

Formulario de auto-evaluación

Modulo 2. Sesión 1. Llamadas al sistema para el S.Archivos. Parte I

Nombre y apellidos:

Juan Carlos Ruiz García

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:

Si, estube atento a la explicación de clase y solvente todas mis dudas antes de comenzar con la práctica.

2. Tengo que trabajar algo más los conceptos sobre:

Lo entiendo todo por el momento.

3. Comentarios y sugerencias:

Ninguna

b) Cuestionario de conocimientos adquiridos.

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

```
juanka1995@juanka-laptop ~/practicass/so/practical_II $ gcc tareal.c -o tareal
juanka1995@juanka-laptop ~/practicass/so/practical_II $ ./tareal
juanka1995@juanka-laptop ~/practicass/so/practical_II $ ls -l
total 20
-rw----- 1 juanka1995 juanka1995  50 Oct 31 18:13 archivo
-rwxr-xr-x 1 juanka1995 juanka1995 9056 Oct 31 18:13 tareal
-rw-r--r-- 1 juanka1995 juanka1995  859 Oct 31 18:07 tareal.c
juanka1995@juanka-laptop ~/practicass/so/practical_II $ cat archivo && echo
abcdefghijklABCDEFGHIJ
juanka1995@juanka-laptop ~/practicass/so/practical_II $ od -c archivo
00000000  a  b  c  d  e  f  g  h  i  j  \0  \0  \0  \0  \0  \0
00000020  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0
00000040  \0  \0  \0  \0  \0  \0  \0  \0  A  B  C  D  E  F  G  H
00000060  I  J
00000062
```

La finalidad de este programa es abrir un fichero llamado archivo, el cuál en caso de no existir se creará con permisos de escritura y lectura para el usuario creador del dicho fichero. Una vez abierto/creado se introducirán dos cadenas de caracteres en este fichero, la primera contendrá "abcdefghijkl" que al ser introducida dejará la *posición de lectura actual* en la 10ª posición. Después de esto mediante la orden *lseek* modificará la *posición de lectura actual* a la posición 40, es decir 30 posiciones más a la que había quedado anteriormente. Y por último escribirá la segunda cadena en dicha posición.

Esto a la hora de mostrar un fichero con la orden *cat* no lo apreciaríamos, pero utilizando la orden *od -c* podemos darnos cuenta que efectivamente existen 30 posiciones de diferencia entre una cadena de caracteres y otra.

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

```
/*
tareal.c
Trabajo con llamadas al sistema del Sistema de Archivos 'POSIX 2.10 compliant'
Probad tras la ejecución del programa: $>cat archivo y $> od -c archivo
*/

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

int main(int argc, char *argv[]){
    char caracter[1],texto_bloque[20],num_bloques[30];
    int contador = 1, fentrada, fsalida, end_file, current_pos;
```

```
if(argc == 2){ //Si hay dos argumentos coje el segundo como el fichero con la
informacion
    fentrada = open(argv[1], O_RDONLY);
}
else{ // Si no coje la entrada estandar
    printf("Escriba el texto deseado: \n");
    fentrada = STDIN_FILENO;
}

if(fentrada < 0){ //En caso de que no se abra bien el fichero dará error
    printf("Error al abrir el fichero de entrada.\n");
    exit(-1);
}

if( (fsalida = open("salida.txt", O_CREAT|O_TRUNC|O_WRONLY, S_IRUSR|S_IWUSR)) < 0) {
//Crea el fichero salida.txt o lo abre si no existe
    printf("Error al abrir el fichero de salida.\n");
    exit(-1);
}

while( (end_file = read(fentrada, caracter, 1)) != 0 ){ //Read devolvera 0 cuando
llegue al final del fichero
    current_pos = lseek(fentrada, 0, SEEK_CUR); // Calculo la posicion actual del
puntero
    if(current_pos == 1 || current_pos % 80 == 0){
        if(current_pos == 1){ //Escribo el total de bloques MAL, que posteriormente se
cambiara
            sprintf(num_bloques, "El numero de bloques es <%d>\n", contador);
            if(write(fs salida, num_bloques, strlen(num_bloques)) != strlen(num_bloques)){
                printf("Error escribir nº de bloques en el fichero de salida.\n");
                exit(-1);
            }
        }
        sprintf(texto_bloque, "\nBloque %d\n", contador); //Escribo el numero de bloque
        if(write(fs salida, texto_bloque, strlen(texto_bloque)) != strlen(texto_bloque)){
            printf("Error escribir bloque en el fichero de salida.\n");
            exit(-1);
        }
        contador++;
    }
    if(write(fs salida, caracter, 1) != 1){ //Escribo caracter a caracter hasta llegar a
```

```

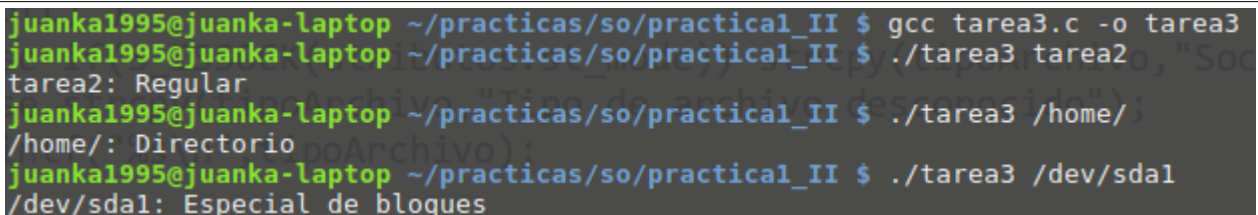
modulo 80
    printf("Error escribir caracter en el fichero de salida.\n");
    exit(-1);
}
}

    sprintf(num_bloques, "El numero de bloques es <%d>\n", --contador); //Calculo el
num_bloques correcto
    lseek(fsalida, 0, SEEK_SET); //Vuelvo al inicio del fichero de salida
    if(write(fsalida, num_bloques, strlen(num_bloques)) != strlen(num_bloques)){ //Escribo
el total de bloques correcto
        printf("Error escribir nº de bloques en el fichero de salida.\n");
        exit(-1);
    }

    close(fentrada); //Cierro ambos ficheros
    close(fsalida);
    return 0;
}

```

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



```

juanka1995@juanka-laptop ~/practicasso/practical_II $ gcc tarea3.c -o tarea3
juanka1995@juanka-laptop ~/practicasso/practical_II $ ./tarea3 tarea2, "Soc
tarea2: Regular
juanka1995@juanka-laptop ~/practicasso/practical_II $ ./tarea3 /home/
/home/: Directorio
juanka1995@juanka-laptop ~/practicasso/practical_II $ ./tarea3 /dev/sda1
/dev/sda1: Especial de bloques

```

Ese ejercicio recibe mediante la sintaxis `./tarea3 <nombre-archivo>` te muestra que tipo de archivo es el introducido. En la captura anterior podemos ver varios ejemplos del uso de este programa.

Internamente esto funciona utilizando los flags, cuando el flag correspondiente al archivo que hemos introducido se activa nos muestra por el tipo.

Mi solución a la **ejercicio 4** ha sido:

```
#define S_ISREG2(mode) (mode & S_IFMT == S_IFREG)
```

Anteriormente le he mostrado como se declararía una macro y ahora le muestro un breve ejemplo de programa para usarla.

```

/*
tarea4.c
Trabajo con llamadas al sistema del Sistema de Archivos 'POSIX 2.10 compliant'
*/

```

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

#define S_ISREG2(mode) ((mode & S_IFMT) == S_IFREG)

int main(int argc, char *argv[]){
    int i;
    struct stat atributos;
    char tipoArchivo[30];
    if(argc<2) {
        printf("\nSintaxis de ejecucion: tarea4 [<nombre_archivo>]+\n\n");
        exit(-1);
    }
    if(lstat(argv[1],&atributos) < 0) {
        printf("\nError al intentar acceder a los atributos de %s",argv[1]);
        perror("\nError en lstat");
        exit(-1);
    }
    if(S_ISREG2(atributos.st_mode))
        strcpy(tipoArchivo, "Regular");
    else
        strcpy(tipoArchivo, "No Regular");

    printf("El archivo %s es de tipo: %s\n",argv[1],tipoArchivo);

    return 0;
}
```