Capítulo 1

Manipulación de Archivos

En este capítulo presentaremos los conceptos básicos necesarios para manipular archivos de texto. Un archivo es un conjunto de información que se ha almacenado usando algún medio físico para preservar su integridad. ???

Para usar la información almacenada en algún dispositivo físico se asocia el dispositivo con flujo—un flujo es una secuencia formada por los carácteres almacenados en o que se quieren almacenar en algún dsipositivo, junto con un conjunto de operaciones que permiten manipularlos.

1.1. Algunos Sub-programas

FILE * fopen(const char *filename, const char *mode)
fopen abre el archivo de nombre indicado en la modalidad indicada por mode, le asocia a un flujo y devuelve un apuntador a dicho flujo, en caso de error devuelve NULL.

int feof(FILE * flujo)
feof devuelve un valor no-nulo si se está en el fin del archivo, y cero si no.

int fclose(FILE * flujo) :

Para leer o escribir en un archivo se requiere:

1. Declarar un apuntador a un FILE que un tipo definido en la librería stdio.h
2. Abrir el archivo en cuestión usando la función fopen cuya firma es `FILE * fopen( char s[], char t[])` donde s es una cadena que representa el nombre físico del archivo y t es una cadena que indica para qué se usará el archivo
3.

Como primer ejemplo mostraremos un programa que escribe el contenido de un archivo en la pantalla.

Ejemplo 1.1 *Escribir un programa en C que permita mostrar en la pantalla el contenido de un archivo de texto cuyo nombre se sollicita.*

Explicación:

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

int main(){
   FILE  * pe;
   char nameArch[30];
   printf("Dime el nombre del archivo que quieres visualizar: ");
   scanf("%s", nameArch);
   pe = fopen(nameArch, "r");
   if (pe != NULL) {
      while (!feof(pe)) putchar(getc(pe));
   }
   fclose(pe);
   return 0;
}
```

•

Ejercicio 1.1 *Escriba un programa que permita hacer una copia de un archivo.*

Como segundo ejemplo mostraremos cómo leer información de un archivo y escribirla en otro.

Ejemplo 1.2 *Escriba un programa en C que permita leer un archivo de texto, que luce como el ejemplo que se muestra luego, y escriba en otro archivo los mismos datos con dos columnas más—la suma de segunda más la tercera columna, y el promedio de dichas colunas. El formato debe ser como se muestra en el ejemplo. Los nombres de los archivos de entrada y salida debe suministrarlo el usuario.*

Ejemplo de archivo de entrada:

```
Petra 12 18
Carolina 16 16
Jose 20 12
Ana 18 17
Juan 15 16
```

El archivo de salida para el archivo de entrada anterior debe lucir como sigue:

```
Nombre          P1 P2 su mean
Petra           12 18 30 15.00
Carolina        16 16 32 16.00
Jose            20 12 32 16.00
Ana             18 17 35 17.50
Juan            15 16 31 15.50
```

Explicación: Las tareas básicas son: leer los nombres de los archivos de entrada y saliada, abrir dichos archivos, y si se pudieron abrir, mientras no se haya acabado el archivo de entrada, leer una linea en él y escribirla en el archivo de salida. Por último se deben cerrar los archivos.

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

int main(){
   FILE  *pe, *ps;
   char nEnt[30], nSal[30], name[15];
   int n1, n2;
   printf("Amo dime los nombres de los archivos de entrada y de salida: ");
   scanf("%s%s", nEnt, nSal);
   pe = fopen(nEnt, "r");
   ps = fopen(nSal, "w");
   if (pe != NULL && ps != NULL) {
      fprintf(ps, "%-15s%s\n", "Nombre", " P1 P2 su mean" );
      while (!feof(pe)) {
         fscanf(pe, "%s%d%d", name, &n1, &n2);
         fprintf(ps,"%-15s%3d%3d%3d%6.2f\n",name,n1,n2,n1+n2,(float)(n1+n2)/2);
      }
   } else printf("Error abriendo algun archivo!!!");
   fclose(pe);
   fclose(ps);
   return 0;
}
```

•

Ejemplo 1.3 *Escriba un programa en C que permita leer de un archivo cuyo nombre lo sumisnistra el usuario los datos varios estudiantes, y escribirlos en otro archivo, cuyo nombre también debe suministrar el usuario, los datos ordenados de mayor a menor en base a la suma de la segunda y la tercera columna. Para la entrada del ejercicio anterior la salida debería ser:*

```
Nombre          P1 P2 su mean
Ana             18 17 35 17.50
```

```
Jose            20 12 32 16.00
Carolina        16 16 32 16.00
Juan            15 16 31 15.50
Petra           12 18 30 15.00
```

Explicación: A diferencia del ejemplo anterior, no podemos leer los datos en variables sencillas sino que tenemos que definir una estructura para almacenar los datos que llamaremos **Estudiante**. Estudiante es un nuevo tipo de datos. En el *main* declaramos un arreglo de **Estudiantes** en el que leeremos los datos, luego lo ordenaremos, y finalmente lo escribiremos en otro archivo.

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

struct est {
   char nom[15];
   int n1, n2, s;
};

typedef struct est Estudiante;


int main(){
   FILE  *pe, *ps;
   char nEnt[30], nSal[30];
   Estudiante e[30], aux;
   int n = 0, i, j;
   printf("Amo dime los nombres de los archivos de entrada y de salida: ");
   scanf("%s%s", nEnt, nSal);
   pe = fopen(nEnt, "r");
   ps = fopen(nSal, "w");
   if (pe != NULL && ps != NULL) {
      fprintf(ps, "%-15s%s\n", "Nombre", " P1 P2 su mean" );
      while (!feof(pe)) {
         fscanf(pe, "%s%d%d", e[n].nom, &e[n].n1, &e[n].n2);
         e[n].s = e[n].n1 + e[n].n2;
         n++;
      }
      for (i = 0; i < n-1; i++)
         for (j = i+1; j < n; j++)
            if (e[j].s > e[i].s) {
               aux = e[j]; e[j] = e[i]; e[i] = aux;
            }
      for (i = 0; i< n; i++)
       fprintf(ps,"%-15s%3d%3d%3d%6.2f\n",e[i].nom,e[i].n1,e[i].n2,e[i].s,(float)e[
```

```
   } else printf("Error abriendo algun archivo!!!");
   fclose(pe);
   fclose(ps);
   return 0;
}
```

•

En el siguiente ejemplo mostraremos cómo ordenar un arreglo de estructuras en base un capo que sea una cadena de carácteres. Además de cómo separar el programa en sub-programas: un para cada sub-tarea.

Ejemplo 1.4 *Escriba un programa en C que permita leer un archivo de texto, que luce como el ejemplo que se muestra luego, en una estructura de datos para luego ordenarlos en base a los apellidos y volverlo a escribir sobre otro archivo de texto. Los nombres de los archivos de entrada y de salida deben perdirsele al usuario. Debe definirse un registro adecuado, y declararse un tipo de datos concreto para almacenar en memoria principal los registros leidos del archivo de entrada. Todas las sub-tareas deben ejecutarse usando sub-programas adecuados, de tal manera que en el programa principal sólo se tenga que declarar las estructuras de datos necesarias e invocar a los sub-programas adecuados.*

Explicación: Hay tres tareas básicas que ejecutar; ellas son:

- Leer el archivo de entrada
- Ordenar los datos
- Escribir los datos ordenados en el archivo de salida

Ellas se van a ejecutar usando tres sub-programas, pero antes hay que definir la estructuta de dats que usaremos. Necesitamos definir un **struct** con los capos *nom, ape, not* por nombre, apellido y nota. Note que se incluye la librería *string.h*. Note también que el programa principal sólo declara como variables un arreglo de Items y un entero n en el que se almacenará el tamaño del archivo, e invoca a los sub-programas. A continuación se muestra el código. Analicelo cuidadosamente.

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

typedef struct {
   char nom[12], ape[12];
   int not;
} Item;

void leeArc(Item e[], int *n){
```

```
   FILE *pe;
   char nomA[20];
   printf("Archivo de entrada: ");
   scanf("%s", nomA);
   pe = fopen(nomA, "r");
   if (pe != NULL){
      *n = 0;
      while (!feof(pe)) {
          fscanf(pe, "%s%s%d", e[*n].nom, e[*n].ape, &e[*n].not);
          (*n)++;
      }
   } else printf("Error abriendo el archivo %s!!!", nomA);
   fclose(pe);
}

void ordena(Item e[], int n){
   int i,j;
   Item aux;
   for (i = 0; i < n-1; i++){
      int m = i;
      for (j = i + 1; j < n; j++)
         if (strcmp(e[j].ape, e[m].ape) < 0) m = j;
      if (m != i) {aux = e[i]; e[i] = e[m]; e[m] = aux;}
   }
}

Item max(Item e[], int n){
   Item m = e[0];
   int k;
   for (k = 1; k < n; k++) if (e[k].not < m.not) m = e[k];
   return m;
}

void escribeArc(Item e[], int n){
   FILE *ps;
   char nomA[20];
   printf("\nArchivo de salida: ");
   scanf("%s", nomA);
   ps = fopen(nomA, "w");
   if (ps != NULL){
      int k;
      for (k = 0; k < n; k++)
          fprintf(ps, "%-12s%-12s%3d\n", e[k].ape, e[k].nom, e[k].not);
   } else printf("Error abriendo el archivo %s!!!", nomA);
   fclose(ps);
}
```

```
int main(){
   Item e[30];
   int n;
   leeArc(e,&n);
   ordena(e,n);
   escribeArc(e,n);

   printf("Mision cumplida... revisa el archivo de salida.\n");
   return 0;
}
```

•