Université de Limoges | FACULTÉ DES SCIENCES ET TECHNIQUES

Licence 3ème année

Parallélisme I

janvier 2014

Durée : 2h — Documents autorisés

Threads & Sémaphores – (10 points)

1 – 6pts Le responsable d'un parking disposant de plusieurs entrées équipées de portail d'accès informatisé et géré par un même système central vous contacte.

Il voudrait disposer d'une solution logicielle pour assurer **indépendemment chacun** des fonctionnements suivants :

a. La solution doit garantir qu'il ne laisse pas entrer dans son parking, plus de n voitures pour les n places disponibles.

b. Chaque voiture dispose d'un emplacement de parking réservé. La solution doit garantir qu'il n'y ait pas plus d'une voiture sur l'emplacement qui lui est réservé.

c. Une offre famille a été définie : un emplacement peut être associé à deux voitures de la même famille et seule une de ces deux voitures peut accèder à la place de parking.

d. Une offre moto a été définie : un emplacement peut être utilisé par deux motos simultanément ou par une seule voiture.

e. Une offre « famille & moto » : un emplacement est associé à une seule famille qui peut y ranger soit deux motos soit une voiture.

Vous respecterez les consignes suivantes :

- ⋆ Chaque solution aux questions a) à e) est indépendante l'une de l'autre.
- ⋆ Vous donnerez le code correspondant :
  - ◇ à l'occupation de la place
  - ◇ à sa libération.
- ⋆ Si un véhicule, moto ou voiture, n'a pu obtenir une place, il est « mis en attente » pour être prioritaire lorsque la place se libérera.

2 – 4pts Questions sur :

a. l'utilisation des threads : Quelles sont les **risques de corruption** de données ?

b. l'utilisation des threads : Comment est gérée la « **pile ou stack** » et à quoi sert-elle ?

c. l'utilisation des sémaphores : Comment sont gérés les problèmes de **famine** et d'**équité** ?

d. l'utilisation des threads : Comment se passe le « **changement de contexte** » et comment évolue « **l'ordonnancement** » ?

Création de processus avec fork – (4 points)

3 – Soit le programme C suivant :

4pts

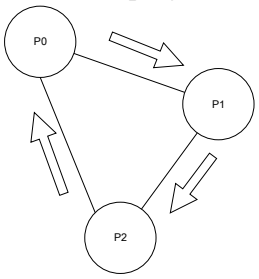
```
#include <stdio.h>

void traitement(int a, int b)
{
  printf("-->%d\n", a*b);
  if (a)
  {
    if (fork())
    {
      traitement(a-1, 2);
    }
    else
    {
      traitement(a-1, 3);
    }
  }
}
int main()
{
  traitement(3,1);
}
```

Que va-t-il afficher lors de son exécution ?

Signaux & tubes & fork – (6 points)

4 – On veut écrire un programme pour créer une structure arborescente de processus communicant.

6pts



1. Un processus P_0 va :
   - ◇ créer deux processus enfants P_1 et P_2 ;
2. on va créer 3 tubes entre les 3 processus :
   - ◇ un tube de $P_0 \longrightarrow P_1$;
   - ◇ un tube de $P_1 \longrightarrow P_2$;
   - ◇ un tube de $P_2 \longrightarrow P_0$;
3. P_0 va écrire dans le tube $P_0 \longrightarrow P_1$ les chiffres de 0 à 5 ;
4. P_1 lit un entier depuis le tube $P_0 \longrightarrow P_1$ et écrit cet entier dans le tube $P_1 \longrightarrow P_2$;
5. P_2 lit un entier depuis le tube $P_1 \longrightarrow P_2$ et écrit cet entier dans le tube $P_2 \longrightarrow P_0$;
6. P_0 lit dans le tube $P_2 \longrightarrow P_0$ et affiche la valeur lue.

a. Quel processus doit créer chacun des « tubes » et pourquoi ?

b. Écrire un « squelette » de programme C permettant de créer la structure circulaire décrite précédemment (3 processus et 3 tubes) ;

c. Si on veut que les opérations d'écriture/lecture soient synchronisées comment faut-il procéder ? A-t-on besoin d'utiliser des signaux pour le faire et comment ?

d. Si on voulait réécrire le programme avec des *threads*, avec une thread au lieu d'un processus :
   - ◇ A-t-on toujours besoin des tubes ?
   - ◇ Comment synchroniser les échanges entre threads ?
   - ◇ Est-ce que les signaux sont utiles ?