Programmation C avancée
TP 7

Non, sérieux WTF* !

L'objectif de ce TP est de produire un exécutable récupérant des informations sur les fichiers d'un système de type `Unix`. Ces exécutables devront être produits dans l'esprit des fonctionnalités `Unix` : bien documentées et relativement user-friendly.

La fonction `ftw` de la bibliothèque <`ftw.h`> permet de se promener dans l'arborescence des fichiers. `ftw` signifie en anglais *File Tree Walk* et permet de se promener dans l'arborescence des fichiers à partir d'un répertoire racine précisé `dirpath`. Se promener, c'est sympa mais récupérer des informations c'est mieux. Pour cela, à chaque nouvel objet rencontré (il faut comprendre par objet, un fichier, un répertoire, un lien symbolique, une socket, ...), `ftw` lance un postprocess (traitement postérieur) via l'appel à une sous fonction `fn` qui devra être programmée par vos soins. Le prototype de `ftw` est ainsi le suivant :

```
1 int ftw(const char *dirpath,
2         int (*fn) (const char *fpath,
3                    const struct stat *sb,
4                    int typeflag),
5         int nopenfd);
```

Pour comprendre le rôle de chaque intervenant, il vous sera nécessaire de lire des pages de documentation (il n'y a pas tant de choses à lire, et la quête d'information est le quotidien du développeur). Habituez-vous dès maintenant à utiliser des choses programmées par autrui, c'est le seul moyen de ne pas réinventer la roue tout le temps. Les pages à regarder seront `ftw` en section 3 du manuel et `stat` en section 2.

```
man 3 ftw
man 2 stat
```

Produire un exécutable `larger_file` qui prend en argument un nom de répertoire et affiche sur la sortie standard le chemin des 10 plus gros fichiers, dans l'ordre décroissant des tailles, contenus dans ce répertoire (ou un de ces sous-répertoires) ainsi que leurs tailles en octet.

Voici un exemple recherchant les 10 plus gros fichiers contenus dans un dossier dédié aux bibliothèques:

```
nborie@perceval:~> ./larger_file /usr/lib/
/usr/lib/libreoffice/program/libmergedlo.so : 51143680 octets
/usr/lib/thunderbird/libxul.so : 51100024 octets
/usr/lib/firefox/libxul.so : 48333968 octets
/usr/lib/nvidia-304/libnvidia-glcore.so.304.88 : 35454032 octets
/usr/lib/x86_64-linux-gnu/libQtWebKit.so.4.10.0 : 34590848 octets
/usr/lib/x86_64-linux-gnu/libQtWebKit.so.4.10 : 34590848 octets
/usr/lib/x86_64-linux-gnu/libQtWebKit.so.4 : 34590848 octets
```

```
/usr/lib/jvm/default-java/jre/lib/rt.jar : 32083857 octets
/usr/lib/chromium-browser/libs/libwebkit.so : 31256800 octets
/usr/lib/nvidia-304/libnvidia-compiler.so.304.88 : 27730224 octets
```

Commençant à devenir des adultes de la programmation, il n'est pas question que votre programme puisse produire une seule segfault. Il faut gérer tous les cas d'erreur et vous devez identifier tous les cas pathologiques : absence d'argument, argument invalide, moins de 10 fichiers présents, ...

* La WTF est bien sûr la fédération mondiale de Taekwondo : *the World Taekwondo Federation*