TD de Système n° 5 : Manipuler fichiers et répertoires avec POSIX

Les exercices 1, 2 et 3 sont les mêmes que ceux du TD3, à la différence qu'il faudra utiliser les équivalents bas niveau POSIX des fonctions de manipulation de fichiers.

Exercice 1 : La commande « `cat` »

Nous nous proposons de réécrire ici la commande « `cat` », qui recopie son entrée standard sur sa sortie standard. Cette commande peut éventuellement recevoir un ou plusieurs noms de fichiers en paramètre : ce sont alors ces fichiers (et non plus l'entrée standard) qui sont copiés séquentiellement sur la sortie standard.

Vous pouvez utiliser les fonctions de `fcntl.h` comme :

- `int open(const char *path, int oflag, ...);`
- `int close(int fildes);`

et de `unistd.h` comme :

- `ssize_t read(int fildes, void *buf, size_t size);`
- `ssize_t write(int fildes, void *buf, size_t size);`

Si vous le souhaitez, vous pouvez aussi utiliser la fonction de `string.h` `char *strchr(const char *s, int c);` qui renvoie un pointeur vers la prochaine occurrence de `c` dans la chaîne `s` ou `NULL` si `c` n'apparaât pas.

1. Écrire une fonction qui prend en argument un descripteur de fichier et le copie sur la sortie standard (en utilisant un tampon).
2. Écrire la fonction `main` correspondante, qui appelle la fonction précédente sur chacun des fichiers listés sur la ligne de commande ou l'entrée standard si aucun argument n'est fourni.
3. Modifier la première fonction pour qu'elle puisse numéroter chaque ligne (c'est-à-dire préfixer chaque ligne de son numéro et d'une espace).
4. Modifier la fonction `main` pour qu'elle analyse les options de la ligne de commande et accepte l'option `-n`.

Exercice 2 : La commande « `tr` »

Écrire un programme « `montr` » tel que `montr` *fichier1 fichier2* a b lise un fichier *fichier1* et le recopie dans le fichier *fichier2* en remplaçant tous les caractères a par des caractères b.

Exercice 3 : La commande « `cp` » à l'envers

Écrire un programme permettant de copier le contenu d'un fichier dans un autre en inversant l'ordre des caractères. Vous aurez besoin des fonctions

- `ssize_t read(int fildes, void *buf, size_t size);`
- `ssize_t write(int fildes, void *buf, size_t size);`
- et `off_t lseek(int fd, off_t offset, int whence);`.

Exercice 4 : Union de listes triées

Écrire une commande qui prend en argument trois noms de fichiers et qui écrit dans le troisième l'union des listes triées contenues dans les deux premiers : on suppose que les deux premiers fichiers contiennent chacun une liste croissante d'entiers séparés par des espaces ou des retours à la ligne ; le troisième fichier doit contenir en sortie la liste de tous ces entiers, sans repétition et classés par ordre croissant, séparés par des retours à la lignes. On utilisera les appels systèmes « `read` » et « `write` ».