
	Manual de prácticas del Laboratorio de Fundamentos de programación	Código:	MADO-17
		Versión:	03
		Página	1/12
		Sección ISO	8.3
		Fecha de emisión	26 / agosto / 2021
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

Guía práctica de estudio 12: Lectura y escritura de datos



Elaborado por	Actualizado por:	Revisado por:
M.C. Edgar E. García Cano Ing. Jorge A. Solano Gálvez	M.C. Cintia Quezada Reyes	M.C. Laura Sandoval Montaño

	Manual de prácticas del Laboratorio de Fundamentos de programación	Código:	MADO-17
		Versión:	03
		Página	2/12
		Sección ISO	8.3
		Fecha de emisión	26 / agosto / 2021
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

Guía práctica de estudio 12: Lectura y escritura de datos

Objetivo:

El alumno elaborará 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:

- A través de programas en C, emplear las funciones para crear, leer, escribir y sobrescribir archivos de texto plano.
- Manipular archivos empleando los diferentes tipos de acceso a ellos.

Introducción

Un archivo es un conjunto de datos estructurados en una colección de entidades elementales o básicas denominadas registros que son del mismo tipo, pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso.

Lenguaje C permite manejar la entrada y la salida de datos desde o hacia un archivo, respectivamente, a través del uso de la biblioteca de funciones de la cabecera *stdio.h*.

	Manual de prácticas del Laboratorio de Fundamentos de programación	Código:	MADO-17
		Versión:	03
		Página	3/12
		Sección ISO	8.3
		Fecha de emisión	26 / agosto / 2021
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

Licencia GPL de GNU

El software presente en esta práctica es libre bajo la licencia GPL de GNU, es decir, se puede modificar y distribuir mientras se mantenga la licencia GPL.

```

/*
 *
 * This program is free software: you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program. If not, see <http://www.gnu.org/licenses/>.
 *
 * Author: Jorge A. Solano
 *
 */

```


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;
```

	Manual de prácticas del Laboratorio de Fundamentos de programación	Código:	MADO-17
		Versión:	03
		Página	4/12
		Sección ISO	8.3
		Fecha de emisión	26 / agosto / 2021
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

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.

Cerrar archivo

La función *fclose()* cierra una secuencia que fue abierta mediante una llamada a *fopen()*. Escribe la información que se encuentre en el *buffer* al disco y realiza un cierre formal del archivo a nivel del sistema operativo.

Un error en el cierre de una secuencia puede generar todo tipo de problemas, incluyendo la pérdida de datos, destrucción de archivos y posibles errores intermitentes en el programa. La firma de esta función es:

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

	Manual de prácticas del Laboratorio de Fundamentos de programación	Código:	MADO-17
		Versión:	03
		Página	5/12
		Sección ISO	8.3
		Fecha de emisión	26 / agosto / 2021
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

Donde *apArch* es el apuntador al archivo devuelto por la llamada a *fopen()*. Si se devuelve un valor cero significa que la operación de cierre ha tenido éxito. Generalmente, esta función solo falla cuando un disco se ha retirado antes de tiempo o cuando no queda espacio libre en el mismo.

Código (abrir cerrar archivo)

Este programa permite abrir un archivo en modo de lectura, de ser posible ya que el archivo debe existir o debe tener permiso de lectura.

Programa1.c

```
#include<stdio.h>

int main()
{
    FILE *archivo;
    archivo = fopen("archivo.txt", "r");


    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;
}
```

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);
```

	Manual de prácticas del Laboratorio de Fundamentos de programación	Código:	MADO-17
		Versión:	03
		Página	6/12
		Sección ISO	8.3
		Fecha de emisión	26 / agosto / 2021
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

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

Código (fgets)

Este programa permite leer el contenido de un archivo, de ser posible, a través de la función *fgets()*.


Programa2.c

```
#include<stdio.h>

int main()
{
    FILE *archivo;
    char caracteres[50];
    archivo = fopen("gets.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;
}
```

	Manual de prácticas del Laboratorio de Fundamentos de programación	Código:	MADO-17
		Versión:	03
		Página	7/12
		Sección ISO	8.3
		Fecha de emisión	26 / agosto / 2021
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

Código (fputs)

Este programa permite escribir una cadena dentro de un archivo, de ser posible, a través de la función *fputs*.

Programa3.c

```
#include<stdio.h>

int main()
{
    FILE *archivo;
    char escribir[]="Escribir cadena en archivo mediante fputs.\n\tFacultad de
Ingeniería.\n";
    archivo = fopen("puts.txt", "r+");

    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;
}
```

Funciones fscanf y fprintf

Las funciones *fprintf()* y *fscanf()* se comportan similarmente a *printf()* (imprimir) y *scanf()* (leer), con la diferencia de que operan sobre un 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

	Manual de prácticas del Laboratorio de Fundamentos de programación	Código:	MADO-17
		Versión:	03
		Página	8/12
		Sección ISO	8.3
		Fecha de emisión	26 / agosto / 2021
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

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.

Código (fscanf)

Este programa permite leer el contenido de un archivo, de ser posible, a través de la función *fscanf*.


Programa4.c

```
#include<stdio.h>

int main()
{
    FILE *archivo;
    char caracteres[50];
    archivo = fopen("fscanf.txt", "r");
    if (archivo != NULL)
    {
        while (feof(archivo)==0)
        {
            fscanf(archivo, "%s", caracteres);
            printf("%s\n", caracteres);
        }
        fclose(archivo);
    }
    else
    {
        printf("El archivo no existe.\n");
    }
    return 0;
}
```

Código (fprintf)

Este programa permite escribir dentro de un archivo, de ser posible, a través de la función *fprintf*.

	Manual de prácticas del Laboratorio de Fundamentos de programación	Código:	MADO-17
		Versión:	03
		Página	9/12
		Sección ISO	8.3
		Fecha de emisión	26 / agosto / 2021
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

Programa5.c

```
#include<stdio.h>

int main()
{
    FILE *archivo;
    char escribir[] = "Escribir cadena en archivo mediante fprintf. \nFacultad de
Ingeniería.\n";
    archivo = fopen("fprintf.txt", "r+");
    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;
}
```

Funciones fread y fwrite

fread y *fwrite* son funciones que permiten trabajar con elementos de longitud conocida. *fread* permite leer uno o varios elementos de la misma longitud a partir de una dirección de memoria determinada (apuntador).


El valor de retorno es el número de elementos (bytes) leídos. Su sintaxis es la siguiente:

```
int fread(void *ap, size_t tam, size_t nelem, FILE *archivo)
```

fwrite permite escribir hacia un archivo uno o varios elementos de la misma longitud almacenados a partir de una dirección de memoria determinada.

El valor de retorno es el número de elementos escritos. Su sintaxis es la siguiente:

```
int fwrite(void *ap, size_t tam, size_t nelem, FILE *archivo)
```

	Manual de prácticas del Laboratorio de Fundamentos de programación	Código:	MADO-17
		Versión:	03
		Página	10/12
		Sección ISO	8.3
		Fecha de emisión	26 / agosto / 2021
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

Código (fread)

Este programa muestra el contenido de un archivo de texto. El nombre del archivo se recibe como argumento de la función principal.

Programa6.c

```
#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;
}
```

	Manual de prácticas del Laboratorio de Fundamentos de programación	Código:	MADO-17
		Versión:	03
		Página	11/12
		Sección ISO	8.3
		Fecha de emisión	26 / agosto / 2021
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

Código (fwrite)

Este programa realizar una copia exacta de dos archivos. Los nombres de los archivos (origen y destino) se reciben como argumentos de la función principal.

Programa7.c

```
#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("Ejecutar 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 sobrescribe 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;
}
```

	Manual de prácticas del Laboratorio de Fundamentos de programación	Código:	MADO-17
		Versión:	03
		Página	12/12
		Sección ISO	8.3
		Fecha de emisión	26 / agosto / 2021
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

Bibliografía



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