



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: M.I. Marco Antonio Martínez Quintana

Asignatura: Fundamentos de Programación

Grupo: 3

No de Práctica(s): #13

Integrante(s): Cuevas Antunez Samantha

*No. de Equipo de
cómputo empleado:* No aplica

No. de Lista o Brigada: 12

Semestre: Primer semestre

Fecha de entrega: 18/01/2021

Observaciones:

CALIFICACIÓN: _____



Lectura y escritura de datos

OBJETIVOS

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

- 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

ARCHIVO

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.

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.

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);
```

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.

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

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.

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)
```

RESULTADOS

Abrir cerra archivo

```
1 #include<stdio.h>
2 /*
3  * Este programa permite abrir un archivo en modo de lectura, de ser posible.
4  */
5
6 int main() {
7     FILE *archivo;
8     archivo = fopen("archivo.txt", "r");
9
10    if (archivo != NULL)
11    {
12        printf("El archivo se abrió correctamente.\n");
13        int res = fclose(archivo);
14        printf("fclose = %d\n", res);
15    } else {
16        printf("Error al abrir el archivo.\n");
17        printf("El archivo no existe o no se tienen permisos de lectura.\n");
18    }
19
20    return 0;
21 }
```

```
C:\Users\La familia\Desktop\Lenguaje C\practicas\P-13>gcc abrirCerarArchi.c -o abrirCerarArchi.exe
C:\Users\La familia\Desktop\Lenguaje C\practicas\P-13>abrirCerarArchi.exe
Error al abrir el archivo.
El archivo no existe o no se tienen permisos de lectura.
C:\Users\La familia\Desktop\Lenguaje C\practicas\P-13>
```

fgets

```
1 #include<stdio.h>
2 /*
3  * Este programa permite lee el contenido de un archivo,
4  * de ser posible, a través de la función fgets.
5  */
6
7 int main() {
8     FILE *archivo;
9     char caracteres[50];
10    archivo = fopen("gets.txt", "r");
11
12    if (archivo != NULL)
13    {
14        printf("El archivo se abrió correctamente.");
15        printf("\nContenido del archivo:\n");
16        while (feof(archivo) == 0)
17        {
18            fgets (caracteres, 50, archivo);
19            printf("%s", caracteres);
20        }
21        fclose(archivo);
22    }
23
24    return 0;
25 }
```

```
C:\Users\La familia\Desktop\Lenguaje C\practicas\P-13>gcc gets.c -o gets.exe
C:\Users\La familia\Desktop\Lenguaje C\practicas\P-13>gets.c
C:\Users\La familia\Desktop\Lenguaje C\practicas\P-13>
```

Abrir cerrar archivo fputs

```
1 #include<stdio.h>
2 /*
3  Este programa permite escribir una cadena dentro de un archivo, de ser
4  posible, a través de la función fputs.
5  */
6
7 int main()
8 {
9     FILE *archivo;
10    char escribir[] = "Escribir cadena en archivo mediante fputs. \n\tFacultad de Ingeniería.\n";
11    archivo = fopen("puts.txt", "r+");
12
13    if (archivo != NULL)
14    {
15        printf("El archivo se abrió correctamente.\n");
16        fputs (escribir, archivo);
17        fclose(archivo);
18    } else {
19        printf("Error al abrir el archivo.\n");
20        printf("El archivo no existe o no se tienen permisos de lectura.\n");
21    }
22
23    return 0;
24 }
```

```
C:\Users\La familia\Desktop\Lenguaje C\practicass\P-13>gcc fputs.
c -o fputs.exe

C:\Users\La familia\Desktop\Lenguaje C\practicass\P-13>fputs.exe
Error al abrir el archivo.
El archivo no existe o no se tienen permisos de lectura.

C:\Users\La familia\Desktop\Lenguaje C\practicass\P-13>
```

fscanf

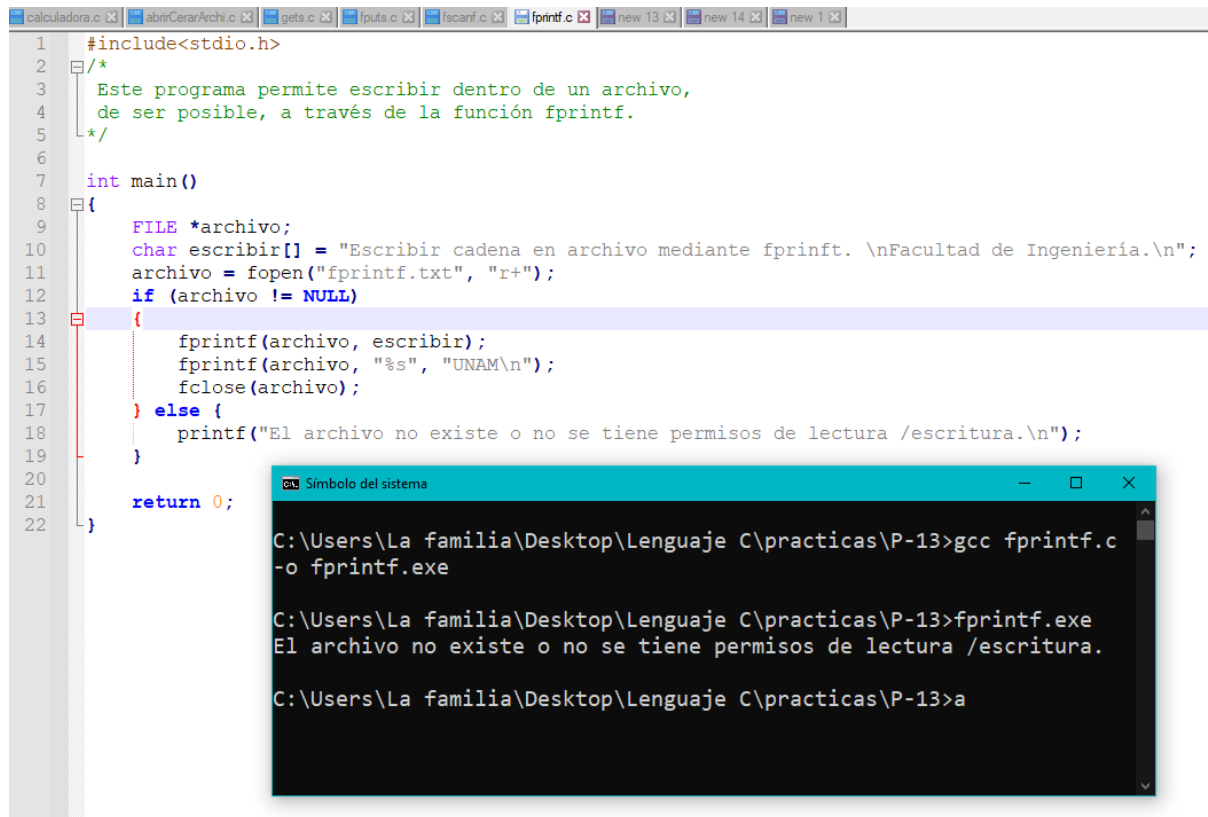
```
1 #include<stdio.h>
2 /*
3  Este programa permite leer el contenido de un archivo,
4  de ser posible, a través de la función fscanf.
5  */
6
7 int main()
8 {
9     FILE *archivo;
10    char caracteres[50];
11    archivo = fopen("fscanf.txt", "r");
12    if (archivo != NULL)
13    {
14        while (feof(archivo)==0)
15        {
16            fscanf(archivo, "%s", caracteres);
17            printf("%s\n", caracteres);
18        }
19        fclose(archivo);
20    } else {
21        printf("El archivo no existe.\n");
22    }
23
24    return 0;
25 }
```

```
C:\Users\La familia\Desktop\Lenguaje C\practicass\P-13>gcc fscanf
.c -o fscanf.exe

C:\Users\La familia\Desktop\Lenguaje C\practicass\P-13>fscanf.exe
El archivo no existe.

C:\Users\La familia\Desktop\Lenguaje C\practicass\P-13>
```

Fprintf



The image shows a C program in a code editor and its execution in a Windows command prompt. The code defines a function `fprintf` that writes to a file named `fprintf.txt`. The `main` function calls `fopen` to open the file in append mode. If the file is successfully opened, it writes the string "Escribir cadena en archivo mediante fprintf. \nFacultad de Ingeniería.\n" followed by "UNAM\n". If the file cannot be opened, it prints an error message. The command prompt shows the compilation of `fprintf.c` into `fprintf.exe` and the execution of `fprintf.exe`, which outputs the error message "El archivo no existe o no se tiene permisos de lectura /escritura."

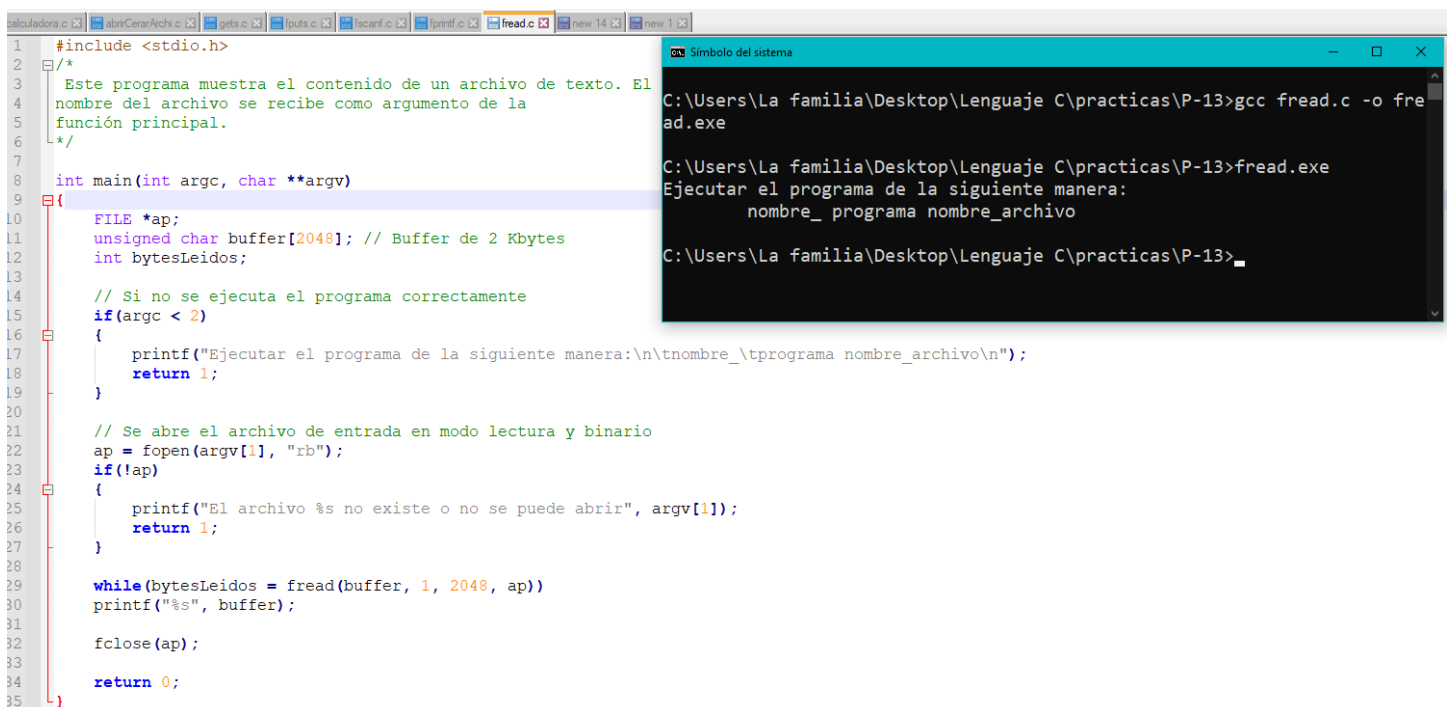
```
1 #include<stdio.h>
2 /*
3  Este programa permite escribir dentro de un archivo,
4  de ser posible, a través de la función fprintf.
5  */
6
7 int main()
8 {
9     FILE *archivo;
10    char escribir[] = "Escribir cadena en archivo mediante fprintf. \nFacultad de Ingeniería.\n";
11    archivo = fopen("fprintf.txt", "r+");
12    if (archivo != NULL)
13    {
14        fprintf(archivo, escribir);
15        fprintf(archivo, "%s", "UNAM\n");
16        fclose(archivo);
17    } else {
18        printf("El archivo no existe o no se tiene permisos de lectura /escritura.\n");
19    }
20
21    return 0;
22 }
```

```
C:\Users\La familia\Desktop\Lenguaje C\practicas\P-13>gcc fprintf.c
-o fprintf.exe

C:\Users\La familia\Desktop\Lenguaje C\practicas\P-13>fprintf.exe
El archivo no existe o no se tiene permisos de lectura /escritura.

C:\Users\La familia\Desktop\Lenguaje C\practicas\P-13>a
```

Fread



The image shows a C program in a code editor and its execution in a Windows command prompt. The code defines a function `fread` that reads from a file into a buffer. The `main` function takes two arguments: the program name and the file name. It opens the file in binary read mode and reads the contents into a buffer of 2048 bytes. It then prints the contents of the buffer. The command prompt shows the compilation of `fread.c` into `fread.exe` and the execution of `fread.exe`, which outputs the message "Ejecutar el programa de la siguiente manera: nombre_programa nombre_archivo".

```
1 #include <stdio.h>
2 /*
3  Este programa muestra el contenido de un archivo de texto. El
4  nombre del archivo se recibe como argumento de la
5  función principal.
6  */
7
8 int main(int argc, char **argv)
9 {
10     FILE *ap;
11     unsigned char buffer[2048]; // Buffer de 2 Kbytes
12     int bytesLeidos;
13
14     // Si no se ejecuta el programa correctamente
15     if(argc < 2)
16     {
17         printf("Ejecutar el programa de la siguiente manera:\n\tnombre_\tprograma nombre_archivo\n");
18         return 1;
19     }
20
21     // Se abre el archivo de entrada en modo lectura y binario
22     ap = fopen(argv[1], "rb");
23     if(!ap)
24     {
25         printf("El archivo %s no existe o no se puede abrir", argv[1]);
26         return 1;
27     }
28
29     while(bytesLeidos = fread(buffer, 1, 2048, ap))
30     printf("%s", buffer);
31
32     fclose(ap);
33
34     return 0;
35 }
```

```
C:\Users\La familia\Desktop\Lenguaje C\practicas\P-13>gcc fread.c -o fread.exe

C:\Users\La familia\Desktop\Lenguaje C\practicas\P-13>fread.exe
Ejecutar el programa de la siguiente manera:
nombre_programa nombre_archivo

C:\Users\La familia\Desktop\Lenguaje C\practicas\P-13>
```

Fwrite

```
C:\Users\La familia\Desktop\Lenguaje C\practicas\P-13\fwrite.c - Notepad++
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins

calculadora.c  abrirCerarArchi.c  gets.c  fputs.c  fscanf.c  fprintf.c  fread.c  fwrite.c

1  #include <stdio.h>
2  /*
3   Este programa realizar una copia exacta de dos archivos. Los
4   nombres de los archivos (origen y destino) se reciben como
5   argumentos de la función principal.
6  */
7
8  int main(int argc, char **argv)
9  {
10     FILE *archEntrada, *archivoSalida;
11     unsigned char buffer[2048]; // Buffer de 2 Kbytes
12     int bytesLeidos;
13
14     // Si no se ejecuta el programa correctamente
15     if(argc < 3)
16     {
17         printf("Ejctuar el programa de la siguiente manera:\n");
18         printf("\tnombre_programa \tarchivo_origen \tarchivo_destino\n");
19         return 1;
20     }
21     // Se abre el archivo de entrada en modo de lectura y binario
22     archEntrada = fopen(argv[1], "rb");
23     if(!archEntrada)
24     {
25         printf("El archivo %s no existe o no se puede abrir", argv[1]);
26         return 1;
27     }
28
29     // Se crea o sobrescribe el archivo de salida en modo binario
30     archivoSalida = fopen(argv[2], "wb");
31     if(!archivoSalida)
32     {
33         printf("El archivo %s no puede ser creado", argv[2]);
34         return 1;
35     }
36
37     // Copia archivos
38     while (bytesLeidos = fread(buffer, 1, 2048, archEntrada))
39         fwrite(buffer, 1, bytesLeidos, archivoSalida);
40
41     // Cerrar archivos
42     fclose(archEntrada);
43     fclose(archivoSalida);
44
45     return 0;
46 }
```

```
Simbolo del sistema
C:\Users\La familia\Desktop\Lenguaje C\practicas\P-13>gcc fwrite.c -o fwrite.exe
C:\Users\La familia\Desktop\Lenguaje C\practicas\P-13>fwrite.exe
Ejctuar el programa de la siguiente manera:
        nombre_programa        archivo_origen  archivo_destino
C:\Users\La familia\Desktop\Lenguaje C\practicas\P-13>
```


CONCLUSIONES

Como vimos, un archivo es un conjunto de datos estructurados en una colección de registros que son del mismo tipo, pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso.

La práctica personalmente me pareció un poco confusa, sobre todo por los resultados de los ejemplos.

REFERENCIAS

Facultad de Ingeniería. (2018, 6 abril). Manual de prácticas de fundamentos de programación MAD0-17_FP. Laboratorio de Computación Salas A y B. <http://lcp02.fi-b.unam.mx/>