Laboratorio de Programación

Lenguaje C : Manejo de Archivos

Luis Garreta
luis.garreta@javerianacali.edu.co

Ingeniería de Sistemas y Computación
IlPontificia Universidad Javeriana – Cali

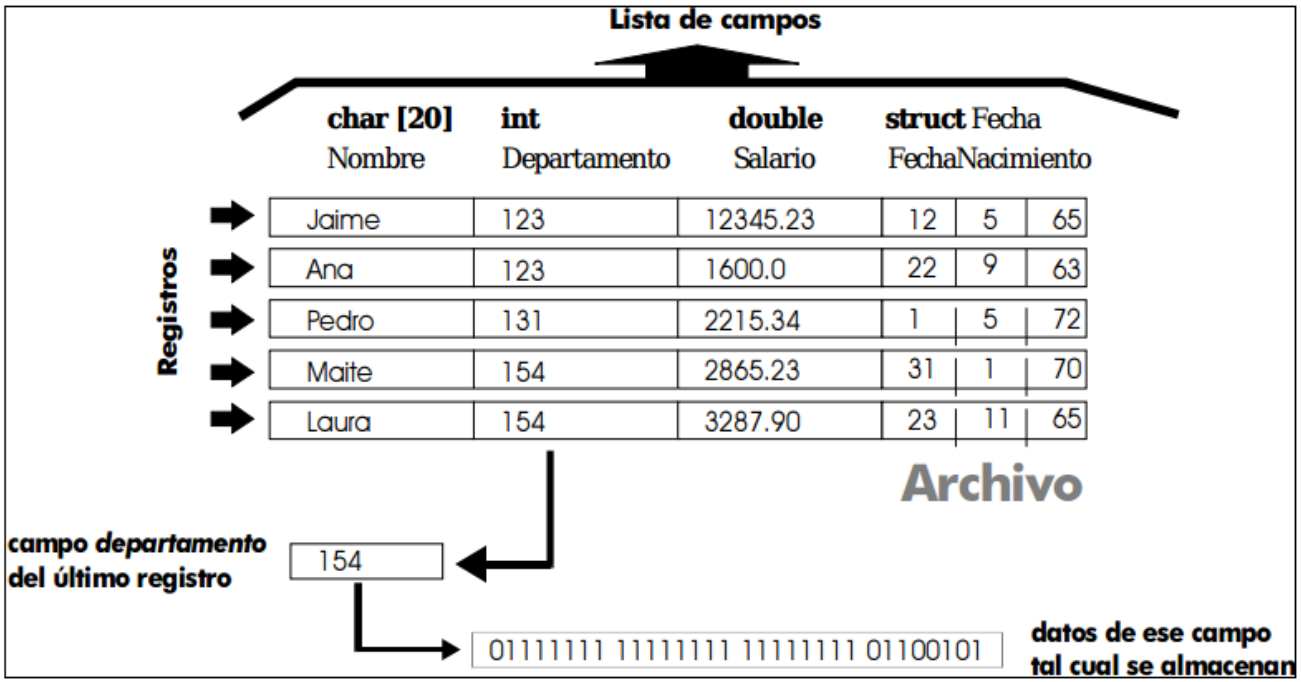
3 de abril de 2017

Archivos

- El problema de los datos en un programa, es **estos se pierden cuando el programa termina.**
- Muchas veces se desean datos que no desaparezcan cuando el programa finaliza.
- Un archivo no es más que una corriente (también llamada stream) de bits o bytes con un terminador de archivo.
- En C un archivo puede ser cualquier cosa, desde un archivo de disco a un terminal o una impresora.
- Se puede asociar un flujo a un archivo mediante una operación de apertura del archivo.

Jerarquia de los Datos

- Al final los datos se guardan en binario pero se representan en distintos tipos: int, float, arreglos, structs, etc.
- Los archivos pueden verse como una colección de **Registros**,
- Donde cada registro tiene la misma estructura y tamanho.
- Entonces los archivos no son más que estructuras secuenciales de datos en un medio externo.



Clasificación de los archivos: Por Contenido

- **Archivos Texto:**
  - Almacenan caracteres y por lo tanto son legibles
  - La información se almacena en líneas
  - Cada línea tiene un terminador de línea (\r\n)
  - Algunos codificaciones:
    - ASCII, ISO-8859-1, Unicode
  - Ejemplos:
    - Archivos de .txt, .py, .c, .xml.
- **Archivos Binarios:**
  - La información está en bits y su contenido no es legible o interpretable
  - Ejemplos:
    - de Imágenes .bmp, tiff, jpg, etc.
    - de sonido: .mp3, ogg, etc.
    - Ejecutables: .exe, .com, .bin
    - Librerías: .lib, .so

Clasificación de los archivos: Por Acceso

Según la forma en la que accedamos a los archivos:

Archivos secuenciales.

- Se trata de archivos en los que el contenido se lee o escribe de forma continua.
- No se accede a un punto concreto del archivo,
- Para leer cualquier información necesitamos leer todos los datos hasta llegar a dicha información.
- Los archivos de texto se utilizan de forma secuencial.

Archivos de acceso directo:

- Se puede acceder a cualquier dato del archivo conociendo su posición en el mismo.
- Los archivos binarios se utilizan mediante acceso directo.

Cualquier archivo en C puede ser accedido de forma secuencial o usando acceso directo.

Estructura FILE y punteros a archivos

- **stdio.h** define una estructura llamada **FILE**
- FILE es la estructura que representa a los archivos
- Un programa requiere tener un puntero de tipo *FILE a cada archivo que se desee leer o escribir
- A este puntero se le llama puntero de archivos.
- Declaración:

```
1 File *archivoX:
```

Operaciones Básicas sobre Archivos

- Abrir
- Leer
- Escribir
- Cerrar
- Fin de Archivo

Apertura y cierre de archivos (I)

```
1 FILE *fopen(const char *nombreArchivo, const char *modo
```

Modo	Significado
“r”	Abre un archivo para lectura de archivo de textos (el archivo tiene que existir)
“w”	Crea un archivo de escritura de archivo de textos. Si el archivo ya existe se borra el contenido que posee.
“a”	Abre un archivo para adición de datos de archivo de textos “
”rb”	Abre un archivo para lectura de archivos binarios (el archivo tiene que existir)
“wb”	Crea un archivo para escritura de archivos binarios (si ya existe, se descarta el contenido)
“ab”	Abre un archivo para añadir datos en archivos binarios

Apertura y cierre de archivos (II)

```
1 FILE *fopen(const char *nombreArchivo, const char *modo
```

Modo	Significado
"r+"	Abre un archivo de texto para lectura/escritura en archivos de texto. El archivo tiene que existir
"w+"	Crea un archivo de texto para lectura/escritura en archivos de texto. Si el archivo tenía datos, estos se descartan en la apertura.
"a+"	Crea o abre un archivo de texto para lectura/escritura. Los datos se escriben al final.
"r+b"	Abre un archivo binario para lectura/escritura en archivos de texto
"w+b"	Crea un archivo binario para lectura/escritura en archivos de texto. Si el archivo tiene datos, éstos se pierden.
"a+b"	Crea o abre un archivo binario para lectura/escritura. La escritura se hace al final de el archivo.

Manejo de Archivos Texto

- Abrir : **fopen**
- Cerrar: **fclose**
- Leer: **fgetc, fgets, fscanf**
- Escribir: **fputc, fputs, fprintf**
- Fin de Archivo: **feof**

Prototipos de la Principales Funciones

```
1 FILE *fopen(const char *nombreArchivo, const char *modo)
2 int fclose(FILE *pArchivo);
3 char *fgets(char *texto, int longitud, FILE *pArchivo)
4 int fputs(const char texto, FILE *pArchivo)
```

Archivos Binarios

Funciones Básicas

fread:

- Utilizamos "fread" para leer un bloque de datos
- size_t fread (void * ptr, size_t size, size_t count, FILE * stream);
- numDatosLeidos = **fread** (apuntdorBuffer, tamañoDeCadaDato, cuantosDatos, fichero);
- Si la "numDatosoLeidos" es menor "cuantosDatos", entonces se llegó al final del archivo

fwrite:

- Para guardar un bloque de datos
- size_t fwrite(void *puntero, size_t tamano, size_t cantidad, FILE *archivo);
- cantidadEscrita = **fread** (apuntadorBuffer, tamañoDeCadaDato, cuantosDatos, fichero);

fseek:

- Para acceder directamente a cualquier posición del archivo
- int fseek(FILE *fichero, long posicion, int valorDesde);
- valorDesde: toma los siguientes valores (constantes):
  - SEEK_SET (0): Ubicarse en la posición "pos" desde el inicio
  - SEEK_CUR (1): Ubicarse en la posición "pos" desde la posición actual
  - SEEK_END (2): Ubicarse en la posición "pos" desde la posición final

Ejemplo:Diccionario con Estructuras