



ACTIVITAT
Objectius: <ul style="list-style-type: none">- Aprendre a programar threads amb JAVA
Instruccions: <ul style="list-style-type: none">- Es tracta d'un treball en grups de dos, no s'admet cap tipus de còpia.- Responen 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: <ul style="list-style-type: none">- Cada pregunta té el mateix pes
Entrega: <ul style="list-style-type: none">- 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:

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](#)



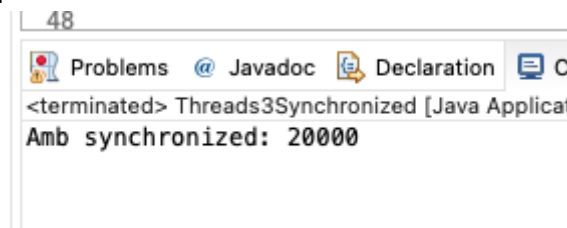
Tasques:

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 "PR220Threads2.java" com el descrit anteriorment, fent ús de exclusió mútua amb 'ThreadsSynchronized' per tal que el resultat final sigui 20.000:

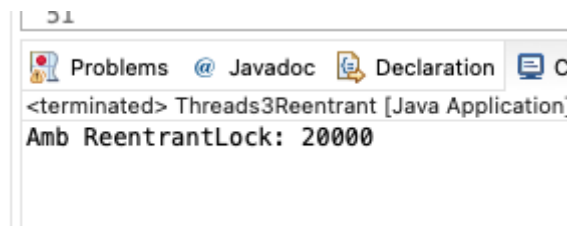
- Arxiu ThreadsSynchronized.java
- Sortida esperada:



- **Exercici 1** - Implementa un programa "PR221Threads2.java" 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);  
        }  
    }  
}
```

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);  
        }  
    }  
}
```

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?
- Com es diuen?
- Per a què serveix la variable enposicio?
- Per a què s'invoca al mètode wait()?
- Per a què s'invoca el mètode notifyAll()?
- El resultat sempre serà el mateix en diferents execucions?
- Per a què es fa servir el "synchronized"?
- Què passaria si no hi hagués en "synchronized"?