TD 4
Sémaphores et Exclusion Mutuelle

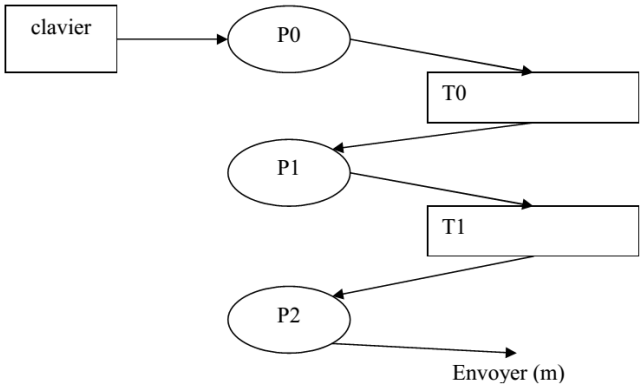
Exercice 1

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 et
- 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 P1 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).

b) Donnez les pseudocodes des trois processus.

Exercice 2

1) Expliquez ce qui peut arriver si la file d'attente d'un sémaphore est gérée selon la discipline LIFO (last in first out).

2) Expliquez un avantage de l'utilisation de moniteurs sur les sémaphores pour la synchronisation de processus.

3) Complétez, en ajoutant les sémaphores et les opérations P et V nécessaires, les codes du producteur et du consommateur suivants. Le producteur produit plusieurs ressources à la fois alors que le onsommateur consomme une seule ressource à la fois.

char T[N]; // tableau de N caractères

Semaphore Plein =0, Vide=N, Mutex=1

```
Producteur {
int ip=0, M;
char ch[N];
Repeter
{
M=Lire(ch,N);
        Pour i=1 à M Pas 1 faire

        Deposer(ch, M, ip);

        Pour i=0 à M-1 Pas 1 faire
        ip = (ip + M) % N;
}
}
```

```
Consommateur {
int ic=0;
char c;
Repeter
{
        c = Retirer( ic);
        ic = (ic+1) %N
}
Traiter(c);
}
```

La fonction « int Lire(char ch[], int N); » construit, dans ch, une chaîne de caractères de longueur comprise entre 1 et N inclusivement. Elle retourne la longueur de la chaîne. La fonction « void Deposer(char ch[], M, int ip); » insère, dans le tampon T, la chaîne de caractères ch. M est la longueur de la chaîne.

La fonction « char Retirer(int ic); » retire un caractère du tampon T. Elle retourne le caractère retiré.

La fonction « void Traiter(char c); » traite le caractère.

Exercice 3

Synchronisez au moyen de sémaphores l'enchaînement des opérations de fabrication de stylos à bille. Chaque stylo est formé d'un corps, d'une cartouche, d'un bouchon arrière et d'un capuchon. Les opérations à effectuer sont les suivantes :

- remplissage de la cartouche avec l'encre (opération RC),
- assemblage du bouchon arrière et du corps (opération BO),
- assemblage de la cartouche avec le corps et le capuchon (opération AS),
- emballage (opération EM).

Chaque opération est effectuée par une machine spécialisée (mRC, mBO, mAS, mEM). Les stocks de pièces détachées et d'encre sont supposés disponibles quand la machine est disponible. Les opérations RC et BO se font en parallèle. L'opération AS doit être effectuée, après ces deux opérations, en prélevant directement les éléments sur les machines mRC et mBO. Le produit assemblé est déposé dans un stock en attente de l'opération EM.

L'opération EM se fait donc après AS, à partir du stock. Le stock est supposé de taille N et de discipline FIFO.

```
mRC( )
{  while (1)
   {
      RC( ) ;

   }
}
```

```
mBO( )
{   while (1)
    {

        BO( ) ;

    }
}
```

```
mAS( )
{    while (1)
     {
        AS( ) ;
     }
}
```

```
mEM( )
{     while (1)
          {
             EM() ;

          }
}
```