	Lectura y escritura de datos
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): Sánchez García Rocío

No. de Equipo de cómputo empleado: No aplica

No. de Lista: 47

Semestre: 2021-1

Fecha de entrega: Domingo 24 de enero de 2020

Observaciones:

CALIFICACIÓN: _____

Práctica 13

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

Un archivo es un conjunto de datos estructurados en una colección de entidades elementales o básicas denominadas registros que son del mismo tipo, pertenecientes 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*

Desarrollo

Apuntador o archivo

Un apuntado r a un archivo es un hilo común que unifica el sistema de Entrda7Salida (E/S) con un buffer donde se transportan los datos.

Un apuntador a archivo señala a la información que tiene 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 es la siguiente:

```
FILE*F;
```

Abrir archivo

La función *fopen()* abre una secuencia para que pueda ser utilizada y la socia 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 nombra valido del archivo y puede incluir una especificación del directorio. La cadena a la que apunta *modo* determina como 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 de *fopen().* Escribe la información que se encuentre en el buffer al disco y realiza un cierre formal del archivo a nivel 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)

```c
#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");
        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;
}
```

```
C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>gcc abrirCerrar.c -o abrirCerrar.exe

C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>abrirCerrar.exe
Error al abrir el archivo.
El archivo no existe o no se tienen permisos de lectura.

C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>
```

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 especificado. Esta función lee un renglón a la vez

Código (fgets)

```c
#include<stdio.h>
/*
    Este programa permite lee 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.");
        printf("\nContenido del archivo:\n");
        while (feof(archivo) == 0)
        {
            fgets (caracteres, 50, archivo);
            printf("%s", caracteres);
        }
        fclose(archivo);
    }
    return 0;
}
```

Código (fputs)

```c
#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\tFacultadde 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;
}
```

```
C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>gcc fputs.c -o fputs.exe

C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>fputs.exe
Error al abrir el archivo.
El archivo no existe o no se tienen permisos de lectura.

C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>
```

Funciones *fscanf* y *fprintf*

Las funciones *fprintf()* y *scanf()* 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 especificaciones 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)

```c
#include<stdio.h>
/*
    Este programa permite leer el contenido de un archivo,
    de ser posible, a través de la función fscanf.
*/
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;
}
```

```
C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>gcc fscanf.c -o fscanf.exe

C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>fscanf.exe
El archivo no existe.

C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>
```

Código (fprintf)

```c
#include<stdio.h>
/*
    Este programa permite escribir dentro de un archivo,
    de ser posible, a través de la función fprintf.
*/
int main() {
    FILE *archivo;
    char escribir[] = ("Escribir cadena en archivo mediante fprinft. \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;
}
```

```
C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>gcc fprintf.c -o fprintf.exe

C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>fprintf.exe
El archivo no existe o no se tiene permisos de lectura / escritura.

C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>
```

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
#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 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;
}
```

```
C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>gcc fread.c -o fread.exe

C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>fread.exe
Ejecutar el programa de la siguiente manera:
        nombre_ programa nombre_archivo

C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>
```

Código (fwrite)

```c
#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 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;
}
```

```
C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>gcc fwrite.c -o fwrite.exe

C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>fwrite.exe
Ejectuar el programa de la siguiente manera:
        nombre_programa         archivo_origen  archivo_destino

C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>
```

➢ Actividades asignadas por el profesor

Archivos

Para crear un archivo y escribir en él nuestros resultados utilizamos la palabra reservada FILE y ejecutamos las siguientes acciones:

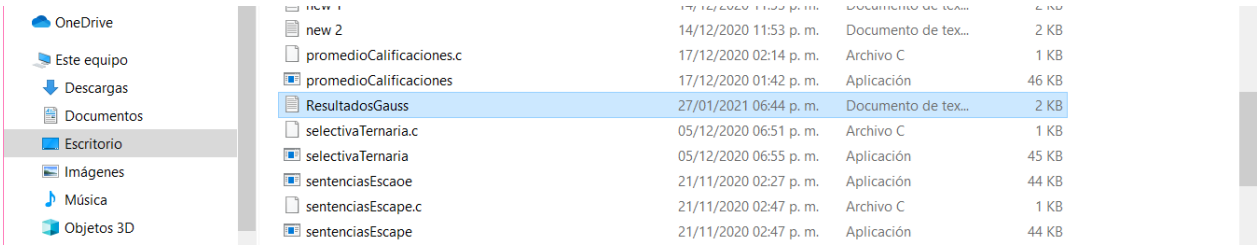
- Crear nuestro apuntador al archivo.
- Abrir nuestro archivo en modo escritura o añadir.
- Escribir en nuestro archivo.
- Cerrar nuestro archivo.

Implementación:

Escribir los resultados de la suma de los primeros n números en un archivo.

```c
#include<stdio.h>
int main()
{
    //Declarar las variables
    char au = 163,sp = 168,aa = 160;
    int n,res;
    //Apuntador a archivo
    FILE *a;
    a=fopen("ResultadosGauss.txt","w");
    //Mensaje de bienvenida
    printf("\n\tSuma de los primeros n n%cmeros\n\n",au);
    //Solicitar el número de elementos a solicitar
    printf("\t%cCu%cntos n%cmeros 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);
    }
    //Mostrar el resultado
    printf("\n\tLa suma de los primeros %d n%Cmeros es: %d \n",n,au,res);
    fclose(a);
    return 0;
}
```

```
C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>gcc gaussArchivo.c -o gaussArchivo.exe

C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>gaussArchivo.exe

        Suma de los primeros n números

        ¿Cuántos números deseas sumar? 100

        La suma de los primeros 100 nmeros es: 5050

C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>
```



ResultadosGauss: Bloc de notas

Archivo Edición Formato Ver Ayuda

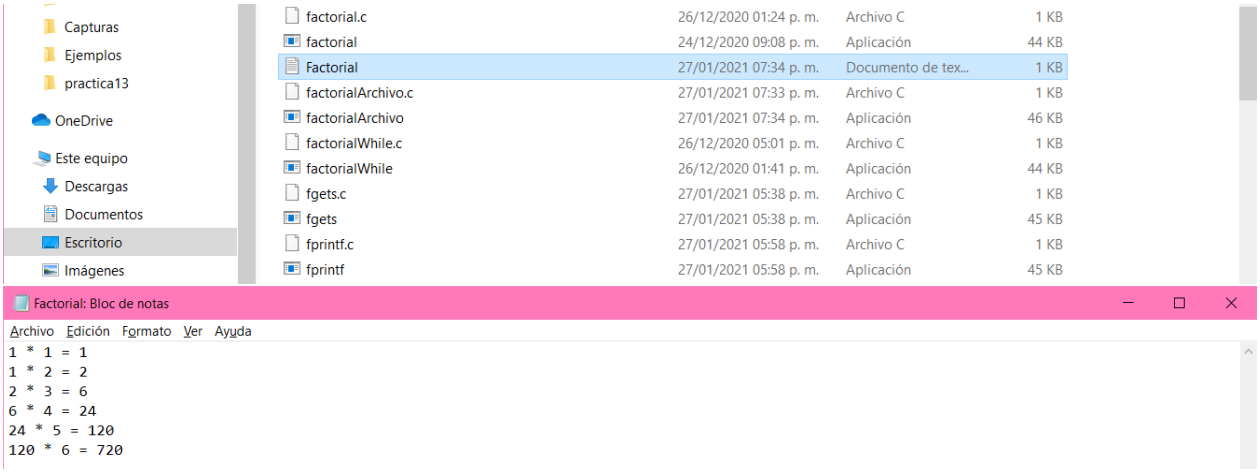
```
0 + 1 = 1
1 + 2 = 3
3 + 3 = 6
6 + 4 = 10
10 + 5 = 15
15 + 6 = 21
21 + 7 = 28
28 + 8 = 36
36 + 9 = 45
45 + 10 = 55
55 + 11 = 66
66 + 12 = 78
78 + 13 = 91
91 + 14 = 105
105 + 15 = 120
120 + 16 = 136
136 + 17 = 153
153 + 18 = 171
171 + 19 = 190
190 + 20 = 210
210 + 21 = 231
231 + 22 = 253
253 + 23 = 276
276 + 24 = 300
300 + 25 = 325
325 + 26 = 351
351 + 27 = 378
378 + 28 = 406
406 + 29 = 435
435 + 30 = 465
465 + 31 = 496
496 + 32 = 528
528 + 33 = 561
561 + 34 = 595
595 + 35 = 630
630 + 36 = 666
666 + 37 = 703
703 + 38 = 741
741 + 39 = 780
780 + 40 = 820
820 + 41 = 861
```

Ejercicio

Crear un programa que escriba los pasos del calculo del factorial de un numero en un archivo llamado factorial.txt

```c
#include<stdio.h>
int main()
{
    //Declarar las variables
    char au = 163;
    int n,fact = 1;
    //Apuntador a archivo
    FILE *a;
    a=fopen("Factorial.txt","w");
    //Mensaje de bienvenida
    printf("\n\tFactorial de un n%cmero\n\n",au);
    //Solicitar un número
    printf("\tDigita el n%cmero: ",au);
    scanf("%d",&n);
    //Multiplicar los números
    fact = 1;
    for(int i=1;i<=n;i++)
    {
        fprintf(a,"%d * %d = ",fact,i);
        fact = fact*i;
        fprintf(a,"%d\n",fact);
    }
    //Mostrar el resultado
    printf("\n\tEl factorial del n%cmero %d es: %d \n",au,n,fact);
    fclose(a);
    return 0;
}
```

```
C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>gcc factorialArchivo.c -o factorialArchivo.exe

C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>factorialArchivo.exe

        Factorial de un número

        Digita el número: 6

        El factorial del número 6 es: 720

C:\Users\rocio\OneDrive\Escritorio\Lenguaje c\Ejemplos>
```



Bibliografía

Manual de prácticas del Laboratorio de Fundamentos de Programación, Facultad de ingeniería UNAM, recuperada el 18 de enero, en http://lcp02.fi-b.unam.mx/