Institut Galilée — Systèmes d'exploitation
Année 2007-2008 — ISPG INFO 1

TD/TP 3

Entrées sorties, compléments

Exercice 1 (Cache et E/S)**.**
Considérer le code suivant :

```
...
fptr = fopen("monfichier","w") ;
fprintf(fptr, "Bonjour le monde") ;
fflush(fptr) ;
...
```

1. *Après l'appel à la fonction* `fprintf`*, combien de caches a traversé la chaîne de caractères "Bonjour le monde" avant d'être recopiée sur le disque ?*
2. *Immédiatement après l'appel à la fonction* `fflush`*, peut-on être sûr que la chaîne de cractères "Bonjour le monde" a été recopiée sur le disque ?*

Exercice 2 (E/S de haut niveau)**.**
Proposer votre propre implantation des fonctions `fopen()`*,* `fprintf()` *(simplifié),* `fflush()` *à partir des E/S de bas niveau.*

Exercice 3 (E/S de haut niveau : répertoires)**.**
Écrire un programme `list` *qui affiche les répertoires et les fichier du répertoire courant. L'option* `-r` *permettra d'afficher uniquement les répertoires et l'option* `-f` *uniquement les fichiers réguliers. Enfin l'option* `-R` *effecturera l'affichage de façon récursive.*

Voici des extraits utiles des pages de man.

```
#include <sys/types.h>
#include <dirent.h>
DIR *opendir (const char *name);
int closedir (DIR *dir);
struct dirent *readdir (DIR *dir);
```

La fonction `opendir()` ouvre un flux répertoire correspondant au répertoire `name`, et renvoie un pointeur sur ce flux. Le flux est positionné sur la première entrée du répertoire.

La fonction `opendir()` renvoie un pointeur sur le flux répertoire ou `NULL` si une erreur se produit.

La fonction `readdir()` renvoie un pointeur sur une structure dirent représentant l'entrée suivante du flux répertoire pointé par `dir`. Elle renvoie `NULL` a la fin du répertoire, ou en cas d'erreur.

Les données renvoyées par `readdir()` sont écrasées lors de l'appel suivant à `readdir()` sur le même flux répertoire.

D'après POSIX, la structure `dirent` contient un champ char `d_name[]` de taille non spécifiée, avec au plus `NAME_MAX` caractères avant le caractère nul final. L'utilisation des autres champs de cette structure compromet la portabilité de votre programme.

Manipulation 1 (Exercices à finir de la feuille précédente)**.**
Faire, ou finir les exercices sur `creat()` *(4.1),* `dup()` *(4.4) et* `lseek()` *(4.5) de la feuille 2. Attention il y a une erreur dans l'énoncé de l'exercice sur* `lseek()`*, il faut utiliser* `du` *(disk usage) et non* `df` *(inutile ici).*

Manipulation 2 (Des E/S de haut niveau aux E/S de bas niveau)**.**
Les entrées sorties de haut niveau fournies par la librairie C standard (`fopen`*,* `fprintf`*, etc.) utilisent le type* `FILE` *comme représentation des fichiers (un* flux*). Mais dans quel fichier d'en-tête est défini ce type ?*

Ces E/S de haut niveau font appel aux E/S de bas niveau c'est à dire aux appels systèmes `open`*,* `write`*, etc. (POSIX.1). Pour cela les E/S de haut niveau doivent manipuler un descripteur de fichier pour chaque fichier ouvert. Voyez-vous comment accéder à ce descripteur de fichier à partir de la variable de type* `FILE *` *renvoyée par un* `fopen()` *? (Cela dépend du système.)*

Écrire un programme qui ouvre un fichier en écriture avec `fopen()` *et y écrit une chaîne à l'aide d'un* `write()` *sur le descripteur de fichier correspondant.*

Manipulation 3 (Des E/S de bas niveau aux E/S de haut niveau)**.**
La fonction `fdopen()` *associe un flux (type* `FILE *`*, E/S haut niveau) à un descripteur de fichier (E/S bas niveau).*

Écrire un programme qui ouvre un fichier en écriture à l'aide de `open()` *et écrit dans ce fichier à l'aide de* `fprintf()`*.*

Manipulation 4 (`atexit()`)**.**
La fonction `int atexit(void (*func)(void))` *de la biblothèque C standard inscrit la fonction* `func` *donnée en argument comme devant être automatiquement invoquée lors d'un appel à* `exit()`*.*

1. *Tester cette fonction par un exemple simple.*
2. *Comment peut-on inscrire plusieurs fonctions à invoquer au moment du* `exit()` *? Dans quel ordre s'exécutent-elles ? (Tester)*
3. *Comment peut-on terminer le processus correctement, sans invoquer les fonctions ainsi inscrites ? (Tester)*
4. *Que se passe t-il si* `func` *fait à son tour appel à* `exit()` *?*