Université Paul Sabatier **Licence Informatique – L3**

31062 Toulouse Cedex 9

Programmation multitâche

Travaux dirigés sur machines

TP n°2 – Processus, fichiers et tubes

1 – open(), read(), write(), close() – Compilation séparée

Encapsuler (.h, .c ; fichier objet) la fonction

```
int afficher(char *nomFichier)
```

écrite en TD qui réalise l'affichage à l'écran (sur le fichier de descripteur 1) du contenu d'un fichier de nom donné en paramètre.

2 – fork(), dup() / dup2()

Utiliser cette fonction (sans la modifier) afin d'écrire la commande `moncat` qui affiche à l'écran de l'utilisateur le contenu d'une liste de fichiers, mais qui peut prendre en compte une éventuelle redirection vers un fichier autre que l'écran.

Exemple

```
moncat fichier1 fichier2 fichier3 ">" fichier_destination
```

Attention : Pour tester la commande, on tapera le signe de redirection encadré par des guillemets.
Voir le manuel de `string.h` pour les comparaisons de chaines.

3 – fork(), open(), write(), exec()

Écrire la commande `compiler` définie de la manière suivante :

```
compiler fichier1.c fichier2.c fichier3.c  ... fichiern.c
```

Sémantique :

La commande `compiler` permet de réaliser, en parallèle, la compilation des fichiers spécifiés. La compilation d'un fichier `fichieri.c` devra générer un fichier objet de nom `fichieri.o` et placera les erreurs dans un fichier de nom `fichieri.err`.

Dans le cas où la compilation de tous ces fichiers est correcte, l'édition de liens doit être immédiatement effectuée par le processus père (à partir de tous les fichiers objets générés) afin de créer le fichier exécutable `a.out` (on jugera inutile de le renommer).

Dans le cas où la compilation d'un fichier `fichieri.c` est erronée, le père affichera, sur la sortie standard, le nom du fichier source C, ainsi que le nom du fichier où sont placées les erreurs de compilation relatives à ce fichier.

Exemple d'exécution :

```
compiler moncat.c afficher.c
```

Cas 1 : on suppose que des erreurs de compilation se trouvent dans afficher.c, l'exécution de la compilation. Exemple l'exécution en parallèle des commandes :

```
cc –c moncat.c
cc –c afficher.c
```

mènera à l'affichage suivant :

```
Des erreurs de compilation ont été détectées dans le fichier afficher.c, les erreurs se trouvent dans afficher.err.
```

Cas 2 : Ces erreurs ont été corrigées, la nouvelle compilation a pu se dérouler normalement et être suivie de l'édition de liens i.e. que la commande :

```
cc moncat.o afficher.o
```

a été exécutée pour générer le fichier exécutable `a.out`.

Remarque :

Afin d'obtenir facilement le nom d'un fichier dérivé à partir du nom d'un fichier donné et d'une extension, écrire une fonction `ChangerExtension` telle que : `char *ChangerExtension (char *NomFichier, char *Extension);` qui retourne le nom créé en remplaçant l'extension de `NomFichier` par l'extension `Extension` (voir *Rappel* en fin d'énoncé). Par exemple, `ChangerExtension("monFichier.c", "o")` retournera "`monFichier.o`", `ChangerExtension("monFichier.c", "err")` retournera "`monFichier.err`".

Écrire le programme correspondant à cette commande de manière **incrémentale** :

- Écrire et tester la fonction `ChangerExtension` ;
- Réaliser la compilation des fichiers paramètres de `compiler` en parallèle ;
- Modifier le programme pour prendre en compte les erreurs éventuelles de compilation et l'affichage des résultats par le père ;
- Modifier le programme pour réaliser l'édition de liens.

3 – fork(), exec()

Tester la ligne de commande suivante : `ps –ef | grep utilisateur.`

Écrire la commande `monps` réalisant un travail équivalent et paramétrable par le nom d'utilisateur recherché.

Questions optionnelles ouvertes

1 – Compléter l'application `monsh` réalisant un shell afin d'utiliser des redirections telles que « < », « > » et « >> ».

2 – Écrire une commande permettant de crypter un fichier ou un répertoire. On procédera de manière incrémentale :

☞ Écrire une fonction permettant de crypter un caractère (par exemple, en remplaçant un caractère par le caractère le suivant dans l'alphabet) ;

☞ Écrire une fonction permettant de crypter un fichier complet ;

☞ Écrire une fonction permettant de crypter tous les fichiers d'un répertoire.

Les options suivantes seront possibles :

- Cryptage et décryptage ;
- Utilisation d'un mot de passe lors du cryptage ; le même mot de passe doit être fourni lors du décryptage ;
- Le cryptage d'un fichier ou d'un répertoire crée un seul fichier dont le nom pourra être précisé dans la commande ; dans le cas contraire, il sera choisi de manière standard ;
- Les noms de fichiers originaux doivent être restitués lors du décryptage.

Des options supplémentaires pourront être envisagées.

Rappel

Des fonctions de traitement de chaînes de caractères sont disponibles dans `<string.h>` telles que `strcpy` (recopie de chaîne), `strncpy` (recopie sur n caractères), `strcmp` (comparaison de chaînes), `strcat`… Consulter le manuel en ligne (ou le support de cours C) pour plus d'informations à propos de ces fonctions.