# Informatica B
# Esercitazione 4

25 ottobre 2021

## Array, Stringhe

**3.1** Si scriva un **programma C** per calcolare la media pesata dei voti di uno studente. Il programma chiede innanzitutto all'utente quanti esami vuole inserire (massimo 60). Quindi, per ciascun esame, **memorizza** il voto (controllando che sia compreso tra 18 e 30) e il numero di crediti. Infine, stampa la media pesata, calcolata secondo questa formula:

$$media\,pesata = \frac{\sum_{i=1}^{n} voto_i * crediti_i}{\sum_{i=1}^{n} crediti_i}.$$

*Suggerimento*: utilizzare un array per i voti e uno per i crediti

```c
#include <stdio.h>

/*
    Es. 3.1
    Media pesata
*/

#define MAX_ESAMI 60

int main()
{
    int num_esami = 0;
    int voti[MAX_ESAMI]; //array
    int crediti[MAX_ESAMI];
    int i; //indice
    float somma_pesata = 0;
    float somma_crediti = 0;
    float media_pesata;

    //Quanti esami?
    do
```

```
22      {
23          printf("Quanti esami vuoi inserire?\n");
24          scanf("%d", &num_esami);
25      } while(num_esami<=0 || num_esami>MAX_ESAMI);
26
27      //Inserimento: per ogni esame ...
28      for(i=0; i<num_esami; i++)
29      {
30          //Inserisci voto
31          do
32          {
33              printf("Esame %d: inserisci voto\n", i+1);
34              scanf("%d", &voti[i]);
35          } while(voti[i]<18 || voti[i]>30);
36
37          //Inserisci crediti
38          do
39          {
40              printf("Quanti crediti vale?\n");
41              scanf("%d", &crediti[i]);
42          } while(crediti[i]<=0);
43      }
44
45      //Calcolo media pesata
46      for(i=0; i<num_esami; i++)
47      {
48          somma_pesata+=voti[i]*crediti[i];
49          somma_crediti+=crediti[i];
50      }
51      media_pesata = somma_pesata/somma_crediti;
52
53      //Risultato
54      printf("\nLa media pesata e' %.2f\n", media_pesata);
55
56  }
```

*Bonus*: l'esercizio richiede esplicitamente di memorizzare i dati inseriti. Questo è davvero necessario per il funzionamento del programma?

Una soluzione alternativa consiste nel calcolare dei risultati parziali man mano che i dati vengono inseriti.

```c
#include <stdio.h>

/*
    Es. 3.1 (bonus)
    Media pesata (senza array)
*/

#define MAX_ESAMI 60

int main()
{
    int num_esami = 0;
    int i; //indice
    int voto, crediti;
    float somma_pesata = 0;
    float somma_crediti = 0;
    float media_pesata;

    //Quanti esami?
    do
    {
        printf("Quanti esami vuoi inserire?\n");
        scanf("%d", &num_esami);
    } while(num_esami<=0 || num_esami>MAX_ESAMI);

    //Raccolta dati
    for(i=0; i<num_esami; i++)
    {
        //Inserisci voto
        do
        {
            printf("Esame %d: inserisci voto\n", i+1);
            scanf("%d", &voto);
        } while(voto<18 || voto>30);

        //Inserisci crediti
        do
        {
            printf("Quanti crediti vale?\n");
            scanf("%d", &crediti);
```

```
41          } while(crediti<=0);
42
43          //Aggiorna risultati parziali
44          somma_pesata+=crediti*voto;
45          somma_crediti+=crediti;
46      }
47
48      //Calcolo media pesata
49      media_pesata = somma_pesata/somma_crediti;
50
51      //Risultato
52      printf("\nLa media pesata e' %.2f\n", media_pesata);
53
54  }
```

**3.2** Si scriva un **programma C** per contare la frequenza di ogni numero in un array letto inserito dall'utente.
*Esempio*: dato {5,100,12,5}, il programma stampa:

- 5 e presente 2 volte.
- 100 e presente 1 volta
- 12 e presente 1 volta

```c
#include <stdio.h>

/*
    Es. 3.2
    Contatore Frequenze
*/

#define MAX_LEN 200

int main()
{
    int arr[MAX_LEN];
    int frequenze[MAX_LEN]; //ci servira per tenere conto delle frequenze
    int n, i, j;

    //Quanti numeri?
    do
    {
        printf("Quanti numeri vuoi inserire?\n");
        scanf("%d", &n);
    } while(n<=0 || n>MAX_LEN);

   printf("Inserire %d elementi nel array :\n",n);
   for(i=0;i<n;i++)
        {
      printf("Inserire elemento - %d : ", i + 1);
      scanf("%d",&arr[i]);
      frequenze[i] = -1; //inizializo le frequenze a -1
    }

    for(i=0; i<n; i++)
    {
        if (frequenze[i] == -1)
        {
            frequenze[i] = 1;
            // controllo gli elementi dopo l'attuale
            for(j=i+1; j<n; j++)
```

```
38              {
39                  // se trovo lo stesso elemento
40                  if(arr[i]==arr[j])
41                  {
42                      frequenze[i]++; //incremento la frequenza
43                      // metto la frequenza di questo elemento a 0
44                      // notare che si mette a 0 la frequenza di j non di i
45                      // in questo caso una frequenza di 0 serve anche come 'flag'
46                      frequenze[j] = 0;
47                  }
48              }
49          }
50      }
51      printf("\nLa frequenza dei elementi dell'array : \n");
52      for(i=0; i<n; i++)
53      {
54          // salto gli elementi con frequenza 0
55          // sono duplicati
56          if(frequenze[i]!=0)
57          {
58              printf("%d e presente %d volte\n", arr[i], frequenze[i]);
59          }
60      }
61  }
```

**3.3** Si scriva un programma che prende in input un array e separata gli elementi pari e dispari in due array dedicati

```c
#include <stdio.h>
#include <stdlib.h>


/*
    Es. 3.3
    Pari e dispari
*/

#define MAX_N 60

int main()
{
    int arr_1[MAX_N], pari[MAX_N], dispari[MAX_N];
    int n ,i, cont_pari = 0, cont_dispari=0;

    do{
        printf("Inserire il numero di elementi!\n");
        scanf("%d", &n);
    }
    while(n <= 0 || n > MAX_N);

    for(i=0; i <n ;i++){
        printf("Inserire il numero %d \n", i+1);
        scanf("%d", &arr_1[i]);
    }

    for(i=0;i<n;i++)
    {
        if(arr_1[i] % 2 == 0)
        {
            // aggiungiamo all'array dei numeri pari
            pari[cont_pari] = arr_1[i];
            cont_pari++;
        }
        else
        {
            // aggiungiamo all'array dei numeri dispari
            dispari[cont_dispari] = arr_1[i];
            cont_dispari++;
        }
    }

```

```
44      printf("Il array contine %d numeri pari e %d numeri dispari \n", cont_pari,
45      printf("I numeri pari sono:\n [");
46      for(i=0; i<cont_pari ;i++){
47          printf("%d ", pari[i]);
48      }
49      printf("]\n");
50
51      printf("I numeri dispari sono:\n [");
52      for(i=0; i<cont_dispari ;i++){
53          printf("%d ", dispari[i]);
54      }
55      printf("]\n");
56
57      return 0;
58  }
```

**3.4** Scrivere un **programma C** che data una stringa dall'utente converte tutti i caratteri minuscoli in maiuscoli e viceversa.

```c
#include <stdio.h>
#include <string.h> //necessario per usare gets
#define LEN 200

/*
    Es. 3.4
    Inversione caratteri
*/

int main()
{
   // variabile per la stringha di input
   char stringa[LEN];
   // molteplici interi che ci serviranno
   // i per scorrere la stringa completa
   // direzione : per decidere che modifica aplicare
   int i = 0, direzione;

   printf("Inserire la stringa completa : \n");
   gets(stringa);

   while(stringa[i] != '\0')
   {
       if (stringa[i] >= 'a' && stringa[i] <= 'z')
       {
           // convertire in maiuscolo
           direzione = -1;
       }
       else if(stringa[i] >= 'A' && stringa[i] <= 'Z')
        {
            // convertire in minuscolo
            direzione = 1;
        }
        else
            {
            direzione = 0;
        }
        stringa[i] = stringa[i] + direzione * ('a' - 'A');
        i++;
   }
   printf("La stringa modificata e:\n");
   printf("%s", stringa);

```

```
44
45  }
```

**3.5** Scrivere un **programma C** che rimuove tutte le occorrenze di una sottostringa da una stringa piu lunga. Considerare lungheza massima delle stringe 200.
*Esempio*: dato "ciao a tutti voi. Come state tutti?" e "tutti", stampa "ciao a voi. Come state ?".

```c
#include <stdio.h>
#include <string.h> //necessario per usare gets
#define LEN 200

/*
    Es. 3.5
    Togliere sotto stringe
*/

int main()
{
   // variabile per le stringhe di input e di output
   char stringa[LEN + 1], sottostringa[LEN + 1], nuova_stringa[LEN + 1];
   // molteplici indici che ci serviranno
   // i per scorrere la stringa completa
   // j per scorrere la sottostringa
   // k per scorrere porzioni di sottostringa nella stringa completa
   // flag per sapere se si sono trovate sottostringe
   // n per tenere il prossimo indice dove aggiungere
   // elementi nella nuova stringa
   int i = 0, j,  k, flag = 0, n=0;

   printf("Inserire la stringa completa : ");
   gets(stringa);

   printf("Inserire la stringa da rimuovere : ");
   gets(sottostringa);

   printf("Dopo avere tolto \"%s\" la stringa e:\n", sottostringa);

   if (strlen(sottostringa) < strlen(stringa))
   {
        // cerchiamo per la presenza della sottostringa
        // partiamo dall'inizio della stringa completa
        // finche non arriviamo alla fine
        while(stringa[i] != '\0')
        {
            // k tiene il punto in cui stiamo cercando per sottostringe
            flag = 1, k = i;
            while(flag == 1)
```

```
41              {
42                  // iniziamo dall'inizio della sottostringa
43                  j = 0;
44                  // finche troviamo caratteri che coincidono andiamo
45                  // avanti in entrambe le stringe
46                  // senza passare oltre la lungezza della stringa
47                  while(stringa[k] == sottostringa[j] &&
48                         sottostringa[j] != '\0' &&
49                          stringa[k] != '\0')
50                  {
51                      k++;
52                      j++;
53                  }
54                  // se siamo arrivati alla fine della sottostringa
55                  // abbiamo trovato un matching
56                  if(sottostringa[j] == '\0')
57                  {
58                      // saltiamo la copia di questa sottostringa nel risultato
59                      i = k;
60                  }
61                  else
62                  {
63                      // altrimenti non abbiamo trovato una sottostringa
64                      // andiamo avanti con la copia
65                      flag = 0;
66                  }
67              }
68              if (stringa[i] == '\0')
69                  break;
70              // copiamo i caratteri tra una e l'altra stringa
71              nuova_stringa[n] = stringa[i];
72              i++;
73              n++;
74          }
75          // chiudiamo la stringa
76          nuova_stringa[n] = '\0';
77          printf("%s", nuova_stringa);
78
79      }
80      else if (strlen(sottostringa) == strlen(stringa))
81      {
82          if (strcmp(sottostringa, stringa) == 0)
83          {
84              // se le due stringe sono uguali
85              // il risultato e la stringa vuota
86              printf("");
```

```
87          }
88      }
89      else
90      {
91          // la sottostringa e piu lungha della stringa originale
92          // quindi non puo essere inclusa
93          printf("%s", stringa);
94      }
95
96
97  }
```