Systèmes Temps-Réel
Correction TD1

Exercice 1 :

Soit P0 et P1 deux processus parallèles se partageant deux ressources R1 et R2. Les algorithmes de ces deux processus sont écrits comme suit :

```
var s1,s2 : sémaphore init 1,1 ;
```

```
Processus P0
Début
 a0: (1) P(s1)
     (2) utiliser R1
     (3) P(s2)
         utiliser R1 et R2
         V(s1)
         V(s2)
         Aller à a0
Fin
```

```
Processus P1
Début
 a1: (1') P(s2)
     (2') utiliser R2
     (3') P(s1)
          utiliser R1 et R2
          V(s2)
          V(s1)
          Aller à a1
Fin
```

1) À quelle situation anormale peut conduire l'exécution de ces deux processus ?
2) Donner une solution à ce problème.

Solution :

1) Considérons la séquence d'exécution : (1) , (1') , (2) , (2') , (3) , (3')
après l'exécution de (1) , (1') $s_1.val$ devient égal à $s_2.val = 0$
lorsque P_0 exécutera (3) il se bloquera , et P_1 se bloquera lorsqu'il exécutera (3') : les deux processus sont bloqués sans aucun moyen d'être réveillés : interblocage.

2) Une solution à ce problème est de procéder comme suit :

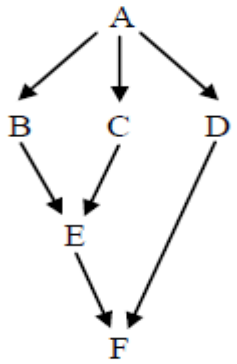
```
var s1,s2 : sémaphore init 1,1 ;
```

```
Processus P0
Début
 a0: P(s1)
     utiliser R1
     V(s1)
     P(s1)
     P(s2)
     utiliser R1 et R2
     V(s2)
     V(s1)
     Aller à a0
Fin
```

```
Processus P1
Début
 a1: P(s2)
     utiliser R2
     V(s2)
     P(s1)
     P(s2)
     utiliser R1 et R2
     V(s2)
     V(s1)
     Aller à a1
Fin
```

Exercice 2 :

On considère un ensemble de six tâches séquentielles {A, B, C, D, E, F}. La tâche A doit précéder les tâches B, C, D. Les tâches B et C doivent précéder la tâche E. Les tâches D et E doivent précéder la tâche F. Réaliser la synchronisation de ces tâches en utilisant les sémaphores ?




```
var SA,SB,SC,SD,SE : sémaphore init 0,0,0,0,0 ;

Tâche A
Début
 Exécution
 V(SA) ;  V(SA) ;  V(SA)
Fin

Tâche B
Début
 P(SA)
 Exécution
 V(SB)
Fin

Tâche C
Début
 P(SA)
 Exécution
 V(SC)
Fin

Tâche D
Début
 P(SA)
 Exécution
 V(SD)
Fin

Tâche E
Début
 P(SB) ; P(SC)
 Exécution
 V(SE)
Fin

Tâche F
Début
 P(SE) ; P(SD)
 Exécution
Fin
```

Exercice 3 :

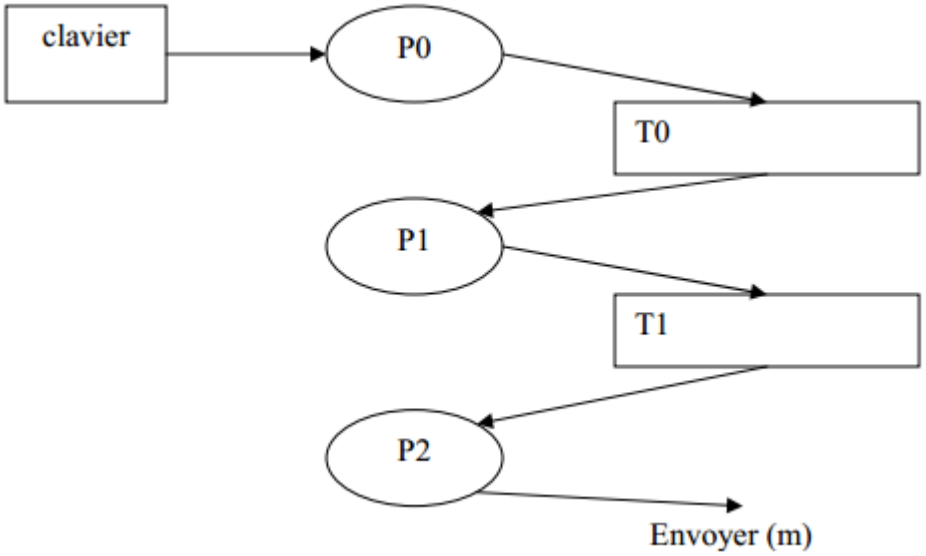
Considérez un système multicouche composé de trois couches P0, P1 et P2. Les couches sont des processus concurrents qui communiquent au moyen de deux tampons T0 et T1 de même taille N :

- P0 et P1 partagent le tampon T0,
- P1 et P2 partagent le tampon T1.

Chaque couche se charge d'un traitement particulier :

- Le processus P0 se charge de lire du clavier des messages qu'il traite avant de les déposer dans le tampon T0. Le traitement d'un message par la couche P0 consiste à l'encrypter. Il est réalisé par la fonction Encrypter suivante : **Message Encrypter (Message);** La fonction **Message Lire ();** permet de lire un message du clavier.
- Le processus P1 se charge de transférer directement les messages du tampon T0 vers le tampon T1,

- Le processus P2 récupère les messages du tampon T1 pour les envoyer à un destinataire. L'envoi d'un message est réalisé par la fonction Envoyer : **Envoyer (Message);**



a) Expliquez comment les processus peuvent utiliser les sémaphores pour contrôler les accès aux tampons partagés (exclusion mutuelle, pas d'interblocage).

Mutex0 et mutex1 pour contrôler les accès aux tampons.
Vide1, Vide0, Plein1 et Plein0 pour bloquer un processus si le tampon est vide ou plein.

Mutex0, mutex1 : Sémaphores pour ressources partagées
Plein0, plein1 : Sémaphores pour relation de précédence
Vide0, Vide1 : Sémaphores pour bloquer un processus s'il est vide

b) Donnez les pseudocodes des trois processus.

```
Semaphore
        mutex1=1, mutex0=1, Vide0=N, Vide1=N, Plein0 = 0, Plein1 = 0;

P0
Message m, mc;
int ip=0;
 Répéter
{       m= lire();
        mc = Encrypter(m);
        P(Vide0);
```

```
        P(mutex0);
        T0[ip] = mc;
        V(mutex0);
        ip = ip+1 mod(N)
        V(Plein0);
}


P1
int icp=0;
 Répéter
{
        P(Plein0);
        P(Vide1)
        P(mutex0);
        P(mutex1)
        T1[icp] = T0[icp];
        V(mutex1);
        V(mutex0)
        icp = icp+1 mod(N)
        V(Plein1);
        V(Vide0);
}
  P2
  Message mc;
  int ic=0;
   Répéter
  {

          P(Plein1)
          P(mutex1);
          mc = T1[ic];
          V(mutex1);
          ic = ic+1 mod(N)
          V(Vide1);
          Envoyer(mc);
  }
```