Generalitat de Catalunya
Departament d'Educació
Institut Baix Camp

c/Jacint Barrau,1 – 43201 Reus
Tf.977310953 / Fax.977314721
http://www.insbaixcamp.org
email: e3002594@xtec.cat

CFGS - Desenvolupament d'aplicacions multiplataforma
Mòdul 9 – Programació de serveis i processos
UF2 – Processos i fils

Per l'activitat heu de lliurar un zip emb el següent contingut:

- el codi generat (arxiu/s *.java amb comentaris i mètodes separats i clarament diferenciats).
- un arxiu pdf que contindrà exemples de la seva execució i les explicacions que creieu necessàries així com les comandes utilitzades.

L'arxiu .zip s'ha d'anomenar Cognom1Cognom2.nom.zip (p ex: GrangeBorrasFede.zip).
L'arxiu .pdf s'ha d'anomenar Cognom1Cognom2Nom.pdf (GrangeBorrasFede.pdf).

Enllaços d'interes:

Objectius:
Identificar les paraules reservades del llengautge Java per gestió de processos/fils

Activitat 1

A partir del següent codi subministrat, fes els canvis que calgui per a que:

- L'operació ha de ser Sumar en lloc de Multiplicacio.

```
    @Override
    public Integer call() throws Exception {
        return operador1 + operador2;
        }
    }

public static void main(String[] args) throws
```

- Executor ha de llançar 5 fils.

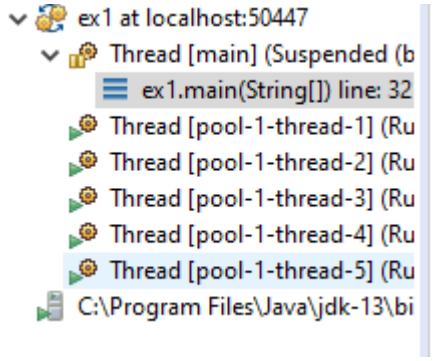
```
ThreadPoolExecutor executor = (ThreadPoolExecutor) Executors.newFixedThreadPool(5);
```

- Cal llençar 25 tasques que sumin números aleatoris.

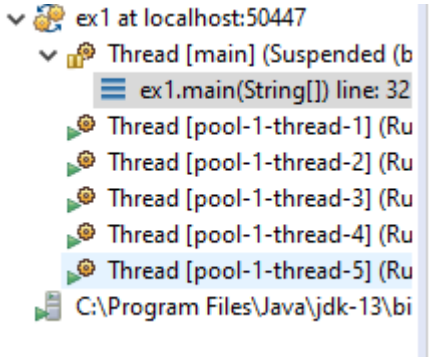
```
for (int i = 0; i < 25; i++) {
    Multiplicacio calcula = new M
    llistaTasques.add(calcula);
    }
```

c/Jacint Barrau,1 – 43201 Reus
Tf.977310953 / Fax.977314721
http://www.insbaixcamp.org
email: e3002594@xtec.cat

- Les tasques es realitzen de manera paral·lela.



- Cal comentar el codi explicant sobretot el que fa referència a concurrència
- Executeu el programa i, mitjançant el debugger mireu els fils que es generen



c/Jacint Barrau,1 – 43201 Reus
Tf.977310953 / Fax.977314721
http://www.insbaixcamp.org
email: e3002594@xtec.cat

```
package multiplicallista;

import java.util.*;
import java.util.concurrent.*;

public class MultiplicaLlista {
        static class Multiplicacio implements Callable<Integer> {
                private int operador1;
                private int operador2;

                public Multiplicacio(int operador1, int operador2) {
                        this.operador1 = operador1;
                        this.operador2 = operador2;
                        }

                @Override
                public Integer call() throws Exception {
                        return operador1 * operador2;
                        }
                }

        public static void main(String[] args) throws
                InterruptedException, ExecutionException {
                        ThreadPoolExecutor executor = (ThreadPoolExecutor) Executors.newFixedThreadPool(3);
                        List<Multiplicacio> llistaTasques= new ArrayList<Multiplicacio>();
                        for (int i = 0; i < 10; i++) {
                                Multiplicacio calcula = new Multiplicacio((int)(Math.random()*10), (int)
(Math.random()*10));
                                llistaTasques.add(calcula);
                                }
                        List <Future<Integer>> llistaResultats;
                        llistaResultats = executor.invokeAll(llistaTasques);

                        executor.shutdown();

                        for (int i = 0; i < llistaResultats.size(); i++) {
                                Future<Integer> resultat = llistaResultats.get(i);
                                try {
                                        System.out.println("Resultat tasca "+i+ " és:" +
                                        resultat.get());
                                }
                                catch (InterruptedException | ExecutionException e)
                                        {
                                        }
                                }
                }
        }
```