Université Djillali Liabès
Faculté des Sciences Exactes
Département d'Informatique

16 Janvier 2019

Epreuve de la matière
Système d'Exploitation 2
Licence SI (S5)

Nom :	Prénom :

Questions de cours (10 pts) : Soulignez la réponse correcte :

Q_1 : La relation T -> T' signifie : 1- T<T' et pour tout k, on a T<T_k<T'. 2- T<T' et il existe un k, tel que T<T_k<T'. 3- T<T' et pour tout k, on n'a pas T<T_k<T'. (0.5pt)	**Q_7 : La structure moniteur utilise une file d'attente supplémentaire si :** 1- 'signal and continue' est utilisé. 2- la version de Brinch-Hansen est utilisée. 3- 'signal and wait' est utilisé. (0.5pt)
Q_2 : Un système est de parallélisme maximal ssi la suppression de : 1- tous ses arcs induit l'interférence des tâches concernées. (0.5pt) 2- tous ses arcs induit la non-interférence des tâches concernées. 3- certains de ses arcs induit l'interférence des tâches concernées.	**Q_8 : la notion de moniteur peut être intégrée dans un langage évolué sous forme de :** 1- de bibliothèques. 2- structure de données. (0.5pt) 3- d'appels système.
Q_3 : Soit S un système de tâches, S' est le système de parallélisme maximal, < et <' sont les relations de précédence respectives : 1- <' a autant de contraintes de précédence que <. 2- <' a moins de contraintes de précédence que <. (0.5pt) 3- <' a plus de contraintes de précédence que <.	**Q_9 : L'appel 'receive' de la technique des boites aux lettres bloque :** 1- l'émetteur jusqu'à ce qu'un message soit retiré de la boite en question. 2- le récepteur jusqu'à ce qu'un message soit envoyé. 3- le récepteur jusqu'à ce qu'un message soit délivré à la boite en question. (0.5pt)
Q_4 : Parmi les informations suivantes, laquelle n'est pas contenue dans le contexte individuel d'un thread ? 1- un identificateur unique. 2- une pile d'exécution. 3- un espace d'adressage. (0.5pt)	**Q_{10} : La capture d'un signal permet :** 1- d'exécuter le traitement par défaut de tous les signaux. 2- de redéfinir le traitement de tous les signaux. 3- de redéfinir le traitement de certains signaux. (0.5pt)
Q_5 : la solution du masquage des interruptions est généralement utilisée par : 1- le SE pour de très courtes sections critiques. (0.5pt) 2- les processus utilisateurs pour de très courtes sections critiques. 3- le SE pour de longues sections critiques.	**Q_{11} : Les tubes anonymes peuvent être utilisés pour des communications entre :** 1- des processus indépendants. 2- des processus dépendants. (0.5pt) 3- des threads indépendants.
Q_6 : L'algorithme de Peterson a comme inconvénient : 1- l'attente active infinie. 2- la consommation du temps processeur. (0.5pt) 3- la violation de l'E.M.	**Q_{12} : Les tubes nommés représentent une version avancée des tubes anonymes assurant des communications entre des processus :** 1- dépendants s'exécutant sur la même machine. 2- indépendants s'exécutant sur des machines différentes. 3- indépendants s'exécutant sur la même machine.

	(0.5pt)
Q_{13} : L'instruction TSL(bool b) permet : 1- d'affecter la valeur FALSE au paramètre b. 2- de retourner la nouvelle valeur du paramètre b. 3- de récupérer l'ancienne valeur du paramètre b. (0.5pt)	**Q_{17} : Un interblocage se produit éventuellement si en utilisant l'algorithme du banquier on constate que :** 1- tous les processus ont été marqués. 2- certains processus n'ont pas été marqués. (0.5pt) 3- tous les processus P_i vérifient la condition $Req_{Pi}<=A$.
Q_{14} : Les primitives SLEEP et WAKEUP n'assurent pas l'E.M si : 1- les tests et les mises à jour des conditions d'entrée en <sc> ne sont pas exécutés en E.M. (0.5pt) 2- la primitive WAKEUP est exécutée avant la primitive SLEEP. 3- les tests et les mises à jour des conditions d'entrée en <sc> sont exécutés en E.M .	**Q_{18} : Dans un système d'exploitation :** 1- toutes les ressources peuvent être réquisitionnées. 2- certaines ressources peuvent être réquisitionnées. (0.5pt) 3- aucune ressource ne peut être réquisitionnée.
Q_{15} : L'exécution indivisible ou atomique ? 1- garantit qu'un processus ne se bloque pas durant l'exécution. 2- garantit qu'un processus ne soit pas suspendu durant l'exécution. (0.5pt) 3- garantit qu'un processus soit suspendu durant l'exécution.	**Q_{19} : L'algorithme du Banquier a pour but :** 1- de prévenir les interblocages. 2- d'éviter les interblocages. (0.5pt) 3- de détecter les interblocages.
Q_{16} : Les compteurs d'événements ont été conçus pour assurer : 1- la synchronisation entre processus. (0.5pt) 2- l'E.M et la synchronisation entre processus. 3- l'E.M entre processus.	**Q_{20} : Le test ($Req_{Pi}>A$) de l'algorithme du banquier signifie que :** 1- les ressources demandées par le processus P_i peuvent être allouées. 2- les ressources demandées par le processus P_i ne peuvent pas être allouées. (0.5pt) 3- les ressources allouées au processus P_i peuvent être réquisitionnées.

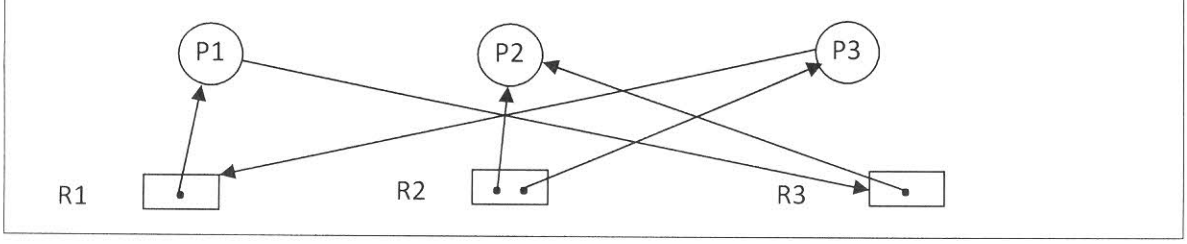
Exercice 1 Interblocage (5 points) :

On dispose de trois ressources R_1=segment mémoire (un seul exemplaire), R_2=imprimante (deux exemplaires), R_3=modem (un seul exemplaire)) nécessaires à l'exécution de trois processus P_1, P_2 et P_3. Ces ressources sont libérées au bout d'un temps fini et au pire en fin d'exécution.

Processus P_1	*Processus P_2*	*Processus P_3*
demander(segment mémoire) demander(modem)	demander(modem) demander(imprimante)	demander(imprimante) demander(segment mémoire)
utilisation_ressources() libérer(modem) libérer(segment mémoire)	utilisation_ressources() libérer(imprimante) libérer(modem)	utilisation_ressources() libérer(segment mémoire) libérer(imprimante)

1. A l'aide de la politique du Tourniquet (le quantum est prévu pour exécuter une seule instruction), exécutez la partie encadrée de chaque processus, en construisant au fur et à mesure un graphe d'allocation de ressources.

Graphe d'allocation de ressources (1.5 pt) (0.25pt par arc)



2. Le système se trouve-t-il dans une situation d'interblocage ? Justifiez votre réponse. (0.5pt)

Le graphe ne contient pas de cycle (0.25pt). Par conséquent, le système ne se trouve pas dans une situation d'interblocage. (0.25pt)

3. Déterminez les vecteurs E et A ainsi que les matrices Alloc et Req **après** la même exécution. (1pt)

E=(1 2 1) (0.25pt) , A=(0 0 0) (0.25pt)

$$Alloc = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad (0.25pt), \qquad Req = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad (0.25pt)$$

4. En utilisant l'algorithme du banquier, déterminez l'état du système. Justifiez. (2 pts)

Algorithme du Banquier :

1- Marquer le processus P2 : ReqP2<=A (0.25pt), A=A+AllocP2=(0 1 1) (0.25pt)
2- Marquer le processus P1 : ReqP1<=A (0.25pt), A=A+AllocP1=(1 1 1) (0.25pt)
3- Marquer le processus P3 : ReqP3<=A (0.25pt), A=A+AllocP3=(1 2 1)=E. (0.25pt)

Etat du système (avec justification) :
Tous les processus ont été marqués (0.25pt). Par conséquent, le système est dans un état sûre. (0.25pt)

Exercice 2 : Synchronisation – Problème de passage (5points)

Dans un hôtel, deux grandes salles, A et B, sont séparées par une porte étroite qui ne peut être franchie que par une seule personne à la fois. Une personne de la salle A qui veut franchir la porte est représentée par un processus Personne A. De même, une personne de la salle B qui veut passer dans la salle A est représentée par un processus Personne B. Proposez un schéma de synchronisation des processus Personne A et Personne B en utilisant des sémaphores sachant que s'il y a des personnes dans les deux salles en attente, il faut assurer une alternance stricte entre le passage des personnes de A et B. C'est à dire, qu'après le passage d'une personne de A, il faut faire passer une personne de B s'il y en a en attente, sinon on continue avec les personnes de A. Même raisonnement pour les personnes B.

1- **Variables globales (rôle et initialisation)** (2pts)

nbr_A, nbr_B : le nombre de personnes A et B respectivement (initialisé à 0). (0.5pt+0.5pt)
mutex_A, mutex_B : sémaphores d'E.M pour les variables nbr_A et nbr_B (initialisés à 1). (0.25pt+0.25pt)
Sens_AB, Sens_BA : sémaphores pour bloquer les personnes A et B respectivement voulant franchir la porte (initialisés à 0). (0.25pt+0.25pt)

2- **Codes des processus Personne A et Personne B :** (3pts)

Personne A (1.5pt)	**Personne B** (1.5pt)
Début	**Début**
Tant que (vrai) faire	**Tant que (vrai) faire**
P(mutex_A) ;	**P(mutex_B) ;**
nbr_A=nbr_A+1 ;	**nbr_B=nbr_B+1 ;**
V(mutex_A) ; 0.25pt	**V(mutex_B) ;** 0.25pt
P(mutex_B) ;	**P(mutex_A) ;**
Si(nbr_B<>0) alors P(Sens_AB) ;	**Si(nbr_A<>0) alors P(Sens_BA) ;**
V(mutex_B) ; 0.5pt	**V(mutex_A) ;** 0.5pt
Franchir_porte() ;	**Franchir_porte() ;**
P(mutex_A)	**P(mutex_B)**
nbr_A=nbr_A-1 ;	**nbr_B=nbr_B-1 ;**
V(mutex_A) ; 0.25pt	**V(mutex_B) ;** 0.25pt
P(mutex_B)	**P(mutex_A)**
Si(nbr_B<>0) alors V(Sens_BA) 0.5pt	**Si(nbr_A<>0) alors V(Sens_AB)** 0.5pt
V(mutex_B)	**V(mutex_A)**
Fin ;	**Fin ;**
Fin.	**Fin.**