USTHB **Année 2024/2025**
Faculté d'Informatique **SYS02**
Département SIQ **LIC. ISIL B**

TP N°2
(Manipulation de Fichiers sous Linux : Partie II)

L'objectif de ce TP est la manipulation de fichiers et répertoires à l'aide des appels systèmes et des fonctions de la bibliothèque C.

Placez-vous dans le répertoire TP2.

Exercice 1

1. Avec un éditeur de texte (Ex. gedit), créez un fichier **`creat.c`** contenant le programme C suivant :

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {

    int fd;
    if (argc != 2) {
            printf("Usage: %s <File name#>\n", argv[0]);
            exit(1);
    }

    mode_t mode = S_IRUSR | S_IWUSR | S_IXUSR | S_IRGRP | S_IWGRP
                  | S_IROTH | S_IWOTH;

   fd = open(argv[1], O_WRONLY | O_EXCL | O_CREAT, mode);

   if (fd == -1) {
           perror("open");
           exit(-1);
   }
   return EXIT_SUCCESS;
}
```

2. Interprétez les différentes instructions de ce code. Déduisez le rôle de ce programme.
3. Créez l'exécutable creat et exécutez-le en donnant comme argument en entrée **fich1.txt**.
4. Refaites l'exécution de ce programme en donnant encore une fois **fich1.txt** comme argument. Que remarquez-vous ?
5. Quelle est la valeur du masque par défaut des droits d'accès ?
6. Listez les droits d'accès du fichier **fich1.txt**.
7. Comparez ces droits à ceux de la variable mode du programme **`creat.c`**. Quelle relation existe-t-elle entre eux ?

Exercice 2

1. Quel est le rôle de l'appel système **`stat()`** ?
2. Avec un éditeur de texte (Ex. gedit), créez un fichier **`attributsInode.c`** contenant le programme C suivant :

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <pwd.h>

int main(int argc, char *argv[]) {

    struct stat StatusBuffer;
    if (stat(argv[1], &StatusBuffer) == -1) {
          perror(argv[1]);
          exit(1);
    }
    printf("Nom de fichier : %s\n",argv[1]);
    printf("Droits d'accès (mode octal) : %o\n",StatusBuffer.st_mode
&0777);

    if (S_ISREG(StatusBuffer.st_mode)) {
            printf("%s est un fichier régulier\n", argv[1]);
    } else if (S_ISDIR(StatusBuffer.st_mode)) {
                  printf("%s est un répertoire\n", argv[1]);
    };

    printf ("Les droits d'accès de USER en mode symbolique sont :");
    printf ("%c", StatusBuffer.st_mode & S_IRUSR ? 'r' : '-');
    printf ("%c", StatusBuffer.st_mode & S_IWUSR ? 'w' : '-');
    printf ("%c\n", StatusBuffer.st_mode & S_IXUSR ? 'x' : '-');

    printf("Owner:  %s\n",getpwuid(StatusBuffer.st_uid)->pw_name);

    printf("Taille: %ld\n",StatusBuffer.st_size);

    return EXIT_SUCCESS;
}
```

3. Interprétez les différentes instructions de ce code. Déduisez le rôle de ce programme.
4. Créez l'exécutable **`attributsInode`** et exécutez-le.
5. Modifiez ce programme afin de simuler l'exécution de la commande **`ls -l`** sur le fichier passé en paramètre.

Exercice 3

Ecrivez un programme C qui copie le contenu d'un fichier source vers un fichier destination, en simulant la commande **`cp`**.

Exercice 4

1. Ecrivez un programme C, que vous nommez **`listdir`**, qui affiche le contenu d'un répertoire passé comme paramètre. Si aucun répertoire n'est spécifié, le programme affichera le contenu du répertoire courant.

   Utilisez les fonctions **`opendir`**, **`readdir`** et **`closedir`**, de la bibliothèque C, pour parcourir le contenu d'un répertoire.

2. Modifiez le programme précèdent pour qu'il accepte l'utilisation des deux options suivantes :
   - l'option **`-r`** qui permet d'afficher uniquement les répertoires, et
   - l'option **`-f`** qui permet d'afficher uniquement les fichiers réguliers.

     Utilisez la fonction **`getopt`** de la bibliothèque C (Consultez **man 3 getopt**), afin de récupérer les options de la ligne de commande.