

```

#include <iostream>
using namespace std;
#include <string.h>
#include <fstream>

#define DIM 12
#define SP ' '

struct Atleti {
    char NOME[DIM];
    int punti;
};

int main(){
    ofstream fout_t ("classifica.txt");
    ofstream fout_b ("classifica.dat", ios::binary);

    Atleti atleta;

    char NOME[DIM];
    int punti;

    for (int i=0; i<4; i++){
        cout << "\natleta? ";
        cin >> NOME >> punti;

        fout_t << NOME << SP << punti << endl;

        for (int k=0; k<DIM; k++)
            atleta.NOME[k] = SP; //-- spazio in NOME --

        strcpy (atleta.NOME, NOME);
        atleta.punti = punti;

        fout_b.write( (char *) &atleta, sizeof(Atleti) );
    }
    fout_t.close(); fout_b.close();
    return 0;
}

```

Il programma **classifica\_txt\_bin.cpp** acquisisce da tastiera nome e punteggio di 4 persone e li memorizza in un **file di testo** "classifica.txt" (nome logico **fout\_t**) e in un **file binary** "classifica.dat" (nome logico **fout\_b**).

I dati vengono letti nella variabile elementare **punti** e nell'array di caratteri **NOME** (secondo la gestione delle stringhe nel C), dati che vengono poi copiati nei corrispondenti campi del record **atleta** di tipo **Atleti** (dati strutturati); non c'è ambiguità per i nomi delle variabili definite nel main e nella struttura perchè per riferirsi ai campi di un record occorre indicare prima il record (es: **atleta.punti**).

ATTENZIONE: per gli array di caratteri **non è possibile usare gli operatori di assegnazione e confronto**; al loro posto vanno utilizzate le funzioni di copia (**strcpy**) e confronto (**strcmp**) e includere **<string.h>**

La funzione **strcpy** (<http://www.cplusplus.com/reference/cstring/strcpy/>) copia dall'array **NOME** all'array **atleta.NOME** un byte alla volta fino al terminatore di stringa ('\0' NULL character 0<sub>hex</sub> = 0x00). Se non si rimette **spazio nei byte successivi**, i nomi dal secondo al quarto non puliscono le celle successive.

### INPUT da tastiera

```

C:\> Prompt dei comandi
>>classifica_txt_bin.exe
atleta? Giangiaco 26
atleta? Anna 16
atleta? Desdemona      12
atleta? Ugo      3
>>

```

2

FILE

di OUTPUT

```

classifica.txt - Blocco note di...
File Modifica Formato Visualizza ?
Giangiaco 26
Anna 16
Desdemona 12
Ugo 3

classifica.dat - Blocco note ...
File Modifica Formato Visualizza ?
Giangiaco | Anna |
Desdemona ↑ Ugo |

```

**OUTPUT** su file ASCII: tutti i caratteri sono scritti in chiaro (anche le cifre), rappresentati in **codice ASCII**

classifica.txt - Gvim

```

00000000: 4769 616e 6769 6163 6f6d 6f20 3236 0d0a Giangiaco 26..
00000010: 416e 6e61 2031 360d 0a44 6573 6465 6d6f Anna 16..Desdemo
00000020: 6e61 2031 320d 0a55 676f 2033 0d0a na 12..Ugo 3..
  
```

3 caratteri di "Ugo" - 1 carattere a spazio - 1 byte per il carattere '3' - A CAPO (endl) ←file ASCII

**OUTPUT** su file binary **ripulendo atleta.NOME** (istruzione `atleta.NOME[k] = SP;`)

classifica.dat - Blocco note d...

```

Giangiaco  Anna
Desdemona  Ugo
  
```

Aprendo il file con l'editor Vim e selezionando Strumenti > Converti a esadecimale, sulla destra si vedono i singoli caratteri; al centro, per ogni carattere si vede il codice ASCII espresso in esadecimale.

classifica.dat - Gvim

```

00000000: 4769 616e 6769 6163 6f6d 6f00 1a00 0000 Giangiaco.....
00000010: 416e 6e61 0020 2020 2020 2020 1000 0000 Anna. ....
00000020: 4465 7364 656d 6f6e 6100 2020 0c00 0000 Desdemona. ....
00000030: 5567 6f00 2020 2020 2020 2020 0300 0000 Ugo. ....
00000040: 0d0a
  
```

3 caratteri di "Ugo" - terminatore di stringa - 8 caratteri a spazio - 4 byte per l'intero (3<sub>16</sub>)

**OUTPUT** su file binary **SENZA ripulire atleta.NOME**

classifica.dat - Blocco note

```

Giangiaco  Anna iacomo
Desdemona o  Ugo emona o
  
```

terminatore di stringa: 00

classifica.dat - Gvim

```

00000000: 4769 616e 6769 6163 6f6d 6f00 1a00 0000 Giangiaco.....
00000010: 416e 6e61 0069 6163 6f6d 6f00 1000 0000 Anna.iacomo.....
00000020: 4465 7364 656d 6f6e 6100 6f00 0c00 0000 Desdemona.o.....
00000030: 5567 6f00 656d 6f6e 6100 6f00 0300 0000 Ugo.emona.o.....
00000040: 0d0a
  
```

0x00 cioè '\0' Null character