Eserci su vettori e matrici

Esercizio 1 (23 Luglio 2015)

Scrivere una funzione che, data una matrice mat di interi passata come argomento alla funzione, restituisca la somma degli elementi in posizione (i,j), in cui almeno uno tra l'indice di riga i e l'indice di colonna j siano multipli di 2. La matrice mat ha dimensione nxm, dove n e m sono variabili. Per esempio, data la matrice seguente

1 2 3
4 5 6
7 8 9
2 3 0

la funzione restituisce 33 (=3+6+7+8+9+0)

Esercizio 2 (2 Luglio 2015)

Scrivere una funzione che, dati due vettori v1 e v2 di elementi di tipo int ordinati in ordine crescente, restituisca un vettore con tutti gli elementi di v1 e v2 ordinato in ordine decrescente. I vettori v1 e v2 possono contenere duplicati e possono avere dimensione diversa.

Esercizio 3 (11 Giugno 2015)

Scrivere una funzione che, data una matrice `mat` di $n \times m$ elementi di tipo intero ed un intero p, restituisce `true` se esistono almeno **3 elementi adiacenti** uguali a p in una riga e in una colonna della matrice, `false` altrimenti.

Per esempio, data la matrice `mat` (4×6), la funzione restituisce `true` se p=1 oppure p=0; la funzione restituisce `false` altrimenti:

$$\texttt{mat} = \begin{matrix} 0\,1\,1\,1\,7\,0 \\ 0\,0\,0\,0\,1\,0 \\ 2\,0\,1\,0\,1\,3 \\ 1\,6\,2\,0\,1\,8 \end{matrix}$$

Esercizio 4 (13 Gennaio 2015)

Scrivere una funzione che prende in ingresso una matrice di interi di dimensione nxm. La funzione deve restituire una nuova matrice che è la trasposta della matrice passata alla funzione.

Per esempio, se la matrice passata alla funzione ha dimensioni 4x3 ed è inizializzata come:

$$\begin{pmatrix} 2 & 4 & 8 \\ 3 & 2 & 0 \\ 5 & 3 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

La matrice restituita dalla funzione è:

$$\begin{pmatrix} 2 & 3 & 5 & 0 \\ 4 & 2 & 3 & 1 \\ 8 & 0 & 1 & 0 \end{pmatrix}$$

Esercizio 5 (23 Luglio 2014)

Sia data la struttura seguente:

```
struct punto {double x; double y;}
```

Scrivere una funzione che, dati due vettori v1 e v2 di lunghezza n > 0 di elementi di tipo reale, restituisca un vettore di elementi di tipo punto di dimensione uguale alla dimensione massima fra quella di v1 e v2.
L'elemento i-esimo del vettore restituito deve avere come parte x l'elemento i-esimo di v1 e come parte y l'elemento i-esimo di v2. Se i-esimo elemento del vettore non esiste, assumere come valore 0.
Ad esempio, dati i vettori v1 = [1.0, 3.2] e v2 = [0.2, 5.1, 7.3], la funzione restituirà il vettore [(1.0,0.2), (3.2,5.1), (0.0, 7.3)].

Esercizio 6 (2 Luglio 2014)

Scrivere una funzione che dati due interi N e K, passati come argomento alla funzione, legge da tastiera N stringhe di lunghezza <=K e le memorizza in un array che viene restituito dalla funzione. Se N <=0, vengono lette 5 stringhe. Se K <=0, la lunghezza massima della stringa è 99. In caso di errore nella lettura da tastiera di una stringa, viene salvata nel vettore la stringa "Ciao". L'array di stringhe restituito dalla funzione deve poter essere utilizzato correttamente nel chiamante.

Esercizio 7 (19 Febbraio 2014)

Scrivere una funzione che prende in ingresso una matrice di interi ordinata per righe per valori non decrescenti e un intero `k`. La funzione incrementa di 5 l'elemento in posizione (k,k), e riordina gli elementi della matrice. Se la posizione `k` non è valida, la funzione restituisce `false` e lascia la matrice inalterata; altrimenti modifica la matrice e restituisce `true`.

Per esempio, se la funzione viene chiamata con la matrice seguente di dimensione `3x4` e `k=1`:

```
0  1  2  2
2  3  4  6
7  7  7  9        3+5 = 8
```

la matrice verrà modificata come segue e la funzione restituisce `true`:

```
0  1  2  2
2  4  6  7
7  7  8  9
```

Esercizio 8 (2013, appello I)

Scrivere una funzione che prende in ingresso un vettore di interi di dimensione n e restituisce una matrice quadrata di interi di dimensione nxn inizializzata come segue. Tutti gli elementi di riga i e colonna j>=i sono uguali al valore dell'elemento del vettore nella posizione di indice i. Tutti gli elementi di colonna j e riga i>=j sono uguali al valore dell'elemento del vettore nella posizione di indice j. Le righe e le colonne sono numerate a partire da 0.

Per esempio, se la funzione viene chiamata con il vettore di dimensione 5 seguente:
3 1 7 9 2
la matrice restituita è (dimensione 5x5):

3 3 3 3 3
3 1 1 1 1
3 1 7 7 7
3 1 7 9 9
3 1 7 9 2

Soluzione Esercizio 1

```
int somma(int * mat, int n, int m) {
    int s = 0;
    for (int i = 0; i < n; i++)
        for (int j = 0; j < m; j++)
            if (((i != 0) && (i % 2 == 0)) ||
               ((j != 0) && (j % 2 == 0)))
                s += mat[i * m + j];
    return s;
}
```

Soluzione Esercizio 2

```
int * creavettore(int * v1, int l1, int * v2, int l2) {
  int * v = new int[l1 + l2];
  int i, j, k;
  i = 0;
  j = 0;
  k = l1 + l2 - 1;
  while (i < l1 && j < l2)
     v[k--] = (v[i] < v[j]) ? v[i++] : v[j++];
  if (i == l1)
     while (j < l2) v[k--] = v[j++];
  else
     while (i < l1) v[k--] = v[i++];

  return v;
}
```

Soluzione Esercizio 3

```
bool cerca(int * mat, int n, int m, int p) {
    bool trovator = false;
    bool trovatoc = false;
    for (int i = 0; i < n; i++)
      for (int j = 0; j < m - 2; j++)
        if ( mat[i * m + j] == p &&
             mat[i * m + j + 1] == p && mat[i * m + j + 2] == p)
           trovator = true;
    if (!trovator)
        return false;
    for (int i = 0; i < n - 2; i++) {
      for (int j = 0; j < m; j++)
        if ( mat[i * m + j] == p && mat[(i + 1) * m + j] == p &&
             mat[(i + 2) * m + j] == p)
           trovatoc = true;

    return trovatoc;
}
```

Soluzione Esercizio 4

```
int * creamatrice(int * mat, int r, int c) {
   int * p = new int[r * c];
   for (int i = 0; i < r; i++) // colonne matrice nuova
      for (int j = 0; j < c; j++) // righe matrice nuova
         p[j * r + i] = mat[i * c + j];
   return p;
}
```

Soluzione Esercizio 5

```
punto * nuovoVettore(double * v1, int d1, double * v2, int d2) {
  int dim = d1;
  if (d2 > d1) dim = d2;

  punto * v = new punto[dim];

  for (int i = 0; i < dim; i++) { // inizializzo campo x della
    // struttura punto dell'elemento i-esimo
    if (i < d1)
      v[i].x = v1[i];
    else v[i].x = 0;

    // inizializzo campo y della
    // struttura punto dell'elemento i-esimo
    if (i < d2)
      v[i].y = v2[i];
    else
      v[i].y = 0;
  }
  return v;
}
```

Soluzione Esercizio 6

```
char ** leggi_stringhe(int N, int K) {
  char ** v;
  if (N <= 0)
    N = 5;
  if (K <= 0)
    K = 99;

  v = new char * [N];
  for (intj = 0; j < N; j++) {
    v[j] = new char[K + 1];
    cin >> v[j];
    if (!cin)
      strncpy(v[j], "Ciao", K);

  }
  return v;

}
```

Soluzione Esercizio 7

```
void modifica(int* mat, int r, int c, int k){
    if (k>=r || k >= c)
        return;
    int indice = k*c+k;
    int nuovo = m[indice] + 5;
    for(int i=indice+1; (i<r*c) && m[i] <= nuovo; i++)
         m[i-1]=m[i];
    m[i-1]=nuovo;
    return;
}
```

Soluzione Esercizio 8

```
int* creamatrice(int v[], int n){
    int* m = new int [n * n];
    for (int i=0; i< n; i++)
        for (int j=i; j < n; j++)
            m[i*n +j] = v[i];

        for (int j=0; j< n; j++)
             for (int i = j+1; i < n; i++)
                m[i*n +j] = v[j];
     return m;
}
```