

# Tècniques de programació

---

Juan Mendoza Bravo  
juanmenbra@yahoo.es

# C++

## Unitat 10

### Entrada i sortida per arxiu

## Introducció

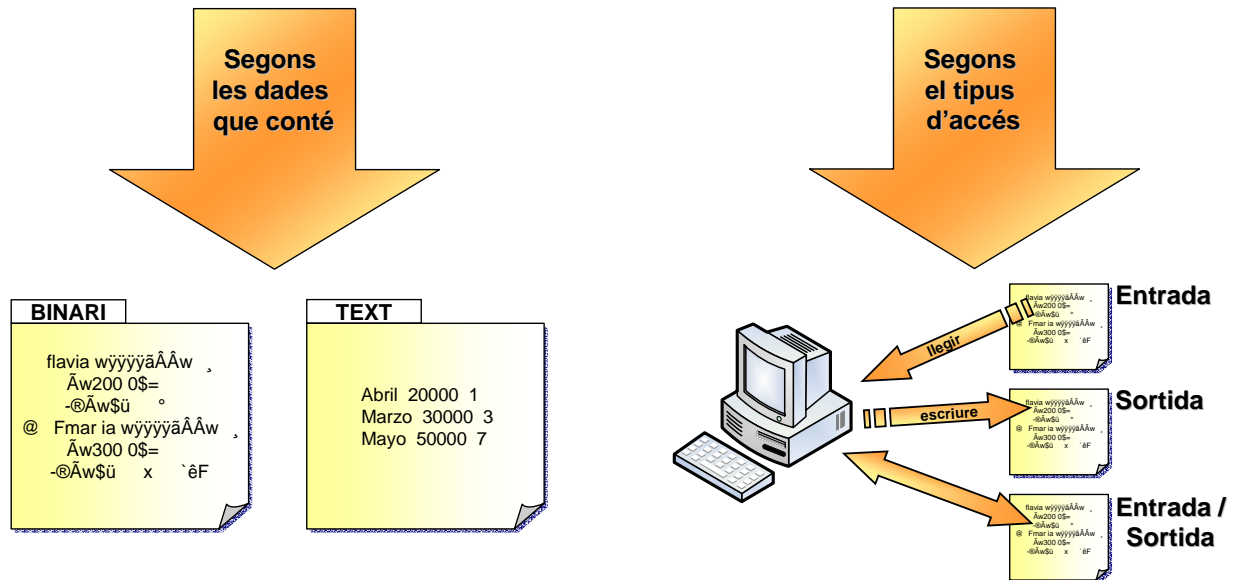
---

- En la majoria d'aplicacions de mitjana entitat necessitem que la informació s'emmagatzemi de forma persistent.
- Moltes aplicacions treballen amb gran quantitat d'informació que pot omplir la memòria principal.

**Aquestes necessitats les resollem  
utilitzant els arxius.**

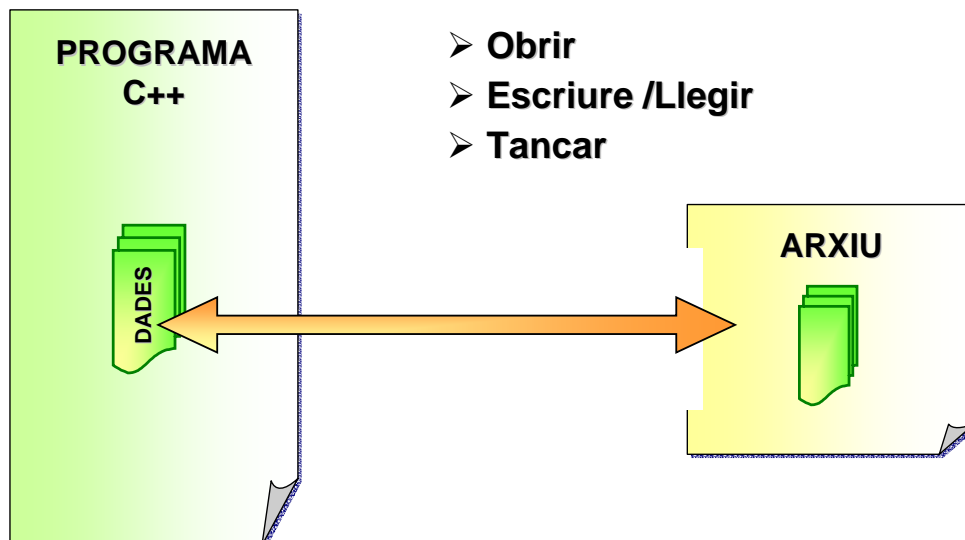
## Introducció

### TIPUS D'ARXIS



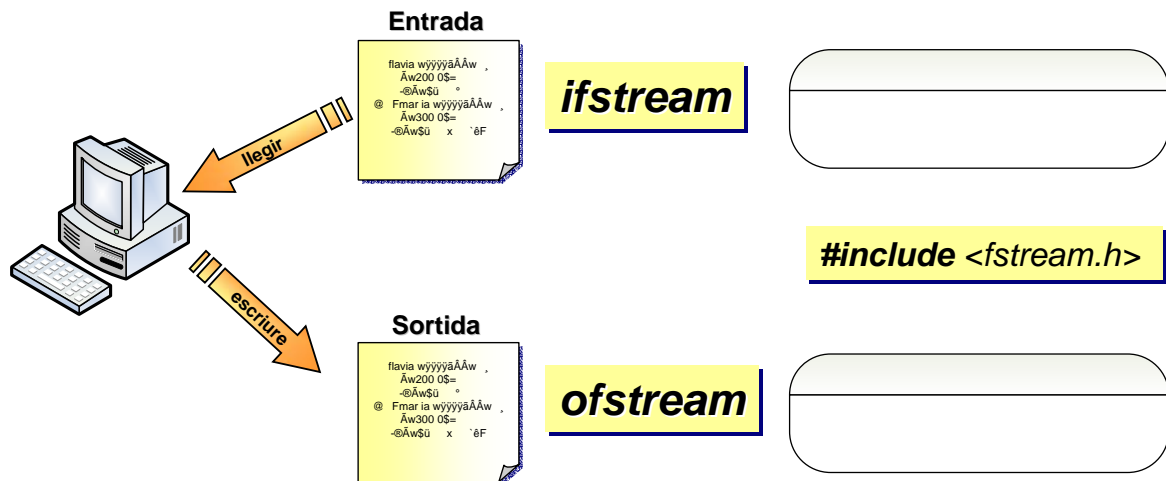
## Introducció

### OPERACIONS BÀSIQUES DELS ARXIS



## Declaració d'un arxiu

Com qualsevol variable, la variable arxiu ha de ser declarada.



## Obertura d'un arxiu de text

Amb la declaració de l'arxiu no és suficient.  
Es necessiten dos paràmetres.

**`.open( "nomArxiu.ext" , tipustreball )`**

### Obertura d'un arxiu de text entrada

```
#include <fstream.h>
...
ifstream arxiu;

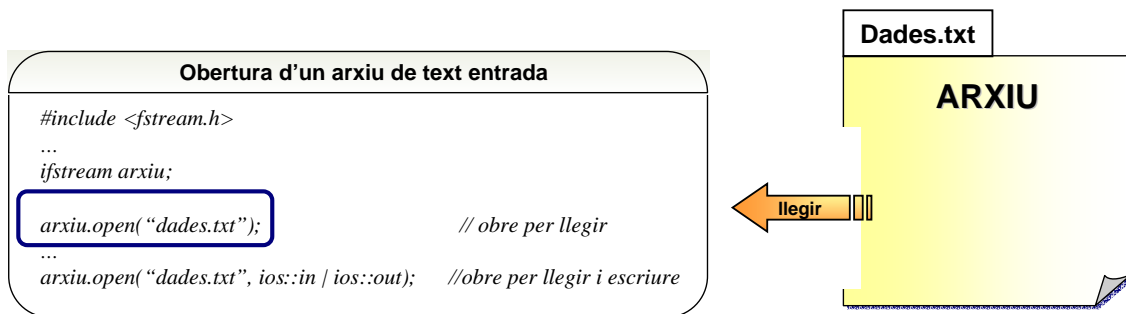
arxiu.open("dades.txt");           // obre per llegir
...
arxiu.open("dades.txt", ios::in | ios::out); // obre per llegir i escriure
```

ARXIU

## Obertura d'un arxiu de text

Amb la declaració de l'arxiu no és suficient.  
Es necessiten dos paràmetres.

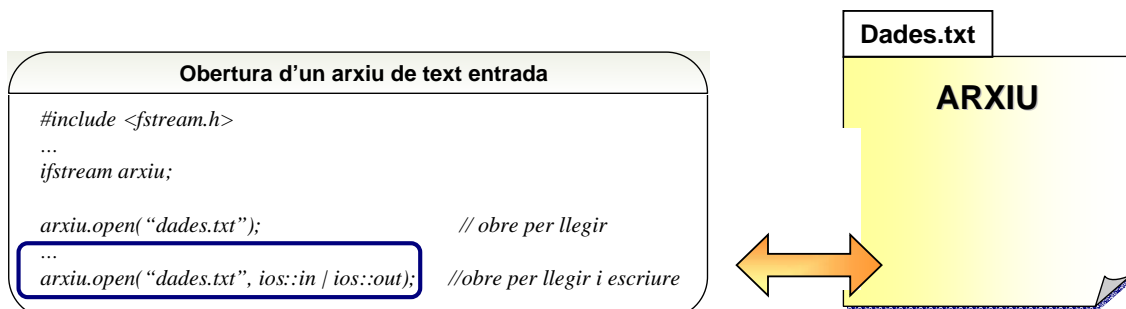
**`.open( "nomArxiu.ext" , tipustreball )`**



## Obertura d'un arxiu de text

Amb la declaració de l'arxiu no és suficient.  
Es necessiten dos paràmetres.

**`.open( "nomArxiu.ext" , tipustreball )`**



## Obertura d'un arxiu de text

Amb la declaració de l'arxiu no és suficient.  
Es necessiten dos paràmetres.

**.open( "nomArxiu.ext" , tipustreball )**

### Obertura d'un arxiu de text sortida

```
#include <fstream.h>
...
ofstream arxiu;
arxiu.open( "dades.txt" );           // obre per escriure i esborra
                                     el contingut si estava creat
arxiu.open( "dades.txt", ios::app);  // obre per afegir dades al final
arxiu.open( "dades.txt", ios::in | ios::out); // obre per llegir i escriure
```

ARXIU

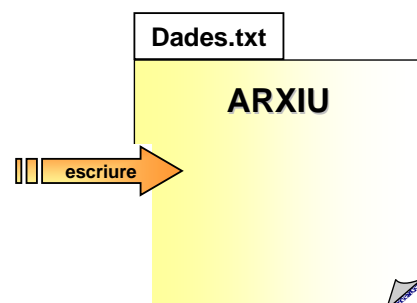
## Obertura d'un arxiu de text

Amb la declaració de l'arxiu no és suficient.  
Es necessiten dos paràmetres.

**.open( "nomArxiu.ext" , tipustreball )**

### Obertura d'un arxiu de text sortida

```
#include <fstream.h>
...
ofstream arxiu;
arxiu.open( "dades.txt" );           // obre per escriure i esborra
                                     el contingut si estava creat
arxiu.open( "dades.txt", ios::app);  // obre per afegir dades al final
arxiu.open( "dades.txt", ios::in | ios::out); // obre per llegir i escriure
```



## Obertura d'un arxiu de text

Amb la declaració de l'arxiu no és suficient.  
Es necessiten dos paràmetres.

**`.open( "nomArxiu.ext" , tipustreball )`**

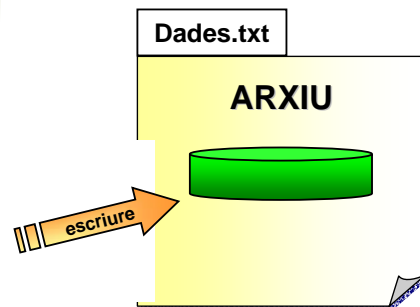
### Obertura d'un arxiu de text sortida

```
#include <fstream.h>
...
ofstream arxiu;

arxiu.open( "dades.txt" );           // obre per escriure i esborra
                                     // el contingut si estava creat

arxiu.open( "dades.txt", ios::app ); // obre per afegir dades al final

arxiu.open( "dades.txt", ios::in | ios::out ); // obre per llegir i escriure
```



## Obertura d'un arxiu de text

Amb la declaració de l'arxiu no és suficient.  
Es necessiten dos paràmetres.

**`.open( "nomArxiu.ext" , tipustreball )`**

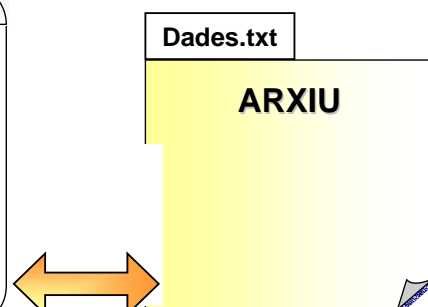
### Obertura d'un arxiu de text sortida

```
#include <fstream.h>
...
ofstream arxiu;

arxiu.open( "dades.txt" );           // obre per escriure i esborra
                                     // el contingut si estava creat

arxiu.open( "dades.txt", ios::app ); // obre per afegir dades al final

arxiu.open( "dades.txt", ios::in | ios::out ); // obre per llegir i escriure
```



## Obertura d'un arxiu binari

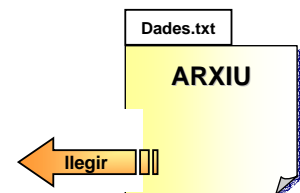
Amb la declaració de l'arxiu no és suficient.  
Es necessiten dos paràmetres.

**`.open( "nomArxiu.ext" , ios::binary )`**

### Obertura d'un arxiu binari entrada

```
#include <fstream.h>
...
ifstream arxiu;

arxiu.open( "dades.txt", ios::binary);           //obre per llegir
...
arxiu.open( "dades.txt", ios::binary | ios::in | ios::out); //obre per llegir i escriure
```



## Obertura d'un arxiu binari

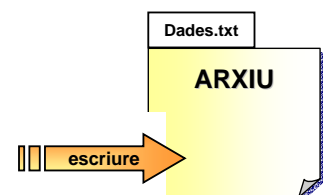
Amb la declaració de l'arxiu no és suficient.  
Es necessiten dos paràmetres.

**`.open( "nomArxiu.ext" , ios::binary )`**

### Obertura d'un arxiu binari sortida

```
#include <fstream.h>
...
ofstream arxiu;

arxiu.open( "dades.txt", ios::binary);           //obre per escriure i esborra
                                                    el contingut si estava creat
arxiu.open( "dades.txt", ios::binary | ios::app); //obre per afegir dades al final
arxiu.open( "dades.txt", ios::binary | ios::in | ios::out); //obre per llegir i escriure
```



## Tancar un arxiu

Allibera el canal de flux per permetre obrir un altre arxiu.  
Buida el buffer associat a l'arxiu (alliberant memòria).

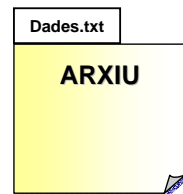
**.close ( )**

```

Tancar un arxiu

#include <fstream.h>
...
ifstream arxiu;
...
arxiu.close(); // Tancar l'arxiu

```



## Funcions amb arxius

Abans de llegir/escriure en un arxiu, s'ha de comprovar  
que l'arxiu s'ha obert correctament.

**.bad ( )**

```

Comprovar el flux de dades

#include <fstream.h>
...
ifstream arxiu;
arxiu.open("noms.txt");

if(arxiu.bad())
{
    cout << "Error per obrir l'arxiu";
}
else
{
    ... // Podem treballar amb l'arxiu
}

```

Torna *true* si el flux de dades  
està corrupte.

## Funcions amb arxius

Quan llegim hem de comprovar on està el final de l'arxiu, per saber que no hi ha més per llegir.

**.eof ( )**

### Llegir un arxiu fins al final

```
#include <fstream.h>
...
ifstream arxiu;
arxiu.open("noms.txt");

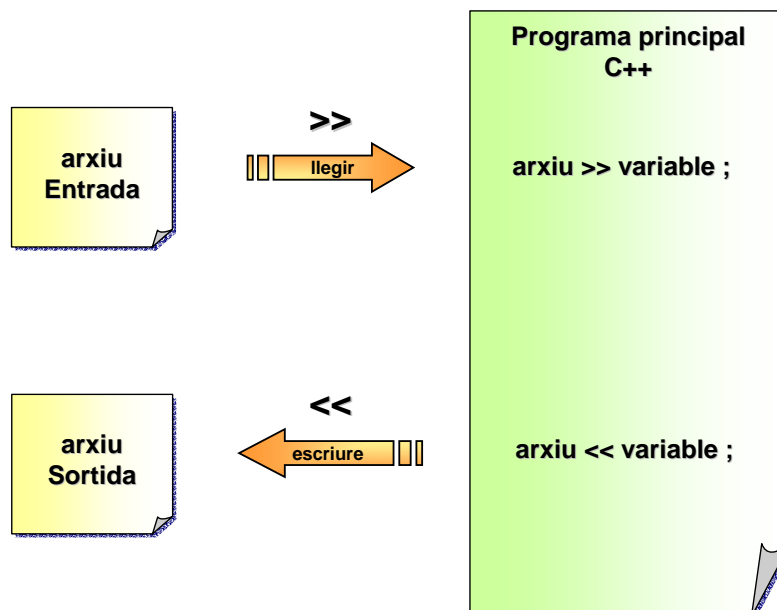
arxiu>>..... //lectura avançada
while (!arxiu.eof())
{
    .... // Llegir l'arxiu
}
```

Torna *true* si estem al final de l'arxiu.

S'ha de fer una lectura avançada.

## Escriure i llegir en arxius de text

Es pot realitzar directament amb els operadors << i >>.



## Escriure en un arxiu de text

Es pot realitzar directament amb els operadors << i >>.

### Escriure en un arxiu de text

```
#include <fstream> // Biblioteca per treballar amb arxius
#include <stdio> // Biblioteca estàndar de C

using namespace std;

int main()
{
    ofstream arxiu;
    arxiu.open("direccionsIP.txt");
    if (arxiu.bad())
    {
        cout << "Error per crear o obrir l'arxiu" << endl;
    }
    else
    {
        // Escriure en l'arxiu
        arxiu << "Clase A" << " " << 8 << " Red " << endl;
        arxiu << "Clase B" << " " << 16 << " Red " << endl;
        arxiu << "Clase C" << " " << 24 << " Red " << endl;
        arxiu.close();
    }
}
```

ARXIU

## Escriure en un arxiu de text

Es pot realitzar directament amb els operadors << i >>.

### Escriure en un arxiu de text

```
#include <fstream> // Biblioteca per treballar amb arxius
#include <stdio> // Biblioteca estàndar de C

using namespace std;

int main()
{
    ofstream arxiu;
    arxiu.open("direccionsIP.txt");
    if (arxiu.bad())
    {
        cout << "Error per crear o obrir l'arxiu" << endl;
    }
    else
    {
        // Escriure en l'arxiu
        arxiu << "Clase A" << " " << 8 << " Red " << endl;
        arxiu << "Clase B" << " " << 16 << " Red " << endl;
        arxiu << "Clase C" << " " << 24 << " Red " << endl;
        arxiu.close();
    }
}
```

direccionsIP.txt

Clase A	8	Red
Clase B	16	Red
Clase C	24	Red

## Llegir d'un arxiu de text

Es pot realitzar directament amb els operadors << i >>.

### Llegir d'un arxiu de text

```
#include <fstream> // Biblioteca per treballar amb arxius
#include <stdio> // Biblioteca estàndar de C

using namespace std;

int main()
{
    ifstream arxiu; // declaració de l'arxiu
    int bits;
    char classe[10];
    char tipus[5];

    arxiu.open("direccionsIP.txt");
    if (arxiu.bad())
    {
        cout << "Error per crear o obrir l'arxiu" << endl;
    }
    else
    {
        arxiu >> classe; // Lectura avançada
        while (!arxiu.eof())
        {
            cout << classe << " "; // Lectura de valors en l'arxiu
            arxiu >> bits;
            cout << bits << " ";
            arxiu >> tipus;
            cout << tipus << endl;
            arxiu >> classe;
        }
        arxiu.close();
    }
}
```

## Llegir d'un arxiu de text

Es pot realitzar directament amb els operadors << i >>.

```
char tipus[5];

arxiu.open("direccionsIP.txt");
if (arxiu.bad())
{
    cout << "Error per crear o obrir l'arxiu" << endl;
}
else
{
    arxiu >> classe; // Lectura avançada
    while (!arxiu.eof())
    {
        cout << classe << " "; // Lectura de valors en l'arxiu
        arxiu >> bits;
        cout << bits << " ";
        arxiu >> tipus;
        cout << tipus << endl;
        arxiu >> classe;
    }
    arxiu.close();
}
```

### direccionsIP.txt

Clase A	8	Red
Clase B	16	Red
Clase C	24	Red

### Taula variables memòria

bits → ?  
 classe → **Clase A**  
 tipus → ?

## Llegir d'un arxiu de text

Es pot realitzar directament amb els operadors << i >>.

```
char tipus[5];

arxiu.open("direccionsIP.txt");
if (arxiu.bad())
{
    cout << "Error per crear o obrir l'arxiu" << endl;
}
else
{
    arxiu >> classe;           // Lectura avançada
    while (!arxiu.eof())
    {
        cout << classe << " "; // Lectura de valors en l'arxiu
        arxiu >> bits;
        cout << bits << " ";
        arxiu >> tipus;
        cout << tipus << endl;
        arxiu >> classe;
    }
    arxiu.close();
}
```

direccionsIP.txt

Clase A	8	Red
Clase B	16	Red
Clase C	24	Red

Taula variables memòria

bits → ?  
 classe → Clase A  
 tipus → ?

Clase A

## Llegir d'un arxiu de text

Es pot realitzar directament amb els operadors << i >>.

```
char tipus[5];

arxiu.open("direccionsIP.txt");
if (arxiu.bad())
{
    cout << "Error per crear o obrir l'arxiu" << endl;
}
else
{
    arxiu >> classe;           // Lectura avançada
    while (!arxiu.eof())
    {
        cout << classe << " "; // Lectura de valors en l'arxiu
        arxiu >> bits;
        cout << bits << " ";
        arxiu >> tipus;
        cout << tipus << endl;
        arxiu >> classe;
    }
    arxiu.close();
}
```

direccionsIP.txt

Clase A	8	Red
Clase B	16	Red
Clase C	24	Red

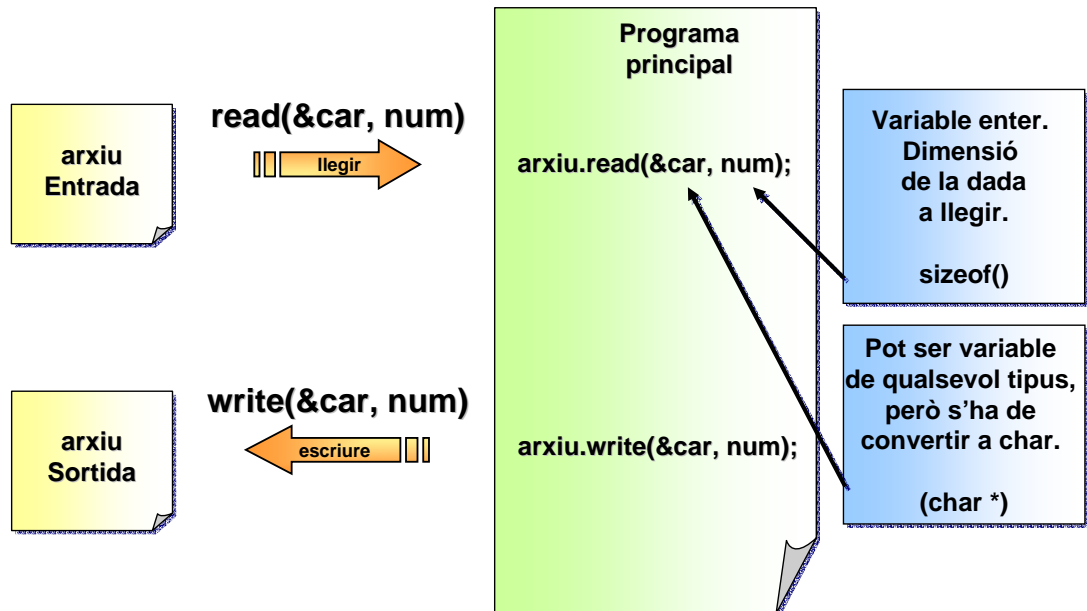
Taula variables memòria

bits → 24  
 classe → Clase C  
 tipus → Red

Clase A 8 Red  
 Clase B 16 Red  
 Clase C 24 Red

## Escriure i llegir en arxius binaris

S'utilitza per escriure i llegir blocs de dades.  
Taules, vectors, estructures....



## Escriure en arxius binaris

Per escriure en arxius binaris → write (&car, num)

### Escriure en un arxiu binari

```
#include <fstream> // Biblioteca per treballar amb arxius
#include <stdio> // Biblioteca estàndar de C

using namespace std;

int main()
{
    int taula[2][2]={1, 2, 3, 4};

    ofstream arxiu;
    arxiu.open("numeros.txt");
    if (arxiu.bad())
    {
        cout << "Error per crear o obrir l'arxiu"<<endl;
    }
    else
    {
        // Escriure en l'arxiu
        arxiu.write((char*)&taula, sizeof(int));
        arxiu.close();
    }
}
```

## Escriure en arxius binaris

Per escriure en arxius binaris → write (&car, num)

### Escriure en un arxiu binari

```
#include <fstream> // Biblioteca per treballar amb arxius
#include <stdio> // Biblioteca estàndar de C

using namespace std;

int main()
{
    int taula[2][2]={1, 2, 3, 4};

    ofstream arxiu;
    arxiu.open("numeros.txt");
    if (arxiu.bad())
    {
        cout << "Error per crear o obrir l'arxiu"<<endl;
    }
    else
    {
        // Escriure en l'arxiu
        arxiu.write((char*)&taula, sizeof(int));
        arxiu.close();
    }
}
```

numeros.txt

ÿÿÿÂw

Taula variables memòria

taula → 1 2  
3 4

## Escriure en arxius binaris

Per llegire en arxius binaris → read (&car, num)

### Llegir d'un arxiu binari

```
#include <fstream> // Biblioteca per treballar amb arxius
#include <stdio> // Biblioteca estàndar de C

using namespace std;

int main()
{
    int taula[2][2];

    ifstream arxiu;
    arxiu.open("numeros.txt");
    if (arxiu.bad())
    {
        cout << "Error per crear o obrir l'arxiu"<<endl;
    }
    else
    {
        // Llegir de l'arxiu
        arxiu.read((char*)&taula, sizeof(taula));
        while(!arxiu.eof())
        {
            arxiu.read((char*)&taula, sizeof(taula));
        }
        arxiu.close();
    }
}
```

## Escriure en arxius binaris

Per llegir en arxius binaris → read (&car, num)

### Llegir d'un arxiu binari

```
#include <fstream> // Biblioteca per treballar amb arxius
#include <stdio> // Biblioteca estàndar de C

using namespace std;

int main()
{
    int taula[2][2];

    ifstream arxiu;
    arxiu.open("numeros.txt");
    if (arxiu.bad())
    {
        cout << "Error per crear o obrir l'arxiu" << endl;
    }
    else
    {
        // Llegir de l'arxiu
        arxiu.read((char*)&taula, sizeof(taula));
        while(!arxiu.eof())
        {
            arxiu.read((char*)&taula, sizeof(taula));
        }
        arxiu.close();
    }
}
```

numeros.txt

ÿÿÿÄÄw -@Aw\$ü @ Gbmw

### Taula variables memòria

taula → 1 2  
3 4