Le système de gestion des fichiers

Exercice 1. Écrire un programme `identique` qui prend deux chemins en arguments et examine si les deux chemins référencent le même fichier. Que se passe-t'il si un fichier est un lien symbolique ?

Exercice 2. Trouver une méthode pour donner le droit de lecture et d'écriture uniquement à la personne de votre choix.

Exercice 3. Écrire une fonction `int autorisation(char *chemin, int mode)` qui renvoie 1 si l'accès à chemin est possible dans le mode donné. On définira les constantes `READ`, `WRITE` et `EXEC` pour désigner le mode d'accès. Indication : on pourra utiliser les fonctions `int getuid(void)` et `int getgid(void)` pour déterminer le propriétaire et le groupe réel du processus.

Exercice 4.

```
 1  #include <stdio.h>
 2  #include <dirent.h>
 3  #include <assert.h>
 4  #include <unistd.h>
 5  #include <sys/types.h>
 6  #include <sys/stat.h>
 7  #include <string.h>

 9  #define MAX_CHARS 100

11  int
12  main(int argc, char *argv[])
13  {
14    DIR *dir;
15    struct dirent *lien;
16    char chemin_fic[MAX_CHARS];
17    struct stat buf;

19    assert(argc == 2);
20    assert(strlen(argv[1]) < MAX_CHARS);
21    strcpy(chemin_fic, argv[1]);
22    chemin_fic[strlen(chemin_fic)] = '/';

24    dir = opendir(argv[1]); assert(dir != NULL);
25    while((lien = readdir(dir)) != NULL)
26      {
27        assert(
28               (strlen(argv[1])+strlen(lien->d_name)+2)
29               <= MAX_CHARS
30               );
31        strcpy(chemin_fic+strlen(argv[1])+1, lien->d_name);
32        assert(stat(chemin_fic,&buf) == 0);
33        printf("%s ", chemin_fic);
34        if(S_ISREG(buf.st_mode))
35          printf("(fichier regulier)  %ld octets\n",buf.st_size);
36        else
37          printf("(autre)\n");
38      }

40    closedir(dir);
41    return 0;
42  }
```

Que se passe-t'il pendant l'exécution du programme précèdent ?

Exercice 5. Le fichier `toto.txt` contient deux lignes :

```
abcd
efgh
```

Que se passe-t'il pendant l'exécution du programme suivant :

```
 1  #include <stdlib.h>
 2  #include <stdio.h>
 3  #include <unistd.h>
 4  #include <fcntl.h>

 6  char tampon[6];

 8  int main(int argc, char *argv[])
 9  {
10    int fd ;
11    pid_t pid;

13    if((fd = open("toto.txt",O_RDONLY)) < 0 ) return EXIT_FAILURE;
14    pid=fork(); sleep(1);

16    switch(pid)
17      {
18      case -1:
19        return EXIT_FAILURE;
20      case 0 :
21        read(fd,tampon,5);
22        break;
23      default:
24        read(fd,tampon,5);
25      }
26    sleep(1); printf("%s",tampon); close(fd);

28    return EXIT_SUCCESS;
29  }
```

Que se passe-t'il pendant l'exécution si on échange les lignes 13 et 14 ?

Exercice 6.
– Il n'est pas possible de ouvrir un répertoire en écriture. Pourquoi ?
– Il n'est pas possible de créer un lien sur un répertoire. Pourquoi ?

Exercice 7. Lorsque l'on crée un lien supplémentaire sur un fichier `fic.txt`, la date du dernier changement du i-node est changée. Pourquoi ?

Exercice 8. Écrire un programme `list` qui affiche les répertoires et les fichier du répertoire courant. De plus, l'option `-r` permet d'afficher uniquement les répertoires et l'option `-f` uniquement les fichiers réguliers. On ajoute aussi l'option `-R` qui effectue l'affichage de façon récursive.