C

Manejo de Archivos

Ejemplo de uso

```
#include <stdio.h>

int main( void )
{
        FILE* fp;
        int caracter;

        if      ( (fp=fopen("test.cpp","r"))==NULL )
        {
                printf("ERROR\n");
                exit(1);
        }

        fseek(fp,0,SEEK_END);
        printf("Tamaño del archivo : %d bytes.\n",ftell(fp));

        rewind(fp);

        while( (caracter=fgetc(fp))!=EOF )
                putchar(caracter);

        fclose(fp);
        return 0;
}
```

FILE (“... FILE* fp;...”)

Estructura (struct) que contiene datos como:

- La dirección de un buffer con el contenido del archivo
- La posición en ese buffer
- Errores que se pudieran haber producido en la lectura y/o escritura.

```
typedef struct _iobuf
{
        char    * _ptr;
        int     _cnt;
        char    * _base;
        int     _flag;
        int     _file;
        int     _charbuf;
        int     _bufsiz;
        char    * _tmpfname;
} FILE;
```

fopen (“...fopen("test.cpp","r")...”)

Parámetros:

- Path al archivo (depende de la plataforma)
- Modo de Apertura del archivo

Modificador	Modo
r	**Lectura**. El archivo **debe existir**.
w	**Escritura** (**crea** el archivos si no existe; **sobreescribe** uno existente)
a	**Append** (**crea** el archivo si no existe. Si existe **continúa al final**)
r+	L**ectura y escritura**, empieza **al principio**. El archivo **debe existir.**
w+	**Lectura y escritura** (**sobreescribe** el archivo si existe)
a+	**Lectura y escritura** (hace append si existe el archivo)
b	Para lectura escritura de archivos binarios (usar con alguno de los anteriores)
t	Para lectura escritura de archivos en modo texto (la contrapartida del “b”)

Otras Funciones:

- int fgetc(FILE *stream);
- int fputc(int char, FILE *stream):
- char *fgets(char *str, int n, FILE *stream):
- int fputs(const char *str, FILE *stream):
- size_t fread(void *ptr, size_t tam_elemento, size_t cant_elem, FILE *stream):
- size_t fwrite(const void *ptr, size_t tam_elemento, size_t cant_elem, FILE *stream):
- int fseek(FILE *stream, long int offset, int desde):
  - SEEK_SET: Inicio
  - SEEK_CUR: Posición actual
  - SEEK_END: Fin
- void rewind(FILE *stream):
- long int ftell(FILE *stream);
- int fclose(FILE *stream);
- int feof(FILE *stream);
- ftruncate(FILE *stream, long int offset);

Ejercicio - Enunciado:

Se cuenta con un **archivo que contiene los datos de empleados** de la compañía, se quiere hacer una aplicación que **lea desde el disco** los mismos, **y que muestre por pantalla todos los nombres** de los empleados y luego el **promedio de edad** de todos ellos.

La estructura de registro del archivo es la siguiente :

- Nombre 30 caracteres
- Apellido 30 caracteres
- edad short unsigned
- sexo char
- domicilio 30 caracteres
- dni int unsigned

Nota: Se recibe **por línea de comandos el nombre del archivo** a procesar.

Ejercicio - Resolución 1/3:

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

typedef struct datos_personales
{
    char            nombre[30];
    char            apellido[30];
    unsigned short  edad;
    char            sexo;
    char            domicilio[30];
    unsigned int    dni;
} dpersonales;
```

Ejercicio:

```
int main(int argc, char** argv)
{
    char            filename[40];
    FILE            *fsource;
    dpersonales     registro;
    unsigned short SumaEdad, TotalRec;

    SumaEdad = TotalRec = 0;
    strcpy(filename, argv[1]); /* Restriccion: filename <= 39 caracteres */

    fsource = fopen(filename, "r+b"); /* Lectura binaria */

    if (fsource == NULL)
        return -2; /* No se pudo abrir archivo */
```

Ejercicio:

```
    while ( !feof(fsource) )
    {
        if (fread((void*)&registro, sizeof(registro), 1, fsource))
        {
            SumaEdad += registro.edad;
            ++TotalRec;
            printf("%s %s \r\n", registro.nombre, registro.apellido);
        }
    }
    printf("El promedio de edades es : %i", SumaEdad / TotalRec);

    fclose(fsource); /* Liberamos los recursos */

    return 0;
}
```