Tema 7: Ficheros

1 Introducción.

Un fichero es un elemento de almacenamiento de datos sobre un medio permanente, como pueden ser discos duros, cd-rom, lápices USB, etc.

Podemos distinguir entre dos tipos de archivos:

-Archivos binarios
-Archivos de texto (ASCII).

El tipo de datos que nos permite implementar un archivo en C es **FILE.** Por tanto, de ahora en adelante, un fichero será para nosotros una variable de tipo FILE.

Las operaciones básicas que podemos realizar sobre un fichero son la apertura, el cierre y la comprobación del final del mismo. Dichas operaciones son comunes tanto a ficheros de texto como a ficheros binarios. En lo que difieren estos tipos de ficheros es a la hora de realizar las operaciones de lectura y escritura.

2 Operaciones básicas con ficheros

2.1 Apertura

Sintaxis:

```
FILE* fopen(char *nombre, char *modo)
```

La función devuelve un puntero a una variable de tipo fichero (FILE) que simboliza el fichero creado. Si dicho puntero es vacío (NULL) significa que el fichero no ha podido ser abierto convenientemente.

Modo de apertura:

r Sólo lectura.
w Sólo escritura. Si el archivo no existe se crea. Si existe se sobrescribe.
a Añadir. Abre para escritura al final del archivo o lo crea si no existe.
r+ Abre un archivo existente para lectura o escritura.

w+ Crea un archivo para lectura o escritura. Si existe se sobreescribe.
a+ Abre un archivo para lectura o escritura al final del mismo.

Estos modos se combinan con el tipo de archivo:

b Archivo binario
t Archivo de texto

Por ejemplo, con el modo "w+b" creamos un archivo binario para lectura o escritura y si existe se sobrescribe.

Veamos un ejemplo de apertura de un archivo:

```
FILE *archivo;
archivo = fopen("c:\\prueba.txt", "at");
if (archivo == NULL){
        printf("Error el archivo no se puede abrir o crear \n");
}else{
        printf("Archivo creado correctamente\n");
}
```

Como podemos observar en el ejemplo, para especificar el carácter '\' en la ruta del archivo, se hace escribiendo '\\', debido a que se trata de un carácter especial. Por ejemplo, para especificar la ruta del documento data.txt que se encuentra en c:\tmp\docum, tendríamos que escribir:

```
"c:\\tmp\\docum\\data.txt"
```

2.2 Cierre

Sintaxis:

```
int fclose(FILE *fich);
```

Esta función devuelve 0 si se cierra correctamente o la constante EOF (End of File) si el cierre falla. El argumento que recibe es un puntero al fichero a cerrar (tipo FILE).

Veamos un ejemplo de utilización de esta función:

```
FILE *archivo;
….
if (fclose(file)==EOF)
     printf("Error el archivo no se puede cerrar\n");
```

2.3 Comprobación de final de fichero

Sintaxis:

```
int feof(FILE * fich);
```

Devuelve:

0 si no se ha llegado al final del archivo
Distinto de cero si hemos llegado al EOF.

Argumentos:

**fich* puntero de tipo FILE, con el archivo a comprobar

Veamos un ejemplo de funcionamiento:*:*

```
FILE *archivo;
…
if (feof(file)!=0)
    printf("Fin de archivo alcanzado\n");
```

3 Ficheros binarios

Son archivos cuyo contenido son caracteres en formato binario, por tanto NO son legibles ni editables. Para asegurarnos que la apertura de un archivo se hace en modo binario se debe emplear el modificador "b" en el modo de apertura del archivo. Si no ponemos nada se asume el modo binario por defecto.

Las funciones básicas de escritura y lectura asociadas a este modo son: *fwrite* y *fread*.

3.1 Escritura

Sintaxis:

```
int  fwrite(void *datos, int tam, int ndatos, FILE *fich);
```

Devuelve:

Número de elementos (no bytes) escritos en el archivo.

Parámetros:

**ptr* = puntero al origen de los datos.
tam = tamaño de cada elemento.
ndatos = número de elementos.
**fich* = puntero a FILE (archivo donde escribir).

Ejemplo**:**

```
FILE *archivo;
int valor;
Tficha ficha;
int n;
…
archivo=fopen("c:\\archivo.dat","w");
n = fwrite(&valor, sizeof(int), 1, archivo);
```

```
if (n!=1)  printf("Error: escritura incompleta.");
n = fwrite(&ficha, sizeof(TFicha), 1, archivo);
if (n!=1)  printf("Error: escritura incompleta.");
```

3.2 Lectura

Sintaxis:

```
int  fread(void *datos, int tam, int ndatos, FILE *fich);
```

Devuelve el número de elementos (no bytes) leídos del archivo.

Parámetros:

**ptr* = puntero al destino de los datos.
tam = tamaño de cada elemento.
ndatos = número de elementos.
**fich* = puntero a FILE (archivo de donde leer).

Ejemplo:

```
FILE *archivo;
int valor;
TFicha ficha;
int n;
…
archivo=fopen("c:\\archivo.dat","r");
n = fread(&valor, sizeof(int), 1, archivo);
if (n!=1)  printf("Error: lectura incompleta.");
n = fread(&ficha, sizeof(TFicha), 1, archivo);
if (n!=1)  printf("Error: lectura incompleta.");
```

Veamos un ejemplo de lectura de una agenda desde un archivo.

```
TFicha      fichas[100];
int         nfichas;
FILE        *agenda;
nfichas =0;
agenda = fopen("agenda.dat", "rb");
while (!feof(agenda)) {
     fread(&fichas[nfichas], sizeof(TFicha), 1, agenda);
     nfichas ++;
}
fclose(agenda);
```

De igual forma, veamos cómo realizar la escritura:

```
TFicha fichas[100];
int   nfichas;
FILE *agenda;
…
agenda = fopen("agenda.dat", "wb");
```

```
fwrite(fichas, sizeof(TFicha), nfichas, agenda);
fclose(agenda);
```

También se podría haber hecho de la forma:

```
…
for(n=0;n<nfichas;n++)
    fwrite(&fichas[n],sizeof(TFicha),1,agenda);
```

Veamos ahora un ejemplo para leer una imagen desde archivo:

```
int   ancho, alto,i,j;
FILE *fimagen;
unsigned char Mimagen[200][200];
fimagen = fopen("foto.img", "rb");
fread(&alto, sizeof(int), 1, fimagen);
fread(&ancho, sizeof(int), 1, fimagen);
for (i=0; i<alto; i++) {
    for (j=0; i<ancho; j++) {
            fread(&Mimagen[i][j], sizeof(unsigned char), 1, fimagen);
    }
}
fclose(fimagen);
```

4 Ficheros de texto

Se trata de archivos cuyo contenido son caracteres en formato ASCII. Por tanto son legibles y editables por cualquier editor de texto. Usualmente tienen la extensión "txt".

Para abrir un archivo en modo texto se debe emplear el modificador "t" en el modo de apertura del archivo.

Existen dos formas de tratar ficheros de texto:
-Carácter a carácter (*fgetc* y *fputc*)
-Con formato (*fprintf* y *fscanf*)

4.1 Funciones de manejo carácter a carácter

Lectura:

```
char fgetc(FILE *fich)
```

Lee un carácter del archivo (EOF si estamos al final del mismo)

Escritura:

```
char fputc(char c, FILE *fich)
```

Escribe el carácter c en el archivo. Si es correcto devuelve el mismo carácter y si no EOF.

4.2 Funciones de manejo con formato

Lectura:

```
int fscanf(FILE *fich, char *formato, arg1,arg2,…argN)
```

Su uso es el mismo que ***scanf*** pero con la salvedad de que lee desde un archivo, en vez desde el teclado.

Escritura:

```
int fprintf(FILE *fich, char *formato, arg1,arg2,…argN)
```

Su uso es igual al de ***printf*** pero con la salvedad de que escribe en un archivo en vez de en la pantalla.

Veamos un ejemplo para grabar una cadena de caracteres en un fichero:

```
char        cadena[255];
FILE        *salida;
…
salida = fopen(“salida.txt”, “wt”);
strcpy(cadena, “Prueba de escritura”);
fprintf(salida,”%s”,cadena);
fclose(salida);
```

Veamos un nuevo ejemplo que realiza la lectura de un archivo carácter a carácter empleando un buffer:

```
char buffer[255], lineas[100][80], c;
int   n, t;
FILE *entrada;

entrada = fopen(“ejemplo.txt”, “rt”);
while (!feof(entrada)) {
      t = 0;
      do {
            c = fgetc(entrada);
            buffer[t] = c;
            t++;
      } while (c!=EOF && c!=’\n’); //fin de fichero o de línea
      strcpy(lineas[n], buffer);
      n++;
}
fclose(salida);
```