Contrôle terminal Session 2 - 9/6/2020

Durée : 2h

Barème donné à titre indicatif : Ex.1 : 4 - Ex.2 : 4 - Ex.3 : 4 - Ex.4 : 2 - Ex.5 : 6

Exercice 1. Considérons le programme suivant, qui se termine parfois par un interblocage :

Conditions initiales
a=1
b=1
c=1

Processus 1
P(a);
P(b);
V(b);
P(c);
V(c);
V(a);

Processus 2
P(c);
P(b);
V(b);
V(c);
P(a);
V(a);

Processus 3
P(c);
V(c);
P(b);
P(a);
V(a);
V(b);

1. Lister toutes les paires de sémaphores que chaque processus cherche à obtenir et donner l'ordre dans lequel le processus les demande.
2. On cherche à éviter les interblocages en ordonnant les réservations selon l'ordre `a` $<$ `b` $<$ `c`, ce qui signifie que `a` est demandé avant `b`, lui-même demandé avant `c`. Discuter les réservations de chaque processus selon ce critère.
3. On sait que lorsque les requêtes sont ordonnées, il n'y a pas d'interblocage. En modifiant seulement le code de P3, proposer un ordre de réservation des requêtes `a, b` et `c` pour tous les processus qui garantit alors l'absence d'interblocage.

Exercice 2. Considérons le problème du barbier qui s'énonce comme suit : *La boutique du barbier est composée d'une salle d'attente contenant* `n` *chaises et du salon où se trouve la chaise du barbier. Lorsque le barbier a fini de raser un client, il fait entrer le client suivant dans le salon. Si la salle d'attente est vide, le barbier s'y installe pour dormir. Si un client trouve le barbier endormi, il le réveille. Si non, il s'installe dans la salle d'attente s'il reste de la place. S'il n'y a pas de place, il rentre chez lui.*

Une solution correcte pour ce problème est la suivante :

```
INITIALISATION
Sémaphores clients, barbier, exclusion
clients = 0  // clients dans la salle d'attente
barbier = 0  // barbier prêt à couper
exclusion = 1 // pour l'exclusion mutuelle
int places = nb_chaises
```

```
     BARBIER
     while (true) {
(1)     P(clients)
(2)     P(exclusion)
(3)     places = places + 1
(4)     V(barbier)
(5)     V(exclusion)
(6)     coupe_cheveux
}
```

```
     CLIENT
(1)  P(exclusion)
(2)  si (place > 0) alors
(3)     places = places - 1
(4)     V(clients)
(5)     V(exclusion)
(6)     P(barbier)
(7)     se_faire_couper_les_cheveux
(8)  sinon
(9)     V(exclusion)
(10)    rentrer_chez_soi_les_cheveux_longs
(11) fsi
```

1. Quel schéma de synchronisation vu en cours est étendu ici ?
2. Dans quelle mesure peut-on permuter les instructions (4) et (5) dans le code de `BARBIER` ?
3. Même question si on permute les instructions (5) et (6) dans le code de `CLIENT` ?

Exercice 3. On se place dans un système de mémoire de 1700 Ko de mémoire haute (i.e. au-delà de la partie utilisée par l'OS) répartie en 5 partitions de 100 Ko, 500 Ko, 200 Ko, 300 Ko et 600 Ko.

On suppose que le système d'exploitation doit allouer des processus de taille 212 Ko, 417 Ko, 112 Ko et 426 Ko dans cet ordre. Une partition ne doit être occupée que par un seul processus. Pour chacun des algorithmes suivants, donner l'allocation obtenue et le taux de fragmentation : (1) prochain bloc libre, (2) meilleur ajustement, (3) plus grand bloc libre.

Quel algorithme utilise le meilleur ajustement sur cet exemple ?

Exercice 4. Combien de défauts de page sont produits avec l'algorithme de remplacement FIFO pour la suite de références suivantes, en supposant qu'on dispose de 4 blocs en mémoire centrale ?

1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6

Exercice 5. On considère 4 processus A, B, C et D et on suppose que leur exécution est comme suit :

pour A : 7 unités de temps CPU, 3 unités de temps d'E/S et 5 unités de temps CPU.
pour B : 6 unités de temps CPU, 4 unités de temps d'E/S, 4 unités de temps CPU.
pour C : 5 unités de temps CPU.
pour D : 1 unité de temps CPU, 4 unités de temps d'E/S, 2 unités de temps CPU.

On suppose que A se présente à l'instant 0, B se présente à l'instant 1, C se présente à l'instant 9 et D se présente à l'instant 12.

Montrer comment les 4 processus utilisent le processeur dans chacun des cas suivants :

1. chaque processus a son propre périphérique d'E/S et l'ordonnanceur fonctionne selon l'ordre FIFO (sans préemption).
2. chaque processus a son propre périphérique d'E/S et l'ordonnanceur utilise l'algorithme du tourniquet, avec un quantum de temps égal à 5. Le temps de commutation est nul.
3. Les trois processus A, B et D utilisent le même périphérique d'E/S et la file d'attente est gérée par le principe du premier arrivé, premier servi. L'ordonnanceur utilise l'algorithme du tourniquet, avec un quantum de temps égal à 5. Le temps de commutation est nul.