ARCHIVOS EN C++ = FLUJOS DE ENTRADA/SALIDA.

Ejemplo: cin y cout

Tipos de ficheros: **DE TEXTO y BINARIOS**

Biblioteca de gestión de ficheros (a incluir en los programas)

```
#include <fstream.h>
```

Apertura de ficheros (1) : Ficheros de entrada o salida.

(A) **Como fichero de entrada:** Para ello empleamos la sentencia

```
ifstream descriptor ("nombre.extensión");
```

(B) **Como fichero de salida:** Para ello empleamos la sentencia

```
ofstream descriptor ("nombre.extensión");
```

Comentario: EMPLEAR EL DESCRIPTOR A PARTIR DE AQUÍ.

Apertura de ficheros (2) : Ficheros de entrada o salida.

PRIMERO: declaramos la dirección del flujo de entrada

```
ifstream descriptor;        // Para ficheros de entrada

ofstream descriptor;        // Para ficheros de salida
```

SEGUNDO: Asociamos el descriptor de ficheros al fichero en sí.

```
descriptor.open("nombre.extensión",int modo);
```

Donde modo puede ser una combinación (empleando |) de lo siguiente:

```
ios:: in     // Modo entrada
ios:: out    // Modo salida (por defecto)
ios:: app    // Modo añadir, o sea, posicionar el cursor del fichero (ver abajo)
                  // al final del fichero antes de cada escritura
ios:: binary      // El archivo se abre en modo binario
ios:: ate         // El archivo se abre y el cursor se posiciona al final
ios:: nocreate    //  Genera un error si el fichero no existe
ios:: noreplace   //  Genera un error si el fichero existe ya
```

Apertura de ficheros (3) : Combinando ambas formas

```
ifstream descriptor("nombre.extensión",int modo);        // para entrada

ofstream descriptor("nombre.extensión",int modo);        // para salida
```

Ejemplo :

Las dos líneas siguientes abren el fichero “mio.txt” como fichero de entrada (para lectura) y lo asocian al descriptor *in*.

```
ifstream in;                // descriptor del fichero a abrir
in.open(“mio.txt”);     // Apertura del fichero;
```

Esas dos líneas equivalen a la siguiente:

```
ifstream in (“mio.txt”);     // Apertura del fichero;
```

Ejemplo :

Para abrir el fichero “salida.dat” en modo modo salida (si el fichero no existe lo crea, y si existe borra su contenido) asociándolo al descriptor *out* podemos usar la siguiente sentencia;

```
ofstream out("salida.txt");
```

o la siguiente

```
ofstream out("salida.txt", ios::out);
```

o también

```
ofstream out;
out.open("salida.txt");
```

Ficheros de entrada/ salida (1): Declaración-apertura

```
fstream descriptor;
descriptor.open("nombrefichero.ext", ios::in | ios::out)
```

Ficheros de entrada/ salida (1): Declaración-apertura

```
fstream descriptor("nombre.extensión",ios::in | ios:: out);        // para entrada-salida
```

Comprobación de apertura correcta.

```
if (descriptor){
        \\Buen estado. Continuamos
        \\ operando  sobre el fichero
        .....
}
```

```
if (! descriptor){
        \\ Mal estado. Escribimos un mensaje
        \\ a la salida estándar
   cout << "Error en la apertura "
   .....
}
```

Ejemplo:

```
ifstream in("F1.dat");
if (!in)
{
  cout << "\n Incapaz de crear este o abrir el fichero ";   // salida estándar
  cout << " para salida " << endl;
}
else
{
    .... // Se opera sobre el fichero

  }
```

Cierre de ficheros.

```
descriptor.close()
```

Detección de fin de fichero y otras funciones.

descriptor.función();

Donde ***función*** es una de las siguientes:

- La función ***eof()*** que devuelve un valor distinto de cero si se ha alcanzado el final del fichero.

  LECTURA ADELANTADA!!

- La función ***fail().***
- La función ***good()***

LECTURA-ESCRITURA EN FICHEROS DE TEXTO: con << y >>.

Comentario:
El operador << omite los espacios en blanco.

Ejemplo :

El siguiente programa escribe tres líneas en un fichero llamado "EJEMPLO5.TXT" que se crea en el programa (si ya existe borramos su contenido). Cada línea consta de un entero, un real y una cadena de caracteres. Los datos en cada línea están separados por espacios en blanco.

```
#include <fstream.h>   // Biblioteca para el manejo de ficheros
#include <iostream.h> // Bibliooteca para la entrada-salida estándar

int main(){
  ofstream fichout("EJEMPLO5.TXT",ios::out);
  if (!fichout)
    cout << "\n Incapaz de crear este o abrir el fichero \n";
  else {
    fichout << 1 << " " << 5.0 << " APROBADO" << endl;
    fichout << 2 << " " << 1.1 << " SUSPENSO" << endl;
    fichout << 3 << " " << 8.0 << " NOTABLE " << endl;
    fichout.close();
   }
}  // Fin del main
```

Ejemplo :

El siguiente programa lee el fichero de texto llamado “EJEMPLO5.TXT” y visualiza su contenido en el monitor.

```
#include <fstream.h>   // Libreria para el manejo de ficheros
#include <iostream.h>
typedef char TCadena[30];

int main(){
  int i;
  char c;
  float r;
  TCadena cad;

  ifstream fichin("EJEMPLO5.TXT");   // declaracion y apertura del fichero
  if (!fichin)
    cout << "\n Incapaz de crear este o abrir el fichero ";
  else{
    fichin >> i;              // Observese la lectura adelantada!!!
    while (!fichin.eof()){
      cout << i << " ";
      fichin >> r;
      cout << r << " ";
      fichin >> cad;
      cout << cad << "\n";
      fichin >> i;
    }
    fichin.close();
  }  // Fin del main
```

FICHEROS BINARIOS.

Lectura byte a byte.

descriptor.get(ch);

Escritura byte a byte.

descriptor.put(ch);

El siguiente programa escribe un texto (byte a byte) en el fichero "Ejemplo8.dat".

```
#include <fstream.h>
#include <iostream.h>

int main(){
char  cad[17]="TEXTO A ESCRIBIR";
ofstream fichout("Ejemplo8.dat", ios::out | ios::binary);
if (!fichout)
  cout << "\n Incapaz de Crear o Abrir el fichero ";
else{
  for (int i=0;i<=16;i++)
    fichout.put(cad[i]);              // Escritura en el fichero

  fichout.close();
  }
}  // Fin del main
```

FICHEROS BINARIOS.

Escritura por bloque de bytes.

descriptor.read(&c, num);

Escritura por bloque de bytes.

Descriptor.write(&c, num);

Recomendación: Usar sizeof() y pasar c por referencia

Acceso aleatorio a ficheros.

- ***seekg(pos)*** y ***seekp(pos)*** mueven la posición del cursor del fichero a la posición relativa del fichero indicada por *pos*, donde pos es un entero (o variable conteniendo un entero).
- ***tellg()*** y ***tellp()*** devuelven la posición relativa actual del cursor asociado al fichero de entrada y devuelve un –1 en caso de existir algún error.

Ficheros pasados como argumentos: POR REFERENCIA!!.

El siguiente programa declara el fichero “F.dat” para entrada-salida, graba en dicho fichero el valor 1234.86 en binario y después los veinte primeros enteros. Posteriormente, lee el fichero visualizando su información en la salida estándar (el monitor).

```
#include <fstream.h>
#include <iostream.h>

int main(){
  float R=1234.86;
  int i,N;

  fstream fichbin("F.dat",ios::binary | ios::out);    // Apertura como salida
  fichbin.write(&R,sizeof(float));
  for (i=1;i<=20;i++)
    fichbin.write(&i,sizeof(int));

  fichbin.close();

  fichbin.open("F.dat",ios::binary | ios::in);        // Apertura como entrada
  fichbin.read(&R,sizeof(float));
  cout <<endl << "R= " << R << endl;
  for (i=1;i<=20;i++){
     fichbin.read(&N,sizeof(float));
     cout <<endl << i << "= " << N << endl;
  }
} // Fin del main
```

El siguiente programa almacena en un fichero los 10 primeros enteros, luego muestra por pantalla el quinto entero (o sea el 5), posteriormente lo reemplaza por el valor 100, y al final visualiza en el monitor el contenido del fichero.

```
#include <fstream.h>
#include <iostream.h>

int main(){
  int i,N;

  fstream fichbin("ejemplo11.dat",ios::binary | ios::in | ios::out);
  for (i=1;i<=10;i++)
     fichbin.write(&i,sizeof(int));         // Almacena los 10 primeros enteros

  fichbin.seekp(4*sizeof(int));  // se posiciona al principio del quinto/ entero

  fichbin.read(&N,sizeof(float));    // Lee dicho entero
  cout <<endl << "Quinto= " << N << endl;  // visualiza el valor 5

  fichbin.seekp(4*sizeof(int));  // se posiciona de nuevo en el quinto entero
                                 // pues el cursor había avanzado.
  i=100;
  fichbin.write(&i,sizeof(int));   // Modifica el valor 5 por el  valor 100;

  fichbin.seekp(0*sizeof(int));  // se posiciona de nuevo al principio del fichero
  for (i=1;i<=10;i++){
     fichbin.read(&N,sizeof(float));
     cout <<endl << i << "= " << N << endl;        // Se visualiza el contenido
  }
  fichbin.close();
}  // Fin del main
```