FACULTE DES SCIENCES DE TUNIS

TD 2

Module : Programmation système – Section ICE3

Enseignants : Chiheb Ameur ABID
Islem DENDEN

Année universitaire : 2023-2024

Primitives de gestion des threads

```
int pthread_create(pthread_t * thread,const pthread_attrt * attr, void *(*start routine)(void*), void * arg);
void pthread_exit(void *value ptr);
int pthread_join(pthread_t thread, void **value ptr);
int pthread_detach(pthread_t thread);
```

Primitives de gestion des mutex

```
int pthread_mutex_init(pthread_mutex_t *, pthread_mutexattr_t *);
int pthread_mutex_destroy(pthread_mutex_t *mutex);
int pthread_mutex_lock(pthread_mutex_t *mutex);
int pthread_mutex_trylock(pthread_mutex_t *mutex);
int pthread_mutex_unlock(pthread_mutex_t *mutex);
```

Exercice 1.

Ecrire un programme qui créent trois threads. Chacun de deux threads incrémente son compteur à chaque 100ms et ceci durant 10 secondes.

Le troisième thread est un thread détaché qui affichera à chaque seconde le caractère * une fois chacun de deux threads a effectué 10 opérations d'incrémentation.

Exercice 2.

1) On aimerait implémenter un programme créant un thread pour calculer la somme d'un tableau d'entiers. Ce thread doit calculer la somme des entiers du tableau et le programme principal (main) s'occupera d'afficher le résultat.

2) Nous aimerions implémenter une autre variante du programme. Celui-ci sera fonctionnellement identique, mais ne devra utiliser aucune variable globale. Vous pouvez par contre passer un argument à votre thread et celui peut retourner une valeur.

Exercice 3.

1°) Considérons le programme suivant :

```
int main() {
   printf("A\n");
   fork();
   printf("B\n");
   fork();
   printf("C\n");
   exit(EXIT_SUCESS);
}
```

Combien de processus sont créés pae ce programme ?

Combien de fois sera affiché le message "A", le message "B" et le message "C" ?

2°) a) On veut écrire un programme qui recherche un élément dans un tableau d'entiers de grande taille (1 million de valeurs). L'idée retenue est d'écrire un programme qui crée un processus fils par fork puis :
- Le processus père explore la 1ere moitié du tableau
- Le processus fils explore la 2éme moitié du tableau

Celui qui trouve affiche "Le père (resp fils) a trouvé en position : x"
On ne se préoccupera pas de l'initialisation du tableau qui aura pu être faite par une lecture dans un fichier par exemple. On prévoira de saisir au clavier la valeur à rechercher avant de créer le processus fils.
Pour éviter les processus zombies on prévoira que le père attende la terminaison de son fils.
b) Modifier ce programme pour qu'il affiche un message quand la valeur n'a pas été trouvée.

3°) On va maintenant reprendre le même problème en utilisant des threads.
Le programme (processus) saisit au clavier la valeur à rechercher comme précédemment puis crée deux threads.
- Le 1er thread explore la 1ere moitié du tableau
- Le 2éme thread explore la 2éme moitié du tableau

Celui qui trouve place la position où il a trouvé dans une variable globale du programme.
Le processus attend la fin des 2 threads puis affiche en quelle position la valeur a été trouvée ou un message indiquant qu'elle n'a pas été trouvée.
On écrira une fonction différente pour chacun des 2 threads.