Pr FARAH Le 25/01/2016
Département d'informatique
Promotion Licence 3eme année
Documents non autorisés Durée 1h30mn
Module : SE2

EMD1

Question 1 : (5 pts)

Qu'est-ce que l'attente active ? quel est le problème qui se pose ?

A quoi servent les moniteurs ? Qu'est-ce qu'ils apportent en plus par rapport aux autres solutions?

Exercice 1 :(7 pts)

On veut installer une variante du producteur consommateur qui permette à deux producteurs P1 et P2 (processus clients) de partager un même consommateur C (processus serveur), tout en ayant chacun son tampon de dépôt de message (chacun a un tampon de 5 cases). On ajoute en plus que les messages déposés par P1 sont prioritaires par rapport aux messages déposés par P2. Cela donne la gestion de données suivante :

```
processus P1; -- producteur prioritaire
X1: Message;
Queue1 : Integer:= 0;
Debut
Boucle
        preparer(X1);
        T1(Queue1) := X1;
        Queue1 := (Queue1 + 1) mod 5;
        Nombre := Nombre+ 1;
fin boucle;
Fin P1 ;
```

```
Processus P2; -- deuxième producteur
X2 : Message;
Queue2: Integer := 0;
Debut
Boucle
        preparer(X2);
        T2(Queue2) := X2;
        Queue2 := (Queue2 + 1) mod 5;
fin boucle;
Fin P2 ;
```

```
Processus C ; -- c'est le consommateur
Prioritaire : Boolean; -- permet de choisir entre T1 et T2
Tete1, Tete2 : Integer := 0; Y : Message;
Debut
Boucle
        Prioritaire := Nombre > 0; -- vrai s'il y a au moins un message dans T1
        Si Prioritaire alors Nombre := Nombre - 1; fin si; -- et C va le consommer
        Si Prioritaire alors
                Y := T1(Tete1); Tete1 := (Tete1 + 1) mod 5;
        Sinon
                Y := T2(Tete2); Tete2 := (Tete2 + 1) mod 5;
        fin si;
        Fin boucle;
Fin C;
```

Compléter les programmes en ajoutant le contrôle de concurrence et la synchronisation par sémaphores (ne mettre que ceux qui sont nécessaires, tout sémaphore en plus sera sanctionné).

Exercice 2 : (8 pts)

Les supporteurs de deux équipes locales veulent assister à un match de football, les deux équipes appartiennent à la même ville mais sont localisées de deux cotés différents d'une rivière, celle-ci est traversée par un pont P à voie unique, on peut le traverser en sens inverse (sens A et sens B). A tout moment, le pont ne doit contenir que des supporteurs allant dans un sens uniquement. On assumera que le nombre de supporteurs pouvant traverser le pont dans un sens est illimité. On assimilera les supporteurs à des processus parallèles synchronisés par sémaphores.

Ecrire les algorithmes de chaque classe de processus.

Question 1

L'attente active permet d'assurer l'exclusion mutuelle entre processus tout en restant actif au niveau du processeur, le fait de garder le processeur sans travailler est un problème, mauvaise gestion de la ressource. (2 pts)

Les moniteurs sont un moyen évolué pour'assurer l'exclusion mutuelle entre les processus, ils ont étés développés pour éviter les problèmes de programmation posés par les semaphores (mettre un P à la place de V ou le contraire) (3 pts)

Exercice 1 :

```
Processus P1;
X1: Message;
Queue1 : Integer;:= 0;
begin
loop
P(Vide1);  (1 pt)
preparer(X1);
T1(Queue1) := X1;
Queue1 := (Queue1 + 1) mod 5;
V(Plein); -- unique semaphore de consommation (0,5 pt)

P(Mutex); -- si l'incrementation n'est pas atomique (1 pt)
Nombre := Nombre+ 1;
V(Mutex);  (1 pts)
end loop;
end P1 ;
```

```
Processus P2 ;
X2 : Message;
Queue2: Integer := 0;
begin
loop
P(Vide2); (1 pt)
preparer(X2);
T2(Queue2) := X2;
Queue2 := (Queue2 + 1) mod 5;
V(Plein); -- meme semaphore de
consommation (0,5 pt)

-- P2 n'incrémente jamais Nombre

end loop;
end P2 ;
```

```
task C ; -- c'est le consommateur
task body C is
Prioritaire : Boolean; -- permet de choisir entre T1 et T2
Tete1, Tete2 : Integer := 0; Y : Message;
begin
loop
P(Plein); -- attendre un message (0,5 pt)
P(Mutex); -- exclusion mutuelle pour cohérence de la lecture et decrementation (1  pt)
Prioritaire := Nombre > 0; -- vrai s'il y a au moins un message dans T1
if Prioritaire then Nombre := Nombre - 1; end if; -- et C va le consommer
V(Mutex); (1 pt)
if Prioritaire then
Y := T1(Tete1); Tete1 := (Tete1 + 1) mod 5; V(Vide1); (0,5 pt)
else
Y := T2(Tete2); Tete2 := (Tete2 + 1) mod 5; V(Vide2); (0,5 pt)
end if;
end loop;
end C;
```

Exercice 2

Ce qui donne la solution suivante :

```
var Sa,Sb,Sab :
sémaphore init 1,1,1 ;
na,nb :
entier init 0 ;
```

```
Processus A
Début
P(Sa)    (1 pt)
na := na + 1
si na=1 alors P(Sab) finsi (1,5pt)
V(Sa)   (0,5 pt)
T r a v e r s e r l e p o n t
```

```
Processus B
Début
        P(Sb)  (1 pt)
                nb := nb + 1
                si nb=1 alors P(Sab) finsi  (1,5 pt)
        V(Sb)  (0,5 pt)
        T r a v e r s e r l e p o n t
```

P(Sa) (0,5 pt)
na := na – 1
si na=0 alors V(Sab) finsi (1pt)
V(Sa) (0,5 pt)
Fin

P(Sb) (0,5 pt)
nb := nb – 1
si nb=0 alors V(Sab) finsi (1 pt)
V(Sb) (0,5 pt)
Fin