Commencé le	samedi 27 avril 2024, 15:21
État	Terminé
Terminé le	samedi 27 avril 2024, 15:22
Temps mis	56 s
Note	**10,00** sur 10,00 (**100**%)

Question 1

Correct

Note de 1,00 sur 1,00

Un moniteur est une structure de données composée d'attributs et de méthodes qui accèdent en exclusion en mutuelle aux attributs du moniteur.

Veuillez choisir une réponse.

- Vrai ✓
- Faux

La réponse correcte est « Vrai ».

Question 2

Correct

Note de 1,00 sur 1,00

Les moniteurs permettent à eux seuls (sans les variables de condition) d'implémenter des solutions aux problèmes de synchronisation comme les producteurs-consommateurs, les philosophes et les lecteurs-rédacteurs.

Veuillez choisir une réponse.

- Vrai
- Faux ✓

Par exemple, considérez le problème d'un producteur et d'un consommateur. Si le thread consommateur rentre dans le moniteur pour consommer alors que le tampon est vide, il devrait se bloquer (attente passive) dans le moniteur et laisser place au thread producteur.

L'attente active dans le moniteur n'est pas une solution car elle serait infinie.

La réponse correcte est « Faux ».

Question 3

Correct

Note de 1,00 sur 1,00

Une variable de condition d'un moniteur est une variable ...

Veuillez choisir au moins une réponse.

- [x] a. qui n'a pas de valeur mais a une file d'attente gérée FIFO. ✓
- [ ] b. locale déclarée dans une méthode du moniteur.
- [ ] c. booléenne dont la valeur est true ou false.
- [x] d. qui représente un événement ou un état d'un objet. ✓

Votre réponse est correcte.

Les réponses correctes sont :
qui n'a pas de valeur mais a une file d'attente gérée FIFO. ,

qui représente un événement ou un état d'un objet.

Question 4

Correct

Note de 1,00 sur 1,00

Les variables de condition d'un moniteur (c-à-d déclarées dans le moniteur) sont un mécanisme qui permet de bloquer temporairement le thread actif dans le moniteur pour libérer l'accès au moniteur à un autre thread.

Veuillez choisir une réponse.

- (•) a. vrai ✓
- ( ) b. faux

Votre réponse est correcte.

La réponse correcte est :
vrai

Question 5

Correct

Note de 1,00 sur 1,00

La fonction signal d'une variable de condition permet de débloquer un thread en attente de cette variable de condition. En quoi la sémantique Signal-and-Continue diffère-t-elle de la sémantique Signal-and-Wait ?

Veuillez choisir une réponse.

a. Signal-and-Wait place le thread débloqué à la fin de la file d'attente du moniteur alors qu' avec Signal-and-Continue, le thread débloqué devient actif dans le moniteur.

b. Signal-and-Continue place le thread débloqué dans la file d'attente des threads suspendus alors qu'avec Signal-and-Wait, le thread débloqué est mis en attente dans la file d'attente du moniteur.

c. Signal-and-Continue met le thread débloqué dans la file d'attente du moniteur alors qu'avec Signal-and-Wait, le thread débloqué devient actif dans le moniteur. ✓

d. Signal-and-Wait met le thread débloqué dans la file d'attente du moniteur alors qu'avec Signal-and-Continue, le thread débloqué devient actif dans le moniteur.

Votre réponse est correcte.

La réponse correcte est : Signal-and-Continue met le thread débloqué dans la file d'attente du moniteur alors qu'avec Signal-and-Wait, le thread débloqué devient actif dans le moniteur.

Question 6

Correct

Note de 2,00 sur 2,00

Considérez le pseudo-code suivant qui propose une implémentation des sémaphores en utilisant un moniteur et une variable de condition (toutes les files d'attente sont supposées de discipline FIFO (First In First Out) et la sémantique de la fonction signal est Signal-and-Continue) :

```
Moniteur  Sem( int v0) {
        int cp = v0; // valeur du sémaphore
        boolc wq;  // variable de condition pour simuler la file d'attente du sémaphore
        void  P()  {  cp=cp-1;
                      if (cp<0)  wait(wq);
                       printf("cp=%d\n",cp);
         }
        void  V()  {    if (cp<0)  signal(wq);
                          cp = cp+1;
         }
}
```

Deux threads th1 et th2 d'un même processus partagent un objet S de type Sem initialisé à 1. Supposez que les threads réalisent dans l'ordre ce qui suit :

1) th1: S.P();
2) th2: S.P() -> wait(wq);
3) th1: S.V()-> signal(wq)-> débloque th2, met th2 en attente du moniteur puis poursuit l'exécution de V;
4) th2: printf(...).

Quelle est la valeur de cp affichée par le thread th2 ?

Veuillez choisir une réponse.

- a. 0 ✓
- b. -1

Votre réponse est correcte.

La réponse correcte est :
0

Question 7

Correct

Note de 2,00 sur 2,00

Considérez le pseudo-code suivant. La file d'attente de la variable de condition est de discipline FIFO (First In First Out). La sémantique de la fonction signal est Signal-and-Wait :

```
Moniteur  Sem( int v0) {
        int cp = v0; // valeur du sémaphore
        boolc wq;  // variable de condition
        void  P()  {  cp = cp-1;
                        if (cp<0)  wait(wq);
                        printf("cp=%d\n",cp);
        }
        void  V()  {    if (cp<0)  signal(wq);
                          cp = cp+1;
        }
}
```

Deux threads th1 et th2 d'un même processus partagent un objet S de type Sem initialisé à 1. Supposez que les threads réalisent dans l'ordre les appels suivants :

1) th1: S.P();
2) th2: S.P() -> wait(wq);
3) th1: S.V()-> signal(wq)-> débloque th2 puis met en attente th1;
4) th2: printf(...).

Quelle est la valeur de cp affichée par th2?

- a. -1 ✓
- b. 0

Votre réponse est correcte.

La réponse correcte est :
-1

Question 8

Correct

Note de 1,00 sur 1,00

L'implémentation suivante d'un verrou actif au moyen du moniteur spinlock_t suivant est correcte :

```
Moniteur spinlock_t
{   int lock =0;
     void spinlock_lock()
    {  while (lock!=0);
       lock=1;
    }
    void spinlock_unlock()
    {    lock=0;
    }
}
```

Veuillez choisir une réponse.

- Vrai
- Faux ✓

Si un processus/threads demande le verrou alors qu'il est déjà pris, il va renter dans une active infinie dans le moniteur. Aucun autre processus/thread ne pourra accéder au moniteur (y compris celui qui détient le verrou).

La réponse correcte est « Faux ».