

Série 5

30 mai 2005

A rendre le: 13 juin 2005

A rendre dactylographiée

Exercice 1 : Ecriture d'un noyau. Familiarisation avec la classe `Coroutine`.

Le paquetage `exo1` contient un exemple d'utilisation de la classe `Coroutine` du paquetage `epfl.lsr.SysExpl.couche0` (le code est sur la page web du cours). Dans cet exemple, le programme principal crée une seule coroutine (`Co`) et lui passe le contrôle. La coroutine `Co` exécute le code suivant:

```
loop
    écrire le texte "La coroutine Co s'exécute"
    sleep 1 seconde
end loop
```

Compilez et exécutez cet exemple pour voir le résultat.

Utilisez la classe `Coroutine` pour réaliser le programme suivant. Le programme créera trois coroutines `Co1`, `Co2`, `Co3`, et passera initialement le contrôle à `Co1` (en utilisant la méthode statique `transfer` de `Coroutine`).

La coroutine `Coi` exécutera le code suivant:

```
loop
    écrire le texte "La coroutine Coi s'exécute"
    sleep 1 seconde
    passer le contrôle à la coroutine Co(i mod 3)+1
        /* utiliser Coroutine.transfer() */
end loop
```

Indication: Pour implémenter une coroutine, faites comme dans l'exemple: une classe qui hérite de `Coroutine` et qui surcharge la méthode `run()`.

Classes à rendre: `co1.java`, `co2.java`, `co3.java`, ainsi que le fichier du programme principal.

Exercice 2 : Ecriture d'un noyau implémentant les sémaphores.

Complétez la classe `Semaphore` (qui définit les méthodes `P` et `V`) et `NoyauSemaphore` (qui définit les méthodes `createThread` et `demarrerSysteme`) du paquetage `epfl.lsr.SysExpl.couche1Semaphore`. Utilisez pour cela la classe `Coroutine` du paquetage `epfl.lsr.SysExpl.couche0`.

Testez les deux classes que vous avez complétées en utilisant le programme Java appelé `ProdCons` du paquetage `exo2`.

Remarque. La variable `threadsPrets` de la classe `NoyauSemaphore` est déclarée *protected* ce qui permet d'y avoir accès depuis la classe `Semaphore`.

Classes à rendre: Fichiers `Semaphore.java` et `NoyauSemaphore.java` complétés.

Exercice 3 : ioTransfer

Utiliser la classe `Coroutine` du paquetage `epfl.lsr.SysExpl.couche0` pour réaliser le programme suivant. Le programme créera deux coroutines `CO1`, `CO2` et passera initialement le contrôle à `CO1` (utiliser la méthode `transfer`).

La coroutine `CO1` exécutera le code suivant:

```
loop
    Écrire le texte "La coroutine CO1 s'exécute" ;
    suspendre la coroutine CO1 jusqu'à ce qu'un caractère soit entré au clavier, et
        passer le contrôle à CO2 /* utiliser la méthode ioTransfer */ ;
end loop
```

La coroutine `CO2` exécutera le code suivant :

```
loop
    Écrire le texte "La coroutine CO2 s'exécute" ;
    sleep 1 seconde ;
end loop
```

Exercice 4 : TimeSlicing

Compléter la classe `NoyauTimeSlicing` (qui définit les méthodes `createThread` et `demarrerSysteme`) du paquetage `epfl.lsr.SysExpl.couche1TimeSlicing`. Une fois `demarrerSysteme` appelé, le noyau passera le contrôle à tour de rôle à chacun des threads créés. La commutation aura lieu chaque fois que l'utilisateur appuiera sur <RETURN> au clavier.

Tester la classe complétée en utilisant le programme Java appelé `TestTimeSlicing`.

Indication: Utiliser `ioTransfer` dans `demarrerSysteme`.

Remarque: Contrairement à l'exercice 2 ce noyau n'implémente pas les sémaphores.

Exercice 5 : TimeSlicing et Sémaphores

Adapter les classes `NoyauSemaphore` et `Semaphore` de l'exercice 2 pour y ajouter du timeslicing. Autrement dit, le noyau passera le contrôle à un autre thread à chaque fois que l'utilisateur appuiera sur <RETURN> au clavier.

Indication : adapter les méthodes `demarrerSysteme`, ainsi que `P` et `V`. Que se passe-t-il si <RETURN> est entré au clavier alors qu'une des deux méthodes `P` ou `V` s'exécute ?

Tester les deux classes que vous avez complétées en utilisant le programme Java appelé `ProdConsTimeSlicing`.