Contrôle terminal Session 1 - 7/1/2020

Durée : 2h

Une feuille A4 recto-verso manuscrite (cours et/ou td) autorisée.

Barème donné à titre indicatif : Ex.1 : 4 - Ex. 2 : 4 - Ex. 3 : 8 - Ex 4 : 4

Exercice 1. Synchronisation des processus
Soient trois processus concurrents P_1, P_2 et P_3 qui partagent les variables n et *out*. Pour contrôler les accès aux variables partagées, un programmeur propose les codes suivants :

```
Semaphore mutex1 = 1 ;
Semaphore mutex2 = 1 ;
```

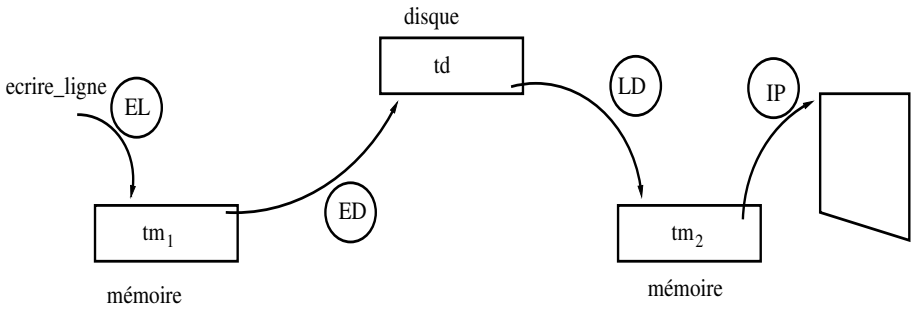
Code du processus P_1	Code du processus P_2	Code du processus P_3
`P(mutex1) ;`	`P(mutex2) ;`	`P(mutex1) ;`
`P(mutex2) ;`	$out = out - 1$;	$n = n + 1$;
$out = out + 1$;	`V(mutex2) ;`	`V(mutex1) ;`
$n = n - 1$;		
`V(mutex2) ;`		
`V(mutex1) ;`		

Cette proposition est-elle correcte ? Justifiez votre réponse.
Si elle n'est pas correcte, indiquer parmi les 4 conditions requises pour réaliser une exclusion mutuelle correcte, celles qui ne sont pas satisfaites et proposer une solution correcte.

Exercice 2. Considérons un système de gestion de fichiers où le disque est géré à l'aide d'une FAT (File Allocation Table). La FAT est un tableau d'entiers qui possède autant de lignes qu'il y a de blocs sur le disque. Un bloc i est libre si $fat[i]$ = `FAT_FREE`. Si le bloc i est le dernier bloc d'un fichier, on a $fat[i]$ = `FAT_EOF`. Toute autre valeur positive sur $fat[i]$ indique le bloc suivant.
Écrire la fonction d'allocation qui recherche un espace libre de n blocs consécutifs et renvoie l'adresse du premier bloc (ou -1 en cas d'échec) en se basant sur le principe du meilleur ajustement (best-fit).

`alloc_bloc_best_fit (entier[] fat, entier` n`) retourne entier`

Exercice 3. Soit le spouleur d'impression schématisé comme suit :



où `EL, ED, LD` et `IP` sont quatre activités autonomes car elles s'exécutent sur des processus distincts. De plus, la procédure `EL (ecrire_ligne)` est une procédure utilisateur.

On pose les hypothèses suivantes :

1. le transfert se fait par blocs de taille fixe, égale à celle d'une ligne. Chaque tampon $\texttt{tm}_1$, $\texttt{tm}_2$ et `td` est respectivement géré par un moniteur $\texttt{tampon}_1$, $\texttt{tampon}_2$ et $\texttt{tampon}_{\texttt{disc}}$ de même structure qui a pour rôle
   - d'assurer l'exclusion mutuelle au tampon,
   - d'assurer la synchronisation sur les conditions *tampon plein* et *tampon vide*
2. $\texttt{tm}_1$, $\texttt{tm}_2$ et `td` sont de taille respective N_1, N_2 et $Ndisc$, les tailles sont exprimées en nombres de blocs,
3. dans le tampon $\texttt{tampon}_{\texttt{disc}}$, les opérations de dépôt et de retrait (`entrer, sortir`) sont des E/S qui utilisent le moniteur de gestion du disque,
4. le système tout entier fonctionne en régime permanent sans limitation de lignes à imprimer et les séquences de traitement des erreurs sont omises.

Questions :

1. Écrire la structure du moniteur $\texttt{tampon}_1$ en supposant qu'on dispose des procédures `entrer(l:ligne)` et `sortir() retourne ligne`.
   (Donner le code des procédures `déposer(b:ligne)` et `retirer() retourne ligne` et préciser les variables et la (les) condition(s) du moniteur.)
2. En définissant $\texttt{tm}_1$, $\texttt{tm}_2$ et `td` comme des tableaux d'éléments de la taille d'une ligne, et des pointeurs `tête` et `queue` locaux à chaque moniteur, écrire les procédures `entrer` et `sortir`
   a) dans le moniteur $\texttt{tampon}_1$,
   b) dans le moniteur $\texttt{tampon}_{\texttt{disc}}$ pour lequel les opérations de dépôt et de retrait sont des E/S qui utilisent la primitive `disque.écrire(source, destination)` du moniteur de gestion du disque.

   `tête` pointe sur la prochaine ligne à prélever et `queue` sur la première place libre dans le tampon.
3. Écrire les processus `IP, ED, LD` et la procédure utilisateur `EL`. Le processus `IP` utilisera la procédure `impr.écrire(l)` du moniteur de gestion de l'imprimante.

Exercice 4. Dans r_0 se trouve l'adresse d'une chaîne de caractères s terminée par l'octet nul et composée de mots. Un mot est un ensemble de caractères ne comprenant pas le caractère espace (32 en décimal). On vous demande d'indiquer ce que calcule ce programme et de préciser où se trouve le résultat de ce calcul.

```
          MVI r1, #0
loop1 :   LDB r2, (r0)
          ADD r0, r0, #1
          JZ r2, fin
          SUB r31, r2, #32
          JZ r31, loop1
          ADD r1, r1, #1
loop2 :   LDB r2, (r0)
          ADD r0, r0, #1
          JZ r2, fin
          SUB r31, r2, #32
          JNZ r31, loop2
          JMP loop1
fin :
```