

## Unità didattica: Applicazione delle liste ad altre strutture dati [8-AT]

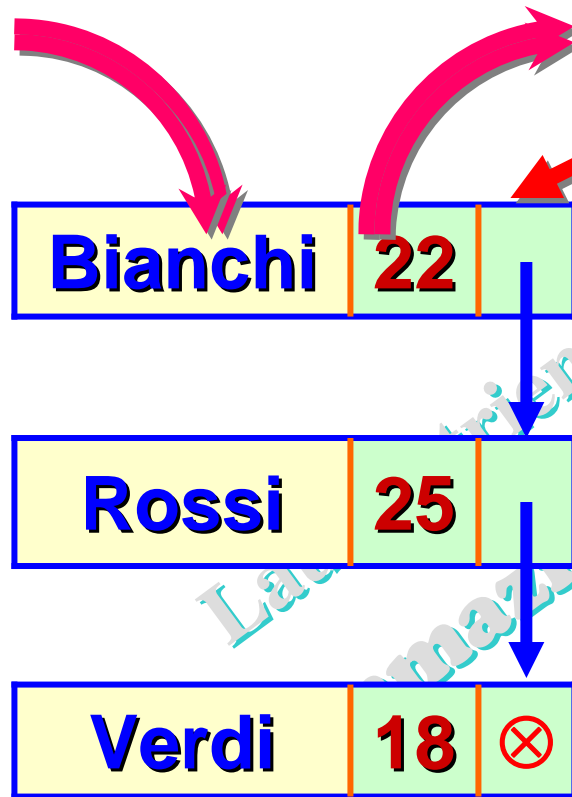
**Titolo:** Uso delle liste per la realizzazione di altre strutture dati lineari

Argomenti trattati:

- ✓ Pila e coda mediante la struttura dati lista lineare
- ✓ Lista circolare e lista bidirezionale
- ✓ Lista multipla (multilista)
- ✓ Matrici sparse rappresentate come liste multiple con accesso indipendente sia per riga e sia per colonna
- ✓ Matrici sparse rappresentate come liste multiple con accesso esclusivo solo per riga oppure solo per colonna

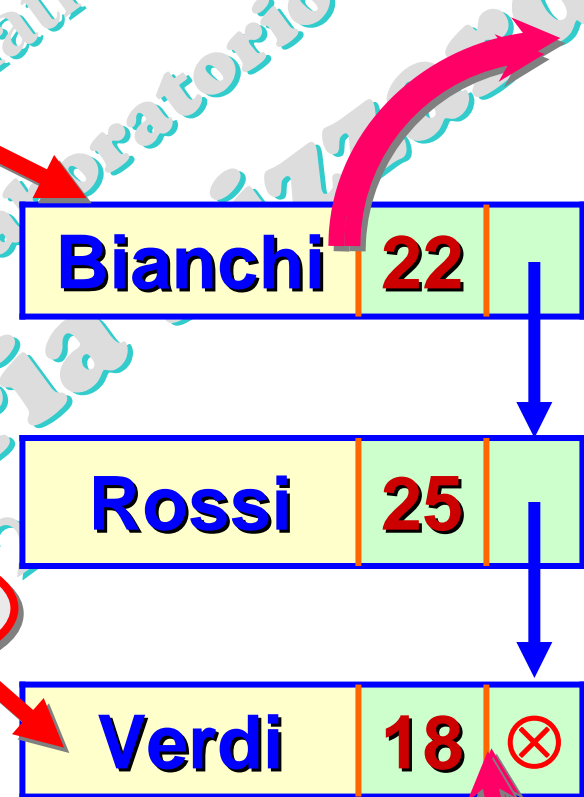
**Prerequisiti richiesti:** implementazione C di una lista lineare, strutture dati dinamiche lineari, matrici

**pila (o stack)**  
(mediante lista)



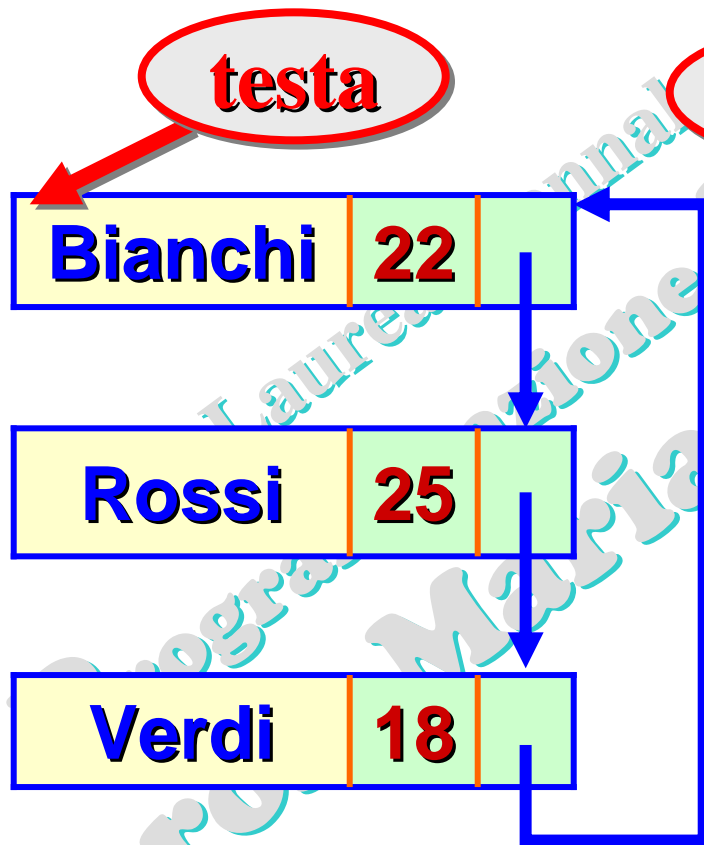
Lista con inserimento / eliminazione sempre all'inizio.

**coda (o queue)**  
(mediante lista)

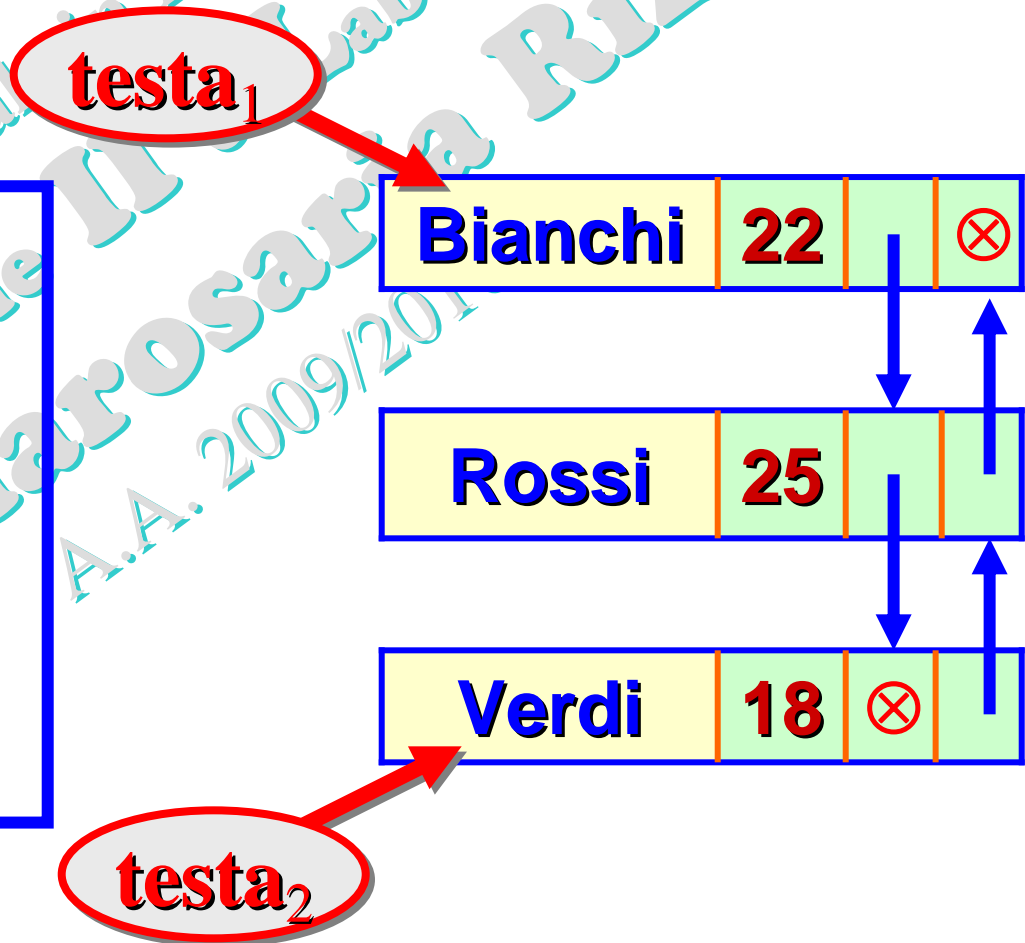


Lista con inserimento sempre all'inizio, eliminazione sempre alla fine.

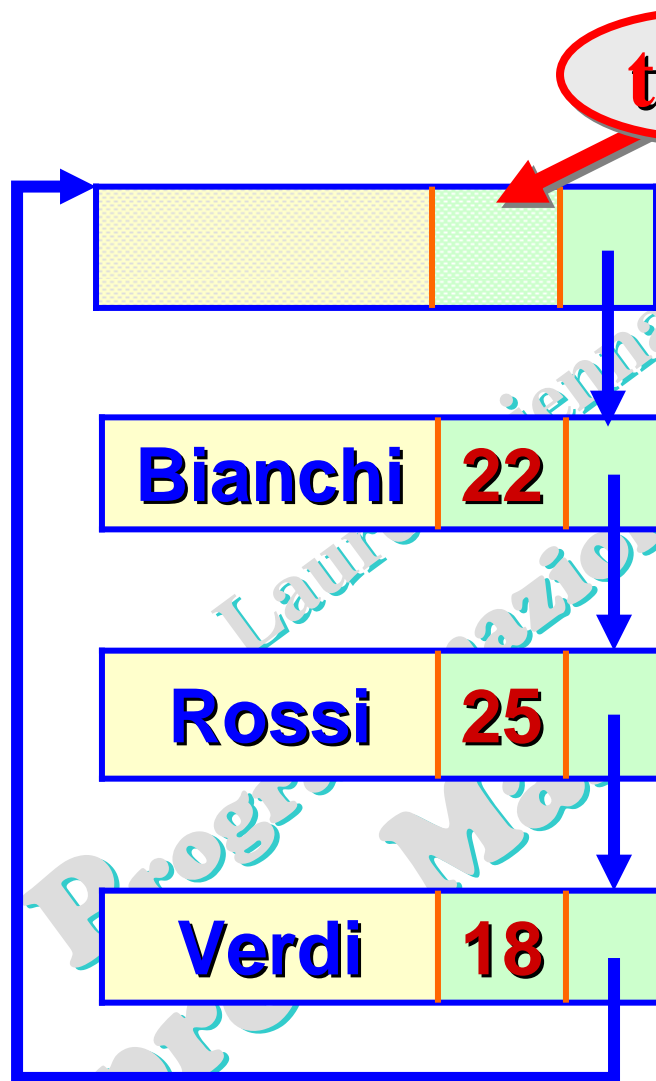
## lista circolare (o catena)



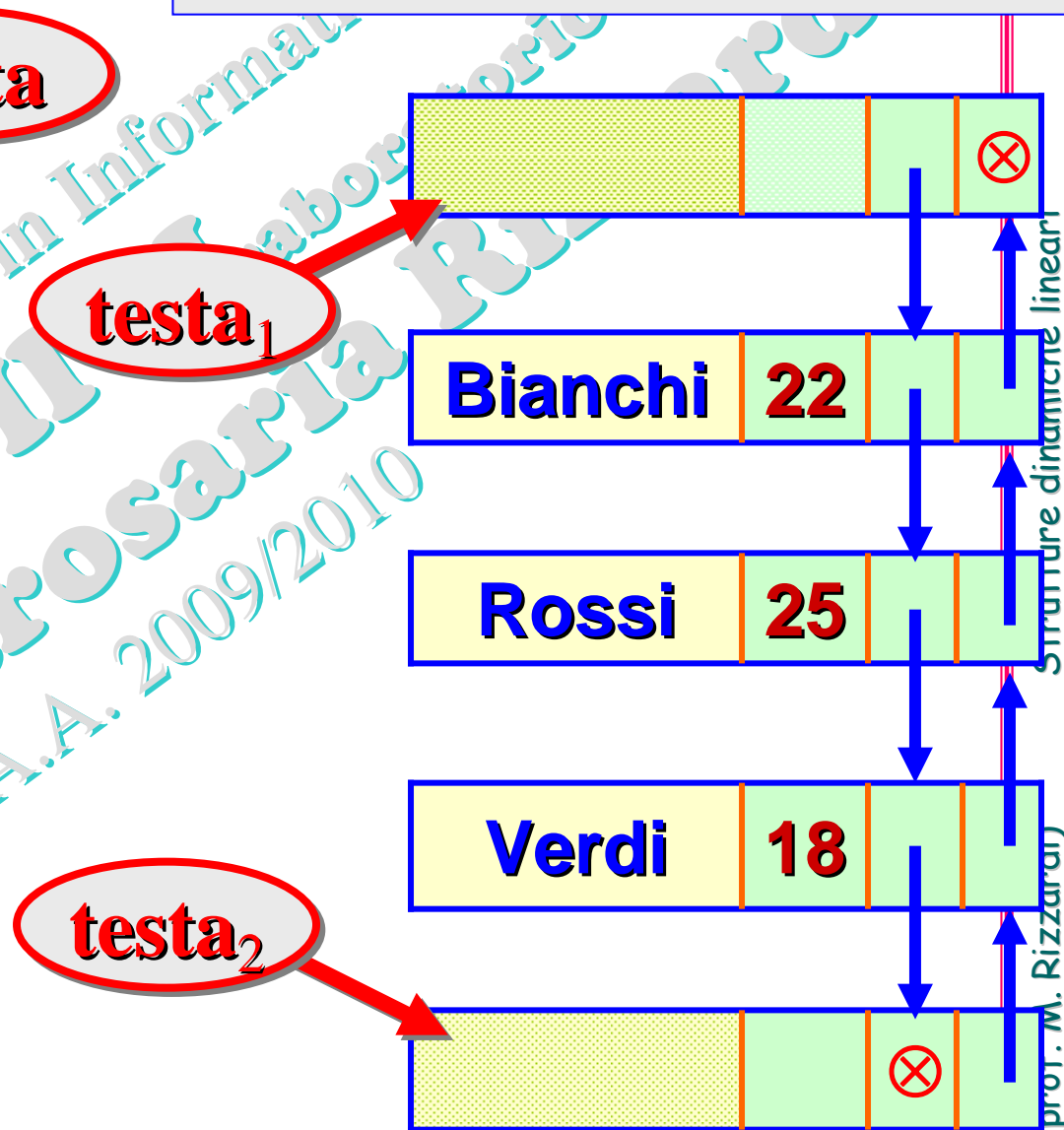
## lista bidirezionale (o simmetrica)



lista circolare (o catena)  
con sentinella



lista bidirezionale (o dop-  
piamente concatenata)  
con sentinella



## Esercizio:

1

Realizzare in *C* le funzioni per la gestione delle precedenti strutture dati:

- pila, coda [liv. 1]
- catena, lista bidirezionale [liv. 2]

mediante *linked list* dinamica e generica con e senza nodo sentinella.

# Esempio di liste multiple: rappresentazione di matrici sparse

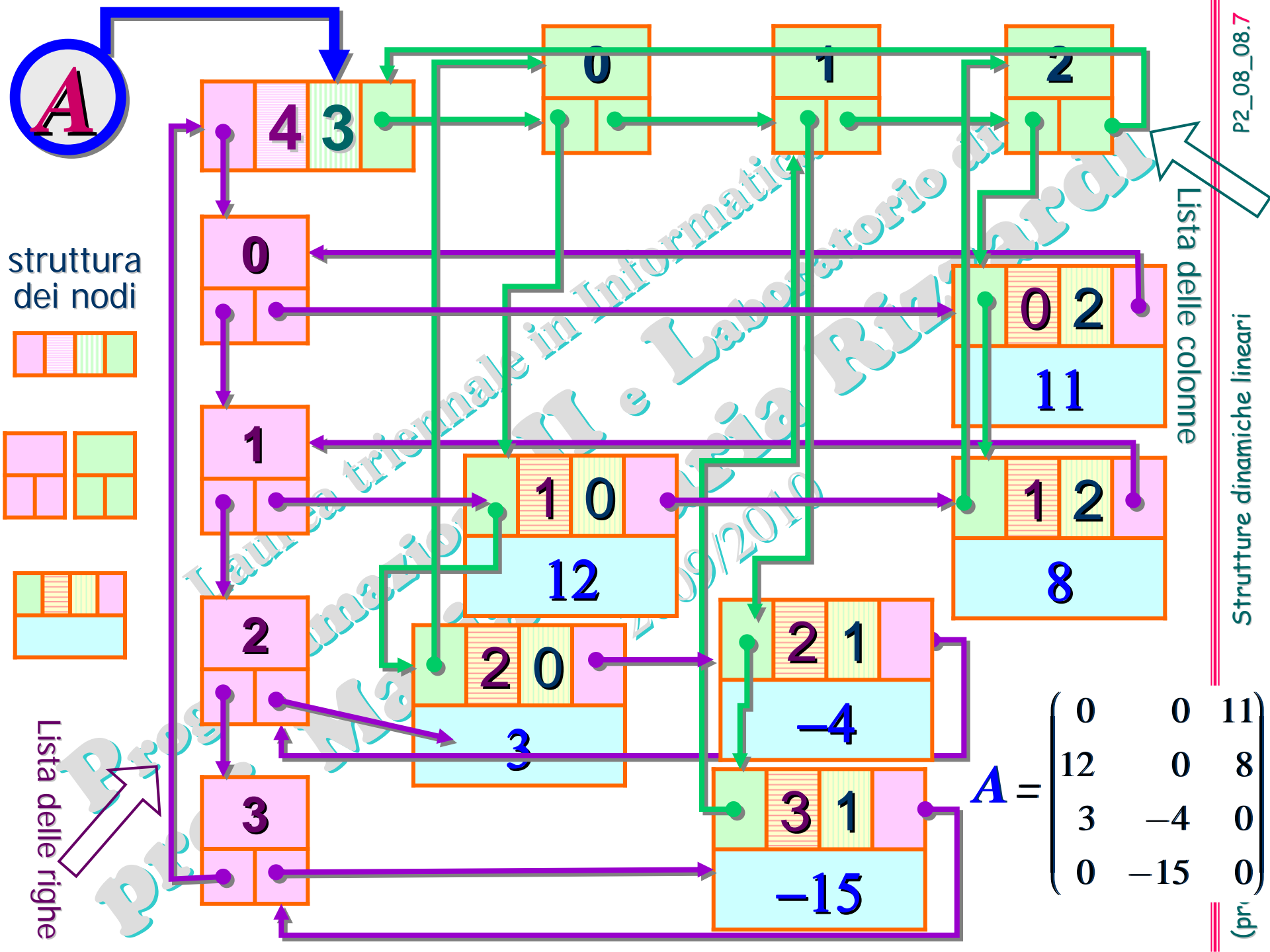
$$A = \begin{pmatrix} 0 & 0 & 11 \\ 12 & 0 & 8 \\ 3 & -4 & 0 \\ 0 & -15 & 0 \end{pmatrix}$$

in generale una  
matrice sparsa ha

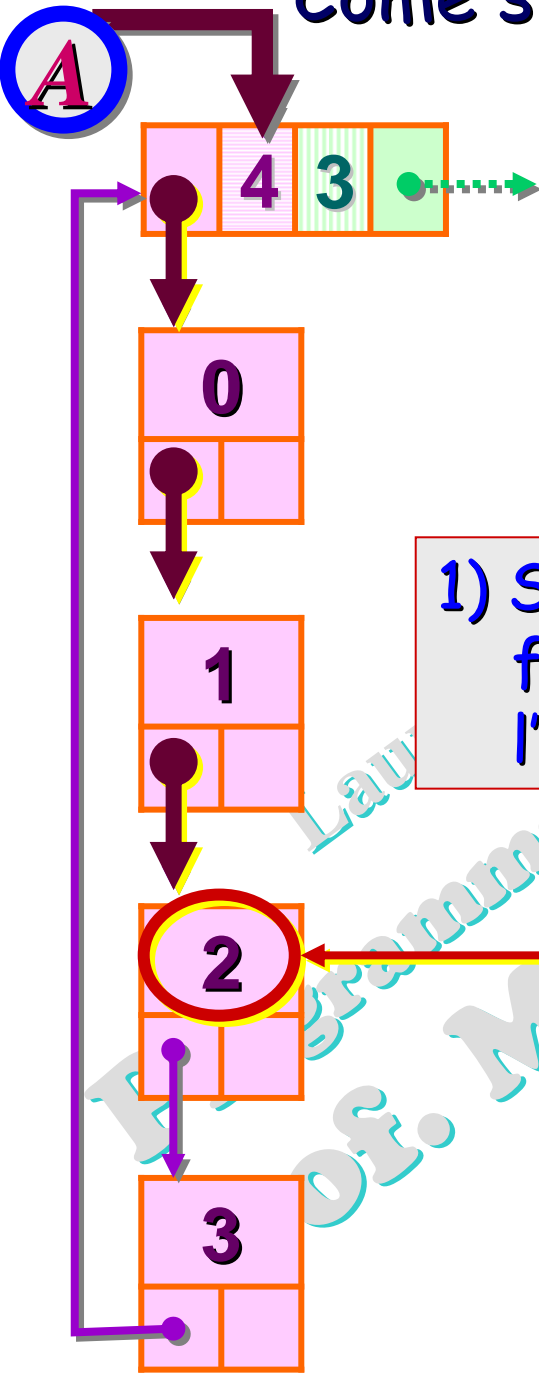
**...molti zeri!!!**

per cui non conviene me-  
morizzarla come array 2D

Come rappresentare una **matrice sparsa** in  
memoria (**senza memorizzare gli zeri**)  
consentendo di accedere ai suoi elementi  
**indifferentemente per righe e per colonne?**



# Come si **accede per righe** ad un elemento?



$$A = \begin{pmatrix} 0 & 0 & 11 \\ 12 & 0 & 8 \\ 3 & -4 & 0 \\ 0 & -15 & 0 \end{pmatrix}$$

$A(2,1)$  points to the element  $-4$  in the matrix.

1) Si avanza sulla lista dei puntatori alle **righe** fino a trovare quella cui appartiene l'elemento cercato



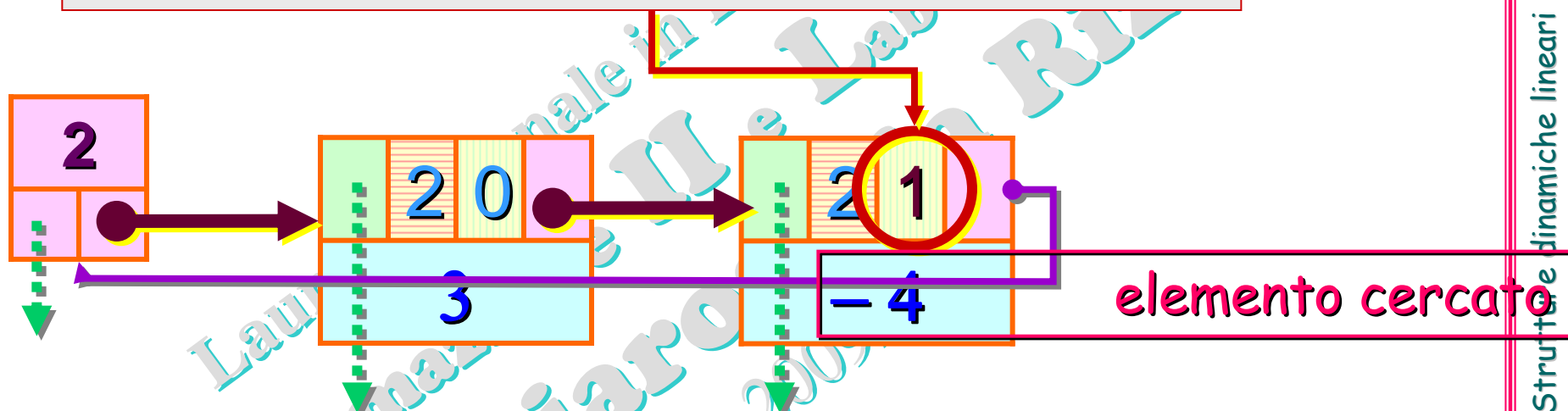
Come si accede per righe ad un elemento?

$A(2,1)$

$$A = \begin{pmatrix} 0 & 0 & 11 \\ 12 & 0 & 8 \\ 5 & -4 & 0 \\ 0 & -15 & 0 \end{pmatrix}$$

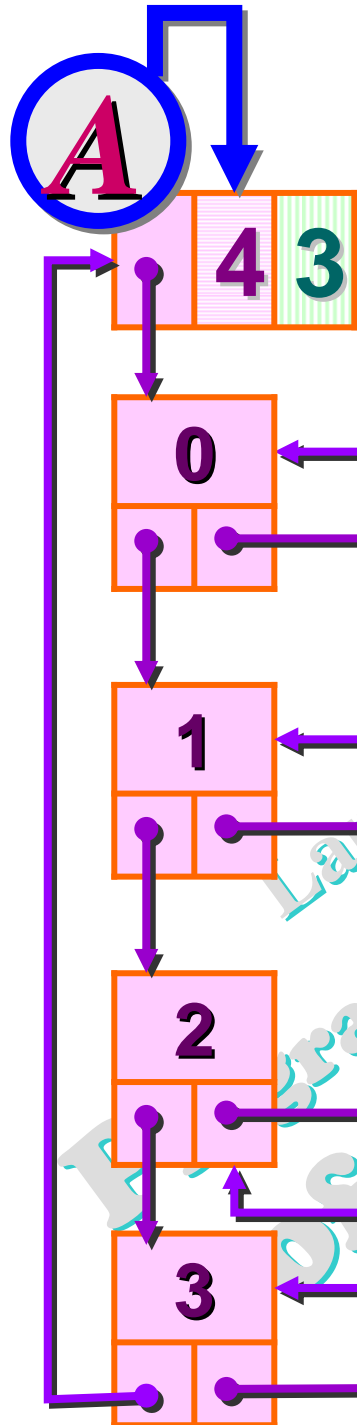
Diagram showing access to element  $A(2,1)$  (row 2, column 1). A red arrow points from the row index '2' to the second row, and another red arrow points from the column index '1' to the first column. The element  $-4$  is highlighted with a red box.

2) Si scorre la lista della riga cercando l'indice di colonna dell'elemento voluto

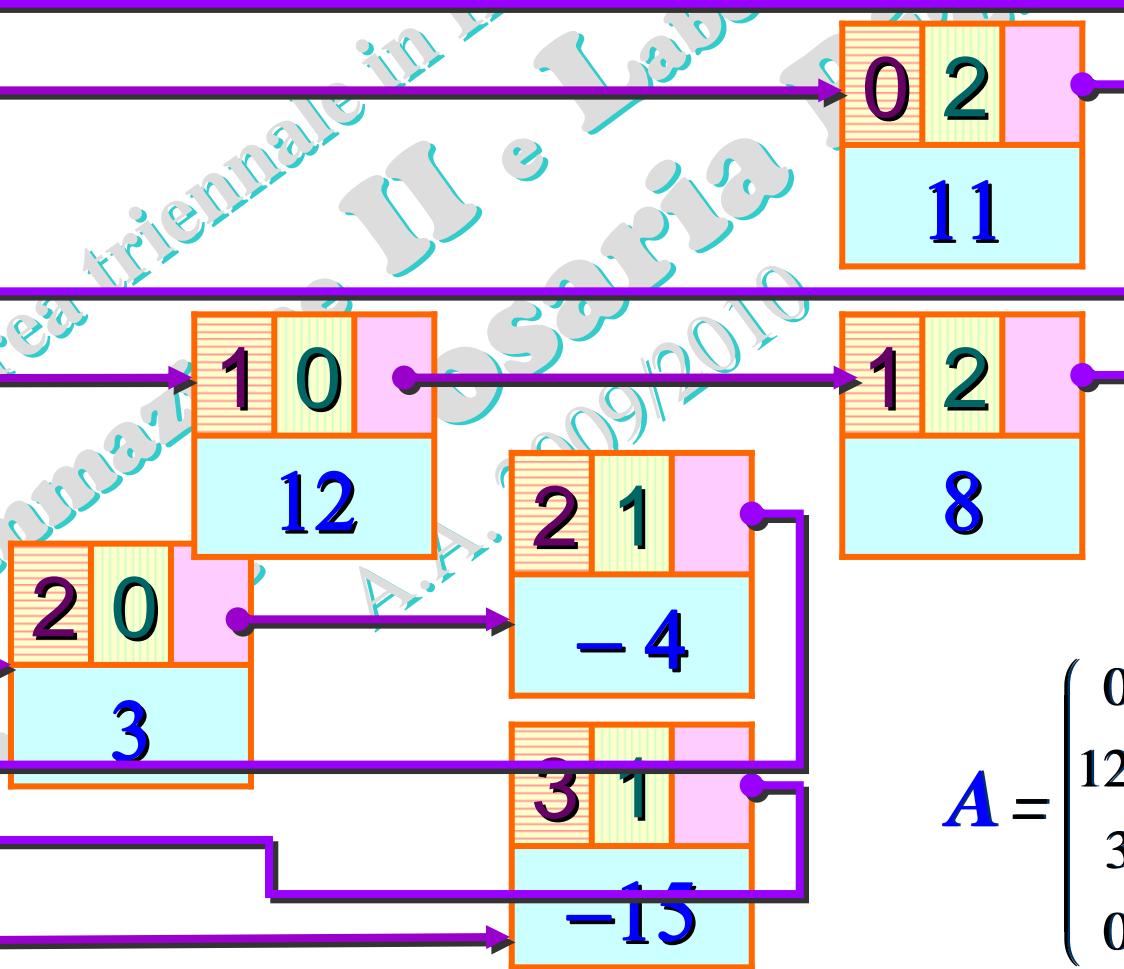


3) Se l'indice di colonna è presente allora si è trovato l'elemento cercato.

Se l'indice di colonna non è presente allora l'elemento cercato è zero!



Invece la rappresentazione di una **matrice sparsa** in memoria che consente l'accesso agli elementi **esclusivamente per righe** è ...



$$A = \begin{pmatrix} 0 & 0 & 11 \\ 12 & 0 & 8 \\ 3 & -4 & 0 \\ 0 & -15 & 0 \end{pmatrix}$$

# Esercizio:

1

Implementare in **C** il prodotto righe $\times$ colonne di due matrici sparse rappresentate tramite liste multiple. Scegliere per ciascuna matrice la rappresentazione più idonea. [liv. 3]