

TP3: Synchronisation

Système D'exploitation

Samir AKNINE, Antoine GRÉA

♣ Compétences À Acquérir

- ✓ Savoir synchroniser l'exécution des processus
- ✓ Connaître les cas de blocages.
- ✓ Comprendre la notion d'atomicité d'exécution

! Les questions marquées d'un drapeau 🚩 seront à rendre dans le
● compte-rendu global.

1 Inter-blocage

1.1 Sémaphores

? Étudiez et compilez le fichier
● [semaphore.c](#). Que se passe-t-il si on inverse le **up** et le **down** ?

1.2 Instinct Parental

? Créez un programme où le processus père s'exécute toujours avant le processus fils pour afficher du texte. Tester cette propriété en rendant le père plus lent avec la fonction **sleep(1)**.



2 Affichage Collaboratif

L e but de cet exercice est de créer un programme avec N processus fils qui vont afficher chacun leur tour une ligne de fichiers respectif.

Cela donne :

```
Le processus 1 affiche la ligne 1 du fichier 1
Le processus 2 affiche la ligne 1 du fichier 2 ...
Le processus N affiche la ligne1 du fichier N
Le processus 1 affiche la ligne 2 du fichier 1
Le processus 2 affiche la ligne 2 du fichier 2 ...
```

2.1 Afficher Un Fichier

- Créez la fonction `print` qui à partir d'un numéro `i` affiche le contenu du fichier `data/fi`. Vous pouvez utiliser le code inclus dans le fichier `lecture.c`

2.2 Contrôle D'exécution

- En utilisant un sémaphore, faites en sorte que votre programme crée 6 processus fils qui afficheront leur fichier respectif avant que le père commun ait affiché "**Résultat :**"

2.3 🚩 Chacun Son Tour !

- Créez 6 sémaphores (en passant 6 en paramètre de `semget` pour le nombre de sémaphores) et faite attendre chaque processus sur leur sémaphore respectif. Ajoutez un affichage avant d'attendre sur le sémaphore.
- Proposez un mécanisme qui permet d'exécuter vos processus dans l'ordre afin que chacun d'eux affiche une seule ligne de leur fichier.
- Justifier et expliquer comment votre solution garantie l'ordre d'exécution. Vous construirez une preuve formelle.