Matrici – Vettori di stringhe

Esercizi risolti

1 Esercizio: “Concorso di intelligenza”

In un concorso di intelligenza, N giudici esprimono il loro giudizio su K candidati. Il giudizio è un valore numerico tra 0 e 5.

Si scriva un programma in linguaggio C per determinare il candidato più intelligente, ed il giudice più severo.

Soluzione

```c
/* PROGRAMMAZIONE IN C */

/* File: concorso_intelligenza.c */
/* Soluzione proposta esercizio "Concorso di intelligenza" */

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

#define MAXK 100 /* max n. candidati */
#define MAXN 10  /* max n. giudici */

int main(void)
{
    int voti[MAXK][MAXN] ;
    int tot[MAXK] ; /* somma dei voti per ogni candidato */
    int totg[MAXN] ; /* somma dei voti di ogni giudice */
    int K, N ;
    int i, j ;
    int min, max, posmin, posmax ;

    printf("Quanti candidati ci sono? ");
    scanf("%d", &K) ;

    printf("Quanti giudici ci sono? ");
    scanf("%d", &N) ;

    for (i=0; i<K; i++)
    {
        printf("Immettere i giudizi per il candidato %d\n", i+1);

        for (j=0; j<N; j++)
        {
            printf("Giudice %d, cosa pensi del candidato %d? ",
                   j+1, i+1 );
            scanf("%d", & voti[i][j] ) ;
        }
    }

    for (i=0; i<K; i++) tot[i]=0 ;
    for (j=0; j<N; j++) totg[j]=0 ;

```

```
42      for (i=0; i<K; i++)
43      {
44          /* già fatto tot[i] = 0 ; */
45          for (j=0; j<N; j++)
46          {
47              tot[i] = tot[i] + voti[i][j] ;
48              totg[j] = totg[j] + voti[i][j] ;
49          }
50      }
51
52      max = tot[0] ;
53      posmax = 0 ;
54      for (i=1; i<K; i++)
55      {
56          if (tot[i]>max)
57          {
58              max = tot[i];
59              posmax = i ;
60          }
61      }
62
63      printf("Il␣vincitore␣e'␣il␣candidato␣numero␣%d\n", posmax+1);
64
65      min = totg[0] ;
66      posmin = 0 ;
67      for (i=1; i<N; i++)
68      {
69          if (totg[i]<min)
70          {
71              min = totg[i];
72              posmin = i ;
73          }
74      }
75
76      printf("Il␣giudice␣piu'␣severo␣e'␣il␣numero␣%d\n", posmin+1);
77      exit(0) ;
78  }
```

2 Esercizio: "Statistiche testo"

Si scriva un programma in C che acquisisca da tastiera un testo libero, composto da più righe (max 1000) di un numero di caratteri non superiore a 100 ciascuna. L'inserimento termina quando l'utente inserirà una riga uguale a `FINE`.

Al termine dell'acquisizione del testo, il programma dovrà stampare le seguenti statistiche:

1. il numero totale di righe inserite [1];
2. il numero totale di caratteri inseriti;
3. il numero totale di caratteri *alfanumerici* inseriti;
4. il numero totale di *parole* inserite.

Soluzione

[1] esclusa quella contentente `FINE`

```c
/* PROGRAMMAZIONE IN C */

/* File: statistiche.c */
/* Soluzione proposta esercizio "Statistiche testo" */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

int main(void)
{
    const int MAX = 1000 ;
    const int LUN = 100 ;

    char testo[MAX][LUN+1] ;
    int N ; /* righe inserite */
    int ncar, nalfa, npar ;
    int end ;

    char riga[300] ;
    int i,j ;

    N = 0 ;
    end = 0 ;
    do
    {
        printf("Testo: ") ;
        gets(riga) ;

        if ( strlen(riga) > LUN )
            printf("Errore: riga troppo lunga (max %d caratteri)\n", LUN) ;
        else if ( strcmp( riga, "FINE" )==0 )
            end = 1 ;
        else
        {
            strcpy( testo[N], riga ) ;
            N++ ;
        }
    }
    while (end==0 && N<MAX) ;

    printf("L'utente ha inserito %d righe\n", N) ;

    ncar = 0 ;
    for (i=0; i<N; i++)
        ncar = ncar + strlen( testo[i] ) ;

    printf("L'utente ha inserito %d caratteri\n", ncar) ;

    nalfa = 0 ;
    for (i=0; i<N; i++)
    {
        for (j=0; testo[i][j]!=0; j++)
        {
            if ( isalnum( testo[i][j] ) )
```

```
57                  nalfa++ ;
58          }
59      }
60
61      printf("L'utente␣ha␣inserito␣%d␣caratteri␣alfanumerici\n", nalfa) ;
62
63      npar = 0 ;
64      for (i=0; i<N; i++)
65      {
66          for (j=0; testo[i][j]!=0; j++)
67          {
68              /* verifico se [i][j] è il carattere
69                 iniziale di una parola */
70              if ( isalpha(testo[i][j]) &&
71                      ((j==0)|| !isalpha(testo[i][j-1])) )
72              {
73                  npar++ ;
74              }
75          }
76      }
77
78      printf("L'utente␣ha␣inserito␣%d␣parole\n", npar) ;
79      exit(0) ;
80  }
```

3 Esercizio: "Rubrica telefonica"

Si realizzi un programma in linguaggio C in grado di gestire una rubrica di nomi e numeri telefonici. La rubrica deve contenere fino a 100 voci diverse. Ciascuna voce è composta da un nome (max 40 caratteri) e da un numero di telefono (max 20 caratteri).

Il programma deve fornire all'utente un menù di scelta, con le seguenti voci:

```
1) Aggiungi nuova voce in rubrica
2) Ricerca esatta per nome
3) Ricerca approssimata per nome
4) Stampa completa rubrica
0) Esci dal programma
```

Una volta che l'utente ha scelto l'operazione desiderata (1-4), il programma acquisirà i dati necessari dall'utente ed eseguirà il comando. Nota: nella rubrica non possono esistere due voci con lo stesso nome.

Soluzione

```
1   /* PROGRAMMAZIONE IN C */
2
3   /* File: rubrica.c */
4   /* Soluzione proposta esercizio "Rubrica telefonica" */
5
6   #include <stdio.h>
7   #include <stdlib.h>
8   #include <string.h>
9
10  int main(void)
11  {
12      const int MAX = 100 ; /* numero max di voci */
13      const int LUNN = 40 ; /* lunghezza del nome */
```

```
14      const int LUNT = 20 ; /* lunghezza n. telefono */
15
16      char nome[MAX][LUNN+1] ;
17      char tel[MAX][LUNT+1] ;
18
19      int N ; /* numero di voci memorizzate */
20
21      int comando ; /* comando dell'utente 0-4 */
22
23      char riga[200] ;
24      char sn[LUNN+1] ;
25      char st[LUNT+1] ;
26      int i, duplicato, trovato, pos ;
27
28      /* INIZIALIZZAZIONI */
29
30      N = 0 ;
31
32      do
33      {
34          /* STAMPA DEL MENU */
35          puts("1) Aggiungi nuova voce in rubrica") ;
36          puts("2) Ricerca esatta per nome") ;
37          puts("3) Ricerca approssimata per nome") ;
38          puts("4) Stampa completa rubrica") ;
39          puts("0) Esci dal programma") ;
40
41          /* LETTURA DEL COMANDO */
42          printf("Inserisci il comando: ") ;
43          gets(riga) ;
44          comando = atoi( riga ) ;
45
46          /* ESECUZIONE DEL COMANDO */
47          switch ( comando )
48          {
49          case 1:
50              /* Acquisisci i dati */
51              printf("Inserisci il nome da aggiungere: ") ;
52              gets(sn) ;
53              printf("Inserisci il numero di telefono corrispondente: ") ;
54              gets(st) ;
55
56              /* Verifica se i dati sono validi */
57              if ( N == MAX )
58              {
59                  puts("ERRORE: rubrica piena") ;
60                  break ;
61              }
62
63              duplicato = 0 ;
64              for ( i = 0 ; i < N ; i++ )
65                  if ( strcmp(sn, nome[i]) == 0 )
66                      duplicato = 1 ;
67
68              if ( duplicato == 1 )
69              {
```

```
70              puts("ERRORE: nome duplicato") ;
71              break ;
72          }
73
74          /* Aggiungi il nome in rubrica */
75          strcpy( nome[N], sn ) ;
76          strcpy( tel[N], st ) ;
77          N++ ;
78
79          break ;
80
81      case 2: /* ricerca esatta */
82          printf("Inserisci il nome da ricercare: ") ;
83          gets(sn) ;
84
85          trovato = 0 ;
86          for ( i = 0 ; i < N && trovato == 0 ; i++ )
87          {
88              if ( strcmp( sn, nome[i] ) == 0 )
89              {
90                  trovato = 1 ;
91                  pos = i ;
92              }
93          }
94
95          if ( trovato == 1 )
96          {
97              printf("Il telefono di %s e': %s\n",
98                     sn, tel[pos] ) ;
99          }
100         else
101         {
102             printf("Nessun %s e' presente in rubrica\n", sn) ;
103         }
104
105         break ;
106
107     case 3: /* ricerca approssimata */
108         printf("Inserisci una parte del nome da ricercare: ") ;
109         gets(sn) ;
110
111         trovato = 0 ;
112         for ( i = 0 ; i < N ; i++ )
113         {
114             if ( strstr( nome[i], sn ) != NULL )
115             {
116                 printf("%s: %s\n", nome[i], tel[i]) ;
117                 trovato = 1 ;
118             }
119         }
120
121         if (trovato==0)
122             printf("Non trovato...\n") ;
123         break ;
124
125     case 4:
```

```
126             printf("CONTENUTO␣DELLA␣RUBRICA␣(%d␣VOCI)\n", N) ;
127
128             for ( i = 0 ; i < N ; i++ )
129                 printf("%s:␣%s\n", nome[i], tel[i] ) ;
130             break ;
131
132         case 0:
133             puts("Arrivederci") ;
134             break ;
135
136         default:
137             printf("ERRORE␣NEL␣PROGRAMMA␣(comando=%d)\n", comando) ;
138         }
139
140     }
141     while ( comando != 0 ) ;
142
143     exit(0) ;
144 }
```

4 Esercizio: “Gestione magazzino”

Un’azienda deve tenere traccia dei beni presenti in un magazzino. L’utente inserisce da tastiera dei “comandi” nel seguente formato:

```
bene EU quantità
```

dove:

- `bene` è il nome di un bene;
- `EU` è la lettera `'E'` per entrata, `'U'` per uscita;
- `quantità` è la quantità di bene entrata o uscita.

L’utente termina il caricamento inserendo un comando pari a `FINE`. In tal caso il programma deve stampare le quantità di beni presenti a magazzino.

Esempio:

```
viti E 10
dadi E 50
viti U 5
viti E 3
FINE
```

Soluzione

```
1  /* PROGRAMMAZIONE IN C */
2
3  /* File: magazzino.c */
4  /* Soluzione proposta esercizio "Gestione magazzino" */
5
6  #include <stdio.h>
7  #include <stdlib.h>
8  #include <string.h>
9
10 #define MAX 100
11 #define LUN 30
```

```c

int main(void)
{
    char prodotti[MAX][LUN] ;
    char prod[LUN] ;
    int quantita[MAX] ;
    int qta ;
    char dir ;
    int i ;
    int trovato ;
    int N ; /* dimensione dei vettori prodotti[] e quantita[] */

    N = 0 ;

    do
    {
        /* acquisisci un comando dall'utente */

        /* NOTA: non si può usare il costrutto
            scanf("%s %c %d", prod, &dir, &qta) ;
           in quanto non funziona per l'ultima riga (FINE) */

        printf("Comando: ") ;
        scanf("%s", prod) ;

        if ( strcmp(prod, "FINE") != 0 )
        {
            scanf(" %c %d", &dir, &qta) ;

            if ( dir=='E' ) /* entrata */
            {
                /* trova la posizione del prodotto nel vettore prodotti[] */
                trovato = -1 ;
                for (i=0; i<N; i++)
                {
                    if ( strcmp(prodotti[i], prod) == 0 )
                        trovato = i ;
                }

                if ( trovato != -1 ) /* prodotto esiste già */
                {
                    /* incrementa la posizione corrispondente del vettore
                       quantita[] */
                    quantita[trovato] = quantita[trovato] + qta ;
                }
                else /* prodotto nuovo */
                {
                    /* aggiungi il prodotto al magazzino in posizione nuova */
                    strcpy(prodotti[N], prod) ;
                    quantita[N] = qta ;

                    N++ ;
                }
            }
            else /* uscita */
            {
```

```
68                  /* trova la posizione del prodotto nel vettore prodotti[] */
69                  trovato = -1 ;
70                  for (i=0; i<N; i++)
71                  {
72                      if ( strcmp(prodotti[i], prod) == 0 )
73                          trovato = i ;
74                  }
75
76                  if ( trovato == -1 )
77                  {
78                      printf("Prodotto␣%s␣non␣trovato␣in␣magazzino\n", prod);
79                  }
80                  else
81                  {
82                      /* decrementa la posizione corrispondente del vettore
83                         quantita[] */
84                      quantita[trovato] = quantita[trovato] - qta ;
85                  }
86              }
87          }
88
89      }
90      while ( strcmp(prod, "FINE") != 0 ) ;
91
92      for (i=0; i<N; i++)
93      {
94          printf("%s␣%d\n", prodotti[i], quantita[i]) ;
95      }
96
97      exit(0) ;
98  }
```