**CFGS - Desenvolupament d’aplicacions multiplataforma**

**Mòdul 9 – Programació de serveis i processos**

**UF2 – Processos i fils**

Per l’activitat heu de lliurar un PDF amb el següent contingut:

* Nom o noms dels membres que fan l’activitat (màxim 2 alumnes).
* URL del repositori on hi ha els fonts (github, bitbucket,...).
* Noms d’usuari utilitzats per cada membre del grup del repositori del núvol.
* Explicacions, línies de codi creades/modificades i bolcats de pantalla de l’execució dels diferents algoritmes que es demanen en la tasca.
* La claredat de la documentació lliurada (a banda del codi) serà directament proporcional a la nota obtinguda.

El pdf s’ha d’anomenar seguint el format NomCognom1Cognom2.pdf del membre que fa el lliurament. Si un grup està format per dos membres NOMÉS UN l’ha de lliurar.

Tasques entregades fora de termini i/o on els fonts del repositori del núvol hagin estat creats/modificats fora de termini tindran, com a màxim un 5 sempre que el lliurament i/o la modificació del repositori sigui de, com a molt, 3 dies després del venciment. Passats 3 dies no s’avaluarà la tasca i es qualificarà com a 0.

No s’acceptarà tampoc cap tasca que no es lliure en el format especificat i/o que no contingui de forma clara els punts que es demanen.

**Enllaços d'interès:**

[**https://docs.oracle.com/javase/7/docs/api/java/util/concurrent/ThreadPoolExecutor.html**](https://docs.oracle.com/javase/7/docs/api/java/util/concurrent/ThreadPoolExecutor.html)

[**https://docs.oracle.com/javase/7/docs/api/java/util/concurrent/ScheduledThreadPoolExecutor.html**](https://docs.oracle.com/javase/7/docs/api/java/util/concurrent/ScheduledThreadPoolExecutor.html)

[**https://docs.oracle.com/javase/8/docs/api/java/util/concurrent/ForkJoinPool.html**](https://docs.oracle.com/javase/8/docs/api/java/util/concurrent/ForkJoinPool.html)

[**https://docs.oracle.com/javase/8/docs/api/java/util/concurrent/RecursiveTask.html**](https://docs.oracle.com/javase/8/docs/api/java/util/concurrent/RecursiveTask.html)

**Objectius:**

Identificar les paraules reservades del llenguatge Java per gestió de processos/fils

**Activitat 4**

Heu d’implementar els algoritmes que es detallen a continuació utilitzant la llibreria Executor i el patró proposat en cada una d’elles.

**Alumnat DUAL:**

Heu d'implementar el tercer algoritme del tercer apartat i qualsevol dels dos primers.

**Apartat 1:**

Mitjançant el patró ThreadPoolExecutor heu de simular el moviment dels caixers d’un supermercat. En concret heu de:

* Crear un client cada 3 segons. En total hi haurà 50 clients.
* Cada client porta un nombre aleatori (d’entre 1 i 30) articles i cada article en qüestió triga un temps a passar per caixa (d’entre 2 i 8 segons) ja que hi ja objectes que pesen poc i d’altres que pesen molt. Podeu utilitzar un vector, una llista o un array per representar aquestes dades (feu-ho senzill, dediqueu-hi temps per veure quina estructura és més senzilla!).
* Proveu diferent nombre de fils (1, 2, 5 i 10) per passar els articles per caixa.
* Proveu, amb 10 caixes, i diferent nombre de clients (100 i 500)

El resultat ha de ser similar al següent:

**Creat el client 1 amb 5 articles (2, 8, 5, 3, 4)**

**Client 1 passa per caixa...**

**Client 1 article 1/5 (2 segons)...**

**Creat el client 2 amb 3 articles (5,7)**

**Client 2 passa per caixa...**

**Client 2 article 1/2 (5 segons)...**

**Client 1 article 2/5 (8 segons)...**

**Client 1 article 3/5 (5 segons)...**

**Client 2 article 2/2 (7 segons)...FINALITZAT**

**Client 1 article 4/ (3 segons)...**

**Client 1 article 5/5 (7 segons)...FINALITZAT**

**Apartat 2:**

Heu d’implementar l’algoritme d’ordenació d’un array pel mètode de la bombolla i s’ha d’anar anotant per la sortida estàndard els canvis que s’hi fan durant l’execució.

Un cop ho tingueu, realitzareu l’ordenació d’un array dividint-lo en tantes parts com processadors tingueu en l’equip i a continuació cada fil ordenarà una part. En acabar totes les ordenacions l’algoritme principal mostrarà per pantalla el resultat ordenat.

**Apartat 3:**

Heu d’implementar de forma recursiva (1) i mitjançant fils (patró Fork-Join) amb la classe RecursiveTask (2) dos dels següents algoritmes:

Opció 3.a Calcular el valor de xn (per x0=1, la resta xn=x\*xn-1)

Opció 3.b Calcular el màxim comú divisor mitjançant el teorema d’Euclides

Opció 3.c Calcular els nombres de Catalan

Heu de comparar totes dues execucions (1 i 2) i comprovar quina és més eficient. Dibuixeu una gràfica comparant els dos algoritmes amb diferents valors d’entrada.

Si resulta més eficient la forma recursiva normal, poseu-hi una penalització (algun càlcul a dins de tots dos algoritmes) i comproveu ara quin dels dos és més eficient i a partir de quins valors d’entrada. Dibuixeu una gràfica comparant els dos algoritmes amb diferents valors d’entrada i on es vegi clarament a partir de quin/s valors és més eficient utilitzar fils.