Modulo: Allocazione dinamica della memoria in C [P2_05]

Unità didattica: Allocazione dinamica di matrici in C [2-C]

Titolo: Gestione di matrici allocate dinamicamente in C

Argomenti trattati:

- ✓ Richiami sul tipo matrice in C
- ✓ Allocazione di matrici in memoria: mappa di memorizzazione
- ✓ Gestione di matrici allocate dinamicamente

Prerequisiti richiesti: programmazione C (array, puntatori, funzioni C per l'allocazione dinamica)

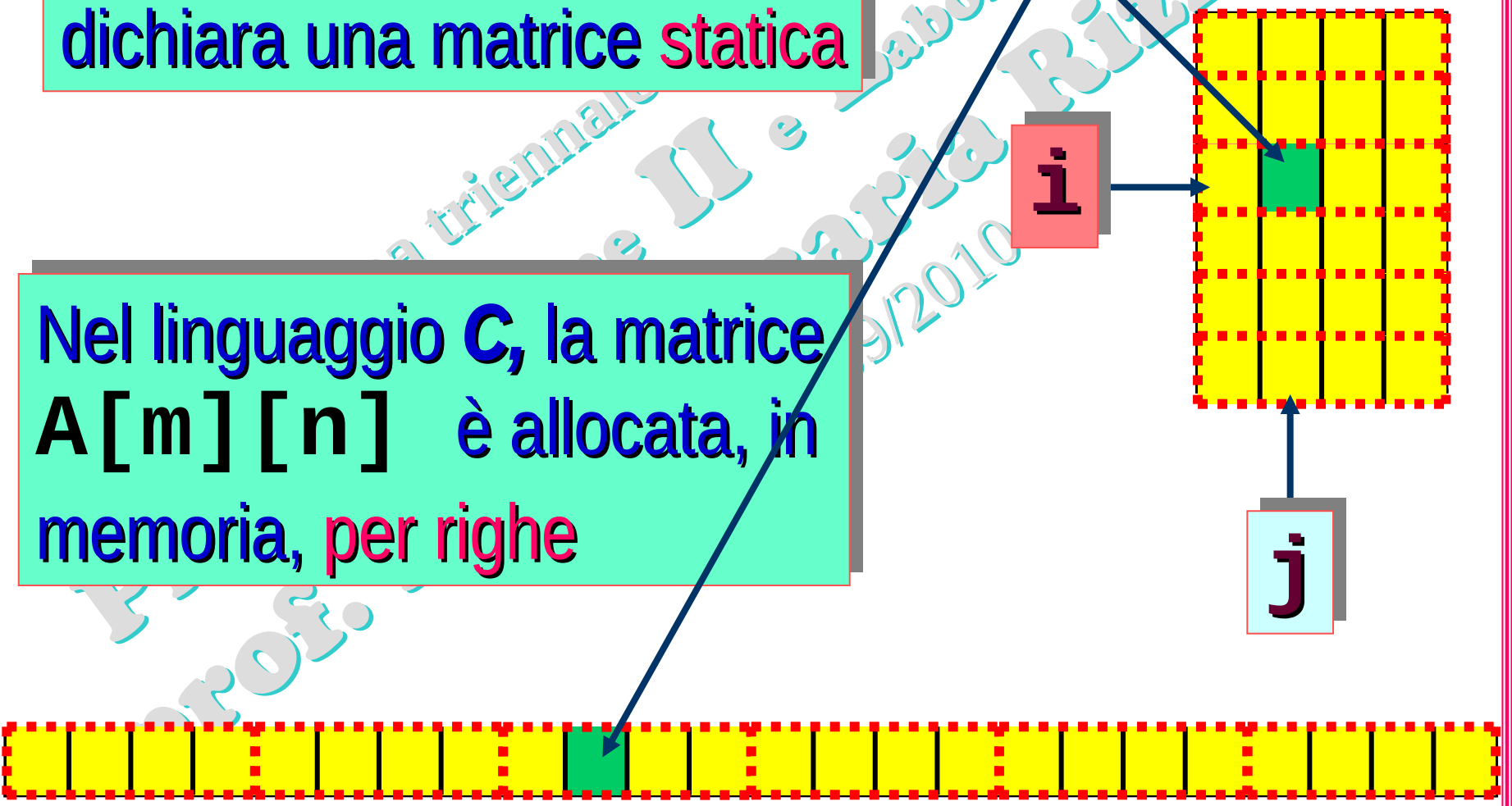
Nel linguaggio **C**

`type A[m][n]`

dichiara una matrice statica

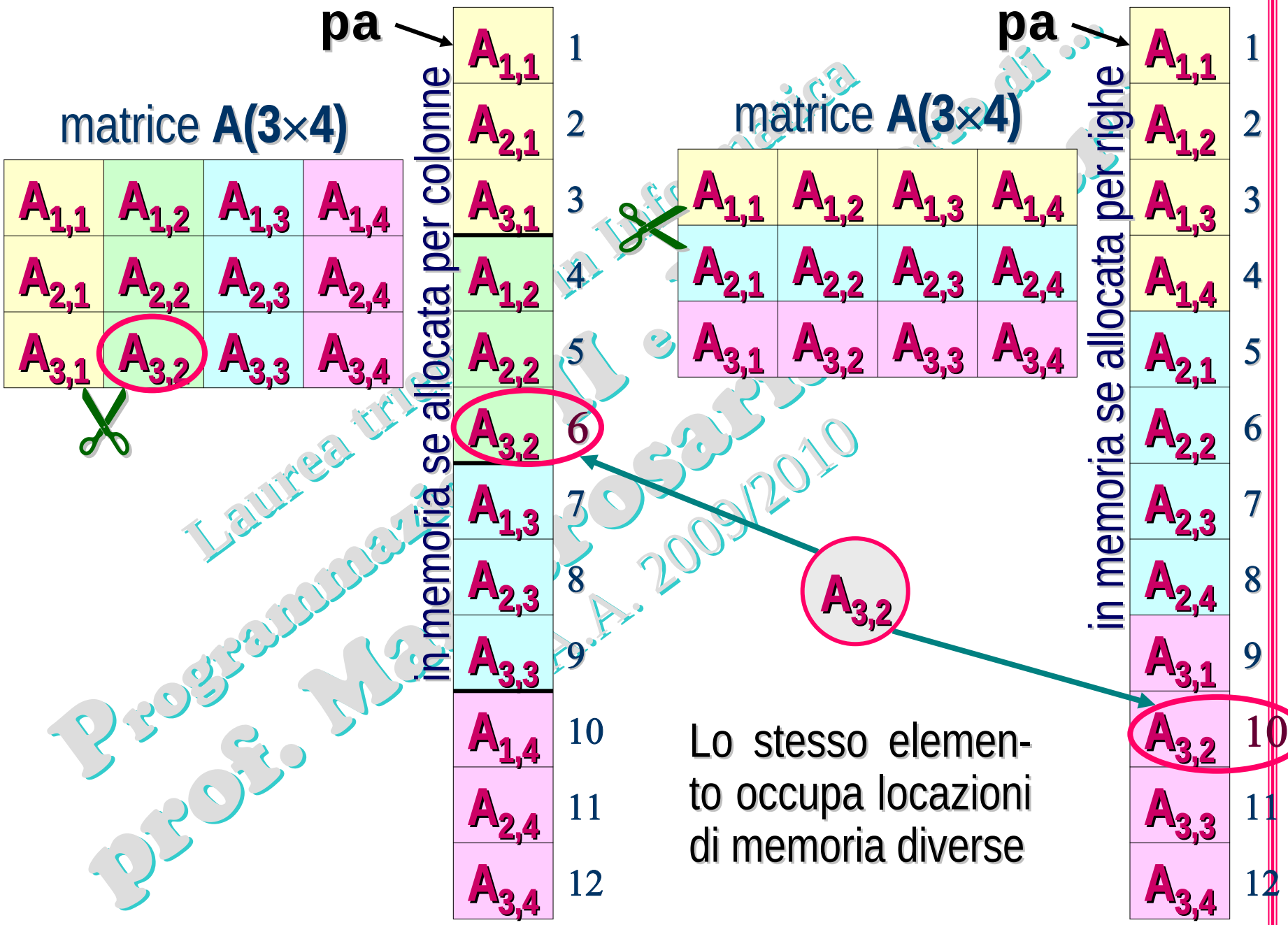
riferimento all'elemento

`A[i][j]`



Nel linguaggio **C**, la matrice `A[m][n]` è allocata, in memoria, per righe

che significa matrice allocata per righe o per colonne?



in C una matrice dinamica va gestita tramite puntatori

`type *pa`

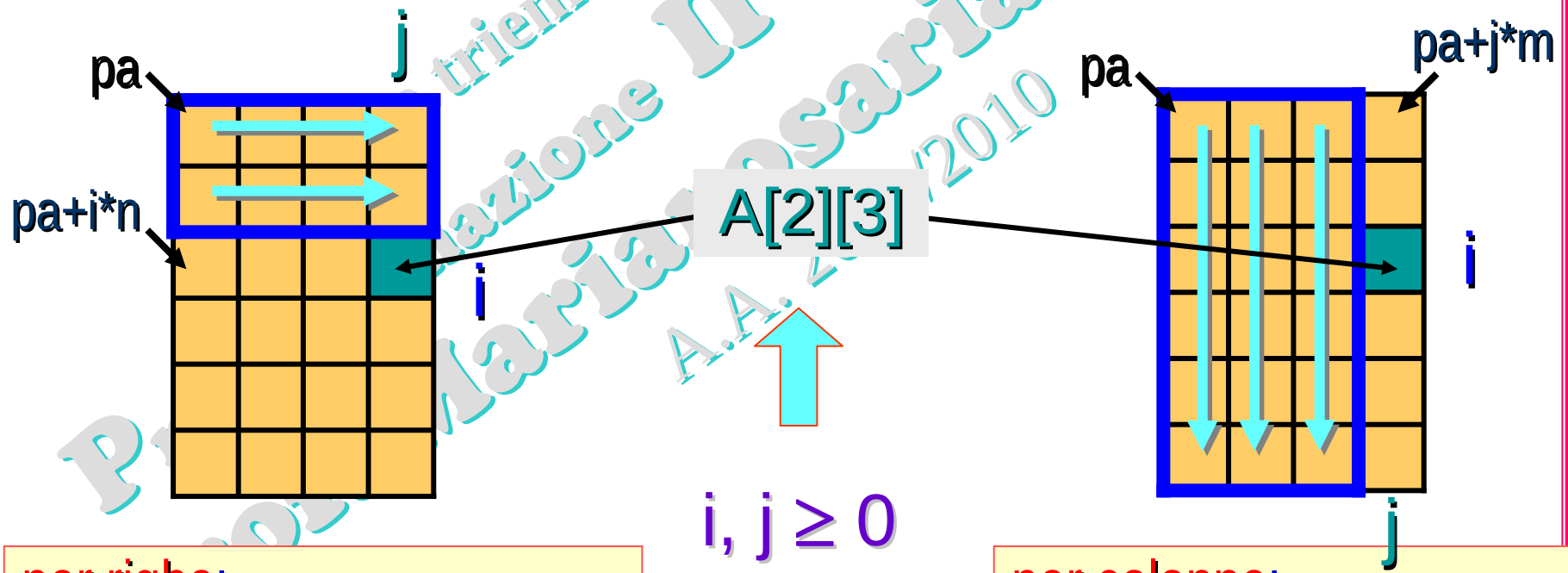
il riferimento all'elemento

`A[i][j]`

va tradotto su un array 1D stabilendo se allocare la matrice per righe o per colonne; se la matrice è A(m×n) allora

per righe: $A_{ij} \longleftrightarrow$ *(pa + i*n + j)

per colonne: $A_{ij} \longleftrightarrow$ *(pa + j*m + i)



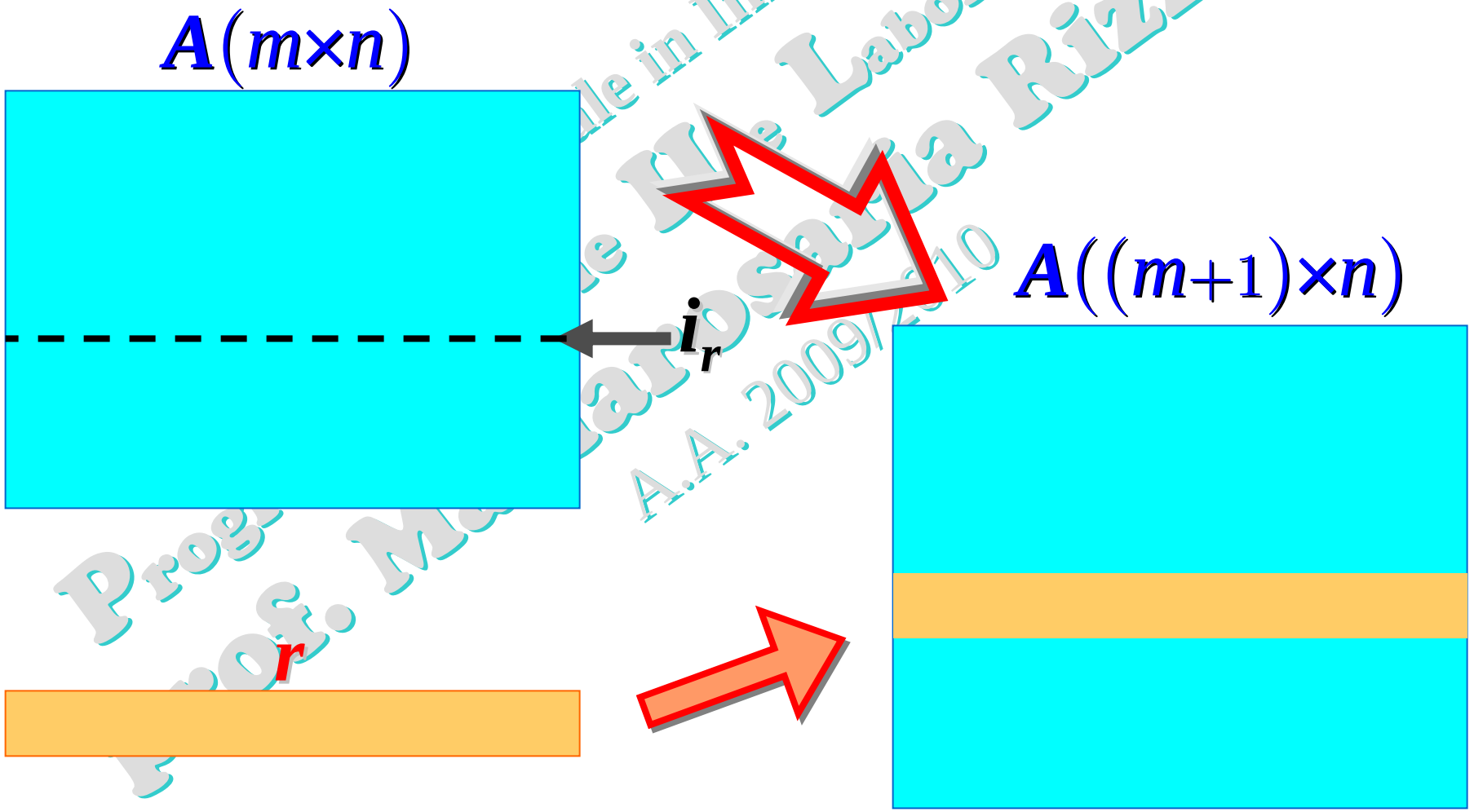
$i, j \geq 0$

per righe: $A_{ij} \longleftrightarrow$ *(pa + i*n + j)

per colonne: $A_{ij} \longleftrightarrow$ *(pa + j*m + i)

Esempio 1

Aggiungere una riga r nella matrice $A(m \times n)$ nella posizione i_r.



```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define M 4
#define N 8
void main()
{int a[M][N],*pa, *row; short i,j,irow;
 row=(int *)calloc(N,sizeof(int)); /* azzera *row  */
 printf("\nnumero riga da inserire  tra 1 e %d =",M);
 scanf("%d",&irow); irow--;
 srand( (unsigned)time( NULL ) );
 printf("RAND_MAX=%d\n",RAND_MAX);
 printf("\nmatrice statica A:\n");
 for (i=0; i<M; i++)
    {for (j=0; j<N; j++)
        {a[i][j]=rand()*100/RAND_MAX;
          printf("%d\t",a[i][j]);
    }
     printf("\n");
    }
...
```

statica

dinamica

Inizializza il seed del generatore di numeri casuali per generare numeri casuali ogni volta diversi

per avere 0 < numeri interi < 100

che differenza c'é con `rand()%100`?

```
 pa=malloc(M*N*sizeof(int));
 printf("\nmatrice dinamica *pa prima:\n");
 for (i=0; i<M; i++)
    {for (j=0; j<N; j++)
        {*(pa+i*N+j)=a[i][j];
         printf("%d\t",*(pa+i*N+j));
        }
     printf("\n");
    }

 pa=realloc(pa,(M+1)*N*sizeof(int));

 memmove(pa+(irow+1)*N,pa+irow*N,
                                  (M-irow)*N*sizeof(int));
memcpy(pa+irow*N, row, N*sizeof(int));

 printf("\nmatrice dinamica *pa dopo:\n");
 for (i=0; i<=M; i++)
    {for (j=0; j<N; j++) printf("%d\t",*(pa+i*N+j));
     printf("\n");
    }
}
```

aumenta le dimensioni

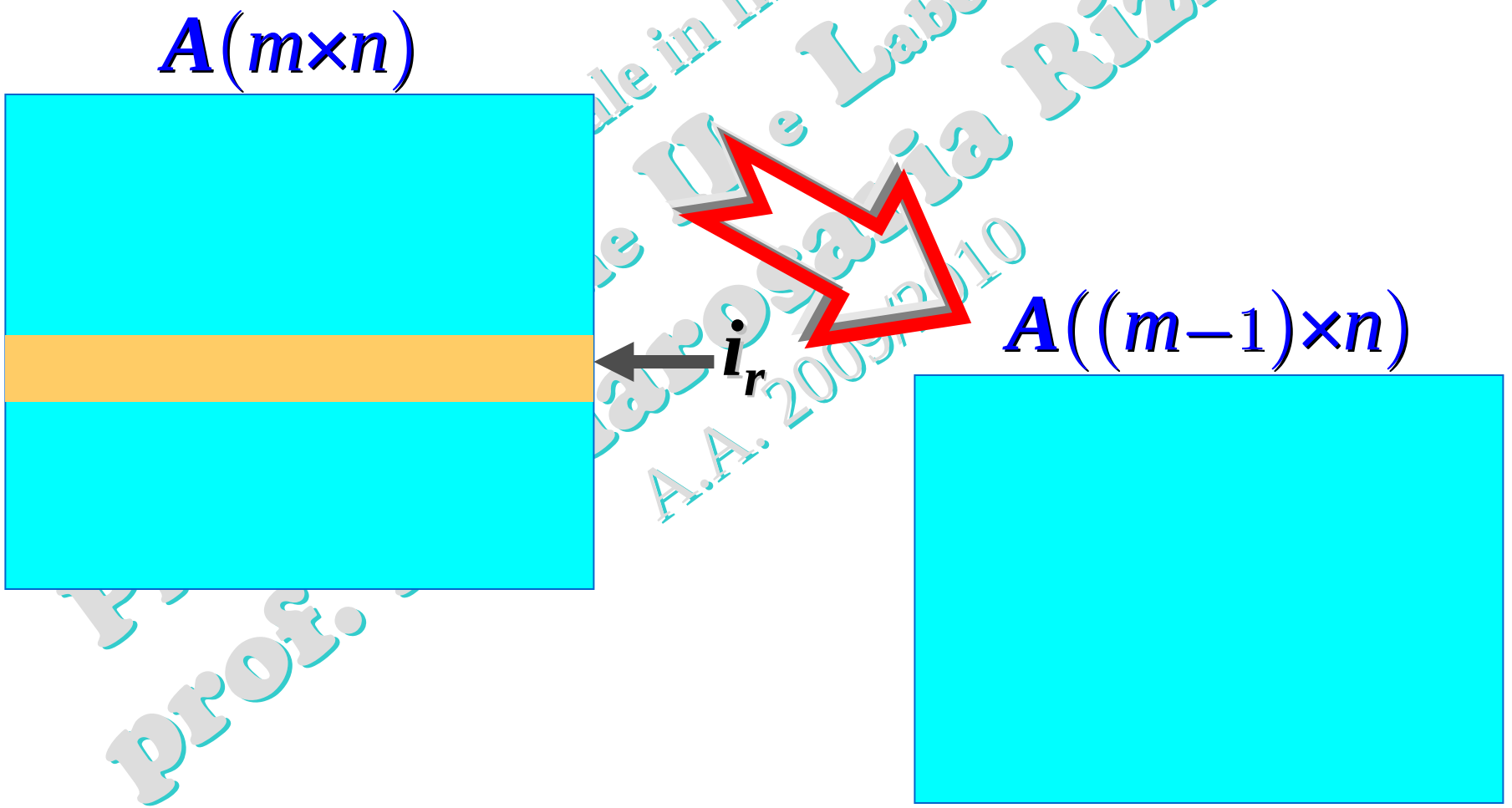
sposta la sottomatrice inferiore

copia la riga

```
0,0,0,0,0,0,0,0,
numero riga da inserire  tra 1 e 4 =2
RAND_MAX=32767

matrice statica A:
15      59      17      17      99      44      82      55
12      39      15      16      90      6       87      78
71      42      97      26      22      38      60      81
9       60      40      39      40      37      26      43

matrice dinamica *pa prima:
15      59      17      17      99      44      82      55
12      39      15      16      90      6       87      78
71      42      97      26      22      38      60      81
9       60      40      39      40      37      26      43

matrice dinamica *pa dopo:
15      59      17      17      99      44      82      55
0       0       0       0       0       0       0       0
12      39      15      16      90      6       87      78
71      42      97      26      22      38      60      81
9       60      40      39      40      37      26      43
```

Esempio 2

Eliminare la riga nella posizione i_r nella matrice $A(m \times n)$.



```
...
#define M 6
#define N 8
void main()
{int *pa; short i,j,irow;
 printf("\nnumero riga da eliminare  tra 1 e %d =",M);
 scanf("%d",&irow); irow--;
 pa=malloc(M*N*sizeof(int));
 srand((unsigned)time(NULL));
 printf("\nmatrice dinamica *pa prima:\n");
 for (i=0; i<M; i++)
    {for (j=0; j<N; j++)
        {*(pa+i*N+j)=rand()*100/RAND_MAX;
         printf("%d\t",*(pa+i*N+j));
        }
     printf("\n");
    }
memmove(pa+irow*N, pa+(irow+1)*N, (M-irow)*N*sizeof(int));
 pa=realloc(pa,(M-1)*N*sizeof(int));
 printf("\nmatrice dinamica *pa dopo:\n");
  ...
```

sposta la sottomatrice inferiore

diminuisce le dimensioni

```
numero riga da eliminare  tra 1 e 6 =2

matrice dinamica *pa prima:
89      13      58      21      52      86      87      84
20      4       2       23      85      66      34      0
44      88      74      85      5       38      64      53
9       58      65      25      14      41      12      79
87      56      84      7       32      83      73      48
10      52      17      79      85      52      59      81

matrice dinamica *pa dopo:
89      13      58      21      52      86      87      84
44      88      74      85      5       38      64      53
9       58      65      25      14      41      12      79
87      56      84      7       32      83      73      48
10      52      17      79      85      52      59      81
```

riga eliminata

Esercizio

A partire da una matrice A($m \times n$), del tipo sotto indicato, allocata per righe

1) staticamente
2) dinamicamente

visualizzarne gli elementi per colonne.

Esempio: $m=4, n=6$

$$A_{4\times 6} = \begin{pmatrix} 11 & 12 & 13 & 14 & 15 & 16 \\ 21 & 22 & 23 & 24 & 25 & 26 \\ 31 & 32 & 33 & 34 & 35 & 36 \\ 41 & 42 & 43 & 44 & 45 & 46 \end{pmatrix}$$