

## TP5 – Mémoire partagée et sémaphores (2)

### Rappel

- 👉 Le TP sera noté. La copie directe des fichiers (maquillée ou non) sera sanctionnée pour les 2 (copieur et copié).
- 👉 Pour les sémaphores, vous utiliserez votre fichier `mes_semaphores.c` du dernier TP : il faut que les fonctions P et V fonctionnent.

### Dépôt du TP

Le TP sera à déposer en 2 temps :

1. à la fin de la séance.
2. avant la date indiquée sur ARCHE.

Réalisez un fichier ZIP avec les fichiers du TP (fichier test de la fin de partie 2 inclus mais **PAS LE FICHIER AVEC LE TEXTE LONG ExtraitLE-Monde2002.txt**. N'oubliez le `.h` pour les fonctions de `mes_semaphores.c`).

**Pour le dernier dépôt**, respectez les consignes suivantes : voir toute fin de l'énoncé... mais pas maintenant. Ces dernières consignes permettront d'alléger la note de quelques points si par malheur elles n'étaient pas respectées.

Le TP est à réaliser progressivement dans l'ordre partie par partie et ► par ►.

## Fichiers et partage...

Le programme à réaliser aura pour objectif final de calculer la fréquence d'apparition des lettres contenues dans un ensemble de phrases d'un (long) fichier texte.

### Partie 1

- Vous trouverez le programme `LectureCorpusLettre.c` sur ARCHE. Ce programme réalise les points suivants :
  - Lit ligne par ligne un fichier dont le nom est donné en argument.

- Utilise un tableau (*histo*) qui stocke le nombre de fois qu'une des 26 lettres (minuscules) de l'alphabet est rencontrée.
- Toutes les 10000 lignes, le programme écrit un fichier contenant le décompte à cet instant. Le fichier est nommé "*nomgénérique\_pid\_compteur.txt*" avec *nomgénérique* donné en paramètre et *compteur* est incrémenté au fur-et-à-mesure.
- À la fin, il génère le fichier avec le décompte final (nom du fichier : "*nomgénérique\_pid\_final.txt*") et affiche le nombre de lignes et de caractères traités.

Compilez le programme et faites-le fonctionner. Vous pouvez utiliser le fichier *ExtraitLeMonde.txt* (ou n'en prendre que quelques lignes).

► Ajoutez comme arguments au programme :

- le numéro de ligne à partir de laquelle le traitement doit commencer
- et le nombre de lignes à traiter.

Adaptez le programme pour qu'il passe les premières lignes qu'il ne doit pas traiter (= lire la ligne et ne rien faire) et qu'il affiche à la fin le nombre de lignes effectivement traitées. Cas particulier : si l'on met un **nombre de lignes à traiter** égal à 0, alors on fait le traitement jusqu'à la fin. Ces arguments seront nécessaires dans la partie suivante.

## Partie 2

Nous allons maintenant répartir le travail entre plusieurs processus.

- Le programme précédent sera pris en charge par un fils : Faites un programme père qui appelle ce programme une fois (par `fork+execl`) pour traiter le fichier en entier. Les arguments du père seront seulement le nom du fichier à traiter et le nom générique des fichiers à générer. Pour les arguments du fils dans `execl`, le nombre de ligne à traiter est donc à 0.
- Modifiez maintenant le programme père et le programme fils pour ce soit le père qui crée un tableau de 26 cases et qui le partage avec le fils en utilisant un segment de mémoire partagée. Mais le fils gardera son tableau (*histo*) pour stocker ses valeurs en attendant de les écrire dans le tableau partagé (juste avant chaque écriture fichiers) : le fils gèrera donc 2 tableaux : 1 local et 1 partagé.
- Le programme père (en fin de la fonction `main`) affichera la fréquence d'apparition de chaque lettre (en pourcentage) par rapport au nombre de lettres lues (il suffit d'utiliser le tableau partagé).
- Enfin, ajoutez un dernier argument au père : le nombre de processus qui devront se partager le calcul de l'histogramme. Le père partagera le travail : cela revient à calculer la taille des blocs à traiter et le numéro de chaque ligne de départ (pour calculer le nombre de ligne total du fichier inspirez-vous de la partie du code du fils qui le fait aussi). Ensuite il créera les processus fils nécessaires. Commencez avec 2 processus (les fils traitent donc une moitié chacun). Il faudra penser à protéger l'accès au tableau partagé dans les fils et attendre la

fin des fils avant que le père n'affiche quelque chose. Vous créerez un fichier de test avec quelques phrases de type :

```
aaaa  
bbbb  
cccc  
dddd
```

Vous pourrez alors plus facilement évaluer le résultat, avant de le faire sur le fichier traiter. Rappel : vous utiliserez votre fichier *mes\_semaphores* du dernier TP.

## Partie 3

- ▶ Comparez le temps d'exécution (sur le fichier à traiter) pour 1, 2, 4 et 8 processus.
    - Vous utiliserez la commande `time` du Shell (ex : `time pere ExtraitLeMonde2002.txt...`). Que signifie le temps "real", "user" et "sys" ?
    - Suivant le nombre de processus, le temps d'exécution est-il différent ? Pourquoi ? (indice : dépend du nombre de coeurs présents sur la machine)
- Vous mettrez vos résultats (les temps obtenus) et explications en haut dans votre fichier *pere.c*.




### Dépôt du TP

Le TP est à déposer en 2 temps :

1. à la fin de la séance.
2. avant la date indiquée sur ARCHE.

Réalisez un fichier ZIP avec les fichiers du TP (fichier test de la fin de partie 2 inclus mais **PAS LE FICHIER AVEC LE TEXTE LONG ExtraitLE-Monde2002.txt**. N'oubliez le `.h` pour les fonctions de *mes\_semaphores.c*).

**Pour le dernier dépôt**, respectez les consignes suivantes :

-  Indiquez comment se compilent vos programmes : idéalement réalisez un `makefile` (si vous savez faire), sinon indiquez les commandes de compilation dans un `README`.
-  **ATTENTION** : pour les clés n'utilisez pas de chemin absolu ou de fichiers particuliers ! qui ne fonctionneraient pas sur une machine différente.
-  Dans le *main*, réalisez un affichage de l'usage lorsque l'on tape le nom de l'exécutable seul (= gérez le nombre d'arguments donnés).

**Ces dernières consignes permettront d'alléger la note de quelques points si par malheur elles n'étaient pas respectées.**