atleti.cpp (1^ parte)

```
#include <iostream>
 using namespace std;
#include <string>
#include <algorithm>
#define DIM  4
#define ACAPO '\n'
#define TAB   '\t'
#define MSG_A "digita nome  atleta : "
#define MSG_B "digita sigla nazione: "
#define MSG_C "digita punti atleta : "
#define MSG_D " atleti registrati\n\n"
#define MSG_E "\n-atleti in ordine di nazione-\n"
#define MSG_F "\n---- GRADUATORIA atleti ----\n"

struct Atleta {
 string nome;
 string naz;
 int    punti;
};
void scrittura (Atleta a[], int tot){
    for (int i=0; i<tot; i++){          //--- per i<tot , tot vale 2
        cout<<"atleta " << i << TAB;    //--- al 1^richiamo, 4 negli altri.
        cout<< a[i].naz       << TAB;   //--- Scrittura dell'elemento
        cout<< a[i].nome      << TAB;   //--- di posto i nell'array a
        cout<< a[i].punti     << ACAPO;//--- di struttura Atleta
    }
}                                       //--- in C/C++ sono obbligatorie le [] per
int lettura  (Atleta aa[], int n){      //--- un parametro formale  array,

    while (n<DIM){                      //--- nella funzione si deve indicare: aa[]
        cout << ACAPO << MSG_A;
        cin  >> aa[n].nome;  //--- valorizzazione dell'elemento di posto n
                    //--- nell'array aa con dimensione fissata a 4 (DIM)
        cout << MSG_B;
        cin  >> aa[n].naz;

        cout << MSG_C;
        cin  >> aa[n].punti;

        n++;
        cout << "\nora registrati ";
        cout << n << " atleti\n\n";
    }
    return n;
}
```

atleti.html (1^ parte)

```
<html>
  <head><title>atleti</title>
    <script>

var DIM   = 4;
var ACAPO = "<br>";
var TAB   = ' ';
var MSG_A = "digita nome  atleta : ";
var MSG_B = "digita sigla nazione: ";
var MSG_C = "digita punti atleta : ";
var MSG_D = " atleti registrati<br><br>";
var MSG_E = "<br>-atleti in ordine di nazione-<br>";
var MSG_F = "<br>---- GRADUATORIA atleti ----<br>";

//--- funzione di scrittura dell'elemento di posto i nell'array a di Oggetti del tipo:
//--- { naz, nome, punti } Al 1^ richiamo tot=2 : vengono scritti gli elementi
//---di posto 0 e 1; negli altri richiami della funzione, invece, tutti gli elementi (4)

function  scrittura ( a, tot ){
    for (i=0; i<tot; i++){
        document.write("atleta " , i , TAB);
        document.write( a[i].naz      , TAB   );
        document.write (a[i].nome     , TAB   );
        document.write( a[i].punti    , ACAPO);
    }
}
function lettura  ( aa , n ){          //--- in JS si indica solo il nome dei

    while (n<DIM){                     //--- parametri formali:  aa  e  n
      nome   = prompt(MSG_A);
      naz    = prompt(MSG_B);
      punti  = prompt(MSG_C);
      aa.push                          //--- valorizzazione (e inserimento in posizione n)
      (   { nome   ,                   //--- di un elemento nell'array  aa  di Oggetti
            naz    ,                   //--- in JS il dimensionamento dell'array è dinamico
            punti }    );
        n++;
        document.write("<br>ora registrati ");
        document.write(n , " atleti<br><br>");
    }
    return n;
}
```

atleti.cpp (2^ parte)

```
//--- la funzione SORT richiede la scrittura di tante funzioni d'ordine
//--- quanti sono i diversi ordinamenti richiesti dal programma
//--- per la struttura Atleta, la funzione deve restituire:
//     true se l'oggetto uno deve precedere l'oggetto due
//     altrimenti false


bool xnazione (const Atleta& uno, const Atleta& due){
      if (uno.naz < due.naz)   //--- se il campo naz della struttura uno
            return true;  //--- precede (è < di) quello della struttura due
      else                  //--- le strutture sono in ordine e restituisce true
            return false;
}


//---  la seconda volta viene ordinato per valore DEcrescente
//--- del campo punti  e, nel caso di atleti con ugual punteggio,
//--- per valore crescente del campo nome (in ordine alfabetico)

bool xpunti(const Atleta& a,  const Atleta& b){

      if (a.punti  > b.punti)
            return true;
      else
      if (a.punti == b.punti  &&   a.nome    < b.nome)
            return true;
      else
            return false;
}
```

atleti.html (1^ parte)

```
//---  l'array atleti viene ordinate, la prima volta, per valore
//---- crescente del campo naz
//---  la funzione (richiamata dal metodo SORT) deve restituire:
//     -1 se l'oggetto uno deve precedere l'oggetto due
//      1 se l'oggetto  uno deve   seguire   l'oggetto due
//      0 se gli oggetti uno e due sono già in ordine
//


function  xnazione  ( uno,  due ) {
      if (uno.naz < due.naz)
          {return -1;}
      if (uno.naz > due.naz)
          {return  1;}
      return 0;
}



function  xpunti( a,  b ) {

      if (a.punti  > b.punti)
          {return -1;}
      else
      if (a.punti == b.punti  &&  a.nome  < b.nome)
          {return -1;}
      else
      if (a.punti == b.punti  &&  a.nome == b.nome)
          {return  0;}

      return 1;
}


    </script>
 </head>
```

NOTA: in questo esempio JS la dichiarazione delle variabili globali e le funzioni sono state poste nella sezione <head>, mentre l'elaborazione principale è stata posta nel <body> solo per poter confrontare meglio il **codice JS con il programma C++,** ma le istruz. **possono stare tutte in <head> o tutte in <body>** indifferentemente

atleti.cpp (3^ parte)

```
int main(){

//-------- atleti è una tabella = Array di strutture di tipo Atleta

   Atleta  atleti[DIM];

   int  num;

      atleti[0].nome  = "Bill";
      atleti[0].naz   = "USA";
      atleti[0].punti = 4;



      atleti[1].nome  = "Aldo";
      atleti[1].naz   = "ITA";
      atleti[1].punti = 7;


      num = 2;

      cout << "\n\n" << num << MSG_D;
      scrittura(atleti, num);




      num  = lettura   (atleti, num);
      scrittura(atleti, num);
//---------------------- richiamo della funzione SORT passando la funzione d'ordine

      sort  (atleti, atleti+num, xnazione);    //--- xnazione
      cout << MSG_E;
      scrittura(atleti, num);

      sort  (atleti, atleti+num, xpunti);      //--- xpunti
      cout << MSG_F;
      scrittura(atleti, num);

}
```

Segue l'esecuzione del programma **atleti.exe** sotto **Prompt dei comandi**

atleti.html (3^ parte)

```
<body style="font-family:Courier">
  <script>

//----------------- Array di OGGETTI  ----------------------------

  var atleti = [];
  var num;

      atleti.push
      (  { nome   : "Bill" ,
           naz    : "USA"  ,
           punti  : 4 }   );

      atleti.push
      (  { nome   : "Aldo" ,
           naz    : "ITA"  ,
           punti  : 7 }   );

      num = 2;

      document.write("<br><br>" , num , MSG_D);
      scrittura(atleti, atleti.length);

//--- length è un attributo dell'array atleti aggiornato automaticamente
//--- (anche dal metodo push). Il valore di length coincide con quello di num
//--- che viene aggiornato dal programma

      num  = lettura   (atleti, num);
      scrittura(atleti, num);

//--------- richiamo del metodo SORT dell'array atleti passando la funzione d'ordine

      atleti.sort  ( xnazione );       //--- xnazione
      document.write(MSG_E);
      scrittura(atleti, num);

      atleti.sort  ( xpunti  );        //--- xpunti
      document.write(MSG_F);
      scrittura(atleti, num);

   </script>
  </body>
</html>
```

OUTPUT a video per atleti.exe eseguito nel Prompt dei comandi (DOS)

```
Prompt dei comandi
>>
>>atleti


2 atleti registrati

atleta 0        USA     Bill    4
atleta 1        ITA     Aldo    7

digita nome  atleta : Bice
digita sigla nazione: ITA
digita punti atleta : 4

ora registrati 3 atleti


digita nome  atleta : John
digita sigla nazione: AUS
digita punti atleta : 2

ora registrati 4 atleti

atleta 0        USA     Bill    4
atleta 1        ITA     Aldo    7
atleta 2        ITA     Bice    4
atleta 3        AUS     John    2

-atleti in ordine di nazione-
atleta 0        AUS     John    2
atleta 1        ITA     Aldo    7
atleta 2        ITA     Bice    4
atleta 3        USA     Bill    4

---- GRADUATORIA atleti ----
atleta 0        ITA     Aldo    7
atleta 1        ITA     Bice    4
atleta 2        USA     Bill    4
atleta 3        AUS     John    2

>>
```

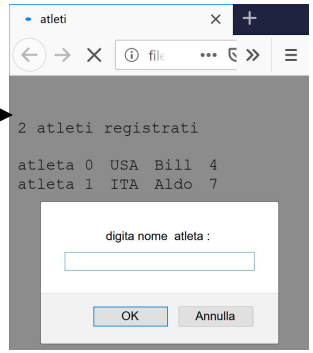
1^ richiamo della funzione:

scrittura(atleti, num);

scrittura(atleti, atleti.length);

il programma C++ richiede (2 volte) la digitazione dei dati di input

per il programma JS si apre (6 volte) la finestra del prompt per acquisire i dati di input; qui la prima per acquisire il campo nome dopo aver esposto il messaggio MSG_A



NOTA: la funzione JS lettura operando con prompt scrive sulla pagina web solo il numero di atleti registrati

2^ richiamo della funzione scrittura

3^ richiamo della funzione scrittura dopo l'ordinamento dell'array atleti mediante la funzione d'ordine xnazione

4^ richiamo della funzione scrittura dopo l'ordinamento dell'array atleti mediante la funzione d'ordine xpunti

PAGINA atleti.html aperta con Mozilla Firefox

```
atleti

2 atleti registrati

atleta 0  USA  Bill  4
atleta 1  ITA  Aldo  7

ora registrati 3 atleti


ora registrati 4 atleti

atleta 0  USA  Bill  4
atleta 1  ITA  Aldo  7
atleta 2  ITA  Bice  4
atleta 3  AUS  John  2

-atleti in ordine di nazione-
atleta 0  AUS  John  2
atleta 1  ITA  Aldo  7
atleta 2  ITA  Bice  4
atleta 3  USA  Bill  4

---- GRADUATORIA atleti ----
atleta 0  ITA  Aldo  7
atleta 1  ITA  Bice  4
atleta 2  USA  Bill  4
atleta 3  AUS  John  2
```