Archivos en C

Ya hemos visto el manejo de registros a partir de los arrays de struct. Ahora, utilizaremos las funciones de manejo de archivos de C para hacer que los datos que hasta ahora son almacenados en memoria sean almacenados en el disco o en la unidad de almacenamiento que deseemos.

Abrir un archivo

fopen

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

Abre un archivo cuyo nombre es indicado por el contenido de la cadena apuntada por *nombre*, el segundo argumento es una cadena que indica el modo de apertura del stream que devolverá.

Parámetros:
nombre – la ruta del archivo, puede ser relativa o absoluta si el sistema lo permite.
modo – el modo de apertura del archivo:

“rb”	Lectura solamente. El archivo debe existir.
“wb”	Crea un archivo para escribir o lo trunca a longitud cero.
“ab”	Añade, abre o crea un archivo para escribir al final del archivo.
“rb+”	Abre un archivo para leer o escribir.

Tabla 1: Modos de apertura para archivos binarios.

Valor de retorno:
La función fopen retorna un puntero al objeto que controla el stream. Si el proceso de apertura no es exitoso, entonces retorna un caracter núlo.

Ejemplo:

```
int existeArchivo(char *strArchivo){
FILE *p;
p = fopen(strArchivo, "rb");
if (p){
fclose(p);
return 1;
}
else
return 0;
}
```

Cerrar un archivo

fclose

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

El stream apuntado por *stream* es liberado y el fichero asociado es cerrado.
Parámetros:
stream – Puntero a un objeto FILE que especifica el stream a cerrar.

Valor de retorno:
La función fclose retorna cero si el stream es cerrado exitosamente, si ocurren errores devuelve EOF.

Leer

fread

```
size_t fread(void *ptr, size_t size, size_t count, FILE *stream);
```

La función fread lee desde el stream apuntado por *stream*, *count* elementos de tamaño *size* y los almacena en el bloque de memoria apuntado por *ptr*.

Parámetros:
ptr – puntero a un bloque de memoria.
size – tamaño en bytes de cada elemento a ser leído.
count – cantidad de elementos a leer, cada uno de tamaño *size*.
stream – puntero a una estructura FILE que especifica el flujo de datos.

Valor de retorno:
La función fread devuelve 1 si pudo leer correctamente y en caso contrario 0.

Ejemplo:

```
void leerCliente(void){
 FILE *p;
 struct Cliente c;
 p = fopen("C:\\clientes.dat", "rb");
 if (p==NULL){
  cout << "Error, no se pudo abrir el archivo.";
  return 0;
 }
 while(fread(&c, sizeof(struct Cliente), 1, p)){
  cout << c.nombre;
 }
 fclose(p);
 return 1;
}
```

Escribir

fwrite

```
size_t fwrite(const void *ptr, size_t size, size_t count, FILE *stream);
```

La función fread escribe en el stream apuntado por *stream*, *count* elementos del tamaño *size* y los almacena en el bloque de memoria apuntado por *ptr*.

Parámetros:
ptr – puntero a un bloque de memoria desde donde se escribirán los datos.
size – tamaño en bytes de cada elemento a ser escrito.
count – cantidad de elementos a escribir, cada uno de tamaño size.
stream – puntero a una estructura FILE que especifica el flujo de datos.

Valor de retorno:
La función fwrite retorna 1 en caso de que se hayan podido escribir correctamente los datos o 0 en caso contrario.

Otras funciones

ftell

```
long ftell(FILE *stream);
```

La función ftell obtiene el valor actual del indicador de posición del archivo apuntado por *stream*.

Parámetros:
stream – puntero a una estructura FILE que especifica el flujo de datos.

Valor de retorno:
La función ftell retorna el valor del indicador de posición del archivo, osea la cantidad de bytes del archivo si el indicador de posición apunta al final del mismo, 0L si el indicador apunta al principio del mismo o la cantidad de bytes leídos o desplazados si el indicador apunta a algún otro lugar.

Ejemplo:

```
long tamanioArchivo(char *strArchivo){
  FILE *p;
  p = fopen(strArchivo, "rb");
  long tamArchivo=0;
  if (p){
    fseek(p, 0, 2);
    tamArchivo=ftell(p);
    fclose(p);
  }
return tamArchivo;
}
```

fseek

```
int fseek(FILE *stream, long int offset, int origin);
```

La función fseek setea el indicador de posición del archivo asociado al puntero *stream* a una nueva posición. Ésta nueva posición se indica en cantidad de bytes a desplazarse por el valor *offset* desde una referencia indicada por el parámetro *origin*.

Parámetros:
stream – puntero a una estructura FILE que especifica el flujo de datos.
offset – tamaño en bytes a desplazarse desde el origen.
origin – posición desde que comienza el desplazamiento:

SEEK_SET	0	Inicio del archivo
SEEK_CUR	1	Posición actual del archivo
SEEK_END	2	Final del archivo

Tabla 2: Valores del parámetro origin.

Valor de retorno:
La función fseek retorna un valor distinto de cero si una petición no se pudo satisfacer.

Ejemplo:

```
#include <iostream>
using namespace std;

int main(void){
  FILE *p;
  p=fopen("C:\\sample.txt", "w");
  fwrite("This is an apple",sizeof(char), strlen("This is an apple"), p);
  fseek (p , 9, SEEK_SET);
  fwrite(" sam", sizeof(char), strlen(" sam"), p);
  fclose (p);
}
```

size_t

Tipo de dato entero sin signo que devuelve el operador sizeof.

size_t equivale a long unsigned int