



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: M.I. MARCO ANTONIO MARTINEZ QUINTANA

Asignatura: FUNDAMENTOS DE PROGRAMACIÓN

Grupo: 3

No de Práctica(s): 13

Integrante(s): CARRILLO CERVANTES IVETTE ALEJANDRA

*No. de Equipo de
cómputo empleado:* NO APLICA

No. de Lista o Brigada: 7

Semestre: PRIMER SEMESTRE

Fecha de entrega: 20 ENERO 2021

Observaciones:

CALIFICACIÓN: _____

Lectura y escritura de datos

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:

- 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

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.

Código (abrir cerrar archivo)

```
chivos.c -o AbrirCerrarArchivos.exe
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>AbrirCerrarArchivos.exe
El archivo se abrió correctamente.
fclose = 0
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>
```

```
#include<stdio.h>
/*
Este programa permite abrir un archivo en modo de lectura, de ser posible.
*/
int main() {
    FILE *archivo;
    archivo = fopen("archivo.txt", "r");

    if (archivo != NULL) {
        printf("El archivo se abrió correctamente.\n", 162);
        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);

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)

```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc fgets.c -o fgets.exe
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>fgets.exe
El archivo se abrió correctamente.
Contenido del archivo:

Holaaaaa (:
Holaaaaa (:
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>
```

```
#include<stdio.h>
/*
Este programa permite leer el contenido de un archivo, de ser posible, a través de la función fgets.
*/
int main() {
    FILE *archivo;
    char caracteres[50];
    archivo = fopen("gets.txt", "r");

    if (archivo != NULL) {
        printf("El archivo se abrió correctamente.", 162);
        printf("\nContenido del archivo:\n");
        while (feof(archivo) == 0) {
            fgets(caracteres, 50, archivo);
            printf("%s", caracteres);
        }
        fclose(archivo);
    }

    return 0;
}
```

Código (fputs)

```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc fputs.c -o fputs.exe
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>fputs.exe
El archivo se abrió correctamente.
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>
```

```
#include<stdio.h>
/*
Este programa permite escribir una cadena dentro de un archivo, de ser posible, a través de la función fputs.
*/
int main() {
    FILE *archivo;
    char escribir[] = "Escribir cadena en archivo mediante fputs. \n\tFacultad de Ingenieria";
    archivo = fopen("puts.txt", "w+");

    if (archivo != NULL) {
        printf("El archivo se abrió correctamente.\n", 162);
        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 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.

Código (fscanf)

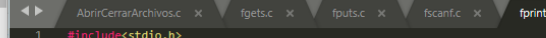
The image shows a Windows development environment. At the top, a terminal window displays the compilation and execution of a C++ program. The command 'gcc fscanf.c -o fscanf.exe' is entered, followed by the execution of 'fscanf.exe', which outputs 'Holaaaaa (: jajaja'. Below the terminal, a Notepad window titled 'fscanf: Bloc de notas' shows the same output. On the right, a code editor displays the source code of 'fscanf.c'. The code includes <stdio.h>, defines a character array 'caracteres' of size 50, opens 'fscanf.txt' in read mode, and uses a while loop with 'fscanf' to read the file's contents and print them. The file 'fscanf.txt' contains the text 'Holaaaaa (: jajaja'.

Código (fprintf)

```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc fprintf.c -o
fprintf.exe

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>fprintf.exe

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>
```



```
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     FILE *archivo;
9     char escribir[] = "Escribir cadena en archivo mediante fprintf. \nFacultad de Ingeniería";
10    if (archivo != NULL) {
11        fprintf(archivo, "%s", "UNAM\n");
12        fclose(archivo);
13    } else {
14        printf("El archivo no existe o no se tiene permisos de lectura / escritura");
15    }
16    return 0;
17 }
```

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)
```

Código (fread)

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc fread.c -o fread.exe

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>fread.exe
nombre_programa nombre_archivo

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>fread.c fread.exe

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>

fread: Bloc de notas

```
#include <stdio.h>

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

int main(int argc, char **argv) {
    FILE *ap;
    unsigned char buffer[2048]; // Buffer de 2 Kbyte
    int bytesLeidos;

    // Si no se ejecuta el programa correctamente
    if(argc < 2) {
```

File Edit Selection Find View Goto Tools Project Preferences Help

AbrirCerrarArchivosc x fgetc x fputc x fscanf x fprintf x fread x

```
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 int main(int argc, char **argv) {
8     FILE *ap;
9     unsigned char buffer[2048]; // Buffer de 2 Kbytes
10    int bytesLeidos;
11
12    // Si no se ejecuta el programa correctamente
13    if(argc < 2) {
14        printf("Ejecutar el programa de la siguiente manera:\n\nnombre_tprograma\n\n");
15        return 1;
16    }
17
18    // Se abre el archivo de entrada en modo lectura y binario
19    ap = fopen(argv[1], "rb");
20    if(!ap) {
21        printf("El archivo %s no existe o no se puede abrir", argv[1]);
22        return 1;
23    }
24
25    while(bytesLeidos = fread(buffer, 1, 2048, ap))
26        printf("%s", buffer);
27
28    fclose(ap);
29    return 0;
30 }
```

Código (fwrite)

```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc fwrite.c -o fwrite.exe
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>fwrite.exe
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>
```

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

    // 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 (bytesLeídos = fread(buffer, 1, 2048, archEntrada))
        fwrite(buffer, 1, bytesLeídos, archivoSalida);

    // Cerrar archivos
    fclose(archEntrada);
    fclose(archivoSalida);
    return 0;
}
```

Actividades

gaussArchivos.c

```
Suma de los primeros n números
¿Cuántos números deseas sumar? 100
La suma de los primeros 100 números es: 5050
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gaussArchivos.exe

Suma de los primeros n números
¿Cuántos números deseas sumar? 10
La suma de los primeros 10 números es: 55
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc gaussArchivos.c -o gaussArchivos.exe
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gaussArchivos.exe

Suma de los primeros n números
¿Cuántos números deseas sumar? 15
La suma de los primeros 15 números es: 120
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>
```

```
#include <stdio.h>
int main ()
{
    //Declarar variables
    char au=163, sp=168, aa=160;
    int n, res;
    //Apuntador a archivo
    FILE *a;
    a=fopen("resultadosGauss.txt", "a");

    //Mensaje de Bienvenida
    printf("\n\n\t\tSuma de los primeros n números\n\n",au);

    //Solicitar el número de elementos a sumar
    printf("%cCuántos números deseas sumar? ",sp,aa,au);
    scanf("%d",&n);

    //Sumar los n números
    res=0;
    for(int i=1; i<=n; i++)
    {
        fprintf(a,"%d+%d=",res,i);
        res=res+i;
        fprintf(a, "%d\n",res);
    }

    /*
     * n = 5
     * i = 5
     * res = 10
     * res = 10 + 5 = 15
     */

    //Mostrar el resultado
    printf("La suma de los primeros %d números es: %d\n",n,au,res);
    fclose(a);

    return 0;
}
```

```
resultadosGauss: Bloc de notas
Archivo Edición Formato Ver Ayuda
0+1=1
1+2=3
3+3=6
6+4=10
10+5=15
15+6=21
21+7=28
28+8=36
36+9=45
45+10=55
0+1=1
1+2=3
3+3=6
6+4=10
10+5=15
15+6=21
21+7=28
28+8=36
36+9=45
45+10=55
55+11=66
66+12=78
78+13=91
91+14=105
105+15=120
```

EJERCICIO 7

- Crear un programa que escriba los pasos del cálculo de la factorial de un número en un archivo llamado factorial.txt

```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc factorialArchivos.c -o factorialArchivos.exe
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>factorialArchivos.exe

Factorial del número n
¿De qué número deseas hacer su factorial? 10
La factorial del número 10 es:
10! = 3628800

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>
```

```
factorial: Bloc de notas
Archivo Edición Formato Ver Ayuda
1!=1
1!=2
2!=6
6!=24
24!=120
120!=720
720!=5040
5040!=40320
40320!=362880
362880!=3628800
```

```
1 #include<stdio.h>
2 int main ()
3 {
4     //Declarar variables
5     char au=163, sp=168, aa=160, ae=130;
6     int n, res;
7     //Apuntador a archivo
8     FILE *a;
9     a=fopen("factorial.txt", "w");
10
11     //Mensaje de Bienvenida
12     printf("\n\n\t\tFactorial del n°cmro n\n\n",au);
13
14     //Solicitar el número del que se desea hacer su factorial
15     printf("%cDe qu°c n°cmro deseas hacer su factorial? ",sp,ae,au);
16     scanf("%d",&n);
17
18     //Factorial de los n números
19     res=1;
20     for(int i=1; i<=n; i++)
21     {
22         fprintf(a,"%d!= ",res,i);
23         res=res*i;
24         fprintf(a,"%d\n",res);
25     }
26
27     //Mostrar el resultado
28     printf("La factorial del n°cmro %d es:\n%d! = %d\n",au,n,n,res);
29     fclose(a);
30
31     return 0;
32 }
```

Conclusiones

Este tema no lo había visto anteriormente, me costo trabajo a un principio entenderlo porque no sabía bien su funcionamiento; sin embargo, al ver el video de Archivos y realizar los ejercicios lo entendí mejor, considero que este tema aun que fue algo corto, es de gran importancia saber cómo leer, crear, escribir y sobrescribir archivos de texto. Me gustó mucho este tema (:

Referencias

- Manual de prácticas del Laboratorio de Fundamentos de programación. 22 Enero 2021, de Facultad de Ingeniería Sitio web: <http://lcp02.fi-b.unam.mx/>