	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: Alejandro Esteban Pimentel Alarcon

Asignatura: Fundamentos de Programación

Grupo: 3

No. de practica: 13

Integrantes:

Jonan Gómez Mendoza 5641
Franco Inglés Carolina 2836
Lucia Nicole Rosette Hernández *2768*

Semestre: *1*

Fecha de entrega: *noviembre 11, 2019*

Observaciones: Recuerda que los números en las prácticas son solo ejemplos, en tus programas siempre debes cambiar las cosas según los casos, el fscanf necesitaba poder leer más que "8" caracteres, eso es muy poco para una palabra, debía corresponder con tu longitud de lista (20)

CALIFICACIÓN: **10**

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.

Introducción
FICHEROS

El estándar de C contiene varias funciones para la edición de ficheros, éstas están definidas en la cabecera stdio.h y por lo general empiezan con la letra f, haciendo referencia a file. Adicionalmente se agrega un tipo FILE, el cual se usará como apuntador a la información del fichero. La secuencia que usaremos para realizar operaciones será la siguiente:

- Crear un apuntador del tipo FILE *
- Abrir el archivo utilizando la función fopen y asignándole el resultado de la llamada a nuestro apuntador.
- Hacer las diversas operaciones (lectura, escritura, etc).
- Cerrar el archivo utilizando la función fclose.

Fopen

Esta función sirve para abrir y crear ficheros en disco.

El prototipo correspondiente de fopen es:

```
FILE * fopen (const char *filename, const char *opentype);
```

Los parámetros de entrada de fopen son:

- filename: una cadena que contiene un nombre de fichero válido. opentype: especifica el tipo de fichero que se abrirá o se creará.

Una lista de parámetros opentype para la función fopen son:

- "r" : abrir un archivo para lectura, el fichero debe existir.
- "w" : abrir un archivo para escritura, se crea si no existe o se sobreescribe si existe.
- "a" : abrir un archivo para escritura al final del contenido, si no existe se crea.
- "r+" : abrir un archivo para lectura y escritura, el fichero debe existir.

- "w+" : crear un archivo para lectura y escritura, se crea si no existe o se sobreescribe si existe.
- "r+b ó rb+" : Abre un archivo en modo binario para actualización (lectura y escritura).
- "rb" : Abre un archivo en modo binario para lectura.

Adicionalmente hay tipos utilizando "b" (binary) los cuales no serán mostrados por ahora y que solo se usan en los sistemas operativos que no pertenecen a la familia de unix.

Fclose

Esta función sirve para poder cerrar un fichero que se ha abierto.

El prototipo correspondiente de fclose es:

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

Un valor de retorno cero indica que el fichero ha sido correctamente cerrado, si ha habido algún error, el valor de retorno es la constante EOF.

Un ejemplo pequeño para abrir y cerrar el archivo llamado fichero.in en modo lectura:

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char** argv)
{
	FILE *fp;
	fp = fopen ( "fichero.in", "r" );
	if (fp==NULL) {fputs ("File error",stderr); exit (1);}
	fclose ( fp );

	return 0;
}
```

Como vemos, en el ejemplo se utilizó el opentype "r", que es para la lectura.

Otra cosa importante es que el lenguaje C no tiene dentro de sí una estructura para el manejo de excepciones o de errores, por eso es necesario comprobar que el archivo fue abierto con éxito "if (fp == NULL)". Si fopen pudo abrir el archivo con éxito devuelve la referencia al archivo (FILE *), de lo contrario devuelve NULL y en este caso se deberá revisar la dirección del archivo o los permisos del mismo. En

estos ejemplos solo vamos a dar una salida con un retorno de 1 que sirve para señalar que el programa terminó por un error.

Feof

Esta función sirve para determinar si el cursor dentro del archivo encontró el final (end of file). Existe otra forma de verificar el final del archivo que es comparar el carácter que trae fgetc del archivo con el macro EOF declarado dentro de stdio.h, pero este método no ofrece la misma seguridad (en especial al tratar con los archivos "binarios"). La función feof siempre devolverá cero (Falso) si no es encontrado EOF en el archivo, de lo contrario regresará un valor distinto de cero (Verdadero).

El prototipo correspondiente de feof es:

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

Rewind

Literalmente significa "rebobinar", sitúa el cursor de lectura/escritura al principio del archivo.

El prototipo correspondiente de rewind es:

```
void rewind(FILE *fichero);
```

LECTURAS

Un archivo generalmente debe verse como un string (una cadena de caracteres) que está guardado en el disco duro. Para trabajar con los archivos existen diferentes formas y diferentes funciones. Las funciones que podríamos usar para leer un archivo son:

```
char fgetc(FILE *archivo)
char *fgets(char *buffer, int tamano, FILE *archivo)
size_t fread(void *puntero, size_t tamano, size_t cantidad, FILE *archivo);
int fscanf(FILE *fichero, const char *formato, argumento, ...);
```

Las primeras dos de estas funciones son muy parecidas entre si. Pero la tercera, por el número y el tipo de parámetros, nos podemos dar cuenta de que es muy diferente, por eso la trataremos aparte junto al fwrite que es su contraparte para escritura.

Fgets

Esta función está diseñada para leer cadenas de caracteres. Leerá hasta n-1 caracteres o hasta que lea un cambio de línea '\n' o un final de archivo EOF. En este último caso, el carácter de cambio de línea '\n' también es leído.

El prototipo correspondiente de fgets es:

```
char *fgets(char *buffer, int tamaño, FILE *archivo);
```

El primer parámetro buffer lo hemos llamado así porque es un puntero a un espacio de memoria del tipo char (podríamos usar un arreglo de char). El segundo parámetro es tamaño que es el limite en cantidad de caracteres a leer para la funcion fgets. Y por ultimo el puntero del archivo por supuesto que es la forma en que fgets sabra a que archivo debe leer.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
	FILE *archivo;

	char caracteres[100];

	archivo = fopen("prueba.txt","r");

	if (archivo == NULL)
		exit(1);
	else
      {
	    printf("\nEl contenido del archivo de prueba es \n\n");
	    while (feof(archivo) == 0)
	    {
		fgets(caracteres,100,archivo);
		printf("%s",caracteres);
	    }
	 system("PAUSE");
      }
      fclose(archivo);
	return 0;
}
```

Este es el mismo ejemplo de antes con la diferencia de que este hace uso de fgets en lugar de fgetc. La función fgets se comporta de la siguiente manera, leerá del archivo apuntado por archivo los caracteres que encuentre y a ponerlos en buffer

hasta que lea un carácter menos que la cantidad de caracteres especificada en tamaño o hasta que encuentre el final de una línea (\n) o hasta que encuentre el final del archivo (EOF). En este ejemplo no vamos a profundizar más que para decir que caracteres es un buffer, los pormenores serán explicados en la sección de manejo dinámico de memoria.

El beneficio de esta función es que se puede obtener una línea completa a la vez. Y resulta muy útil para algunos fines como la construcción de un parser de algún tipo de archivo de texto.

Fscanf

La función fscanf funciona igual que scanf en cuanto a parámetros, pero la entrada se toma de un fichero en lugar del teclado.

El prototipo correspondiente de fscanf es:

```
int fscanf(FILE *fichero, const char *formato, argumento, ...);
```

Podemos ver un ejemplo de su uso, abrimos el documento "fichero.txt" en modo lectura y leyendo dentro de el.

```
#include <stdio.h>


int main ( int argc, char **argv )
{
	FILE *fp;

	char buffer[100];

	fp = fopen ( "fichero.txt", "r" );

	fscanf(fp, "%s" ,buffer);
	printf("%s",buffer);

	fclose ( fp );

	return 0;
}
```

ESCRITURA

Así como podemos leer datos desde un fichero, también se pueden crear y escribir ficheros con la información que deseamos almacenar, Para trabajar con los archivos

existen diferentes formas y diferentes funciones. Las funciones que podríamos usar para escribir dentro de un archivo son:

int fputc(int caracter, FILE *archivo)
int fputs(const char *buffer, FILE *archivo)
size_t fwrite(void *puntero, size_t tamano, size_t cantidad, FILE *archivo);
int fprintf(FILE *archivo, const char *formato, argumento, ...);

Fprintf

La función fprintf funciona igual que printf en cuanto a parámetros, pero la salida se dirige a un archivo en lugar de a la pantalla.

El prototipo correspondiente de fprintf es:

```
int fprintf(FILE *archivo, const char *formato, argumento, ...);
```

Podemos ver un ejemplo de su uso, abrimos el documento "fichero.txt" en modo lectura/escritura y escribimos dentro de el.

```
#include <stdio.h>

int main ( int argc, char **argv )
{
	FILE *fp;

	char buffer[100] = "Esto es un texto dentro del fichero.";

	fp = fopen ( "fichero.txt", "r+" );

	fprintf(fp, buffer);
	fprintf(fp, "%s", "\nEsto es otro texto dentro del fichero.");

	fclose ( fp );

	return 0;
}
```

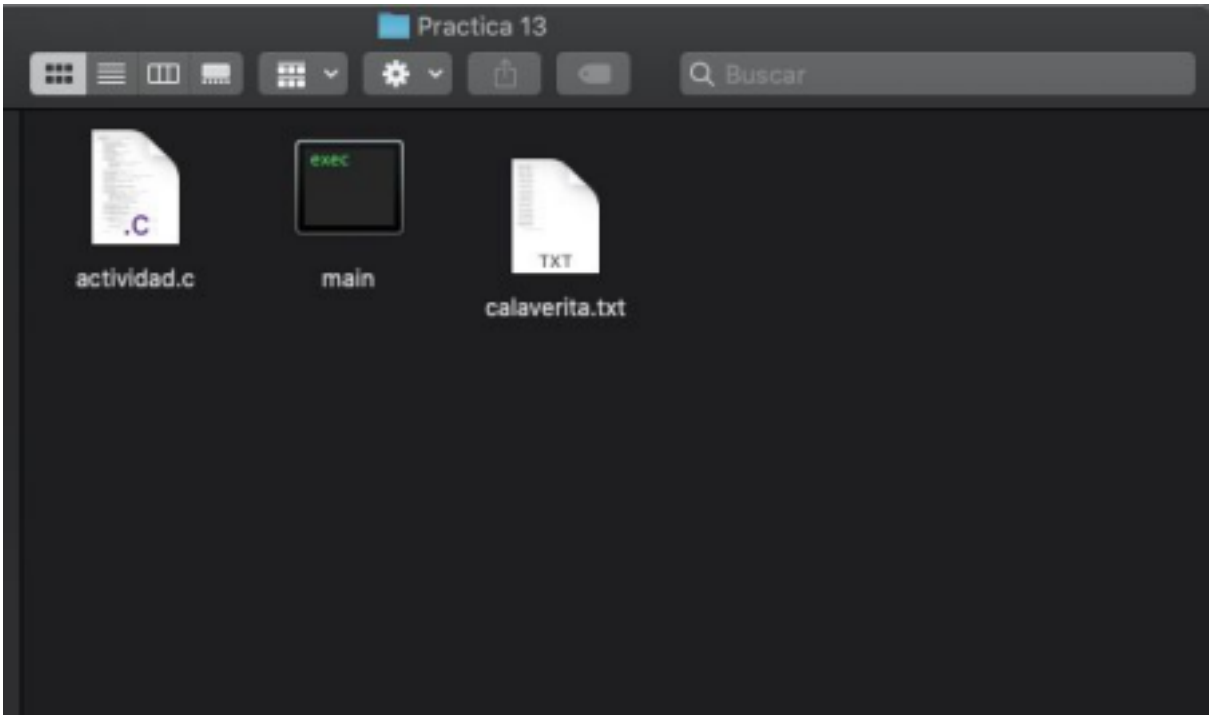
ACTIVIDADES

```c
#include<stdio.h>
//se incluye otra libreria para las nuevas funciones
#include<string.h>
int main(){
    //se declararon dos apuntadores
    FILE *archivo,*archivosalida;
    //con esto manejaremos el texto interior
    char palabra [21], linea[101];
    printf("dime el nombre del archivo\n");
    char nombre[21];
    scanf("%s",nombre);
    char nombresalida[21];
    printf("dame el nombre del archivo nuevo\n");
    scanf("%s",nombresalida);
    //se abre el archivo para lectura
    archivo=fopen(nombre,"r");
    //se declara el contador de lineas
    int contadorlineas=0;
    //con este se imprime cada linea
    while(!feof(archivo)){
        //obtine la linea y la guarda en la lista llamada lista
        fgets(linea,100,archivo);
        //imprime la linea
        printf("%s",linea);
        //cada vez que termina el contador aumenta 1
        contadorlineas++;
    }
    //imprime el contador de lineas
    printf("\nnumero de lineas: %i\n",contadorlineas);
    //se declara el contador de palabras
    int contadorpalabras=0;
    //se vuelve a abrir el archivo para leer para que comience desde el principio
    archivo=fopen(nombre,"r");
    while(!feof(archivo)){
    //obtiene las palabras se para cada que termina una palabra
    fscanf(archivo,"%8s",palabra);
    //aumenta el contador cada que pasa una palabra
    contadorpalabras++;
    }
```

```c
    //imprime el contador de palabras
    printf("numero de palabras: %i\n",contadorpalabras);
    //se vuelve a abrir el archivo desde el principio
    archivo=fopen(nombre,"r");
    //se declara el contador de caracteres y npalabra(numero de la palabra)
    int contadorcaracteres=0,npalabra;
    while(!feof(archivo)){
        //escanea la palabra y lo guarda en la lista palabra
        fscanf(archivo,"%8s",palabra);
        //se guarda cuantos caracteres tiene palabra
        npalabra=strlen(palabra);
        //se hace la sumatoria de las palabras
        contadorcaracteres=contadorcaracteres+npalabra;
    }
    printf("numnero de caracteres %i\n",contadorcaracteres);
    //se abre el archivo para lectura de nuevo
    archivo=fopen(nombre,"r");
    //se crea el archivo para escribir adentro
    archivosalida=fopen(nombresalida,"w");
    //se hace una matriz de matrices para guardar las lineas
    //el tamaño sera de las lineas
    char listaarchivo[contadorlineas][100];
    //se va a empezar alrevez la ultima linea sera la primera
    for(int i=contadorlineas-1;i!=-1;i--){
        fgets(listaarchivo[i],100,archivo);
    }
    //se imprimen las lineas
    for(int i=0;i<contadorlineas;i++){
        //por alguna razon la primera vez no deja un espacio asi que se pone
            manualmente
        if(i==0){
            fprintf(archivosalida,"%s\n",listaarchivo[i]);
        }
        //despues sigue normal
        else{
            fprintf(archivosalida,"%s",listaarchivo[i]);
        }
    }
    return 0;
}
```

COMPROBACIÓN

Primero, la carpeta solo tiene calaverita.txt



Se corre el programa

```
dime el nombre del archivo
calaverita.txt
dame el nombre del archivo nuevo
calaveritainvertida.txt
En este mes singular,
En que todo es fiesta y danza,
Recordamos con amor
Tradiciones y alabanzas.

Es menester recordar
Que aún en la confianza,
De volvernos a encontrar,
Recordamos sus andanzas.

A nuestros seres queridos
Les ponemos un altar,
Pues sus almas y latidos
Los sentimos regresar!

Así que los festejamos
Con grandes piezas de pan,
Colocados en altares
Con flores velas, mezcal!

Estos panes primorosos,
Que de Colores están,
Con formas de cuerno y hueso
Se los hemos de dejar.
```

Para que pasen contentos
En su visita fugaz,
En esta tierra de amores
Los recuerdos que se van.

Para mí es un gran regalo
El poderlos encontrar,
En fechas tan especiales
Y poderlos disfrutar.

Si supieran mis muertitos
Cuánto los echo de menos,
Vendrían todos los días
Y se harían muy amenos!

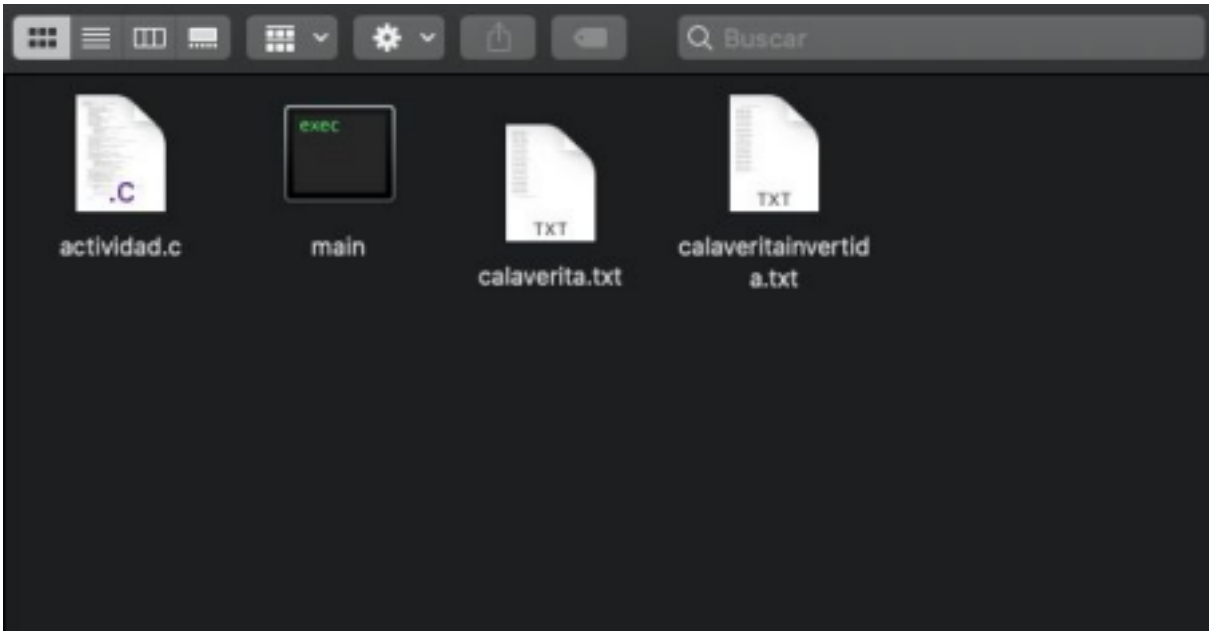
Yo los espero sentada
Pues sé que departiremos
Estas dos noches completas
Con atoles y galletas.

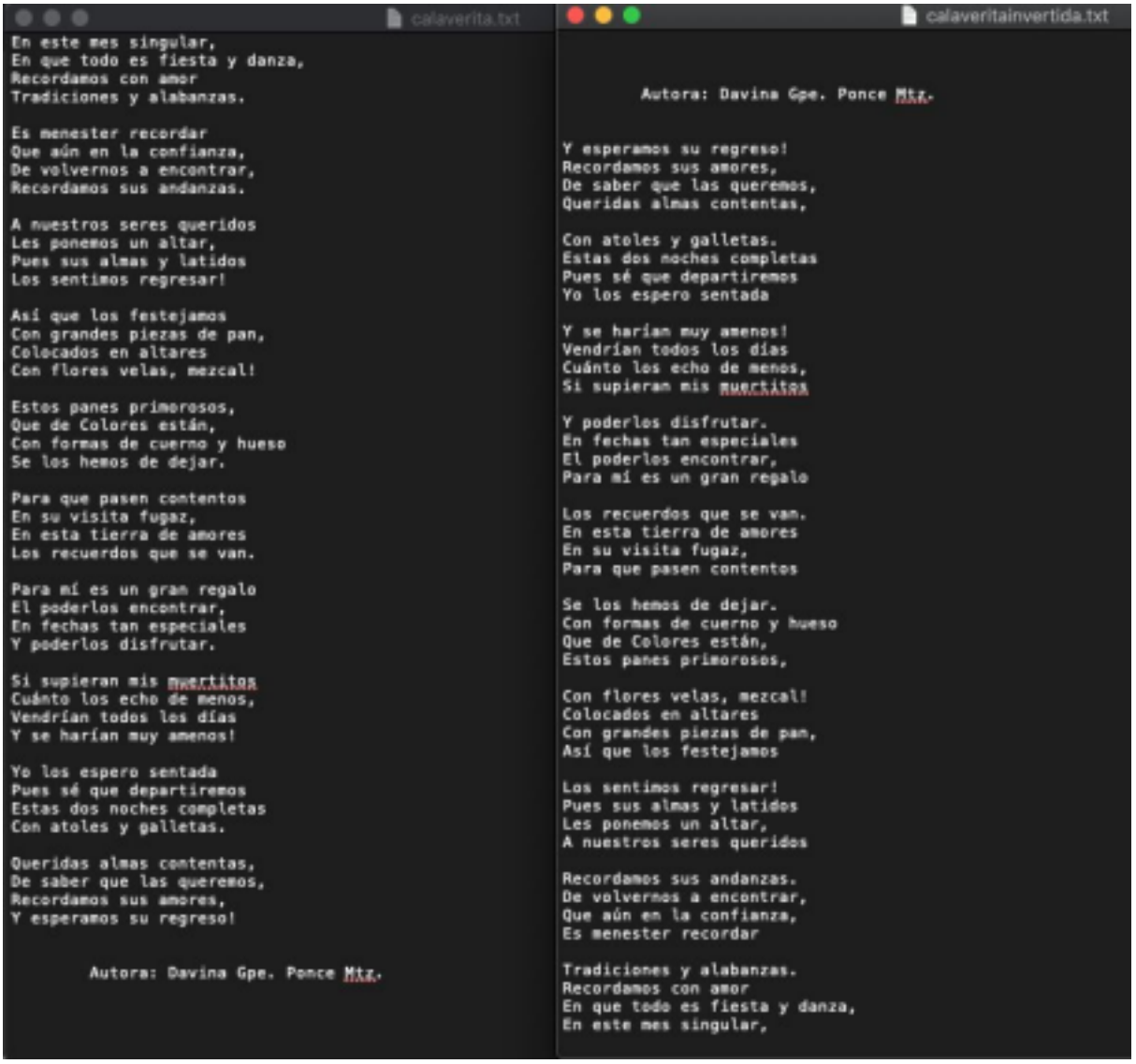
Queridas almas contentas,
De saber que las queremos,
Recordamos sus amores,
Y esperamos su regreso!

Autora: Davina Gpe. Ponce Mtz.

```
numero de lineas: 55
numero de palabras: 198
numnero de caracteres 849
```

Y revisamos el archivo de salida





CONCLUSIÓN:

En esta práctica aprendimos de una manera más completa la manera correcta de lectura, escritura en lenguaje C, además de saber en qué momento poder usar las funciones específicas que se mencionan en la introducción de la práctica.