Sistemas Operativos

Formulario de auto-evaluación

Molulo 2. Sesión 2. Llamadas al sistema para el S.Archivos Parte II

Nombre y apellidos:

David Sánchez Jiménez

a) Cuestionario de actitud frente al trabajo.

El tiempo que he dedicado a la preparación de la sesión antes de asistir al laboratorio ha sido de …... minutos.

1. He resuelto todas las dudas que tenía antes de iniciar la sesión de prácticas: …... (si/no). En caso de haber contestado “no”, indica los motivos por los que no las has resuelto:

2. Tengo que trabajar algo más los conceptos sobre:

3. Comentarios y sugerencias:

b) Cuestionario de conocimientos adquiridos.

Mi solución al **ejercicio 1** ha sido:

El programa crea un archivo1 con permisos de lectura, escritura y ejecución para grupo, pone la máscara a 0 con el comando umask(0) y crea un archivo2 con idénticos permisos.

A continuación comprueba que tiene acceso a los atributos de archivo1 y cambia sus permisos con chmod quitando el permiso de ejecución del grupo y cambiando el GID del propietario al del proceso que ejecuta el archivo. También se utiliza chmod para cambiar los permisos del archivo2 asignandole todos los permisos para el usuario, lectura y escritura para grupo y lectura para otros.

Si lanzamos el comando ls -l archivo* observamos los permisos finales que tienen ambos archivos.

```
$ ls -l archivo*
----r-S--- 1 david users 0 oct  4 10:16 archivo1
-rwxrw-r-- 1 david users 0 oct  4 10:16 archivo2
```

Mi solución a la **ejercicio 2** ha sido:

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

int main(int argc, char *argv[]) {
  struct stat atributos;

  struct dirent *fichero;
  DIR *dir;

  int chmod_result, permisos_antiguos, permisos_nuevos;

  dir = opendir(argv[1]);
  if (dir == NULL) {
    printf("Error: No se puede abrir el directorio: %s\n", argv[1]);
    exit(2);
  }

  while ((fichero = readdir(dir)) != NULL) {
    chdir(argv[1]);
    if (stat(fichero->d_name, &atributos) < 0) {
      printf("Error al intentar acceder a los atributos de %s\n",
            fichero->d_name);

    } else {
      if (!S_ISDIR(atributos.st_mode)) {
        permisos_nuevos = strtol(argv[2], 0, 8) & 0777;
        permisos_antiguos = atributos.st_mode & 0777;
        chmod_result = chmod(fichero->d_name, permisos_nuevos);
        if (chmod_result < 0) {
          printf("%s : \t%d \t%o\n", fichero->d_name, chmod_result,
                permisos_antiguos);
          exit(-1);
        } else {
          printf("%s : \t%o \t%o\n", fichero->d_name, permisos_antiguos,
                permisos_nuevos);
        }
      }
    }
  }

  closedir(dir);

  return 0;
}
```

Mi solución a la **ejercicio 3** ha sido:

```
#include <dirent.h>
#include <errno.h>
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>

#define mymask(mode) ((mode) & ~S_IFMT)
#define S_IFXGRPOTH 011
#define regla1(mode) ((((mode) & ~S_IFMT) & 011) == S_IFXGRPOTH)

void buscar_dir(DIR *direct, char pathname[], int *reg, int *tamanio) {
  struct stat atributos;
  struct dirent *ed;
  DIR *direct_act;
  char cadena[500];

  while ((ed = readdir(direct)) != NULL) {
    if (strcmp(ed->d_name, ".") != 0 && strcmp(ed->d_name, "..") != 0) {
      sprintf(cadena, "%s/%s", pathname, ed->d_name);
      if (stat(cadena, &atributos) < 0) {
        printf("\nError al intentar acceder a los atributos de archivo");
        perror("\nError en lstat");
        exit(-1);
      }
      if (S_ISDIR(atributos.st_mode)) {
        if ((direct_act = opendir(cadena)) == NULL) {
          printf("\nError al abrir el directorio: [%s]\n", cadena);
        } else {
          buscar_dir(direct_act, cadena, reg, tamanio);
        }
      } else {
        printf("%s %ld \n", cadena, atributos.st_ino);
        if (S_ISREG(atributos.st_mode)) {
          if (regla1(atributos.st_mode)) {
            (*reg)++;
            (*tamanio) += (int)atributos.st_size;
          }
        }
      }
    }
  }
  closedir(direct);
}
```

```
int main(int argc, char *argv[]) {
  DIR *direct;
  char pathname[500];
  int reg = 0, tamanio = 0;

  if (argc == 2) {
    strcpy(pathname, argv[1]);
  } else {
    strcpy(pathname, ".");
  }
  if ((direct = opendir(pathname)) == NULL) {
    printf("\nError al abrir directorio\n");
    exit(-1);
  }

  printf("Los inodos son: \n");
  buscar_dir(direct, pathname, &reg, &tamanio);
  printf("Hay %d archivos regulares con permiso x para grupo y otros\n", reg);
  printf("El tamaño total ocupado por dichos archivos es %d bytes\n", tamanio);
  return 0;
}
```