètode no entra a la condició aniria el notifyAll() i si s’executa el notifyAll() sense que hi hagi un process en wait() petaria. Per això la funció del synchronized es que quan un notifyAll() s’executa sense que hi hagi cap wait() espera a que hagi un wait() per executar-ho.

   Si el synchronized no hi fos el programa petaria ja que s’executarian notifyAll() sense averia cap wait()