Université Constantine 2-Abdelhamid Mehri
Faculté NTIC- Département TLSI
3éme année Licence Génie Logiciel
Contrôle du module DAC

Constantine, le 12/02/2020

Nom de l'étudiant(e) : talbi

Prénom : mohamed abdelghafar

Groupe : 2

Questions de cours (7pts) :

1-Quelles sont les différentes méthodes pour implémenter les threads en java ? Justifier quand est ce qu'on utilise chaque méthode **(2pts)**

Il y a deux méthodes pour implémenter les thread en java qu'il sont :
- Par extends Thread
- Par implémentation de l'interface Runnable

La différance entre les deux méthodes est dans utilisation de l'extends Thread on peux pas extends autre class, et chaque Thread a son propre donnes , mais dans l'utilisation de Runnable l'héritage est possible et les ressources sont partager pour tous les Thread .

2- Pour chacune de ces questions, cocher les affirmations qui sont vraies (0,5 point par bonne réponse, 0,5 point par mauvaise réponse) **(5 pts)**

Pour chaque question, cocher les affirmations qui sont vraies (0,5 point par bonne réponse, 0,5 point par mauvaise réponse)

1- Quelle est la méthode qui permet de démarrer un thread :

- ☐ **init() ;**
- ☐ **start() ;**
- ☐ **resume() ;**
- ☐ **run() ;**

2- Pour assurer la synchronisation du point de rendez-vous suivant, déterminer les solutions correctes Pour Pt de rdv (init(SA,0), init (SB,0)):

Processus A	Processus B
... I1; Pt de rdv I2; ...	... J1; Pt de rdv J2; ...

[]

Processus A	Processus B
... I1; I2; ...	... J1; J2; ...

[]

Processus A	Processus B
I1; V(SA); P(SB); I2;	J1; V(SB); P(SA); J2;

[]

Processus A	Processus B
I1; P(SB); V(SA); I2;	J1; V(SB); P(SA); J2;

Université Constantine 2-Abdelhamid Mehri
Faculté NTIC- Département TLSI
3éme année Licence Génie Logiciel

Contrôle du module DAC

Constantine, le 12/02/2020

Nom de l'étudiant(e) : talbi

Prénom : mohamed abdelghafar

Groupe : 2

[vrai]

Processus A	Processus B
I1; P(SB); V(SA); I2;	J1; P(SA); V(SB); J2;

3- Qu'implémente la classe Semaphore :

☐ **Object ; (Tous les class sont hériter du class object)**
☐ **Thread ;**
☐ **Runnable ;**
☐ **Serializable ;**

4- Parmi les propositions suivantes déterminer celles qui sont correctes :

☐ **Le problème d'accès concurrent aux variables peut être résolu en déclarant ces variables** `synchronized ;` **→ vrai**

☐ **Les méthodes synchronized sont remplacées par les méthodes lock(), unlock() etc. de l'interface Lock ; → vrai**

☐ **Dans une classe ayant une partie du code synchronisé, plusieurs threads peuvent accéder à l'autre partie du code non synchronisé ; → vrai**

☐ **Pour une barrière utilisée une seule fois, il vaut mieux utiliser CountDownLatch ; →vrai**
☐ `notify( )` **et** `notifyAll()` **peuvent être appelées dans un contexte non-synchronisé ; → vrai**
☐ **Les variables à conditions sont gérées par le paquetage java.util.concurrent.lock ; → vrai**

☐ **La barrière est cyclique mais pas réutilisable ; → faux**
☐ **Les sémaphores en java sont gérées par les méthodes acquire() et signal() de la classe Semaphore ; → faux**

☐ **Une méthode statique ne peut pas être synchronisée ; → faux**

☐ **Une barrière permet de bloquer plusieurs thread en un point jusqu'à ce qu'un nombre prédéfini de threads atteigne ce point ; → vrai**

☐ **Une barrière permet de bloquer un thread en un point jusqu'à ce qu'un nombre prédéfini de threads atteigne ce point ; → faux**
☐ **Quand le thread est endormi, il libère ses verrous ; → faux**

5- Que va afficher ce code ?

Université Constantine 2-Abdelhamid Mehri
Faculté NTIC- Département TLSI
3éme année Licence Génie Logiciel

Contrôle du module DAC

Constantine, le 12/02/2020

Nom de l'étudiant(e) : talbi

Prénom : mohamed abdelghafar

Groupe : 2

```
class Test extends Thread{
 public void run(){

       for(int i = 0; i < 30; i++)
         System.out.println("LMD GL");
   }
   public static void main(String[] args){
        Test t = new Test();
    }
 }
```

- ☐ **Rien du tout.**
- ☐ **30 fois "LMD GL".**
- ☐ **Ce code ne compilera pas !**
- ☐ **29 fois "LMD GL**

6- Quand un thread est-il considéré comme mort ?

a. Lorsqu'il est mis en attente.
b. Lorsqu'il est bloqué par un autre thread.
c. Lorsqu'il a dépilé la méthode run() de sa pile d'exécution.

Exercice 1 : Les Sémaphores (6pts)

Soient deux villes VILLE1 et VILLE2 sont reliées **par une seule voie** de chemin de fer. Les trains peuvent circuler dans **un seul sens à la fois** de la VILLE1 vers la VILLE2 ou de la VILLE2 vers la VILLE1. On considère deux classes de processus : les trains allant de la VILLE1 vers la VILLE2 (Train V1V2) et les trains allant de la VILLE2 vers la VILLE1 (Train V2V1). Ces processus se décrivent comme suit :

```
Train V1V2 {
Demande d'accès à la voie par VILLE1 () ;
Circulation sur la voie de la ville1 vers
la ville 2 () ;
Sortie de la voie par la VILLE2 ();
}
```

```
Train V2V1{
Demande d'accès à la voie par VILLE2 () ;
Circulation sur la voie de ville2 vers la
ville1 () ;
Sortie de la voie par VILLE1 ();
}
```

1- Proposer une solution permettant de respecter les règles de circulation sur **la voie unique** en utilisant les **sémaphores,** utilisez **deux compteurs** pour calculer **le nombre des trains** voulant aller de **V1V2** et de **V2V1.** (**6pts**)

Université Constantine 2-Abdelhamid Mehri
Faculté NTIC- Département TLSI
3éme année Licence Génie Logiciel
Contrôle du module DAC

Constantine, le 12/02/2020

Nom de l'étudiant(e) : talbi

Prénom : mohamed abdelghafar

Groupe : 2

```
/* Sémaphores et initialisation

Int Nbr_V1V2=0 ;Int Nbr_V2V1=0 ;

  Semaphore Voix =new Semaphore(1,true) ;
  Semaphore mutex =new Semaphore(1,true);

 Demande d'accès à la voie par la VILLE1 () {
  P(mutex)
  if(Nbr_V1V2>0 OR Nbr_V2V1>0 ){
     P(voix) ;
   }

    Nbr_V1V2++ ;
    CirculerV1V2 () ;

}

   Sortie de la voie par la VILLE 2 (){

Nbr_V2V1-- ;

If(Nbr_V2V1 !=0){

  V(mutex) ;

  V(voix) ;

}


3/

}
```

```
Demande d'accès à la voie par la VILLE2 () {


    P(mutex)
    if(Nbr_V2V1>0 OR Nbr_V1V2>0){
     P(voix) ;
     }
    Nbr_V2V1++ ;
    CirculerV2V1 () ;



}


Sortie de la voie par la VILLE1 (){
Nbr_V1V2-- ;

If(Nbr_V1V2 !=0){

   V(mutex) ;

   V(voix) ;

}




}


4
```

Exercice 2 : Les moniteurs en java (7pts)

Soit le problème du producteur/consommateur abordé au cours, avec un tampon de taille fixe N et circulaire, deux pointeurs : **`queue`** pour y produire et **`tete`** pour y consommer.

-Proposer le **code java** des méthodes : **`public void produire (int m)`** et **`public int consommer ()`** en utilisant les **Moniteurs** de **deux façons différentes.**

Université Constantine 2-Abdelhamid Mehri
Faculté NTIC- Département TLSI
3éme année Licence Génie Logiciel

Contrôle du module DAC

Constantine, le 12/02/2020

Nom de l'étudiant(e) : talbi

Prénom : mohamed abdelghafar

Groupe : 2

Moniteur java < 1.5

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Moniteur java > 1.5

```
 Conidtion full= lock.newCondition() ;

Condition empty = lock.newCondition();

public void produire (int m) {

lock.lock() ;

int i=n ;

queue=tampon[i] ;

While(queue != tete ){

    if(queue !=0){
        empty.await() ;

    }

i-- ; queue=t[i] ;

}

Queue=m ;

full.signal() ;

lock.unlock() ;

}
```

```
public int consommer (){
int consume ;
lock.lock;

    if(tete!=0){
     consume=tete ;
     tete=0;
    }else{
     full.await();
    }

   empty.signal() ;
lock.unlock();

return consume ;
}
```

Université Constantine 2-Abdelhamid Mehri
Faculté NTIC- Département TLSI
3éme année Licence Génie Logiciel

Contrôle du module DAC

Constantine, le 12/02/2020

Nom de l'étudiant(e) : talbi

Prénom : mohamed abdelghafar

Groupe : 2