

Examen de rattrapage

Module de Systèmes d'exploitation II

Corrigé

**Exercice 1 :** On considère le problème du Producteur-Consommateur où le buffer possède une seule case. Proposez une solution à ce problème en utilisant les moniteurs de Hoare.

Réponse :

Structure du moniteur :

```

Type Producteur_Consommateur = Monitor

Var
    Plein : boolean ;
    X, Y : Condition
    Buffer : Char ;

Procedure Entry Deposer(M : Char)
Begin
    If Plein Then X.Wait;
    Buffer:=X, Plein := True; Y.Signal
End

Procedure Entry Prelever(M : Char)
Begin
    If Not Plein Then Y.Wait;
    X := Buffer; Plein := False; X.Signal
End

Begin /* Initialisation */
    Plein := False
End.
    
```

Structure des processus : PC étant une instance du type de moniteur Producteur\_Consommateur.

Processus Producteur Début Cycle Produire(Message) ; PC_Deposer(Message) Fin Cycle Fin	Processus Producteur Début Cycle PC_Prelever(Message) ; Consommer(Message) Fin Cycle Fin
----------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------

**Exercice 2 :** On reprend le problème du point de rendez-vous vu en TD. Proposez une solution à ce problème en utilisant les moniteurs de Horare.

Réponse :

Structure du moniteur :

```

Type Rendez_Vous = Monitor

Const N = 10
Var
    I : integer ;
    X : Condition

Procedure Entry Arrivee
Begin
    I := I + 1;
    If I = N Then X.Signal

End

Procedure Entry Franchir
Begin
    X.Wait;
End

Begin /* Initialisation */

    I := 0

End.

```

Structure des processus : RV étant une instance du type de moniteur Rendez\_Vous.

Processus Pi	Processus Maitre
Début	Début
...	...
RV_Arrivee	RV_Franchir
...	...
Fin	Fin

(07 points)

**Exercice 3 :** Ecrire un programme Java qui crée deux (02) threads affichant chacun les nombres entiers compris entre 1 et 100.

Réponse :

```
// LanceCompteurs.java
//

class ThreadCompteur extends Thread {
int no_fin;
// Constructeur
ThreadCompteur (int fin) {
no_fin = fin;
}
// On redéfinit la méthode run()
public void run () {
for (int i=1; i<=no_fin ; i++) {
System.out.println(this.getName i);
}
}
}

// Classe lançant les threads
class LanceCompteurs {
public static void main (String args[]) {
// On instancie les threads
ThreadCompteur cp1 = new ThreadCompteur (100);
ThreadCompteur cp2 = new ThreadCompteur (100);
// On démarre les deux threads
cp1.start();
cp2.start();
// On attend qu'ils aient fini de compter
while (cp1.isAlive() || cp2.isAlive) {
// On bloque le thread 100 ms
try {
Thread.sleep(100);
} catch (InterruptedException e) { return; }
}
}
}
```

(06 points)