Archivos 2

Búsqueda y modificación en archivos

En la lectura y escritura de archivos el indicador de posición se actualiza automáticamente, pero existen casos, por ejemplo en las búsquedas y modificaciones sobre archivos, en los cuales se necesita mover el indicador de posición a algún lugar en particular. Para ello se cuenta con 2 funciones que permiten realizar tal operación, ellas son **fseek()** y **rewind()**. Por otra parte se cuenta con una función que permite obtener el lugar donde se encuentra el indicador de posición del archivo, esa función es **ftell()**. A continuación se detallan las 3 funciones.

rewind()
Esta función permite llevar el indicador de posición al comienzo del archivo. El prototipo es el siguiente:

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

La función rewind() ubica el indicador de posición del archivo referenciado por el puntero arch al principio y limpia los indicadores de fin de archivo y error que se encuentran el la estructura FILE. Ejemplo:

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

#define ARCH "D:\\bin.dat"

typedef struct {
    char nombre[10];
    int edad;
}stPersona;

int main () {
    FILE * bin = fopen (ARCH, "ab" );
    stPersona a;
    int cant;
    if ((bin != NULL){
        do {
            printf ("\nIngrese el nombre: ");
            fflush (stdin);
            gets(pers.nombre);
            printf ("Ingrese la edad: ");
            scanf ("%d", &pers.edad);
           fflush (stdin);
```

```
            fwrite (&pers, sizeof(stPersona), 1, bin);
            printf ("\n Presione ESC para terminar");
        } while ((getche ())!= 27);
        rewind(bin); //se lleva el indicador de posición al principio para
comenzar a leer
        while(fread (&pers, sizeof(stPersona), 1, bin) > 0){
            printf ("\n %s\t %d", pers.nombre, pers.edad);
        }
        fclose(bin);
    }
    else{
        printf("\n ERROR DE DATOS");
    }
    return 0;
}
```

fseek();

Esta función permite desplazar el indicador de posición del archivo a la posición que se le indique. El prototipo es el siguiente:

```
int fseek( FILE *arch, long desplazamiento, int origen);
```

Donde:

arch	Puntero a la estructura FILE asociada con el archivo
desplazamiento	es la cantidad de bytes que se desplazará el indicador de posición a partir de origen
origen	es una constante que determina el punto de referencia a partir del cual se realiza el desplazamiento

Los valores que se le pueden dar a origen figuran en la siguiente tabla. Dichos valores se encuentran definidos en <stdio.h>

SEEK_SET	**0**	**A partir del comienzo del archivo**
SEEK_CUR	**1**	**A partir de la posición actual del cursor**
SEEK_END	**2**	**A partir del final del archivo**

Valor retornado: Si la operación es exitosa **devuelve cero**, caso contrario retorna un valor **distinto de cero.**

La función **fseek()** mueve el indicador de posición del archivo desplazamiento bytes a partir de la posición indicada por origen. Ejemplos:

```
fseek( ptr, 0, SEEK_SET);
```

Mueve el indicador de posición al comienzo del archivo. El origen es **SEEK_SET** que indica el comienzo del archivo y se desplaza 0 bytes, por lo tanto queda al principio. Es aconsejable que cuando se desee llevar el indicador de posición al comienzo del archivo se utilice **rewind()** ya que fseek() no limpia el indicador de error ni el de fin de archivo, por lo tanto cuando en determinada situación se use fseek() no va a dar los resultados esperados.

```
fseek( ptr, 0, SEEK_END);
```

Mueve el indicador de posición al final del archivo. El origen es SEEK_END que indica el final del archivo y a partir de allí se desplaza 0 bytes, por lo tanto está al final del archivo. Si se desea en algún momento agregar datos, simplemente se debe usar esta función para enviar el indicador de posición al final del archivo.

```
fseek( ptr, 20, SEEK_SET);
```

Mueve el indicador de posición 20 bytes a partir del comienzo del archivo.

```
fseek( ptr, (-1) * sizeof (struct x), SEEK_CUR);
```

Mueve el indicador de posición un registro del tamaño de la estructura, para atrás a partir de la posición actual. Normalmente esta forma se utiliza cuando se están editando datos del archivo. Al realizar una búsqueda se va leyendo cada uno de los datos del archivo por medio de fread(), pero cuando encontramos el dato el indicador de posición del archivo está en el dato siguiente al que queremos modificar, con lo cual al hacer fwrite () para escribirlo se modificará otro dato. Por lo tanto antes de escribir se debe mover el indicador de posición una estructura para atrás.

ftell();

La función ftell() me permite obtener la posición actual del indicador de posición. El prototipo es el siguiente:

```
long ftell (FILE * arch);
```

Donde **arch** es el puntero a la estructura FILE asociada al archivo.

Valor retornado: Si la operación es exitosa devuelve la cantidad de bytes que hay desde el comienzo del archivo hasta el lugar en que se encuentra el indicador de posición del archivo, en caso contrario devuelve: –1. Ejemplo: Obtener el tamaño de un archivo en bytes:

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

#define ARCH "d:\\bin.dat"

int main () {
    FILE * bin = fopen (ARCH, "rb");
    int cant;
    if ((bin != NULL) {
        fseek (bin, 0, SEEK_END); //Se envía el indicador de posición al
final del archivo
        cant = ftell (bin);
        printf ("\n\t El archivo tiene %d bytes", cant);
        fclose (bin);
    }
    else{
        printf("ERROR de datos");
    }
    return 0;
}
```

Terminada la descripción de todas las funciones que se utilizarán sobre archivos, podemos escribir 2 ejemplos simples de búsqueda y modificación.

Búsqueda:
En el archivo **C:\bin.dat** que ya contiene datos cargados se desea buscar las edades de varias personas indicando para la búsqueda el nombre, sin que nos importe la existencia de repeticiones.

```
#include <stdio.h>
#include <string.h>
#include <conio.h>


typedef struct {
    char nombre[10];
    int edad;
}stPersona;
```

```
int main() {
    FILE * bin = fopen ("d:\\bin.dat" , "r + b");
    stPersona pers;
    int cant, cont = 0, hay = 0;
    char auxnom[10], rta = 's';
    if ((bin != NULL) {
        while (rta == 's') {
            printf ("\n\n Ingrese nombre a buscar: ");
            fflush (stdin);
            gets (auxnom);
            while ( fread (&pers, sizeof (stPersona), 1 , bin) > 0) {
                cont += cant;
                if (strcmp (pers.nombre, auxnom) == 0) {
                    printf ("\n\n\t\t%s\t%d", pers.nombre, pers.edad);
                    hay++;
                }
            }
            printf ("\n\t Cantidad de registros inspeccionados = %d",cont);
            if ( !hay){
                printf ("\n\n\t No se encontro ningun %s", auxnom);
            }
            rewind (bin);
            cont = 0;
            hay = 0;
            printf ("\n\n ¿ Desea continuar? ('s' es afirmativo): ");
            fflush(stdin);
            if ((rta = getche ()) != 's'){
                printf ("\n\n\n\t\t... F I N ...");
            }
        }
        fclose (bin);
    }
    return 0;
}
```

La forma de buscar en esencia es la misma que cuando se busca en un vector, la diferencia principal es que no podemos usar un **for** ni definir un vector para guardar los datos del archivo debido a que en el momento de la definición del vector no se sabe la cantidad de elementos (del tipo de la estructura) que contiene el archivo. Al igual que en cualquier búsqueda se debe ingresar el dato a buscar, que en nuestro caso es el nombre. Conocido el nombre se va leyendo una a una los datos de tipo estructura que se encuentran en el archivo (por medio de **fread()**) y se va comparando el nombre ingresado con el que estaba guardado en el archivo. Este proceso se hace hasta que la lectura sea errónea.

Modificación: Básicamente tiene la misma forma que en el ejemplo anterior, con el agregado que cuando se encuentra el dato se modifica en memoria, se cambia el indicador de posición del archivo para luego escribirlo.

```
#include <stdio.h>
#include <string.h>
#include <conio.h>

typedef struct {
    char nombre[10];
    int edad;
}stPersona;

int main() {
    FILE * bin = fopen ("d:\\bin.dat" , "r+b");
    stPersona pers;
    char auxnom[10], rta = 's';
    if ((bin != NULL) {
        while (rta == 's') {
            printf ("\n\n Ingrese nombre a buscar: ");
            fflush (stdin);
            gets (auxnom);
            while ( fread (&pers, sizeof (stPersona), 1 , bin) > 0) {
                if (strcmp (pers.nombre, auxnom) == 0) {
                    printf ("\n\n Ingrese nueva edad: );
                    scanf(“%d”, &pers.edad);
                    fseek(bin, (-1)*sizeof(stPersona), SEEK_CUR);
                    fwrite(&pers, sizeof(stPersona), 1, bin);
                }
            }
            rewind(bin);
            printf ("\n\n ¿ Desea continuar? ('s' es afirmativo): ");
            fflush(stdin);
            rta = getche(); ///es como un scanf
        }
        fclose (bin);
    }
    return 0;
}
```