

TD Programmation Système

Synchronisation avec Sémaphores

Série 5 : Correction

Exercice 1 : Synchronisation (barrière)

Question 1 :

Le sémaphore de l'exclusion mutuelle *Sema1* est toujours initialisé à 1. Le sémaphore *Sema2* pour la Synchronisation de la barrière est initialisé à 0 permettant ainsi le blocage du premier processus exécutant la fonction *Wait*. Le compteur permettant de compter le nombre de processus bloqués est initialisé à 0 au départ. Le nombre maximum de processus bloqués est initialisés à N

```
void InitBarrière (Barrière *B; int Maximum)
{
    Init (B→Sema1, 1);
    Init (B→Sema2, 0);
    B→Count    = 0 ;
    B→Maximum   = N;
}
```

Question 2 :

Complétez le code de la procédure d'attente donné ci-dessous :

```
void Wait (Barrière *B)
{
    Boolean Do_Wait = True ;
    int I ;

    P(B→Sema1) ;
    B→Count++ ;
    if (B→Count == B→Maximum ){
        for (I=0 ; I < (B→Maximum-1) ; I++ ) V(B→Sema2) ; ;
        B→Count = 0;
        Do_Wait = False ;
    }
    V(B→Sema1) ;
    if (Do_Wait) P(B→Sema2) ;
}
```

Exercice 2 : Ordonnancement

Deux sémaphores sont nécessaires pour gérer les synchronisations entre P1, P2 et P3 :

- S1 pour gérer les synchronisations entre P1 et P2
- S2 pour gérer les synchronisations entre (P1, P2) et P3

Val(S1)= Val(S2)= 0

