SY5 – Système
TP n° 3 : « mon_ls »

Le but de ce TP est de cloner « `ls` ». On simplifiera la syntaxe sous la forme :

`mon_ls` <*switch*> *nom*

où tous les switchs (s'il y en a) sont groupés et précédés d'un unique tiret, après quoi un seul nom de fichier est obligatoirement donné. Ainsi on pourra exécuter « `mon_ls .` », ou « `mon_ls -lR /usr` », mais pas « `mon_ls -l -R /usr` », ni « `mon_ls /usr /home` ».

Attention, on veut accepter les combinaisons de switchs dans n'importe quel ordre : « `-lR` » est équivalent à « `-Rl` ».

Ce TP peut vous amener à utiliser les fonctions système suivantes : `opendir()`, `readdir()`, `closedir()`, `lstat()`. N'hésitez pas à consulter le manuel si nécessaire.

Lister un répertoire

Un répertoire Unix est une liste chaînée d'*entrées de répertoires* (structure `dirent`). Chaque entrée de répertoire contient le numéro d'inode et le nom d'un fichier du répertoire.

Pour ces trois premiers exercices, le programme doit afficher un message d'erreur si on l'appelle sur autre chose qu'un répertoire.

Exercice 1 : « `ls -a` »

Grâce à `opendir()`, `readdir()` et `closedir()`, programmer un clone de « `ls -a` » qui liste (tout) le contenu d'un répertoire donné en paramètre.

Exercice 2 : « `ls` »

Il se trouve que même les fichiers cachés sont affichés ! Désactiver leur affichage si l'option « `-a` » n'est pas donnée.

Exercice 3 : « `ls -R` »

Ajouter le switch « `-R` » à « `mon_ls` », de sorte qu'il s'appelle récursivement sur les sous-répertoires (en cas d'appel sur un répertoire bien sûr)

À chaque ajout d'un switch, les autres switchs doivent rester fonctionnels. Ainsi on doit pouvoir appeller `ls -aR` qui s'appellera récursivement dans les répertoires cachés, tandis que `ls -R` ne le fait pas.

Attributs d'un fichier

On se propose maintenant d'appliquer « `mon_ls` » à tout type de fichier. Grâce à `lstat()` (variante de `stat` qui, pour les liens symbolique, regarde l'inode du fichier lien et non celui du fichier pointé) on peut obtenir les attributs d'un fichier, qui seront affichés.

Exercice 4 : « `ls -ld` »

Écrire un programme qui affiche les attributs d'un fichier dont le nom est passé en paramètre, plus ou moins comme « `ls -ld` ». Le switch « `-d` » signifie que, si le fichier est un répertoire, on veut ses attributs à lui, et non ceux des fichiers qu'il contient.

Noter qu'il n'y a cette fois aucun appel à `opendir()`, juste un `lstat()` et des manipulations avant affichage.

Les perfectionnistes qui veulent obtenir le comportement exact de « `ls -ld` » doivent s'attendre à bien des surprises, telles que : comment afficher la date en français ? Ou le nom du propriétaire quand on n'a que son UID ?

Exercice 5 : « `ls -l` »

Faire en sorte que, quand on exécute « `mon_ls -l rep` », ou `rep` est un nom de répertoire, les noms et attributs de tous les *fichiers* qu'il contient soient affichés, et non ceux du répertoire lui-même.

Exercice 6 : taille totale

On veut afficher, pour « `mon_ls -l rep` », une première ligne précédant l'affichage des noms et indiquant la taille totale (en kilo-octets) des fichiers contenus (sans plonger dans les sous-répertoires), sous la forme `total xxxx`.

Répertoire courant

S'il reste du temps, implémenter l'exercice 3 du TD n° 2 :

Exercice : « `mon_pwd` »

Le but de cet exercice est d'écrire un ersatz de la commande « `pwd` » qui affiche le chemin du répertoire courant. Par souci de simplification, on supposera que l'arborescence n'est constituée que d'un seul disque logique (*device*).

1. Écrire une fonction `int est_racine(char *ref)` qui teste si `ref` est une référence de la racine de l'arborescence.
2. Écrire une fonction `char *nom_inoeud(int inoeud, char *rep)` qui retourne le (un) nom de l'inœud `inoeud` dans le répertoire `rep` s'il en existe, et `NULL` sinon.
3. Écrire une fonction `char *nom_dans_pere(char *ref)` qui retourne le nom du fichier `ref` dans son père.
4. Écrire une fonction `char *reference_absolue()` qui retourne la référence absolue du répertoire courant.

Note : la longueur d'une référence ne peut pas dépasser la constante `PATH_MAX`.