TP UNIX n°4

Fonctions utiles : pipe, close, read, write

1°) Ecrire un programme qui ouvre un tube (pipe) puis génère un processus fils (fork) avec lequel il va communiquer par ce tube.
Le père lit des caractères au clavier (getchar) et les écrit dans le tube, le fils récupère ces caractères et les affiche après les avoir mis en majuscule (fonction toupper). La saisie du caractère '$' termine le programme. Attention pour éviter que le processus fils ne continue à attendre que des caractères arrivent il faut que le père envoie le caractère $ dans le tube et que le fils s'en serve pour se terminer.

Exécuter ce programme et vérifier qu'il fonctionne correctement.
Remarque : Il faut taper 'entrée' après chaque caractère saisi pour que getchar prenne en compte la saisie.

2°) La méthode utilisée ici pour arrêter le processus fils (envoi du caractère '$') n'est pas toujours possible. En effet cela suppose de pouvoir disposer d'un caractère spécial non utilisé par ailleurs. Si, par exemple, le programme utilisait le tube pour passer le contenu de fichiers à son fils il faudrait être sûr qu'aucun fichier ne contient de caractère '$'. Pour éviter ce problème il vaut mieux que le fils détecte la terminaison du transfert par la fermeture du tube par son père. Ceci est possible car la fonction de lecture (read) retourne 0 lorsqu'elle ne trouve plus rien à lire parce que le tube est fermé en écriture. Modifier le programme précédent pour que le processus fils s'arrête selon ce procédé.

Exécuter ce programme et vérifier qu'il fonctionne correctement.

3°) Modifier le programme du 2°) pour que le processus père s'arrête sur EOF au lieu de '$'. De plus le processus père affichera le message "fin de transfert" lorsqu'il fermera le tube.
En utilisant une redirection (<), faire fonctionner votre programme à partir du contenu d'un fichier au lieu du clavier.
A quel moment apparaît le message "fin de transfert" ?