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

Laboratorios de computación salas A y B

Profesor: Karina García Morales

Asignatura: Fundamentos de programación

Grupo:

No. de Práctica: 12

Integrantes: Ever Ivan Rosales Gómez

No de Lista o brigade:

Semestre: 2023-2

Fecha de entrega: 7/06/23

Observaciones

CALIFICACIÓN

LECTURA Y ESCRITURA DE DATOS

OBJETIVO:

Elaborar programas en lenguaje C incorporando uso de archivos de texto para la resolución de problemas

INTRODUCCIÓN

Cuando se tiene una gran cantidad de información guardada en la computadora, se almacena en dispositivos de memoria secundarias llamados **archivos** que son colecciones de datos.
Otro punto importante a mencionar es definir un fichero. Un **fichero** es una porción de almacenamiento de memoria.
Y como se vio en prácticas pasadas un **apuntador** es un tipo de dato que almacena la dirección de memoria de un objeto.

Para poder usar ficheros es importante declarar un apuntador de tipo FILE. Es importante usar FILE ya que permite guardar o recuperar información de un fichero.

Un apuntador en un archivo unifica el sistema de Entrada-Salida con un buffer (reserva la cantidad de espacio de almacenamiento en el espacio de memoria).

Para **abrir un archivo** se usa **fopen** además se asocia el archivo que se quiere ejecutar de la siguiente manera:

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

Un modo determina cómo se abre el archivo, en esta práctica se utilizarán modos de apertura entre los que destacan los siguientes:

r	Abre un archivo sólo para lectura
w	Crea un archivo sólo para lectura
r+	Abre un archivo para lectura y escritura
w+	Crea un archivo para lectura y escritura
a+	Añade o crea un archivo de texto para lectura y escritura

rb	Abre archivo para lectura y binario
wb	Crea un archivo para escritura y binario

Es de suma importancia mencionar que al usar fopen es necesario cerrar el archivo con ayuda de **fclose** realizando un cierre formal del archivo a nivel del sistema operativo, por ejemplo; cuando dejamos de usar la terminal tecleamos **exit** para que cierre de manera adecuada.

```
[Aldair:~ aldair$ exit
logout

Saving session...
...copying shared history...
...saving history...truncating history files...
...completed.
Deleting expired sessions...none found.

[Proceso completado]
```

Al aplicar el fclose tiene que devolver un valor el cual será cero significando que la operación de cierre fue exitosa.

Otro punto de importancia son las funciones. Con ayuda de funciones podemos asignar si queremos que un dato sea de lectura o escritura

LECTURA	
fgets	Permite leer un renglón de tipo cadena a la vez
fread	Lee una serie de elementos con ayuda de un apuntador
fscanf	Lee el tipo de dato sobre un archivo

ESCRITURA	
fputs	Escribe una serie de tipo cadena
fwrite	Escribe una serie de elementos con ayuda de una dirección de memoria
fprintf	Escribe un dato sobre un archivo

PROGRAMA 1

```
#include<stdio.h>
int main() {
FILE *archivo;
archivo = fopen("archivo.txt", "w"); //AQUI SE DEBÍA EL ERROR POR EL MODO
if (archivo != NULL) {
printf("El archivo se abrió correctamente.\n"); int res = fclose(archivo);
printf("fclose = %d\n", res);
}
else
{
printf("Error al abrir el archivo.\n");
printf("El archivo no existe o no se tienen permisos de lectura.\n"); }
return 0; }
~
```

```
Bulgaria20:~ fp24alu35$ gcc prog1.c -o prog1.out; ./prog1.out
El archivo se abrió correctamente.
fclose = 0
Bulgaria20:~ fp24alu35$
```

En este programa se encontraba un error, se estaba usando un modo **r**. Se cambió por un modo w que permite **crear** un archivo sólo para lectura y como se puede observar se una fopen y termina un fopen.
Al compilar y ejecutar se puede observar un fclose=0 que significa que se cerró correctamente.

PROGRAMA 2

```
#include<stdio.h>
int main() {
FILE *archivo;
char escribir[]="Escribir cadena en archivo mediante fputs.\n\tFacultad de Ingeniería.\n";
archivo = fopen("/Users/fp24alu35/puts.txt", "w");

if (archivo != NULL) {
        printf("El archivo se abrió correctamente.\n");
        fputs (escribir, archivo);
        fclose(archivo);
}
else
{
        printf("Error al abrir el archivo.\n");
        printf("El archivo no existe o no se tienen permisos de lectura.\n");
}
return 0; }
~
~
```

```
Bulgaria20:~ fp24alu35$ ls
Desktop         Library         Pictures        prog1.c         prog2.out
Documents       Movies          Public          prog1.out       puts.txt
Downloads       Music           archivo.txt     prog2.c
Bulgaria20:~ fp24alu35$ cat prog2.c
#include<stdio.h>
int main() {
FILE *archivo;
char escribir[]="Escribir cadena en archivo mediante fputs.\n\tFacultad de Ingeniería.\n";
archivo = fopen("/Users/fp24alu35/puts.txt", "w");

if (archivo != NULL) {
        printf("El archivo se abrió correctamente.\n");
        fputs (escribir, archivo);
        fclose(archivo);
}
else
{
        printf("Error al abrir el archivo.\n");
        printf("El archivo no existe o no se tienen permisos de lectura.\n");
}
return 0; }
Bulgaria20:~ fp24alu35$ cat puts.txt
Escribir cadena en archivo mediante fputs.
        Facultad de Ingeniería.
Bulgaria20:~ fp24alu35$ vi prog2.c
Bulgaria20:~ fp24alu35$
```

Primero hay que buscar la localización para incorporarlo a la parte donde está el fputs que tiene como finalidad escribir una serie de tipo cadena. Posteriormente al compilar y ejecutar el programa arroja que se abrió correctamente, para checar que si se guardó se ve la lista de los archivos ls
Finalmente se muestra únicamente lo que contiene con ayuda del printf

PROGRAMA 3

```
#include<stdio.h>
int main() {
FILE *archivo;
char caracteres[50];
archivo = fopen("/Users/fp24alu35/puts.txt", "r");
if (archivo != NULL) {
printf("El archivo se abrió correctamente."); printf("\nContenido del archivo:\n");
while (feof(archivo) == 0)
{
fgets (caracteres, 50, archivo);
printf("%s", caracteres); }
fclose(archivo); }
return 0; }
~
```

```
Bulgaria20:~ fp24alu35$ gcc prog3.c -o prog3.out; ./prog3.out
El archivo se abrió correctamente.
Contenido del archivo:
Escribir cadena en archivo mediante fputs.
        Facultad de Ingeniería.
        Facultad de Ingeniería.
```

En este programa se usa un fopen en un nombre del archivo específico y contiene **r** que su finalidad es abrir un archivo sólo para lectura

PROGRAMA 5

```
#include<stdio.h>
int main() {
FILE *archivo;
char escribir[] = "Escribir cadena en archivo mediante fprinft.\nFacultad de Ingeniería.\n";
archivo = fopen("/Users/fp24alu35/fprintf.txt", "w");
if (archivo != NULL)
{
        fprintf(archivo, escribir);
        fprintf(archivo, "%s", "UNAM\n");
        fclose(archivo);
}
else
{
        printf("El archivo no existe o no se tiene permisos de lectura/escritura.\n");
}
return 0; }
~
```

```
Bulgaria20:~ fp24alu35$ cat fprintf.txt
Escribir cadena en archivo mediante fprinft.
Facultad de Ingeniería.
UNAM
```

Al igual que el programa anterior se utiliza el mismo nombre de archivo, pero ahora **w** que va a crear un archivo sólo para lectura mostrando solo lo que contenga fprintf. al final se observa el fclose

PROGRAMA 7

```
#include <stdio.h>
int main(int argc, char **argv) {
        FILE *ap;
        unsigned char buffer[2048]; // Buffer de 2 Kbytes
        int bytesLeidos;
        // Si no se ejecuta el programa correctamente
        if(argc < 2) {
printf("Ejecutar el programa de la siguiente manera: \n\tnombre_\tprograma nombre_archivo\n");
return 1; }
// Se abre el archivo de entrada en modo lectura y binario
ap = fopen(argv[1], "rb");
if(!ap)
{
printf("El archivo %s no existe o no se puede abrir", argv[1]);
return 1;
}
while(bytesLeidos = fread(buffer, 1, 2048, ap))
{
printf("%s", buffer);
 }
fclose(ap);
return 0;
}
~
```

```
Bulgaria20:~ fp24alu35$ ./prog6.out fprintf.txt
Escribir cadena en archivo mediante fprinft.
Facultad de Ingeniería.
UNAM
Bulgaria20:~ fp24alu35$
```

```
                ==
1 warning generated.
Ejecutar el programa de la siguiente manera:
        nombre_ programa nombre_archivo
Bulgaria20:~ fp24alu35$ ./prog6.out
Ejecutar el programa de la siguiente manera:
        nombre_ programa nombre_archivo
Bulgaria20:~ fp24alu35$
```

Este programa contiene su fopen y utilizando un modo **rb** que abre archivo para lectura y binario. Va a leer únicamente lo que contiene printf.
Marca algunas sugerencias, pero se puede ejecutar como se muestra en pantalla

PROGRAMA 8

```
#include <stdio.h>
int main(int argc, char **argv)
{
FILE *archEntrada, *archivoSalida;
unsigned char buffer[2048]; // Buffer de 2 Kbytes
int bytesLeidos;

    // Si no se ejecuta el programa correctamente
if(argc < 3) {
        printf("Ejectuar el programa de la siguiente manera:\n");
        printf("\tnombre_programa \tarchivo_origen \tarchivo_destino\n");
        return 1;
}

// Se abre el archivo de entrada en modo de lectura y binario
archEntrada = fopen(argv[1], "rb");

if(!archEntrada) {
        printf("El archivo %s no existe o no se puede abrir", argv[1]);
        return 1; }

// Se crea o sobreescribe el archivo de salida en modo binario
archivoSalida = fopen(argv[2], "wb");
if(!archivoSalida) {
printf("El archivo %s no puede ser creado", argv[2]);
return 1; }

// Copia archivos
while (bytesLeidos = fread(buffer, 1, 2048, archEntrada))
fwrite(buffer, 1, bytesLeidos, archivoSalida);
// Cerrar archivos
fclose(archEntrada); fclose(archivoSalida);
return 0; }
~
```

```
Bulgaria20:~ fp24alu35$ gcc prog7.c -o prog7.out
prog7.c:29:20: warning: using the result of an assignment as a condition without parentheses
      [-Wparentheses]
while (bytesLeidos = fread(buffer, 1, 2048, archEntrada))
       ~~~~~~~~~~~~~^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
prog7.c:29:20: note: place parentheses around the assignment to silence this warning
while (bytesLeidos = fread(buffer, 1, 2048, archEntrada))
                   ^
       (                                               )
prog7.c:29:20: note: use '==' to turn this assignment into an equality comparison
while (bytesLeidos = fread(buffer, 1, 2048, archEntrada))
                   ^
                   ==
1 warning generated.
Bulgaria20:~ fp24alu35$ ./prog7.out
Ejectuar el programa de la siguiente manera:
        nombre_programa         archivo_origen  archivo_destino
Bulgaria20:~ fp24alu35$ ./prog7.out fprintf.txt copia_clase12.txt
Bulgaria20:~ fp24alu35$ cat copia_clase12.txt
Escribir cadena en archivo mediante fprinft.
Facultad de Ingeniería.
UNAM
```

Por último, es este programa se encuentran condicionados y en caso de que se cumplan tiene como bloque de instrucciones diferente modo

EJERCICIOS DE TAREA

Modifica los siguientes programas para emplear el uso de archivos almacenando los datos que ingresa el usuario y el resultado esperado e indica que hace cada programa.

```c
#include <stdio.h>
#include <locale.h>
int main (){
        setlocale(LC_ALL,"ESPAÑOL");
        FILE *archivo; //se agrega un apuntador
        char listaNombre[30],a;
        archivo=fopen("/Users/aldair.txt","w"); //se agrega un fopen con su nombre de archivo y comando w
        int i, n;
        printf("\n\n\tIngresa el nombre del alumno:");
        fgets(listaNombre,30,archivo);
        for (i=0;i<=25;i++)
        {
                a=listaNombre[i];
                if('a'==1||'a'==2||'a'==3||'a'==4||'a'==5||'a'==6||'a'==7||'a'==8||'a'==9||'a'==0)
                {
                        n=i;
                }
                else
                {
                        if(i!=n)
                        printf("%c",listaNombre[i]);
                }
        fclose(archivo); //se cierra el fopen con ayuda de fclose
        }
return 0;
}
```

```
Aldair:~ aldair$ vi tarea1.c
Aldair:~ aldair$ gcc tarea1.c -o tarea1.out
Aldair:~ aldair$ ./tarea1.out


Segmentation fault: 11
Aldair:~ aldair$
```

3.- Completa el cuadro referente a funciones, sintaxis, ejemplo y características de cada una de las funciones vistas en el laboratorio.

Función	Sintaxis	Características	Ejemplo de sintaxis
fputs()	fputs char *fputs(char *Buffer, FILE *apArch);	La función fputs() permite escribir una cadena en un archivo especifico	fputs (listaNombre, archivo);
fgets()	Char*fgets(char*buffer, int tamaño, FILE*nombre_archivo)	Permite leer un renglón de tipo cadena a la vez	Char*fgets (char*buffer, int [20], FILE*archivo)
fopen()	*FILE fopen (char*nombre_archivo, char*modo)	Abre una secuencia de que permite ser utilizada y asociada a un archivo	Fopen (ejercicios,"w");
Siempre un fopen va asociado con un fclose			
fclose()	Fclose (nombre_archivo);	Permite cerrar una secuencia que fue abierta con ayuda de fopen( )	Fclose(ejercicios);
fprintf()	int fprintf (FILE *apArch, char *formato, ...);	Escribe un dato sobre un archivo	Fprintf (archivo, "%s", "UNAM\n");
fscanf()	int fscanf (FILE *apArch, char *formato, ...);	Lee el tipo de dato sobre un archivo	fscanf(archivo, "%s", caracteres);
fread()	int fread (void *ap, size_t tam, size_t nelem, FILE *archivo)	Lee una serie de elementos con ayuda de un apuntador	Fread (buffer, sizeof (char), sizeof(buffer),fp);
fwrite()	Size_t fwrite (void *punter, size_t tamaño, size_t cantidad, FILE, nombre_archivo)	Escribe una serie de elementos con ayuda de una dirección de memoria	Fwrite(buffer, 1, bytesLeidos, archivoSalida)

CONCLUSIÓN

La importancia de los archivos es coleccionar una serie de elementos llamados registros que son de igual tipo, es importante conocer sus funciones o también llamados modos.

BIBLIOGRAFÍA

El lenguaje de programación C. Brian W. Kernighan, Dennis M. Ritchie, segunda edición, USA, Pearson Edu