MATRICI

VETTORI

```
int vet[6] = {0};
```

	0	1	2	3	4	5
vet	0	0	0	~~10~~ 0	0	0

DIMENSIONE 6

```
vet[3] = 10;
```

MATRICI: VETTORE DI VETTORI

[→ = PER LEGGERE UNA MATRICE]

	0	1	2	3	4
0	→	→	→	→	↓
1	→	(1,1)			
2				(2,3)	
3	(3,0)				
4					

← OGNI CELLA (RIGHE, COLONNE)

CREARE UNA MATRICE

```
#define DIM 5
```

- `int matrix[DIM][DIM];` // MATRICE CON VALORI NON DEFINITI
  - DIM ↑ RIGHE
  - DIM ↑ COLONNE

- `int matrix2[4][4] = {{1,2,3,4}, {5,6,7,8}, {9,10,11,12}, {13,14,15,16}};`

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

- `int matrix3[DIM][DIM] = {0};` // MATRICE DI TUTTI ZERI

MODIFICARE UN ELEMENTO DELLA MATRICE

↓

ASSEGNARE

```
int matrix2[4][4] = {{0,1,2,3},
                     {4,5,6,7},
                     {8,9,10,11},
                     {12,13,14,15}};
```

	0	1	2	3
0	0	1	2	3
1	4	5	6	7
2	8	9	10	11
3	12	13	14	15

```
matrix2[1][1] = 10;
int tmp;
tmp = matrix2[0][2];
matrix2[0][2] = matrix2[2][2];
matrix2[2][2] = tmp;
```

SWAP TRA DUE CELLE

FUNZIONI E MATRICI

```
# define DIM 10
void stampaMatrice(int matrix[ ][DIM], int dimX, int dimY);
void randMatrice(int matrix[ ][DIM], int dimX, int dimY, int min,
                                                          int max);
int main( ){
  srand (time(NULL));
  int matrix[DIM][DIM];
  stampaMatrice (matrix, DIM, DIM);
  randMatrice (matrix, DIM, DIM, 1, 10);
  stampaMatrice (matrix, DIM, DIM);
}

void stampaMatrice(int matrix[ ][DIM], int dimX, int dimY){
   int i, j;
   For(i=0; i<dimX; i++){
      For(j=0; j<dimY; j++){
        printF("%4d", matrix[i][j]);
      }
      printf("\n");
   }
   printf("\n");
}
```

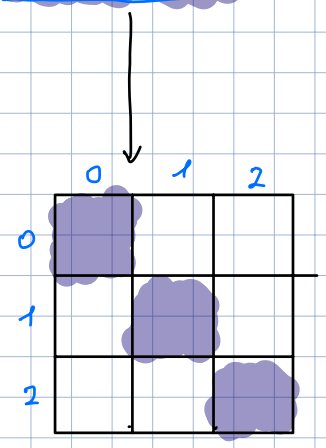
```
void randMatrice(int matrix[][dim], int dimX, int dimY, int min,
                 int max){
  int i, j;
  for(i=0; i<dimX; i++){
    for(j=0; j<dimY; j++){
      matrix[i][j]= rand()%(max-min+1)+min;
    }
  }
}
```

ES. 1

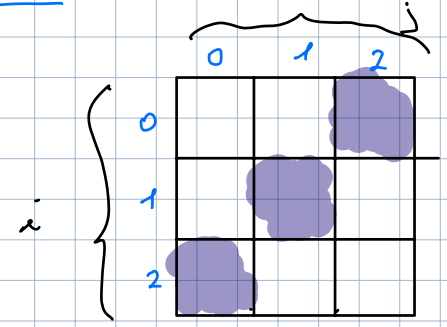
PUNTATORE MATRICE, DIM. RIGHE (x), DIM. COLONNE (y)

Creare una funzione in C che prenda in input una matrice e stampi la diagonale principale della matrice (#define DIM 5)

VOID

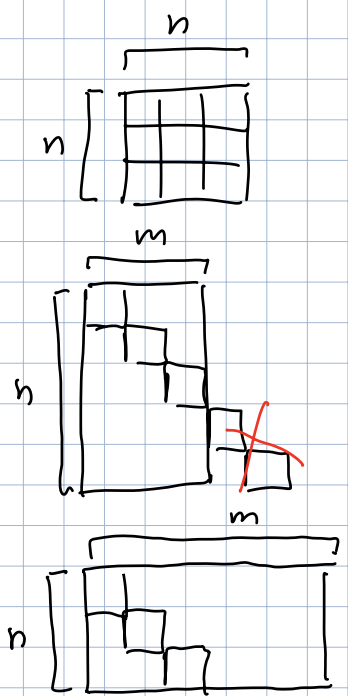


N.B. DIAGONALE SECONDARIA



$i + j == DIM - 1$

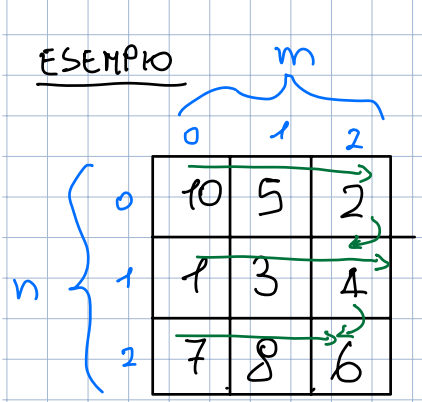
```
void stampaDiagonale(int matrix[][DIM], int dimX, int dimY)
{
  int i, j; // i = RIGHE , j = COLONNE
  for(i=0; i<dimX; i++){
    for(j=0; j<dimY; j++){
      if(i==j){
        printf("%4d", matrix[i][j]);
      }
    }
  }
}
```

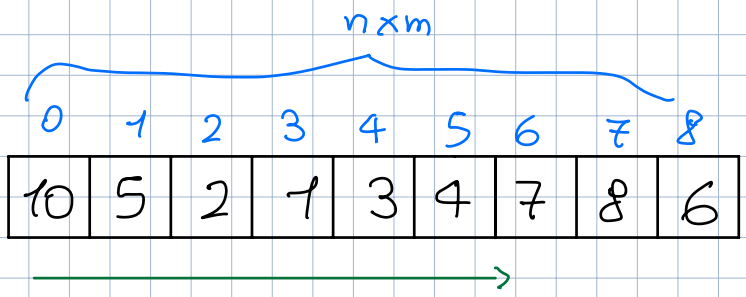


ES. 2

Creare una funzione in C che prenda in input una matrice nxm ed un vettore di dimensioni nxm. La funzione deve inserire all'interno del vettore tutti gli elementi presenti nella matrice, senza modificarne l'ordine.

ESEMPIO






```
void matToVet (int mat[][DIM], int dimX, int dimY, int vet[], int dimV)
{
  int z=0; // x VETTORE
  int i,j; // i = RIGHE , j = COLONNE
  For(i=0; i<dimX ; i++){
    For(j=0; j<dimY; j++){
      vet[z] = mat[i][j];
        z++;
    }
  }
}
```

ES. 3

Creare una funzione in C che prenda in input una matrice. La funzione ricerca il massimo valore e lo assegnerà all'interna matrice. La funzione deve poi tornare tale valore.

	0	1	2
0	10	5	2
1	1	3	4
2	7	8	16

(int)

	0	1	2
0	16	16	16
1	16	16	16
2	16	16	16

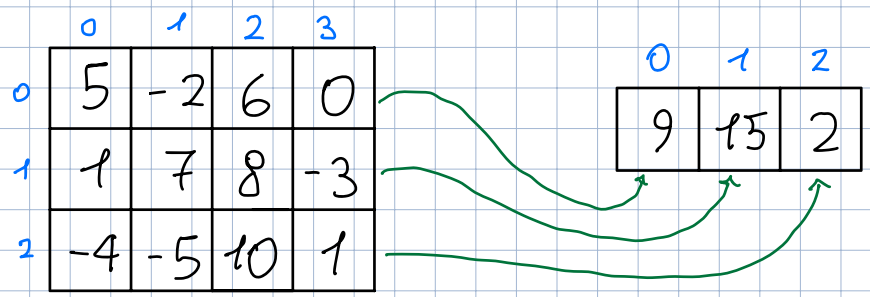
```
int matMax (int mat[ ][DIM], int dimX, int dimY ){
    int max = mat[0][0];  // int max = INT_MIN;
    int i, j;  // i = RIGHE , j = COLONNE
    For(i=0; i<dimX ; i++){
        For(j=0; j<dimY; j++){
            if(max < mat[i][j]){
                max = mat[i][j];
            }
        }
    }
    For(i=0; i<dimX ; i++){
        For(j=0; j<dimY; j++){
            mat[i][j] = max;
        }
    }
    return max;
}
```

TROVO IL MASSIMO

IMPOSTO IL MASSIMO SU TUTTA LA MATRICE

ES. 4

Creare una funzione in C che prenda in input una matrice e dimensioni nxm ed un vettore di dimensioni n.
La funzione dovrà sommare tutti gli elementi di ogni riga e inserirli, sommati, nella medesima posizione nel vettore.




```
void sommaMatToVet(int mat[][DIM], int dimX, int dimY, int vet[], int dimV)
{
    int somma = 0;
    int i, j;
    for(i=0; i<dimX; i++){    // PER LE RIGHE
        somma = 0;
        for(j=0; j<dimY; j++){    // PER LE COLONNE
            somma += mat[i][j];    // SOMMO LA COLONNA
        }
        vet[i] = somma;    // INSERISCO LA SOMMA NEL VETTORE
    }
}
```

ES. 5

Creare una funzione in C che prenda in input una matrice. La funzione dovrà chiedere in input all'utente dei valori interi. Questi valori devono essere univoci e non si devono ripetere all'interno della matrice stessa.

SI!

	0	1	2
0	1	2	5
1	0	-3	7
2	8	10	6

NO!

	0	1	2
0	1	2	5
1	0	1	7
2	8	10	6

```
void inputUnivoci (int mat[][DIM], int dimX, int dimY){
    int i=0, j=0, a, b;
    int val;
    int riga=0, colonna=0;
    int trovato=0; // trovato==1 -> PRESENTE NEL VETTORE
    while(i<dimX){
      while(j<dimY){ <- j=0;
        scanf("%d", &val);
        trovato=0;
        for(a=0; a<=i ; a++){
        ; for(b=0; b<=j ; b++){
        ;   ; if(val==mat[a][b]){
        ;   ;   ;   trovato=1;
        ;   ;   ;}
```

```
        ...}
      ...}
        if(trovato==0){ //NON TROVO IL VALORE NELLA MATRICE
          mat[i][j]=val;
          j++;
        ...}
      } //FINE WHILE PER LE COLONNE
      i++;
    } //FINE WHILE PER LE RIGHE
}
```

ES. 6

Creare una funzione in C che prenda in input una matrice di interi. La funzione deve verificare che la matrice passata sia una matrice simmetrica o meno. Quindi deve dare una stampa del fatto che la matrice sia o non sia simmetrica.

	0	1	2	3
0	-3	7	6	10
1	7	2	5	3
2	6	5	-1	4
3	10	3	4	0

↑

PER VEDERE COSÌ LA MATRICE:

```
For(i=0; i< dimX; i++){
   For(j=i; j<dimY; j++){
```

DEDUZIONI

→ LA MATRICE DEVE ESSERE QUADRATA

$(dimX == dimY)$

→ $\forall i \in dimX, \forall j \in dimY \mid$

$mat[i][j] == mat[j][i]$