TRAVAUX PRATIQUE DE SE

Entrées/Sorties

Exercice 1 :

- Ecrire un programme **Init** qui crée un fichier binaire base.dat qui contient 4 entiers initialisés à 100.
  On utilisera les fonctions d'E/S système : open, close, read, write.
  La commande od permet d'afficher un fichier binaire.
  Exemple : od –d base.dat
- Ecrire un programme **Reserve** qui décrémente dans base.dat le nième entier (passé en argument) si il n'est pas nul.
  Exemple :
  $ Reserve 2
  décrémentera le deuxième entier de base.dat
  On utilisera la fonction lseek pour se positionner à la bonne place dans le fichier
- Que se passe-t-il si deux programmes veulent décrémenter le même entier en même temps ?
  Corriger Reserve en utilisant la fonction : lockf

Exercice 2 :
Ecrire la fonction Ls(char *repertoire) qui affiche la liste des fichiers contenus dans répertoire en exécutant la commande
« ls repertoire ».

Exercice 3 :
Ecrire la fonction LsDansFichier(char *repertoire, char *fichierResultat) qui copie dans fichierResultat la liste des fichiers contenus dans répertoire en exécutant la commande
« ls repertoire > fichierResultat »

Exercice 4 :
Ecrire la fonction NbFichier(char *repertoire) qui **affiche** le nombre de fichiers contenus dans « repertoire » en exécutant la commande
« ls repetoire | wc –l ».

Exercice 5 :
Ecrire la fonction int NbFichier(char *repertoire) qui **retourne** le nombre de fichiers contenus dans « repertoire » en exécutant la commande
« ls repetoire | wc –l » et en lisant le résultat de wc –l.

Exercice 6 : Amélioration du MiniBash
Si dans la ligne de commande on entoure obligatoirement d'un espace les signes <, >, >>, la fonction **Ligne2Argv** du TP1 va découper correctement la commande en mettant le caractère dans un des arguments, et le nom du fichier associé dans le suivant. En tenant compte de cette

contrainte (afin de facilité les choses), modifier le programme **minibash** du TP1 pour qu'il gère les redirections des entrées sorties.

Exemple :

```
Entrer Commande > ls > fichier
Entrer Commande > wc –l < fichier
12
Entrer Commande >
```

Exercice 7 :
Modifier le programme **minibash** du TP1 pour qu'il gère les commandes avec des tubes.
Exemple :

```
Entrer Commande > ls | wc –l
12
Entrer Commande >
```

Exercice 8 :
On souhaite à présent améliorer les programmes **ExecFile**, mais surtout **ExecFileBatch** et **ExecFileBatchLimite** (qui mélangent les sorties des commandes exécutées en même temps), afin de conserver dans des fichiers séparés les sorties standards de chaque commande exécutée. Ces fichiers seront regroupés dans un répertoire dont le nom sera le nom du fichier contenant les commandes (**data** dans l'exemple du TP1) concaténé à la date de lancement du programme ExecFile en secondes (epoch).

Exemple : l'exécution de **`ExecFile data`** produira le répertoire `data.1633599916`

Pour créer le répertoire, il suffit d'exécuter la commande bach pour créer un répertoire avec votre fonction **Execute**.

Ce répertoire contiendra 2 fichiers pour chaque commande exécutée :

- « `numerodepid.out` » qui contiendra sa sortie standard,
- « `numerodepid.err` » qui contiendra sa sortie erreur.

Il contiendra également un fichier `rapport.log` contenant le rapport final d'exécution.

Exemple:

```
>ls data.1633599916
125.out    125.err         126.out    126.err    etc.
rapport.log
```