Université Eshahid Hamma Lakhdar El-Oued
Faculté des sciences exactes
Département d'Informatique

Module : Système Exploitation II
Année : 3ème Année Inf. licence
Le 14/01/2025 – Durée : 1h30

Contrôle Semestriel

Exercice 1 : *Choisir* la ou les b*onnes réponses.* **(05.50 Pts)**

1) Un sémaphore peut être utilisé pour :
a) Contrôler l'accès à une ressource unique.
b) Synchroniser deux processus.
c) Les deux réponses précédentes.
d) Aucune des réponses précédentes

2) Une opération P() sur un sémaphore :
a) Augmente la valeur du sémaphore.
b) Diminue la valeur du sémaphore.
c) Libère une ressource.
d) Crée une variable conditionnelle.

3) Quelle est la principale limitation des sémaphores ?
a) Difficulté d'implémentation.
b) Risque de famine si mal utilisés.
c) Peuvent être utilisés uniquement pour des ressources simples.
d) Ne peuvent pas être utilisés dans les systèmes multi-utilisateurs.

4) Un moniteur est :
a) Une structure de données pour gérer les threads.
b) Un mécanisme de synchronisation de haut niveau.
c) Un algorithme pour éviter les interblocages.
d) Un processus spécifique au système d'exploitation.

5) L'algorithme du banquier peut :
a) Détecter un interblocage.
b) Prévenir un interblocage.
c) Éviter un interblocage.
d) Tous les choix ci-dessus.

6) Une méthode efficace pour éviter un interblocage est :
a) Allouer les ressources dans un ordre spécifique.
b) Utiliser uniquement des ressources partagées.
c) Ignorer les interblocages.
d) Augmenter le nombre de ressources disponibles.

Exercice 2 **(06.50 Pts)**

Une banque gère des ressources sous la forme de crédits alloués à des clients. Chaque client peut demander ou libérer des crédits. Pour éviter les interblocages, la banque utilise l'algorithme du banquier pour s'assurer que l'état du système reste sûr après chaque allocation.

La banque dispose Total = (10, 5, 7) unités disponibles pour trois types de ressources : R1, R2, et R3.
Quatre clients (C1, C2, C3, C4) participent au système.

$Allocation = \begin{bmatrix} 0 & 1 & 0 \\ 2 & 0 & 0 \\ 3 & 0 & 2 \\ 2 & 1 & 1 \end{bmatrix}$	$Max = \begin{bmatrix} 7 & 5 & 3 \\ 3 & 2 & 2 \\ 9 & 0 & 2 \\ 2 & 2 & 2 \end{bmatrix}$

1) Calculer la matrice Need (Need = Max – Allocation) et le vecteur de ressources restées (Disponibles).
2) Vérifiez si l'état initial du système est sûr en utilisant l'algorithme du banquier.
3) Le client C1 effectue la requête (1, 0,2). Peut-elle être satisfaite immédiatement ?

Exercice 3 **(08.00 pts)**

Un pont supporte une charge maximale limitée. Ce pont est traversé par des voitures et par des personnes. L'accès au pont doit être bien géré :

- Le nombre de personnes qui peuvent traverser est de 10, ou une seule voiture.

- La priorité doit être donnée aux personnes : lorsqu'une voiture et une personne demandent de traverser le pont, la personne doit être choisie en priorité, sous réserve, bien sûr, que la capacité maximale du pont soit respectée.

1) De quel type de problème vue au cours il s'agit ?

2) Proposez un schéma de synchronisation des processus voiture et personnes en utilisant les sémaphores (processus personne et processus voiture).

يحمل جســـر حمـــولة قصوى محدودة، و يستطيع أن يعبره أشخـــاص و ســـيارات.

نظام الوصـــول إلى الجســـر يتم حسب القواعد التالية :

-عدد الأشخـــاص الذين يمكـــنهم العـــبور هو 10 أو عبور ســـيارة واحـدة.

-يجب إعطاء الأولوية للأشخـــاص: عندما تتقدم سيارة وشخص لعـبور الجسر، يجب إعطاء الأولوية للشخص، شريطة بالطبع احترام السعة القصوى للجسر.

1) أي نوع من المشكلات شبيهة لها قد تم دراستها في الدروس النظرية؟

2)اقـــترح مخططا لمزامنة عمليتي السيارة والشخص باستخدام االسيمفور.

Bonne Chance

Corrigé-Type

Exercice 1 : *Choisir* La Ou Les *Bonnes Réponses.* **(05.50 Pts)**

1) Un sémaphore peut être utilisé pour :
a) Contrôler l'accès à une ressource unique.
b) Synchroniser deux processus.
c) Les deux réponses précédentes. ✓
d) Aucune des réponses précédentes

2) *Une opération P() sur un sémaphore :*
a) Augmente la valeur du sémaphore.
b) Diminue la valeur du sémaphore. ✓
c) Libère une ressource.
d) Crée une variable conditionnelle.

3) Quelle est la principale limitation des sémaphores ?
a) Difficulté d'implémentation.
b) Risque de famine si mal utilisés. ✓
c) Peuvent être utilisés uniquement pour des ressources simples.
d) Ne peuvent pas être utilisés dans les systèmes multi-utilisateurs.

4) Un moniteur est :
a) Une structure de données pour gérer les threads.
b) Un mécanisme de synchronisation de haut niveau. ✓
c) Un algorithme pour éviter les interblocages.
d) Un processus spécifique au système d'exploitation.

5) L'algorithme du banquier peut :
a. Détecter un interblocage.
b. Prévenir un interblocage.
c. Éviter un interblocage. ✓
d) Tous les choix ci-dessus.

6) Une méthode efficace pour éviter un interblocage est :
a) Allouer les ressources dans un ordre spécifique. ✓
b) Utiliser uniquement des ressources partagées.
c) Ignorer les interblocages.
d) Augmenter le nombre de ressources disponibles.

Exercice 2 **(06.50 Pts)**

1)

-Calcul la matrice Need = Max – Allocation

Need =

7	4	3
1	2	2
6	0	0
0	1	1

- Calcul des ressources disponibles (Available)

Le tableau Disponible est calculé comme suit :

Available=M−Somme des colonnes de Allocation

Total=(10,5,7) Somme des colonnes de Allocation :

Colonne R1 : 0+2+3+2=70+2+3+2=7

Colonne R2 : 1+0+0+1=21+0+0+1=2

Colonne R3 : 0+0+2+1=30+0+2+1=3

Disponible=(10−7,5−2,7−3)=(3,3,4)

2) L'état courant est sûr ?

On applique l'algorithme de banquier.

Étapes de l'algorithme du banquier :

Disponible= [3, 3, 4] et Reste = [C1, C2, C3, C4].

- Trouvez un client Ci tel que Need[i]≤Disponible. (i=1..4)

Si trouvé, allouez ses ressources et mettez à jour Disponible.

Client C2:

Need[2]=(1,2,2)

Need[2]≤Disponible Allouez ses ressources :

Exécuter C2 et récupérer ses ressources.

Disponible=Disponible+Allocation[2]=[3,3,4]+[2,0,0]=[5,3,4]

Reste=[C1,C3,C4]

- Trouvez un client Ci tel que Need[i]≤Disponible. (i=1,3,4)

Client C4 :

Need[4]=(0,1,1)

Need[4]≤Disponible

Exécuter C4 et récupérer ses ressources.

Disponible=Disponible+Allocation[4]=(5,3,4)+(2,1,1)=(7,4,5)

Reste=[C1,C3]

- Trouvez un client Ci tel que Need[i]≤Disponible. (i=1,3)

Client C1 :

Need[1]=(7,4,3)

Need[1]≤Disponible. Allouez ses ressources :

Exécuter C1 et récupérer ses ressources.

Disponible=Disponible+Allocation[1]=(7,4,5)+(0,1,0)=(7,5,5)

Reste=[C3]

- Trouvez un client Ci tel que Need[i]≤Disponible. (i=3)

Client C3 :

Need[3]=(6,0,0)

Need[3]≤Disponible. Allouez ses ressources :

Exécuter C3 et récupérer ses ressources.

Disponible=Disponible+Allocation[3]=(7,5,5)+(3,0,2)=(10,5,7)

État final : Reste=[]

L'état initial est sûr.

3) Demande de ressources (Requête=(1,0,2) par C1)

a) Vérification des conditions nécessaires :

Pour que la demande soit acceptée :

- Requête≤Disponible:

Disponible=(3,3,4), donc (1,0,2)≤(3,3,4) (Condition respectée).

►La demande est acceptable.

b) Allocation temporaire et vérification de l'état sûr :

1. Simulez l'allocation temporaire :

Allocation[1]=[0,1,0]+[1,0,2]=[1,1,2]

Allocation =

1	1	2
2	0	0
3	0	2
2	1	1

Need[1]=[7,5,3]−[1,1,2]=[6,2,1]

Need =

6	2	1
1	2	2
6	0	0
0	1	1

Disponible=[10,5,7]−[8,2,5]=(2,3,2)

2. Vérifiez l'état sûr avec l'algorithme du banquier (comme en 1).

On applique l'algorithme de banquier.

Étapes de l'algorithme du banquier :

Disponible= [2, 3, 2] et Reste = [C1, C2, C3, C4].

- Trouvez un client Ci tel que Need[i]≤Disponible. (i=1..4)

Si trouvé, allouez ses ressources et mettez à jour Disponible.

Client C2:

Need[2]=(1,2,2)

Need[2]≤Disponible Allouez ses ressources :

Exécuter C2 et récupérer ses ressources.

Disponible=Disponible+Allocation[2]=[2,3,2]+[2,0,0]=[4,3,2]

Reste=[C1,C3,C4]

- Trouvez un client Ci tel que Need[i]≤Disponible. (i=1,3,4)

Client C4 :

Need[4]=(0,1,1)

Need[4]≤Disponible

Exécuter C4 et récupérer ses ressources.

Disponible=Disponible+Allocation[4]=(4,3,2)+(2,1,1)=(6,4,3)

Reste=[C1,C3]

- Trouvez un client Ci tel que Need[i]≤Disponible. (i=1,3)

Client C1 :

Need[1]=(6,2,1)

Need[1]≤Disponible. Allouez ses ressources :

Exécuter C1 et récupérer ses ressources.

Disponible=Disponible+Allocation[1]=(6,4,3)+(1,1,2)=(7,5,5)

Reste=[C3]

- Trouvez un client Ci tel que Need[i]≤Disponible. (i=3)

Client C3 :

Need[3]=(6,0,0)

Need[3]≤Disponible. Allouez ses ressources :

Exécuter C3 et récupérer ses ressources.

Disponible=Disponible+Allocation[3]=(7,5,5)+(3,0,2)=(10,5,7)

État final : Reste=[]

L'état initial est sûr.

Conclusion : Requête=[1,0,2] peut être acceptée.

Exercice 3 **(08.00 Pts)**

1)

Problème simulaire : problème des lecteurs/rédacteurs avec priorité des lecteurs.

Rédacteur= 1 une seule voiture; lecteurs : personnes: 10 Personnes

2) Schéma de synchronisation

```
int nbp=0 ;            //Compteur de personnes
semaphore pers=10 ;  //Contrôle les personnes traversés
semaphore voie=1;     // Contrôle la voie
semaphore mutex=1 ;   // Contrôle la mise à jour de compteur nbp par les personnes
semaphore mutex_prio=1 ;// pour la priorité des personnes par rapport aux voitures
```

processus personnes()

```
{
P(pers) ;              // s'il y a déjà 10 personnes,
                       // se bloquer
P(mutex) ;
nbp++ ;
if (nbp ==1 )

  {

   V(mutex) ;
   P(voie) ;
 }
else  { V(mutex) ; }

<Traverser_pont() ;>

p(mutex) ;
nbpers-- ;
if (nbp==0) {V(voie) ; }
V(mutex) ;
V(pers) ;
}
```

processus voitures()

```
{
P(mutex_prio) ;
        // priorité des personnes : les voitures
          traversent // deux barrières (2
          semaphores)

        // de telle façon à avoir une seule voiture
          bloquée // au niveau de p(voit).
P(voie) ;

<Traverser_pont() ;>

V(voie) ;
V(mutex_prio) ;

}
```