Utilisation de la bibliothèque sur une section critique

Question

Créer un programme "excl-mutu-none.c" qui réalise les fonctions suivantes :

1. Création d'un segment de mémoire partagée pouvant contenir un entier E.
2. Initialisation à 0 de l'entier E en mémoire partagée.
3. Création d'un process "fils" partageant le segment précédemment créé avec son "père".

Chaque process (père et fils) exécute ensuite 100 fois le code suivant, dont les quatre premières lignes constituent une section critique :

1. Affecter E à une variable A, entière, locale au process.
2. Attendre entre 20 et 100 ms (Utilisez les fonctions usleep(3) et rand(3V)).
3. Incrémenter A.
4. Affecter la variable locale A à la variable "partagée" E.
5. Attendre entre 20 et 100 ms (Utilisez les fonctions usleep(3) et rand(3V)).
6. Affichage dans le process père de la valeur de E.

Question

La valeur de la variable E est-elle égale à 200 ? Pourquoi ?

Question

Modifiez le programme précédent (nouvelle version nommée "excl-mutu.c") qui utilise la bibliothèque précédemment créée et qui permet de "synchroniser" les modifications de E par les deux process (Utilisation des primitives (P) et (V) pour réaliser une exclusion mutuelle).

Question

Qu'est-ce qui s'exclut mutuellement ? Expliquez le fonctionnement.

Utilisation de la bibliothèque pour un producteur-consommateur

Question

Créer un programme "prod-conso.c" qui utilise la bibliothèque précédemment créée et dont le but est le suivant :

- Un process fils va produire une suite d'entiers qu'il va stocker dans un buffer circulaire de taille 5 entiers.
- Le process père consomme les données de ce buffer circulaire.

Bien entendu, le process fils est bloqué lorsque le buffer est plein (il attend que le père consomme les données et libère de la place), et le père est bloqué losque le buffer est vide (il attend que le fils produise des données). Cette synchronisation entre le père et le fils sera réalisée uniquement à l'aide de deux sémaphores dont on aura soigneusement étudié les valeurs initiales.

Indice

Votre programme réalisera donc les fonctions suivantes :

- Création d'un segment de mémoire partagée pouvant contenir 5 entiers. Ce segment de mémoire partagée sera vu comme un buffer circulaire géré par deux "index" : un index d'écriture et un index de lecture qui évolueront "circulairement" (0 1 2 3 4 0 1 2 ...).
- Création d'un process "fils" partageant le segment précédemment créé avec son "père".
- Le process fils va jouer le rôle du producteur. Il génére les entiers de 1 à 50 qu'il stocke dans le buffer circulaire au fur et à mesure qu'il y a de la place libre. Il gère donc l'index d'écriture du buffer circulaire. Le process père, consommateur, va lire et afficher les données présentes dans le buffer circulaire. Il gère l'index de lecture.