```
/*
 * Corso di Fondamenti di Informatica
 * Esercizio:
 * tema d'esame.
 */

/* Testo dell'esercizio:
Data una matrice di NxM interi, scrivere un programma che scopra se vi sono
numeri comuni a tutte le M colonne e, in caso affermativo, stampi uno degli
elementi trovati in comune (ad esempio il primo elemento trovato).
*/

/* NOTA: nel programma sono state lasciate, a fini didattici, le istruzioni
 * printf "di debug" utilizzate per verificare il suo corretto funzionamento e
 * per facilitare la correzione degli errori.
 * Sono state trasformate in commenti, rendendole di fatto ininfluenti sul
 * funzionamento del programma, ma non eliminate. */


#include <stdio.h>
#define N 4
/* numero delle righe */
#define M 3
/* numero delle colonne */

int main()
{
  typedef enum {false, true} boolean;

  int Matrice[N][M];
  /* la matrice da esaminare: per convenzione stabiliamo che il primo indice
   * si riferisca alla riga, il secondo alla colonna */
  int ContEl, ContR, ContC;
  /* contatori usati per scorrere rispettivamente gli elementi della prima
   * colonna della matrice, le righe della matrice e le colonne della
   * matrice */
  int ElCorr;
  /* valore dell'elemento della matrice di cui si sta correntemente verificando
   * la presenza in tutte le colonne */
  boolean TrovatoMatrice;
  /* vale true se è stato trovato un elemento comune a tutte le colonne della
```

```
 * matrice */
boolean TrovatoColonna;
/* vale true se l'elemento attualmente ricercato è stato trovato nella
 * colonna attualmente in esame */

/************************************************
 *inizializzazione matrice- NON RICHIESTA ALL'ESAME*
 ************************************************/
printf("\n");
ContR = 0;
while (ContR < N)
/* scandisce le righe */
{
  ContC = 0;
  while (ContC < M)
  /* scandisce le colonne */
  {
    printf("Valore elemento (%d, %d) della matrice?  ", ContR+1, ContC+1);
          scanf("%d", &Matrice[ContR][ContC]);
          ++ContC;
       }
       ++ContR;
}

/***************************************
 *stampa matrice - NON RICHIESTA ALL'ESAME*
 ***************************************/
printf("\nMatrice:\n");
ContR = 0;
while (ContR < N)
{
  printf("\n");
  ContC = 0;
  while (ContC < M)
  {
    printf("\t");
    printf("%d", Matrice[ContR][ContC]);
    ++ContC;
  }
  ++ContR;
}
```

```
printf("\n");

/***********************************************
 *ricerca dell'elemento comune a tutte le colonne*
 ***********************************************/
TrovatoMatrice = false;
ContEl = 0;

while ((false == TrovatoMatrice) && (ContEl < N))
/* Ad ogni iterazione, il ciclo prende in esame uno degli elementi della prima
 * colonna (quella avente indice 0) e lo cerca nelle colonne successive.
 * La condizione (false == TrovatoMatrice) è usata per interrompere l'analisi
 * della matrice non appena viene trovato il primo elemento comune a tutte le
 * colonne. */
{
  ElCorr = Matrice[ContEl][0];
  /* Matrice[ContEl][0] è il primo elemento della riga di indice ContEl:
   * verrà cercato in tutte le colonne. Si noti che
   * la variabile ElCorr è stata introdotta soltanto per rendere più chiaro
   * il programma, visto che al suo posto si potrebbe usare semplicemente
   * Matrice[ContEl][0] */

  /* printf("\n##DEBUG: prendo in esame l'elemento %d della colonna di indice 0",
          ElCorr); */
  /* istruzione utilizzata in fase di debugging */

  TrovatoMatrice = true;
  /* supponiamo che l'elemento ElCorr si trovi in tutte le colonne; nel
   * seguito si verificherà se ciò è vero (basta che l'elemento sia assente
   * da una delle colonne perché l'asserzione diventi falsa) */

  ContC = 1;
  /* cominciamo la ricerca dalla seconda colonna (quella avente indice 1) */

  while ((true == TrovatoMatrice) && (ContC < M))
  /* Ad ogni iterazione il ciclo esamina una delle altre colonne per cercare
   * in essa un elemento di valore ElCorr.
   * La condizione (true == TrovatoMatrice) serve a interrompere la ricerca
   * nelle colonne non appena è stata trovata una colonna che non contiene
   * l'elemento cercato, una cosa non richiesta dal testo dell'esercizio ma
```

```
   * apprezzata in fase di correzione */
      {
    /* printf("\n##DEBUG: lo sto cercando nella colonna di indice %d", ContC); */
    /* istruzione utilizzata in fase di debugging */

        TrovatoColonna = false;
        ContR = 0;

        while ( (ContR < N) && (false == TrovatoColonna) )
          /* Ad ogni iterazione esamina uno degli elementi della colonna corrente.
           * La condizione (false == TrovatoColonna) serve a interrompere la ricerca
     * nella colonna non appena è stato trovato l'elemento cercato */
           {
      /* printf("\n##DEBUG: sto confrontando %d con %d", ElCorr,
             Matrice[ContR][ContC]); */
      /* istruzione utilizzata in fase di debugging */

            if (Matrice[ContR][ContC] == ElCorr)
                 {
                   TrovatoColonna = true;
                 }
                 ++ContR;
          }

          if (false == TrovatoColonna)
          /* cio' accade se l'elemento non era presente nella colonna appena
           * esaminata, e dunque non e' presente in tutte le colonne della
           * matrice */
          {
            TrovatoMatrice = false;
          }
          ++ContC;
      }
      ++ContEl;
}

if (true == TrovatoMatrice)
/* ciò accade soltanto se si è usciti dal ciclo più esterno dopo che è stato
 * trovato un elemento presente in tutte le colonne, e non semplicemente
 * perche' tutti gli elementi della prima colonna sono stati cercati senza
 * successo in ciascuna delle altre colonne */
```

```
    {
            printf("\nL'elemento %d e' presente in tutte le colonne.\n\n", ElCorr);
          }
        else
    {
            printf("\nNon esistono elementi comuni a tutte le colonne.\n\n");
          }

  return(0);
}


/*************************************************************************
Esempio di esecuzione con le linee di codice usate per il debug attive:

Valore elemento (1, 1) della matrice?  1
Valore elemento (1, 2) della matrice?  2
Valore elemento (1, 3) della matrice?  3
Valore elemento (2, 1) della matrice?  2
Valore elemento (2, 2) della matrice?  4
Valore elemento (2, 3) della matrice?  5
Valore elemento (3, 1) della matrice?  6
Valore elemento (3, 2) della matrice?  7
Valore elemento (3, 3) della matrice?  8
Valore elemento (4, 1) della matrice?  9
Valore elemento (4, 2) della matrice?  10
Valore elemento (4, 3) della matrice?  2

Matrice:

        1         2       3
        2         4       5
        6         7       8
        9         10    2

##DEBUG: prendo in esame l'elemento 1 della colonna di indice 0
##DEBUG: lo sto cercando nella colonna di indice 1
##DEBUG: sto confrontando 1 con 2
##DEBUG: sto confrontando 1 con 4
##DEBUG: sto confrontando 1 con 7
##DEBUG: sto confrontando 1 con 10
```

```
##DEBUG: prendo in esame l'elemento 2 della colonna di indice 0
##DEBUG: lo sto cercando nella colonna di indice 1
##DEBUG: sto confrontando 2 con 2
##DEBUG: lo sto cercando nella colonna di indice 2
##DEBUG: sto confrontando 2 con 3
##DEBUG: sto confrontando 2 con 5
##DEBUG: sto confrontando 2 con 8
##DEBUG: sto confrontando 2 con 2
L'elemento 2 e' presente in tutte le colonne.
 ************************************************************************/



/************************************************************************
 * Esercizio: riscrivere il programma usando cicli for anziché cicli while.
 ************************************************************************/

/************************************************************************
 * Esercizio: data una matrice di NxM interi, scrivere un programma che scopra
 * se vi sono numeri comuni a tutte le M RIGHE e, in caso affermativo, stampi
 * uno degli elementi trovati in comune (ad esempio il primo elemento
 * trovato). Si cerchi di farlo senza consultare il programma soprastante.
 ************************************************************************/
```