|  |  |  |
| --- | --- | --- |
|  | **Carátula para entrega de prácticas** | |
| Facultad de Ingeniería | | Laboratorio de docencia |

Laboratorios de computación

salas A y B

|  |  |
| --- | --- |
| *Profesor:* | Rodriguez Espino Claudia |
| *Asignatura:* | Fundamentos de Programación |
| *Grupo:* | 1102 |
| *No de Práctica(s):* | Practica número 13 |
| *Integrante(s):* | Gutierrez Orozpe Luis Fernando |
|  |  |
|  |  |
| *Semestre:* | 2018-1 |
| *Fecha de entrega:* | 27 de Noviembre de 2017 |
| *Obervaciones:* |  |
|  |  |

CALIFICACIÓN: \_\_\_\_\_\_\_\_\_\_

OBJETIVO

Elaborar programas en lenguaje C que requieran el uso de archivos de texto plano en la resolución de problemas, entendiendo a los archivos como un elemento de almacenamiento secundario.

ACTIVIDADES

Compilar y ejecutar cada uno de los programas incluidos dentro de esta práctica para la mejor comprensión de los temas al manejar archivos dentro de la programación en lenguaje C.

**APUNTADOR A ARCHIVO**

Un apuntador a un archivo es un hilo común que unifica el sistema de Entrada/Salida (E/S) con un buffer donde se transportan los datos.

Un apuntador a archivo señala a la información que contiene y define ciertas características sobre él, incluyendo el nombre, el estado y la posición actual del archivo.

Los apuntadores a un archivo se manejan en lenguaje C como variables apuntador de tipo FILE que se define en la cabecera *stdio.h*. La sintaxis para obtener una variable apuntador de archivo es la siguiente:

FILE \*F;

**ABRIR ARCHIVO**

La función fopen() abre una secuencia para que pueda ser utilizada y la asocia a un archivo. Su estructura es la siguiente:

\*FILE fopen(char \**nombre\_archivo*, char \**modo*);

Donde *nombre\_archivo* es un puntero a una cadena de caracteres que representan un nombre válido del archivo y puede incluir una especificación del directorio. La cadena a la que apunta *modo* determina cómo se abre el archivo.

Existen diferentes modos de apertura de archivos, los cuales se mencionan a continuación, además de que se pueden utilizar más de uno solo:

r: Abre un archivo de texto para lectura.

w: Crea un archivo de texto para escritura.

a: Abre un archivo de texto para añadir.

r+: Abre un archivo de texto para lectura / escritura.

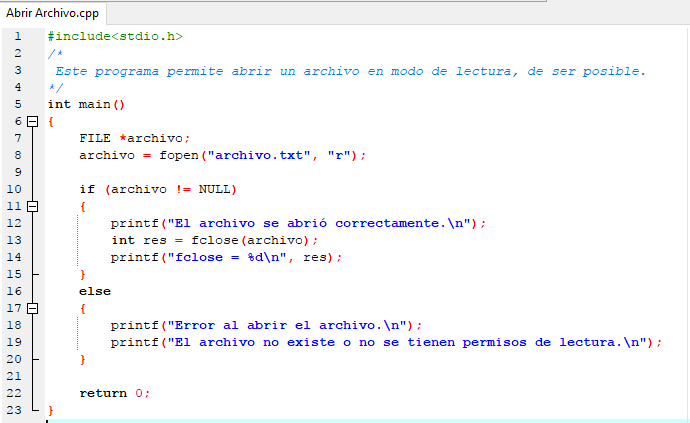
w+: Crea un archivo de texto para lectura / escritura.

a+: Añade o crea un archivo de texto para lectura / escritura.

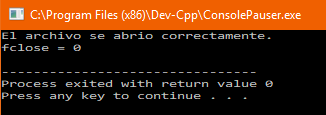
rb: Abre un archivo en modo lectura y binario.

wb: Crea un archivo en modo escritura y binario.

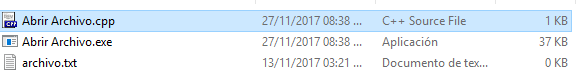
**PROGRAMA 1 : ABRIR PROGRAMA**



Este programa permite abrir un archivo en modo lectura, siempre y cuando este exista, en caso de que no exista el programa enviará el mensaje “Error al abrir el archivo”



En este caso debido a que el archivo si existe, el programa mandó el mensaje “El archivo se abrió correctamente”.



Se puede comprobar que efectivamente el archivo existe, por lo tanto se puede abrir de manera correcta.

**FUNCIONES FGETS Y FPUTS**

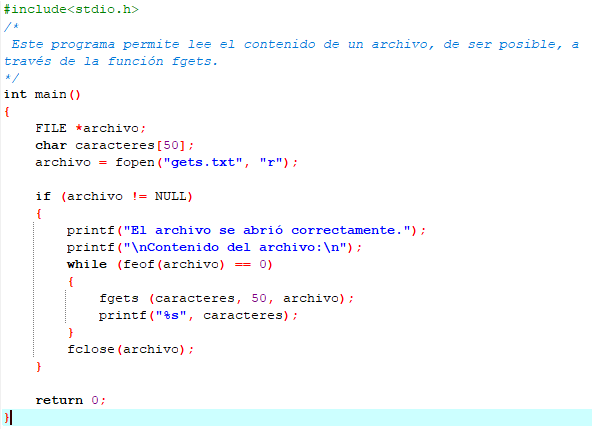
Las funciones fgets() y fputs() pueden leer y escribir, respectivamente, cadenas sobre los archivos. Las firmas de estas funciones son, respectivamente:

char \*fgets(char \*buffer, int tamaño, FILE \*apArch);

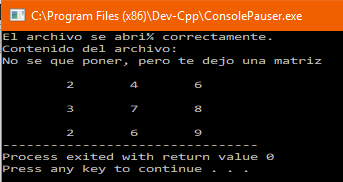
char \*fputs(char \*buffer, FILE \*apArch);

La función fputs() permite escribir una cadena en un archivo especifico. La función fgets() permite leer una cadena desde el archivo especificado. Esta función lee un renglón a la vez.

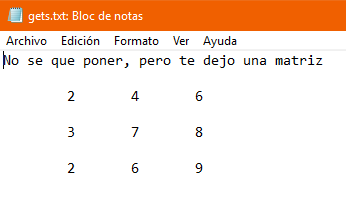
**PROGRAMA 2: FUNCION FGETS**



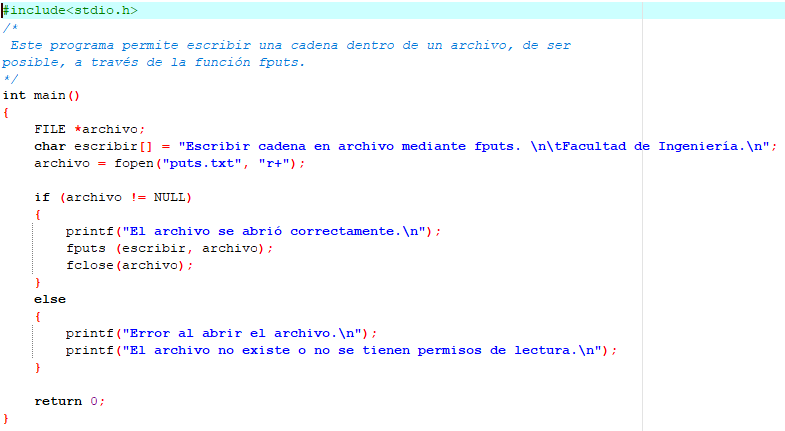
Este programa permite leer el contenido de un archivo a través de la función “fgets” solo si el archivo existe. Si existe el programa mandará el mensaje “El archivo se abrió correctamente” y mostrara su contenido como se puede observar a continuación.



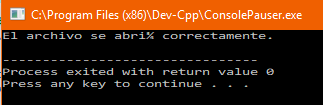
Al existir el archivo, el programa se ejecuta correctamente y muestra el contenido del archivo, en este caso el archivo contenía un mensaje y con él una matriz de orden 3x3.



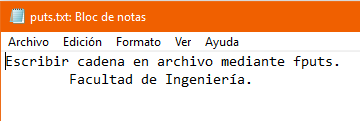
**PROGRAMA 3: FUNCION FPUTS**



Este programa escribe una cadena dentro de un archivo, esto es gracias a la función “fputs”, solo puede escribir la cadena si el archivo existe.



En este caso el archivo existe, por lo tanto escribió la cadena, gracias a esto al abrir el archivo podremos visualizar la cadena que el programa escribió en el.



**FUNCIONES FSCANF Y FPRINTF**

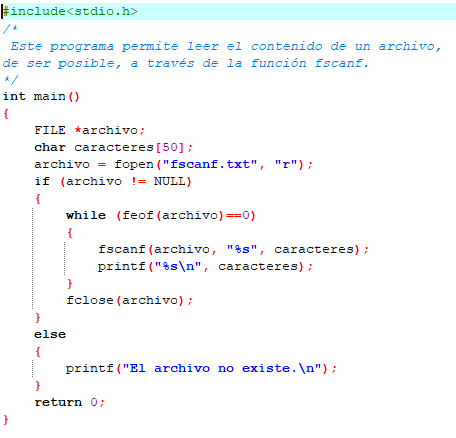
Las funciones fprintf() y fscanf() se comportan exactamente como printf() (imprimir) y scanf() (leer), excepto que operan sobre archivo. Sus estructuras son:

int fprintf(FILE \**apArch*, char \**formato*, ...);

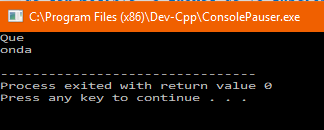
int fscanf(FILE \**apArch*, char \**formato*, ...);

Donde *apArch* es un apuntador al archivo devuelto por una llamada a la función fopen(), es decir, fprintf() y fscanf() dirigen sus operaciones de E/S al archivo al que apunta apArch. *formato* es una cadena que puede incluir texto o especificadores de impresión de variables. En los puntos suspensivos se agregan las variables (si es que existen) cuyos valores se quieren escribir en el archivo.

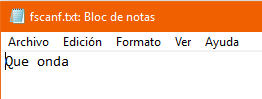
**PROGRAMA 4: FUNCION FSCANF**



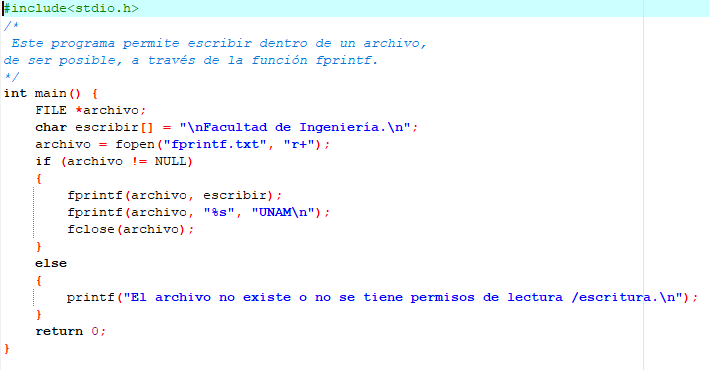
Este programa permitirá leer el contenido de un archivo siempre y cuando este exista. Si existe, el programa mostrará el texto que contiene el archivo, de lo contrario, enviara un mensaje especificando que “El archivo no existe”



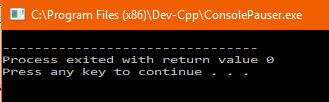
Al existir el archivo el programa lo abre automáticamente y nos muestra el texto que este contenido dentro de él.



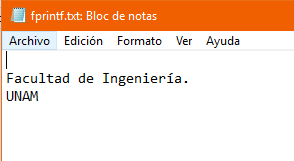
**PROGRAMA 5: FPRINTF**



Este programa nos permitirá escribir cualquier texto dentro de un archivo siempre y cuando este exista y cuente con permisos de lectura y escritura, en este caso se desea escribir dentro del archivo el texto “Facultado de Ingeniería” y posteriormente “UNAM”



Gracias a que el archivo “fprintf” existe el programa se ejecutó sin ningún problema y por lo tanto, al abrir dicho archivo podremos visualizar el texto que se tenía que escribir.



CONCLUSION

Esta práctica se me hizo muy interesante al poder aprender a manipular archivos dentro del lenguaje de programación C, de la misma forma estoy seguro que aprender estos temas me sirve de mucho para poder realizar el proyecto final de la materia Fundamentos de Programacion.