E8 – Esercizi sugli array in C

Esercizio 1. Scrivere una funzione C che riceve come parametro un array di double e la sua dimensione e restituisce la media degli elementi presenti nell'array.

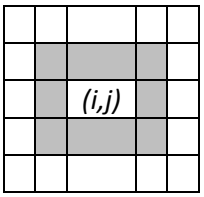
Esercizio 2. Scrivere una funzione C che riceve come parametro un array di interi, la sua dimensione e due indici *h* e *k* e restituisce la somma degli elementi dell'array memorizzati nelle posizioni da *h* a *k*.

Esercizio 3. Scrivere una funzione C che riceve come parametro un array di interi e la sua dimensione e restituisce 1 se tutti gli elementi dell'array sono distinti.

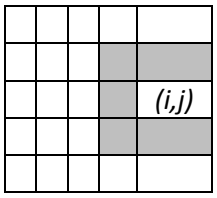
Esercizio 4. Scrivere una funzione C che riceve come parametro una matrice quadrata *m* e la sua dimensione e restituisce 1 se la matrice è diagonale. Si ricorda che una matrice è diagonale se tutti i suoi elementi esclusi quelli della diagonale principale sono uguali a 0.

Esercizio 5. Scrivere una funzione C che riceve come parametro una matrice *m*, le sue dimensioni e un indice di colonna *j* e restituisce l'elemento di valore massimo della colonna *j*.

Esercizio 6. Scrivere una funzione C che riceve come parametro una matrice *m*, le sue dimensioni e due indici *i* e *j* e restituisce il numero di celle adiacenti alla cella (*i*,*j*) che contengono un numero positivo. Le celle adiacenti alla cella (*i*,*j*) sono le celle (*h*,*k*) con $i-1 \leq h \leq i+1$ e $j-1 \leq k \leq j+1$ (quelle evidenziate in grigio in figura). Si noti che una cella che non si trova sul bordo della matrice ha 8 celle adiacenti mentre le celle del bordo ne hanno di meno (vedi figura).



Celle adiacenti



Celle adiacenti ad una cella del bordo

Soluzioni

Esercizio 1 - svolgimento.

```
#include <stdio.h>

double media(double a[], int dim);

int main(void){
   printf("Quanti elementi vuoi inserire?\n");
   int n;
   scanf("%d", &n);

   double b[n];
   for(int i=0; i<n; i++){
      printf("Inserisci l'elemento di posizione %d ",i);
      scanf("%lf",&b[i]);
   }
   double m=media(b,n);
   printf("La media degli elementi inseriti e' %g\n", m);
}

double media(double a[], int dim){
      double somma=0.0;
      for(int i=0;i<dim;i++)
            somma+=a[i];
      return somma/dim;
}
```

Esercizio 2 - svolgimento.

```
#include <stdio.h>

int sommaSottoArray(int a[], int dim, int h, int k);

int main(void){
   printf("Quanti elementi vuoi inserire?\n");
   int n;
   scanf("%d", &n);

   int b[n];
   for(int i=0; i<n; i++){
      printf("Inserisci l'elemento di posizione %d ",i);
      scanf("%d",&b[i]);
   }

   int h,k;
   printf("Inserisci due indici\n");
   scanf("%d%d",&h,&k);

   int somma=sommaSottoArray(b,n,h,k);
   printf("La somma degli elementi da %d a %d e' %d\n", h, k, somma);
}

int sommaSottoArray(int a[], int dim, int h, int k){
      int somma=0;
      for(int i=h;i<=k;i++)
```

```
            somma+=a[i];
      return somma;
}
```

Esercizio 3 - svolgimento.

```
#include <stdio.h>

int tuttiUguali(int a[], int dim);

int main(void){
   printf("Quanti elementi vuoi inserire?\n");
   int n;
   scanf("%d", &n);

   int b[n];
   for(int i=0; i<n; i++){
      printf("Inserisci l'elemento di posizione %d ",i);
      scanf("%d",&b[i]);
   }

   int uguali=tuttiUguali(b,n);

   if(uguali)
       printf("Gli elementi dell'array sono tutti uguali\n");
   else
       printf("Gli elementi dell'array NON sono tutti uguali\n");
}

int tuttiUguali(int a[], int dim){
      int uguali=1;
      int d=a[0];
      int i=1;

      while(i<dim && uguali){

            if(a[i]!=d)
                  uguali=0;
            i++;
      }

      return uguali;
}
```

Esercizio 4 - svolgimento.

```
#include <stdio.h>

int diagonale(int dim, int m[][dim]);

int main(void){
      printf("Inserisci la dimensione della matrice\n");
      int m;
      scanf("%d", &m);

      int mat[m][m];
      for(int i=0; i<m; i++)
```

```
            for(int j=0; j<m; j++){
                  printf("Inserisci l'elem. di posizione (%d,%d) ",i,j);
                  scanf("%d",&mat[i][j]);
            }

   int diag=diagonale(m,mat);

   if(diag)
       printf("La matrice e' diagonale\n");
   else
       printf("La matrice NON e' diagonale\n");
}

int diagonale(int dim, int m[][dim]){
      int diagonale=1;

      int i=0;
      while(i<dim && diagonale){
            int j=0;
            while(j<dim  && diagonale){
                  if(i!=j && m[i][j]!=0){
                        diagonale=0;
                  }
                  j++;
            }
            i++;
      }

      return diagonale;
}
```

Esercizio 5 - svolgimento.

```
#include <stdio.h>

int maxColonna(int rows, int cols, int m[][cols], int j);

int main(void){
      printf("Inserisci le dimensioni della matrice\n");
      int m,n;
      scanf("%d%d", &m, &n);

      int mat[m][n];
      for(int i=0; i<m; i++)
            for(int j=0; j<n; j++){
                  printf("Inserisci l'elem. di posizione (%d,%d) ",i,j);
                  scanf("%d",&mat[i][j]);
            }

      printf("Inserisci un indice di colonna\n");
      int j;
      scanf("%d",&j);

      int max=maxColonna(m,n,mat,j);
      printf("Il max degli elementi della colonna %d e' %d\n", j, max);

}

int maxColonna(int rows, int cols, int m[][cols], int j){
```

```
      int max=m[0][j];

      for(int i=1;i<rows;i++)
            if(m[i][j]>max)
                  max=m[i][j];

      return max;
}
```

Esercizio 6 - svolgimento.

```
#include <stdio.h>

int positiviIntorno(int rows, int cols, int m[][cols], int i, int j);

int main(void){
      printf("Inserisci le dimensioni della matrice\n");
      int m,n;
      scanf("%d%d", &m, &n);

      int mat[m][n];
      for(int i=0; i<m; i++)
            for(int j=0; j<n; j++){
                  printf("Inserisci l'elem. di posizione (%d,%d) ",i,j);
                  scanf("%d",&mat[i][j]);
            }

      printf("Inserisci un indice di riga e uno di colonna\n");
      int i,j;
      scanf("%d%d",&i,&j);

      int num=positiviIntorno(m,n,mat,i,j);
      printf("Il num. di elem. pos. adiac. a (%d,%d) e' %d\n",i,j,num);

}

int positiviIntorno(int rows, int cols, int m[][cols], int i, int j){
   int count=0;

   for(int h=i-1;h<=i+1;h++)
      for(int k=j-1;k<=j+1;k++)
         if((h>=0&&h<rows)&&(k>=0&&k<cols) && (h!=i||k!=j) && m[h][k]>0)
            count++;

   return count;
}
```

Nota: Nell'istruzione **if** della funzione **positiviIntorno** non succede mai che nel valutare la condizione **m[h][k]>0** gli indici siano al di fuori dell'intervallo di validità in quanto essi vengono controllati nelle due condizioni **(h>=0&&h<rows)** e **(k>=0&&k<cols)** dell'if. Poiché gli operatori **&&** e **||** valutano gli operandi successivi solo se necessario, la condizione **m[h][k]>0** verrà valutata soltanto se le due condizioni **(h>=0&&h<rows)** e **(k>=0&&k<cols)** sono vere, cioè se gli indici sono validi.